

ECML 2007 PRDD  
WARSAW POLAND

THE 18<sup>TH</sup> EUROPEAN CONFERENCE ON MACHINE LEARNING  
AND  
THE 11<sup>TH</sup> EUROPEAN CONFERENCE ON PRINCIPLES AND PRACTICE  
OF KNOWLEDGE DISCOVERY IN DATABASES

---

EXPLORING THE POWER  
OF LINKS IN DATA MINING  
TUTORIAL NOTES

---

presented by  
**Jiawei Han, Xiaoxin Yin,  
and Philip S. Yu**

**September 17, 2007**

**Warsaw, Poland**

**Prepared and presented by:**

*Jiawei Han*

University of Illinois at Urbana-Champaign, USA

*Xiaoxin Yin*

Google Inc., USA

*Philip S. Yu*

IBM T.J. Watson Research Center, USA



# Exploring the Power of Links in Data Mining

**Jiawei Han<sup>1</sup>, Xiaoxin Yin<sup>2</sup>, and Philip S. Yu<sup>3</sup>**

1: University of Illinois at Urbana-Champaign

2: Google Inc.

3: IBM T.J. Watson Research Center


ECML/PKDD, Warsaw, Poland, 17–21 September 2007

1

## Outline



Theme: “Knowledge is power, but knowledge is hidden in massive links”

- Link Mining: A General Introduction 
- CrossMine: Classification of Multi-relations by Link Analysis
- CrossClus: Clustering over Multi-relations by User-Guidance
- LinkClus: Efficient Clustering by Exploring the Power Law Distribution
- Distinct: Distinguishing Objects with Identical Names by Link Analysis
- TruthFinder: Conformity to Truth with Conflicting Information
- Conclusions and Future Work

2

# Link Mining: An Introduction

---

- Links: Relationships among data objects
- Link analysis and Web search
  - Web search: A typical similarity query does not work since it may return  $> 10^6$  results! And rank cannot just be based on the keyword similarity
  - Solution: Explore the semantic information carried in hyperlinks
- Exploring the power of links in Web search
  - PageRank: Capturing page popularity (Brin & Page 1998)
  - HITS: Capturing authorities & hubs (Chakrabarti, Kleinberg, et al. 1998 & 1999)
- The success of Google demonstrates the power of links

3

# Information Networks

---

- Information network: A network where each node represents an entity (e.g., actor in a social network) and each link (e.g., tie) a relationship between entities
  - Each node/link may have attributes, labels, and weights
  - Link may carry rich semantic information
- Homogeneous vs. heterogeneous networks
  - Homogeneous networks
    - Single object type and single link type
    - Single model social networks (e.g., friends)
    - WWW: a collection of linked Web pages
  - Heterogeneous networks
    - Multiple object and link types
    - Medical network: patients, doctors, disease, contacts, treatments
    - Bibliographic network: publications, authors, venues

4

## Measure Information Networks: Centrality

- Degree centrality: measure how popular an actor is
  - $C_d(i)$ : Node degree normalized with the maximal degree  $n - 1$  (where  $n$  is the total # of actors)
- Closeness (or distance) centrality: How can the actor  $i$  easily interact with all other actors
- Betweenness centrality: The control of  $i$  over other pairs of actors
  - # of shortest paths that pass  $i$ , ( $p_{jk}(i)$ ) normalized by the total # of shortest paths of all pairs not including  $i$
  - *i.e.*,  $C_b(i) = \sum_{j < k} (p_{jk}(i)/p_{jk})$
  - Can be further normalized

5

## Measure Information Networks: Prestige

- Degree prestige: popular if an actor receives many in-links
  - $P_D(i) = d_i(i)/n-1$ , where  $d_i(i)$  is the in-degree (# of in-links) of  $i$ , and  $n$  is the total # of actors in the network
- Proximity prestige: considers the actors directly or indirectly linked to actor  $l$ 
  - $P_P(i) = (|I_i|/(n-1)) / (\sum_{j \in I_i} (d(j,i)/|I_i|))$ , where  $|I_i|/(n-1)$  is the proportion of actors that can reach actor  $l$ ,  $d(j, i)$  is the shortest path from  $j$  to  $l$ , and  $I_i$  be the set of actors that can reach  $i$ .
- Rank prestige: considers the prominence of individual actors who do the “voting”
  - $P_R(i) = A_{1i} P_R(1) + A_{2i} P_R(2) + \dots + A_{ni} P_R(n)$ , where  $A_{ji} = 1$  if  $j$  points to  $l$ , or 0 otherwise
  - In matrix form for  $n$  actors, we have  $\mathbf{P} = \mathbf{A}^T \mathbf{P}$ . Thus  $\mathbf{P}$  is the eigen vector of  $\mathbf{A}^T$

6

## Co-Citation and Bibliographic Coupling

- Co-citation: if papers  $i$  and  $j$  both cited by paper  $k$ 
  - Co-citation: # of papers that co-cite  $i$  and  $j$
  - $C_{ij} = \sum_{k=1}^n L_{ki} L_{kj}$  where  $L_{ij} = 1$  if paper  $i$  cites paper  $j$ , otherwise 0.
  - Co-citation matrix: a square matrix  $\mathbf{C}$  formed with  $C_{ij}$
- Bibliographic coupling: if papers  $i$  and  $j$  both cite paper  $k$ 
  - Bibliographic coupling: # of papers that are cited by both  $i$  and  $j$
  - $B_{ij} = \sum_{k=1}^n L_{ik} L_{jk}$  where  $L_{ij} = 1$  if paper  $i$  cites paper  $j$ , otherwise 0.
  - Bibliographic coupling matrix  $\mathbf{B}$  formed with  $B_{ij}$
- Hubs and authorities, found by HITS algorithm are directly related to co-citation and bibliographic coupling matrices

7

## PageRank: Capturing Page Popularity (Brin & Page'98)

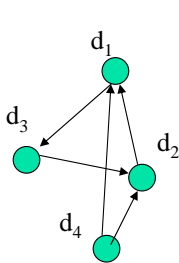
- A hyperlink from page  $x$  to  $y$  is an implicit conveyance of authority to the target page
- Pages point to page  $y$  also have their own prestige scores
- A page may point to many other pages and thus its prestige score should be shared among all the pages that it points to
  - Thus the page rank score of page  $i$ :  $P(i) = \sum_{(j,i) \in E} (P(j)/O_j)$ , where  $O_j$  is the # of out-links of page  $j$ .
- Let  $\mathbf{P}$  be the  $n$ -dimensional column vector of PageRank values
  - $\mathbf{P} = (P(1), P(2), \dots, P(n))^T$
- Let  $A_{ij} = 1/O_i$  if  $(i, j)$  in  $E$ , or 0 otherwise
  - Then  $\mathbf{P} = \mathbf{A}^T \mathbf{P}$ .  $\mathbf{P}$  is the principal eigenvector
- PageRank can also be interpreted as random surfing (thus capturing popularity), and the eigen-system equation can be derived using the Markov chain model

8

# The PageRank Algorithm (Brin & Page'98)

Random surfing model:

At any page,  
 with prob.  $\alpha$ , randomly jumping to a page  
 with prob.  $(1 - \alpha)$ , randomly picking a link to follow



$$M = \begin{bmatrix} 0 & 0 & 1/2 & 1/2 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1/2 & 1/2 & 0 & 0 \end{bmatrix}$$

“Transition matrix”

Same as  $\alpha/N$  (why?)

$$p_{i+1}(d_i) = (1 - \alpha) \sum_{d_j \in N(d_i)} m_{ji} p_i(d_j) + \alpha \sum_k \frac{1}{N} p_i(d_k)$$

$$p(d_i) = \sum_k \left[ \frac{1}{N} \alpha + (1 - \alpha) m_{ki} \right] p(d_k)$$

Stationary (“stable”) distribution, so we ignore time

$$\bar{p} = (\alpha I + (1 - \alpha) M)^T \bar{p} \quad \mathbf{I}_{ij} = 1/N$$

Initial value  $p(d) = 1/N$

Iterate until converge

Essentially an eigenvector problem....

9

## PageRank: Strength and Weakness

- Advantages:
  - Ability to fight spams: It is not easy for a webpage owner to add in-links to his page from an important page
  - A global measure, query-independent: computed offline
- Disadvantages
  - Query-independent: may not be authoritative on the query topic
  - Google may have other ways to deal with it
  - Tend to favor older pages: with more and stable links
- Timed PageRank: Favors newer pages
  - Add a damping factor which is a function  $f(t)$  where  $t$  is the difference between the current time and the time the page was last updated
  - For a complete new page in a website, which does not have in-links, use avg rank value of the pages in the website

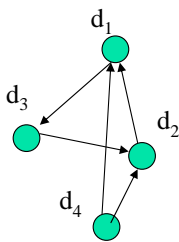
10

# HITS: Capturing Authorities & Hubs (Kleinberg'98)

- Intuition
  - Different from literature citations, many rivals, such as Toyota and Honda, do not cite each other on the Internet
  - Pages that are widely cited (i.e., many in-links) are good **authorities**
  - Pages that cite many other pages (i.e., many out-links) are good **hubs**
  - Authorities and hubs have a **mutual reinforcement** relationship
- The key idea of HITS (Hypertext Induced Topic Search)
  - Good authorities are cited by good hubs
  - Good hubs point to good authorities
  - Iterative reinforcement ...

11

## The HITS Algorithm (Kleinberg 98)



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

“Adjacency matrix”

$$h(d_i) = \sum_{d_j \in OUT(d_i)} a(d_j)$$

$$a(d_i) = \sum_{d_j \in IN(d_i)} h(d_j)$$

Initial values:  $a = h = 1$

Iterate

Normalize:

$$\bar{h} = A\bar{a}; \quad \bar{a} = A^T\bar{h}$$

$$\bar{h} = AA^T\bar{h}; \quad \bar{a} = A^T A\bar{a}$$

$$\sum_i a(d_i)^2 = \sum_i h(d_i)^2 = 1$$

Again eigenvector problems...

12

# HITS: Strength and Weakness

- Advantages: Rank pages according to the query topic
- Disadvantages
  - Does not have anti-spam capability: One may add out-links to his own page that points to many good authorities
  - Topic-drift: One may collect many pages that have nothing to do with the topic — by just pointing to them
  - Query-time evaluation: expensive
- Later studies on HIT improvements
  - SALA [Lemple & Moran, WWW'00], a stochastic alg, two Markov chains, an authority and a hub Markov chain, less susceptible to spam
  - Weight the links [Bharat & Henzinger SIGIR'98]: if there are k edges from documents on a first host to a single document on a second host, give each edge an authority weight of  $1/k$ , ...
  - Handling topic drifting: Content similarity comparison, or segment the page based on the DOM (Document Object Model) tree structure to identify blocks or sub-trees that are more relevant to query topic

13

# Vision-based Page Segmentation and Block-Based Web Page Rank (Cai et al. 04)

- VIPS: Vision based Page Segmentation: partition a web page into blocks
- Block-based web search: A web page contains multiple semantics, separated by blocks
- Block-level link analysis: Extract page-to-block, block-to-page relationships (block-level PageRank and block-level HITS) –One application: Web image search and organization

14

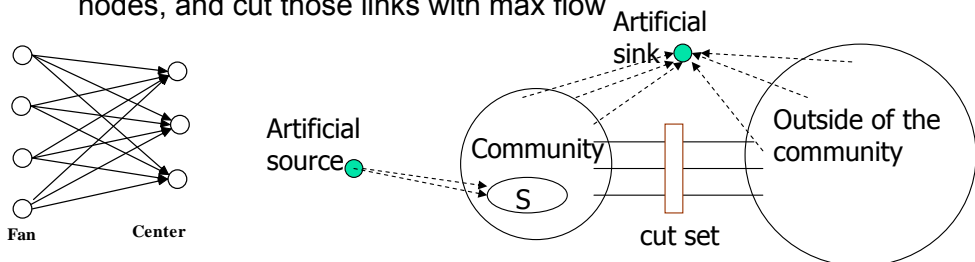
# Community Discovery

- Community: A group of entities,  $G$ , (e.g., people or organization) that shares a common theme,  $T$ , or is involved in an activity or event
  - Communities may overlap, and may have hierarchical structures (i.e., sub-community and sub-theme)
- Community discovery:
  - Given a data set containing entities, discover hidden communities (i.e., theme represented by a set of key words, and its members)
  - Different communities: Web pages (hyperlinks, or content words), e-mails (email exchanges, or contents), and text documents (co-occurrence of entities, appearing in the same sentence and/or document, or contents)

15

## Community Discovery Methods (I)

- HITS finds dense bipartite graph communities based on broad topic queries (but needs eigenvector computation)
- Bi-partite core communities [Kumar et al, WWW'99]
  - (F, C) bi-partite, where Fans are like hubs and Centers authorities
  - 1. pruning by in-degree (too many in-links) and pruning of fans and centers
  - 2. generating all (i, j) cores (but finds only the cores)
- Maximum flow communities [Flake, et al' Computer'02]
  - Use max-flow min-cut algorithm, add artificial source and sink nodes, and cut those links with max flow



16

## Community Discovery Methods (II)

---

- Email communities based on betweenness
  - An edge  $x-y$  if there is a minimum # of messages between them
  - The *betweenness* of an edge is the # of shortest paths that pass it
  - Graph partitioning is based on “max-flow min-cut”
- Overlapping communities of named entities
  - Overlapping community: A person can belong to > 1 community
  - Find entity communities from a text corpus
    - Build a link graph: find named entities in the sentence, link them and attach other keywords
    - Find all triangles: A triangle – three entities bound together
    - Find community cores: a group of tightly bound triangles
    - Cluster around community cores: assign triangles and pairs not in any core to cores according to their textual content similarities
  - Rank entities in each community according to degree centrality
  - Keywords associated with the edges of each community are also ranked, and the top ranked one as the theme of the community

17

## Link-Based Object Classification

---

- Predicting the category of an object based on its attributes, links, and the attributes of linked objects
- **Web:** Predict the category of a web page, based on words that occur on the page, links between pages, anchor text, html tags, etc.
- **Citation:** Predict the topic of a paper, based on word occurrence, citations, co-citations
- **Epidemics:** Predict disease type based on characteristics of the patients infected by the disease
- **Communication:** Predict whether a communication contact is by email, phone call or mail

18

## Other Link Mining Tasks

- Group detection: Cluster the nodes in the graph into groups, e.g., community identification
  - Methods: Hierarchical clustering, blockmodeling, spectral graph partitioning, multi-relational clustering
- Entity Resolution (a.k.a., deduplication, reference reconciliation, object consolidation): Predict whether two objects are the same, based on their attributes and links
- Link prediction: Predict whether a link exists
- Link cardinality estimation: Predict # of links to an object or # of objects reached along a path from an object
- Subgraph discovery: Find common subgraphs
- Metadata mining: Schema mapping, schema discovery, schema reformulation

19

## Link Mining Challenges

- Logical (e.g., links) vs. statistical dependencies
- Feature construction (e.g., by aggregation & selection)
- Modeling instances (more predictive) vs. classes (more general)
- Collective classification: need to handle correlation
- Collective consolidation: Using a link-based statistical model
- Effective use of links between labeled & unlabeled data
- Prediction of prior link probability: model links at higher level
- Closed vs. open world: We may not know all related factors

Challenges common to any link-based statistical model, such as Bayesian Logic Programs, Conditional Random Fields, Probabilistic Relational Models, Relational Markov Networks, Relational Probability Trees, Stochastic Logic Programming.

20

# Outline

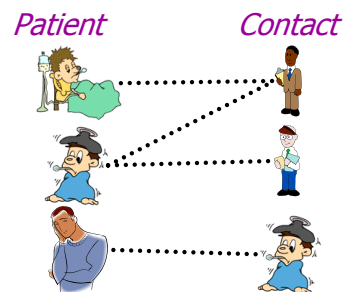
Theme: "Knowledge is power, but knowledge is hidden in massive links"

- Link Mining: A General Introduction
- CrossMine: Classification of Multi-relations by Link Analysis
- CrossClus: Clustering over Multi-relations by User-Guidance
- LinkClus: Efficient Clustering by Exploring the Power Law Distribution
- Distinct: Distinguishing Objects with Identical Names by Link Analysis
- TruthFinder: Conformity to Truth with Conflicting Information
- Conclusions and Future Work

21

## CrossMine: Classification of Multi-Relations

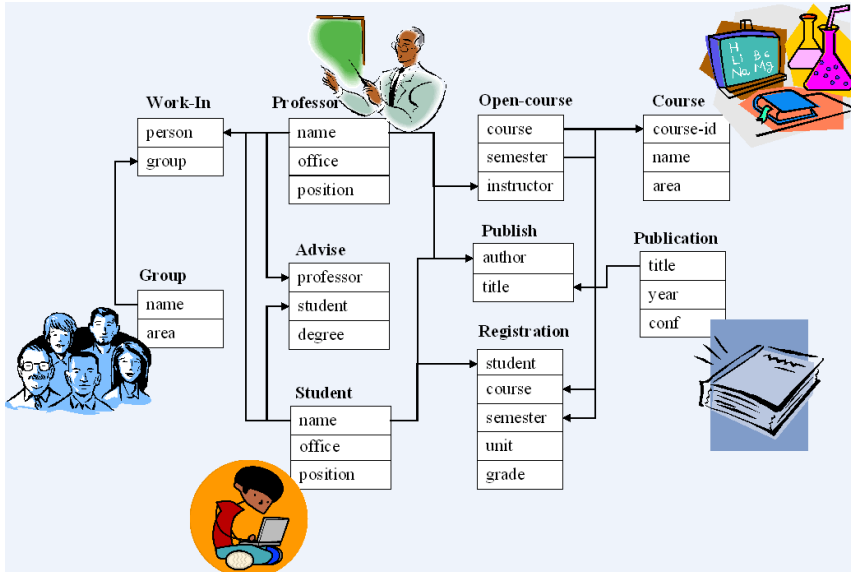
- Multi-relational data mining
  - X. Yin, J. Han, J. Yang, and P. S. Yu, "CrossMine: Efficient Classification across Multiple Database Relations", ICDE'04
- Directly utilize information in multiple relations
  - E.g., predict the disease of each patient
- Multi-relational information
  - Objects joinable in different relations
  - Linkages between different objects



22

# Data Mining May Involve Many Relations

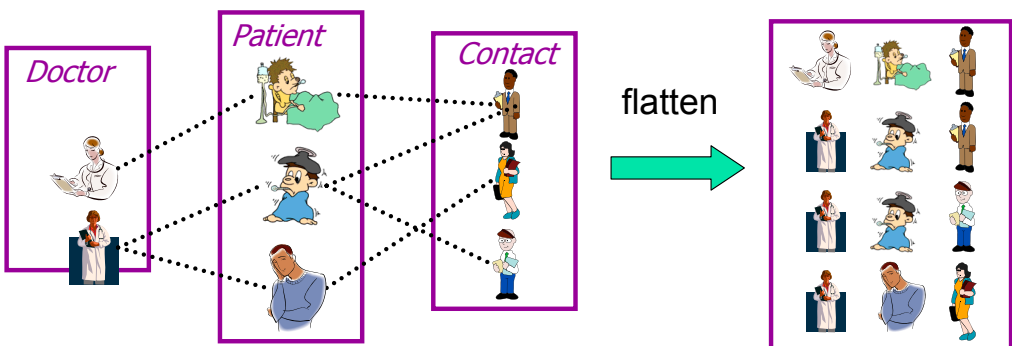
- Most structured data in the real world is stored in multiple relations



23

# Single Table vs. Multiple Relations

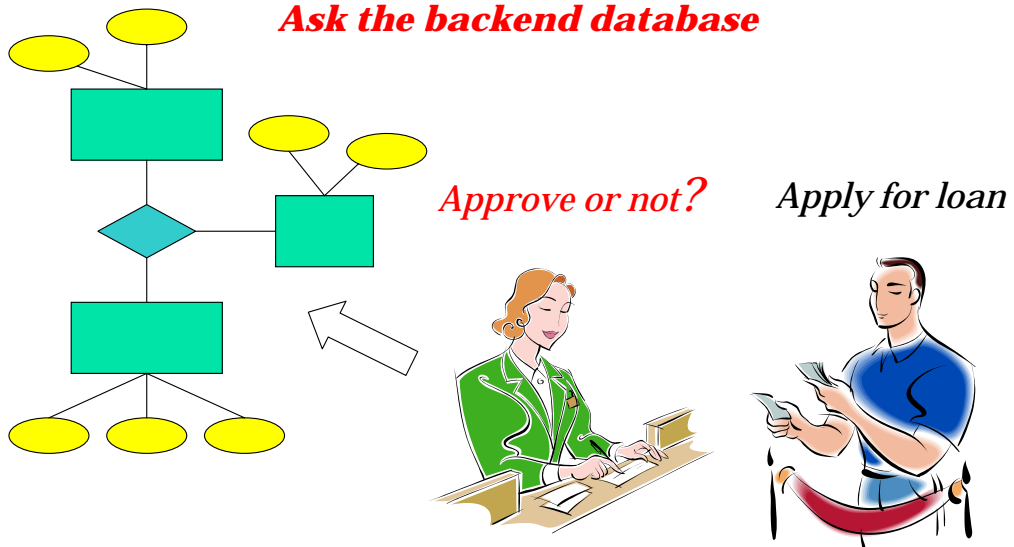
- Should we merge them into one relation in mining?



- Cannot utilize information of database structures or schemas
- Lose linkage information
- Joined relation may be much larger than original relations

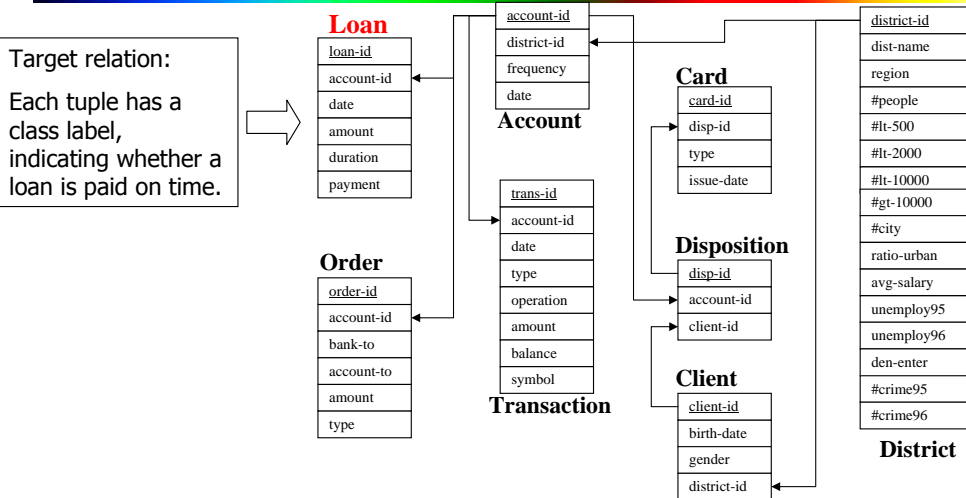
24

# An Example: Loan Applications



25

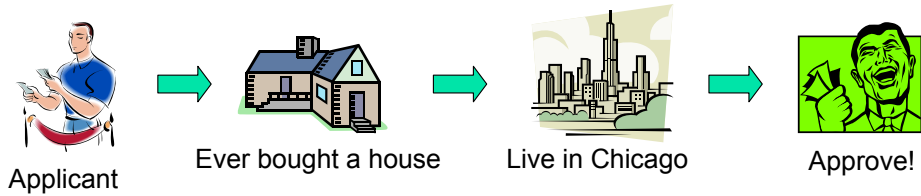
# The Backend Database



How to make decisions to loan applications?

26

# Rule-based Classification



- A rule uses information in multiple relations

27

# Tuple ID Propagation



Loan ID	Account ID	Amount	Duration	Decision
1	124	1000	12	+
2	124	4000	12	+
3	108	10000	24	-
4	45	12000	36	-
5	45	2000	24	+

Account ID	Frequency	Open date	Propagated ID	Labels
124	monthly	02/27/93	1, 2	2+, 0-
108	weekly	09/23/97	3	0+, 1-
45	monthly	12/09/96	4, 5	1+, 1-
67	weekly	01/01/97	Null	0+, 0-

Possible predicates:

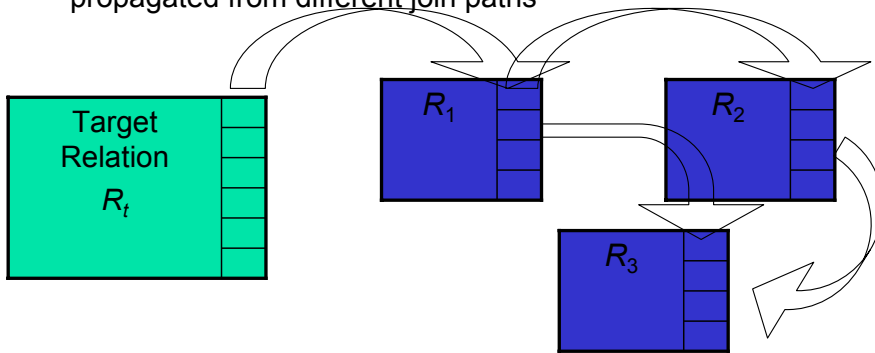
- Frequency='monthly': 3 +, 1 -
- Open date < 01/01/95: 2 +, 0 -

- Propagate tuple IDs of target relation to non-target relations
- Virtually join relations to avoid the high cost of physical joins

28

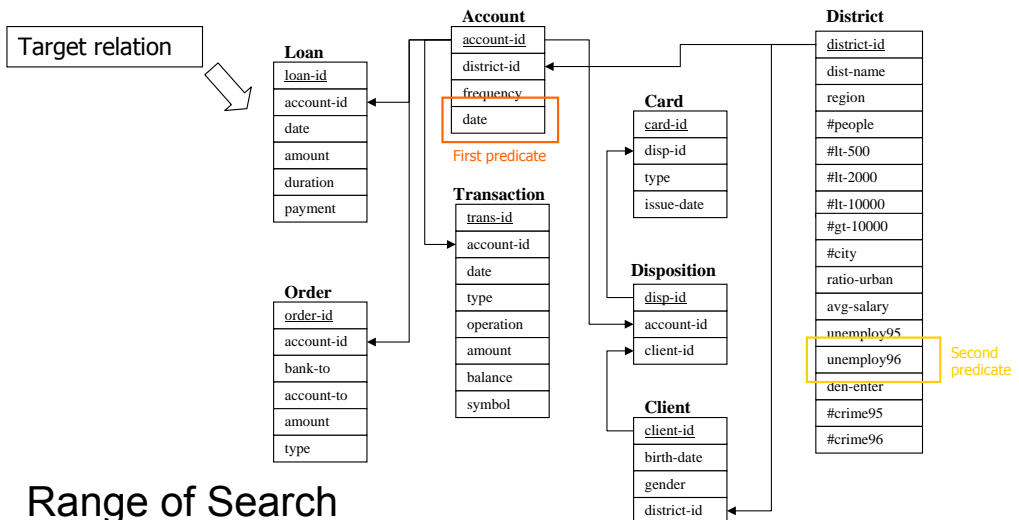
# Tuple ID Propagation

- Efficient
  - Only propagate the tuple IDs
  - Time and space usage is low
- Flexible
  - Can propagate IDs among non-target relations
  - Many sets of IDs can be kept on one relation, which are propagated from different join paths



29

# Rule Generation: Example



Range of Search

Add best predicate to rule

30

# Real Dataset

- PKDD Cup 99 dataset – Loan Application

	Accuracy	Time
FOIL	74.0%	3338 sec
TILDE	81.3%	2429 sec
CrossMine	89.8%	17.4 sec
CrossMine w. sampling	87.5	13.9

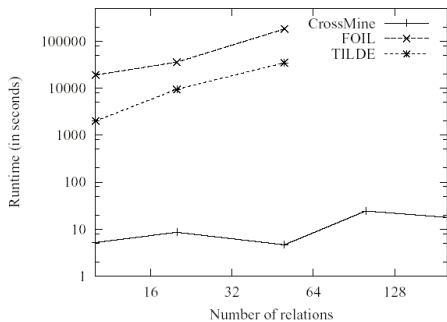
- Mutagenesis dataset (4 relations)

	Accuracy	Time
FOIL	79.7%	1.65 sec
TILDE	89.4%	25.6 sec
CrossMine	87.7%	0.83 sec

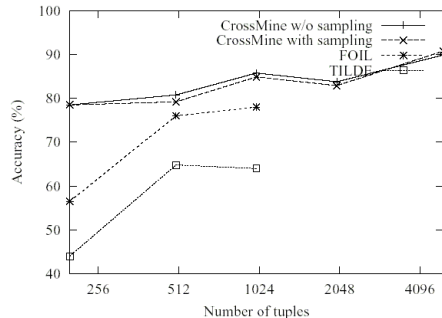
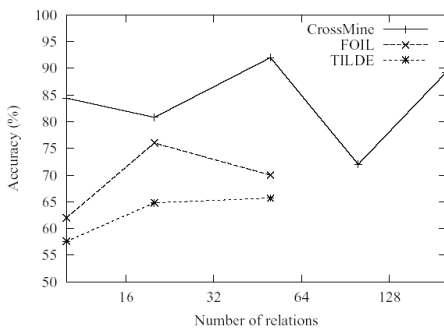
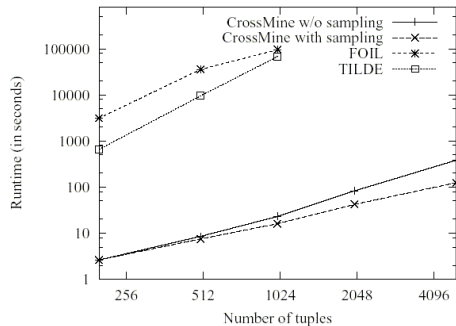
31

## Synthetic datasets:

Scalability w.r.t. number of relations



Scalability w.r.t. number of tuples



32

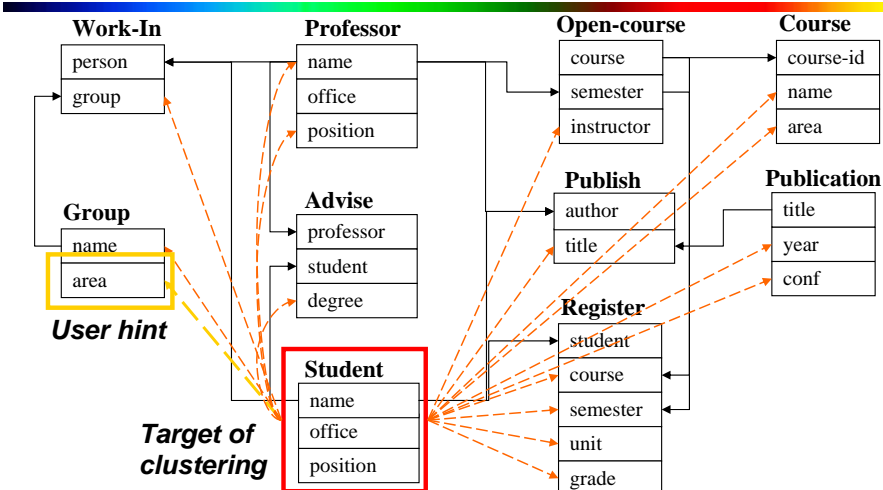
# Outline

Theme: "Knowledge is power, but knowledge is hidden in massive links"

- Link Mining: A General Introduction
- CrossMine: Classification of Multi-relations by Link Analysis
- CrossClus: Clustering over Multi-relations by User-Guidance
- LinkClus: Efficient Clustering by Exploring the Power Law Distribution
- Distinct: Distinguishing Objects with Identical Names by Link Analysis
- TruthFinder: Conformity to Truth with Conflicting Information
- Conclusions and Future Work

33

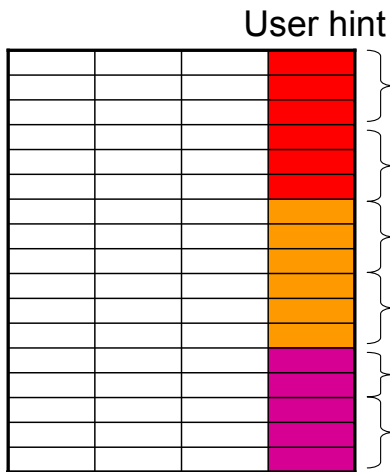
## Why User-Guided Clustering?



- X. Yin, J. Han, P. S. Yu, "Cross-Relational Clustering with User's Guidance", KDD'05
- User usually has a goal of clustering, e.g., clustering students by research area
- User specifies his clustering goal to CrossClus

34

# Comparing with Classification

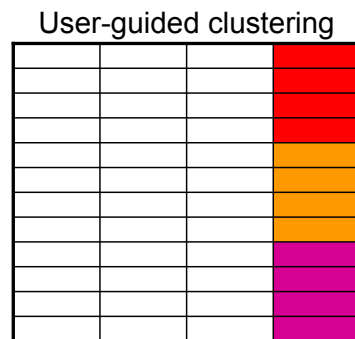
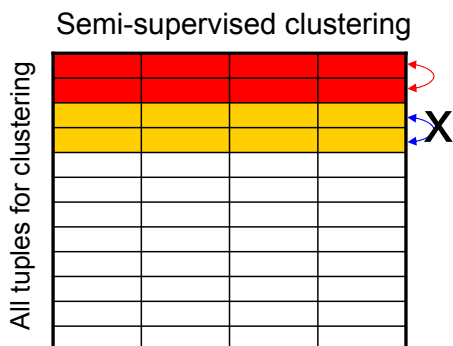


All tuples for clustering

- User-specified *feature* (in the form of *attribute*) is used as a hint, not class labels
  - The attribute may contain too many or too few distinct values
    - E.g., a user may want to cluster students into 20 clusters instead of 3
  - Additional features need to be included in cluster analysis

# Comparing with Semi-Supervised Clustering

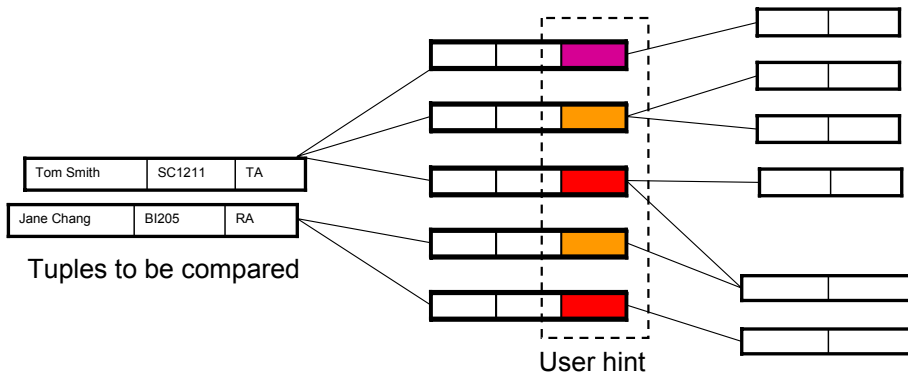
- Semi-supervised clustering [Wagstaff et al' 01, Xing et al.'02]
  - User provides a training set consisting of “similar” and “dissimilar” pairs of objects
- User-guided clustering
  - User specifies an attribute as a hint, and more relevant features are found for clustering



All tuples for clustering

## Semi-Supervised Clustering

- Much information (in multiple relations) is needed to judge whether two tuples are similar
- A user may not be able to provide a good training set
- It is much easier for a user to specify an attribute as a hint, such as a student's *research area*



37

## CrossClus: An Overview

- Measure *similarity between features* by how they group objects into clusters
- Use a heuristic method to search for pertinent features
  - *Start from user-specified feature and gradually expand search range*
- Use *tuple ID propagation* to create feature values
  - Features can be easily created during the expansion of search range, by propagating IDs
- Explore three clustering algorithms: *k*-means, *k*-medoids, and hierarchical clustering

38

## Multi-Relational Features

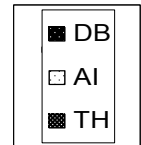
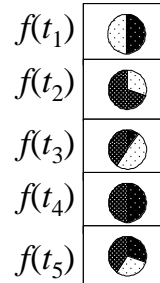
- A multi-relational feature is defined by:
  - A join path, e.g., *Student* → *Register* → *OpenCourse* → *Course*
  - An attribute, e.g., *Course.area*
  - (For numerical feature) an aggregation operator, e.g., sum or average
- Categorical feature  $f = [Student \rightarrow Register \rightarrow OpenCourse \rightarrow Course, Course.area, null]$

areas of courses of each student

Tuple	Areas of courses		
	DB	AI	TH
$t_1$	5	5	0
$t_2$	0	3	7
$t_3$	1	5	4
$t_4$	5	0	5
$t_5$	3	3	4

Values of feature  $f$

Tuple	Feature $f$		
	DB	AI	TH
$t_1$	0.5	0.5	0
$t_2$	0	0.3	0.7
$t_3$	0.1	0.5	0.4
$t_4$	0.5	0	0.5
$t_5$	0.3	0.3	0.4



39

## Similarity between Tuples w.r.t. Categorical Features

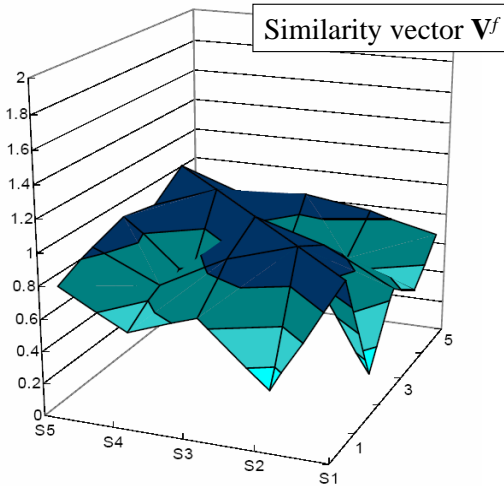
- Similarity between tuples  $t_1$  and  $t_2$  w.r.t. categorical feature  $f$ 
  - Cosine similarity between vectors  $f(t_1)$  and  $f(t_2)$

$$\text{sim}_f(t_1, t_2) = \frac{\sum_{k=1}^L f(t_1) \cdot p_k \cdot f(t_2) \cdot p_k}{\sqrt{\sum_{k=1}^L f(t_1) \cdot p_k^2} \cdot \sqrt{\sum_{k=1}^L f(t_2) \cdot p_k^2}}$$

40

# Representing Features

- Most important information of a feature  $f$  is how  $f$  groups tuples into clusters
- $f$  is represented by similarities between every pair of tuples indicated by  $f$



Similarity between each pair of tuples indicated by  $f$ . The horizontal axes are the tuple indices, and the vertical axis is the similarity. This can be considered as a vector of  $N \times N$  dimensions.

41

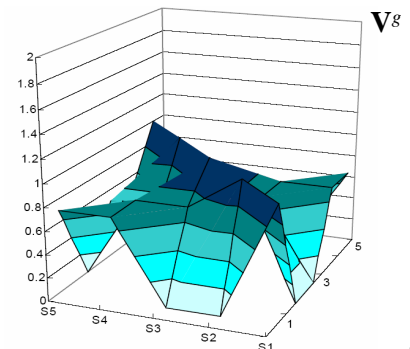
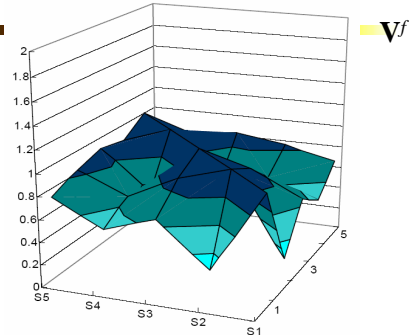
# Similarity Between Features

Values of Feature  $f$  and  $g$

	Feature $f$ (course)			Feature $g$ (group)		
	DB	AI	TH	Info sys	Cog sci	Theory
$t_1$	0.5	0.5	0	1	0	0
$t_2$	0	0.3	0.7	0	0	1
$t_3$	0.1	0.5	0.4	0	0.5	0.5
$t_4$	0.5	0	0.5	0.5	0	0.5
$t_5$	0.3	0.3	0.4	0.5	0.5	0

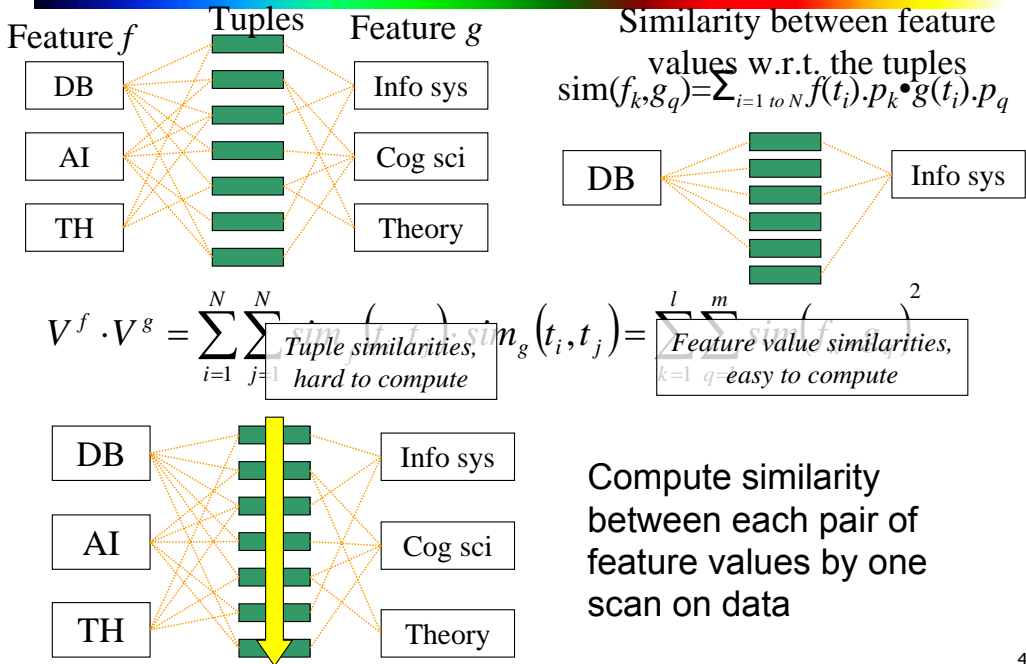
Similarity between two features – cosine similarity of two vectors

$$\text{sim}(f, g) = \frac{V^f \cdot V^g}{|V^f| |V^g|}$$



42

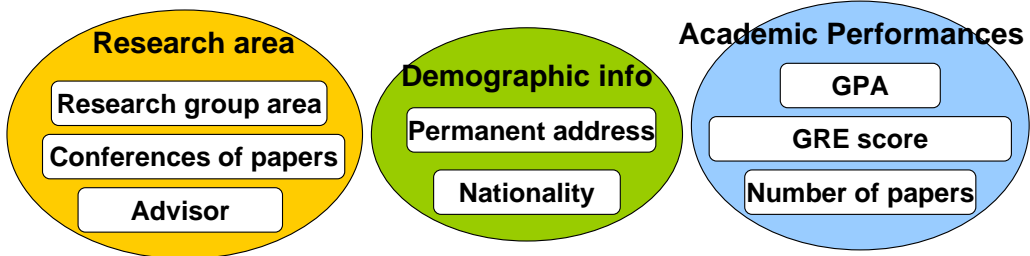
# Computing Feature Similarity



43

# Searching for Pertinent Features

- Different features convey different aspects of information



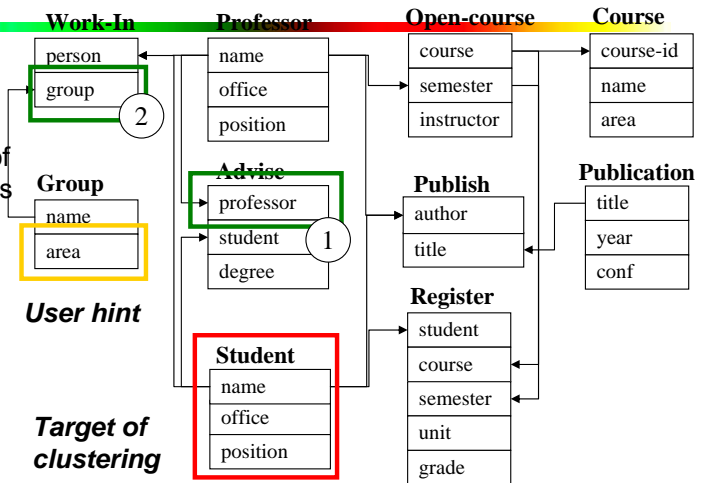
- Features conveying same aspect of information usually cluster tuples in more similar ways
  - Research group areas vs. conferences of publications
- Given user specified feature
  - Find pertinent features by computing feature similarity

44

# Heuristic Search for Pertinent Features

Overall procedure

1. Start from the user-specified feature
2. Search in neighborhood of existing pertinent features
3. Expand search range gradually



- Tuple ID propagation is used to create multi-relational features
  - IDs of target tuples can be propagated along any join path, from which we can find tuples joinable with each target tuple

45

# Clustering with Multi-Relational Features

- Given a set of  $L$  pertinent features  $f_1, \dots, f_L$ , similarity between two tuples

$$\text{sim}(t_1, t_2) = \sum_{i=1}^L \text{sim}_{f_i}(t_1, t_2) \cdot f_i.\text{weight}$$

- Weight of a feature is determined in feature search by its similarity with other pertinent features
- Clustering methods
  - CLARANS [Ng & Han 94], a scalable clustering algorithm for non-Euclidean space
  - K-means
  - Agglomerative hierarchical clustering

46

## Experiments: Compare CrossClus with

---

- Baseline: Only use the user specified feature
- PROCLUS [Aggarwal, et al. 99]: a state-of-the-art subspace clustering algorithm
  - Use a subset of features for each cluster
  - We convert relational database to a table by propositionalization
  - User-specified feature is forced to be used in every cluster
- RDBC [Kirsten and Wrobel'00]
  - A representative ILP clustering algorithm
  - Use neighbor information of objects for clustering
  - User-specified feature is forced to be used

47

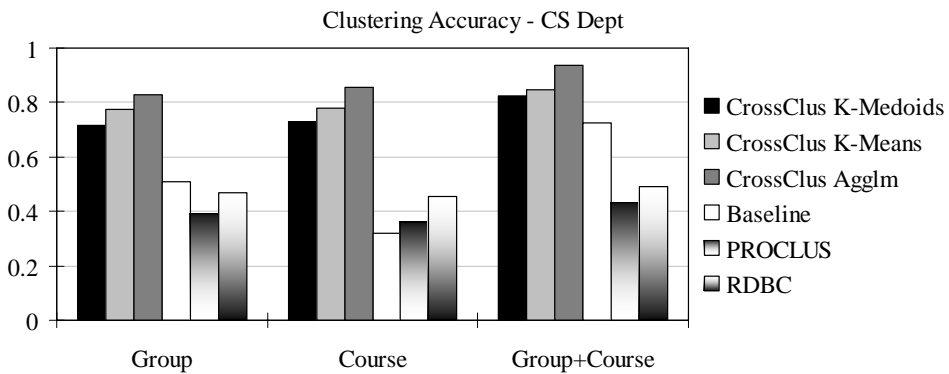
## Measure of Clustering Accuracy

---

- Accuracy
  - Measured by manually labeled data
    - We manually assign tuples into clusters according to their properties (e.g., professors in different research areas)
  - Accuracy of clustering: Percentage of pairs of tuples in the same cluster that share common label
    - This measure favors many small clusters
    - We let each approach generate the same number of clusters

48

# CS Dept Dataset: Clustering Professors by Research Area

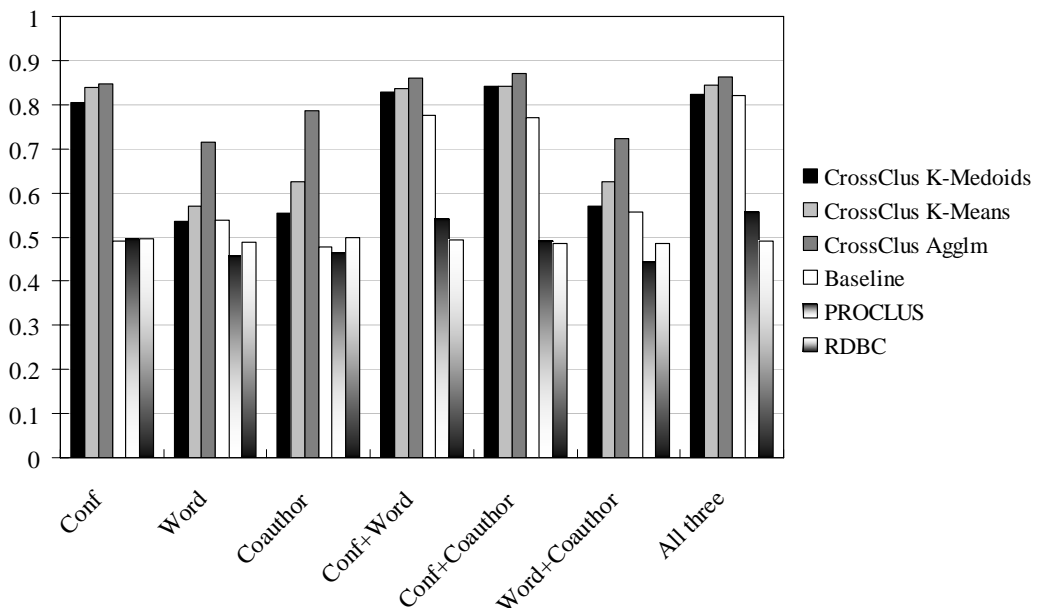


- (Theory): J. Erickson, S. Har-Peled, L. Pitt, E. Ramos, D. Roth, M. Viswanathan
- (Graphics): J. Hart, M. Garland, Y. Yu
- (Database): K. Chang, A. Doan, J. Han, M. Winslett, C. Zhai
- (Numerical computing): M. Heath, T. Kerkhoven, E. de Sturler
- (Networking & QoS): R. Kravets, M. Caccamo, J. Hou, L. Sha
- (Artificial Intelligence): G. Dejong, M. Harandi, J. Ponce, L. Rendell
- (Architecture): D. Padua, J. Torrellas, C. Zilles, S. Adve, M. Snir, D. Reed, V. Adve
- (Operating Systems): D. Mickunas, R. Campbell, Y. Zhou

49

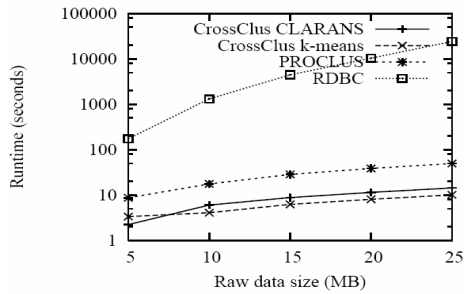
# DBLP Dataset

Clustering Accuracy - DBLP

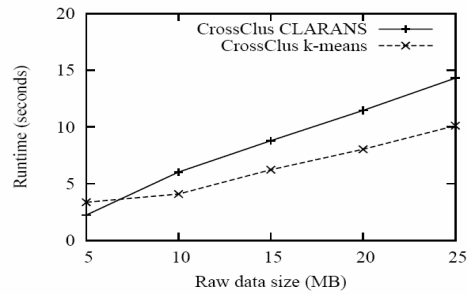


50

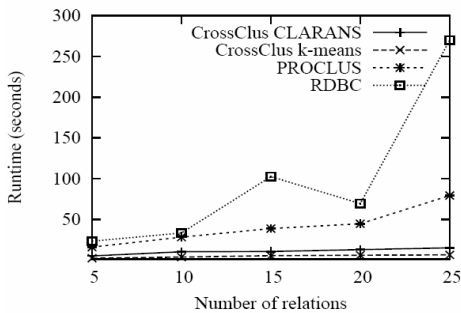
# Scalability w.r.t. Data Size and # of Relations



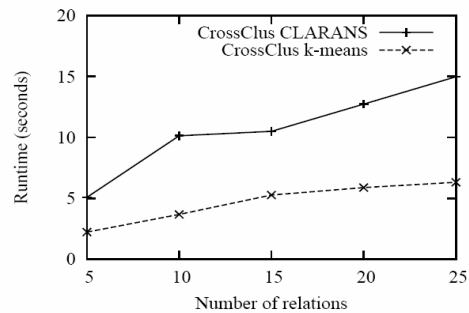
a) All approaches



b) CrossClus



a) All approaches



b) CrossClus

51

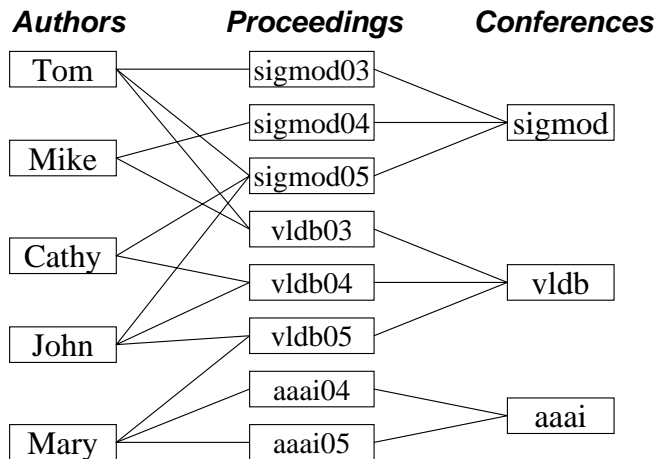
## Outline

Theme: "Knowledge is power, but knowledge is hidden in massive links"

- Link Mining: A General Introduction
- CrossMine: Classification of Multi-relations by Link Analysis
- CrossClus: Clustering over Multi-relations by User-Guidance
- LinkClus: Efficient Clustering by Exploring the Power Law Distribution
- Distinct: Distinguishing Objects with Identical Names by Link Analysis
- TruthFinder: Conformity to Truth with Conflicting Information
- Conclusions and Future Work

52

# Link-Based Clustering: Calculate Similarities Based On Links

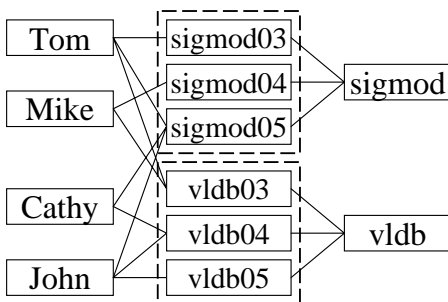
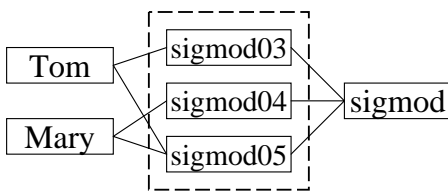


- Q1: How to cluster each type of objects?
- Q2: How to define similarity between each type of objects?

53

# Link-Based Similarities: SimRank

- Two objects are similar if they are linked with the same or similar objects



Jeh & Widom, KDD'2002 - *SimRank*

The similarity between two objects  $x$  and  $y$  is defined as the average similarity between objects linked with  $x$  and those with  $y$ :

$$\text{sim}(a, b) = \frac{C}{|I(a)||I(b)|} \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} \text{sim}(I_i(a), I_j(b))$$

Expensive to compute:

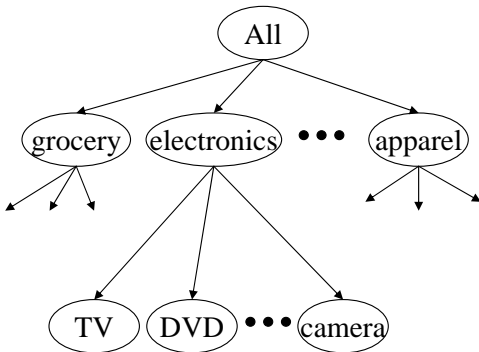
For a dataset of  $N$  objects and  $M$  links, it takes  $\mathcal{O}(M^2)$  space and  $\mathcal{O}(M^2)$  time to compute all similarities.

54

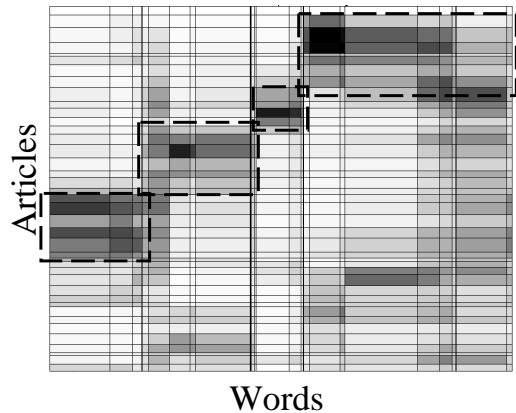
## Observation 1: Hierarchical Structures

- Hierarchical structures often exist naturally among objects (e.g., taxonomy of animals)

A hierarchical structure of products in Walmart

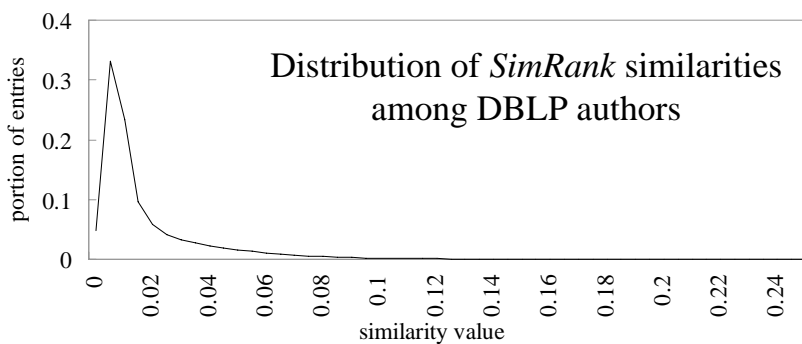


Relationships between articles and words (Chakrabarti, Papadimitriou, Modha, Faloutsos, 2004)



55

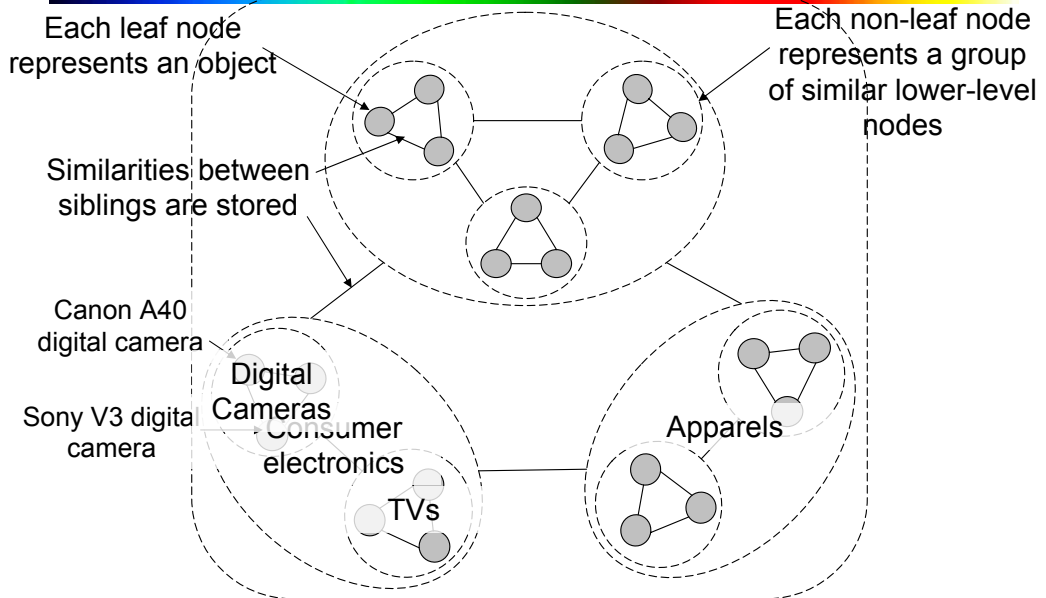
## Observation 2: Distribution of Similarity



- Power law distribution exists in similarities
  - 56% of similarity entries are in [0.005, 0.015]
  - 1.4% of similarity entries are larger than 0.1
  - Our goal: Design a data structure that stores the significant similarities and compresses insignificant ones

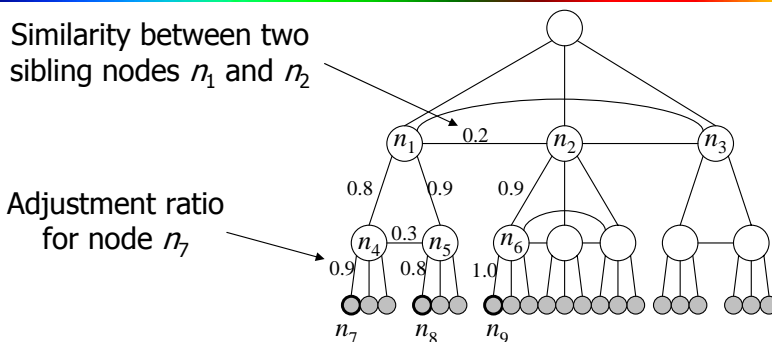
56

# Our Data Structure: SimTree



57

# Similarity Defined by SimTree



- $sim_p(n_7, n_8) = s(n_7, n_4) \times s(n_4, n_5) \times s(n_5, n_8)$ 
  - Path-based node similarity
- Similarity between two nodes is the average similarity between objects linked with them in other SimTrees
- Adjustment ratio for  $x = \frac{\text{Average similarity between } x \text{ and all other nodes}}{\text{Average similarity between } x\text{'s parent and all other nodes}}$

58

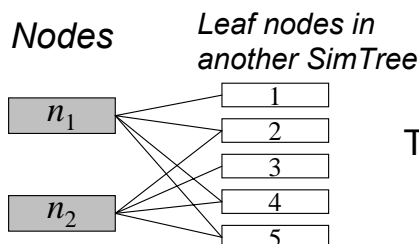
# Overview of LinkClus

- Initialize a SimTree for objects of each type
- Repeat
  - For each SimTree, update the similarities between its nodes using similarities in other SimTrees
    - Similarity between two nodes  $x$  and  $y$  is the average similarity between objects linked with them
  - Adjust the structure of each SimTree
    - Assign each node to the parent node that it is most similar to
- X. Yin, J. Han, and P. S. Yu, "LinkClus: Efficient Clustering via Heterogeneous Semantic Links", VLDB'06

59

# Initialization of SimTrees

- Initializing a SimTree
  - Repeatedly find groups of tightly related nodes, which are merged into a higher-level node
- Tightness of a group of nodes
  - For a group of nodes  $\{n_1, \dots, n_k\}$ , its tightness is defined as the number of leaf nodes in other SimTrees that are connected to all of  $\{n_1, \dots, n_k\}$

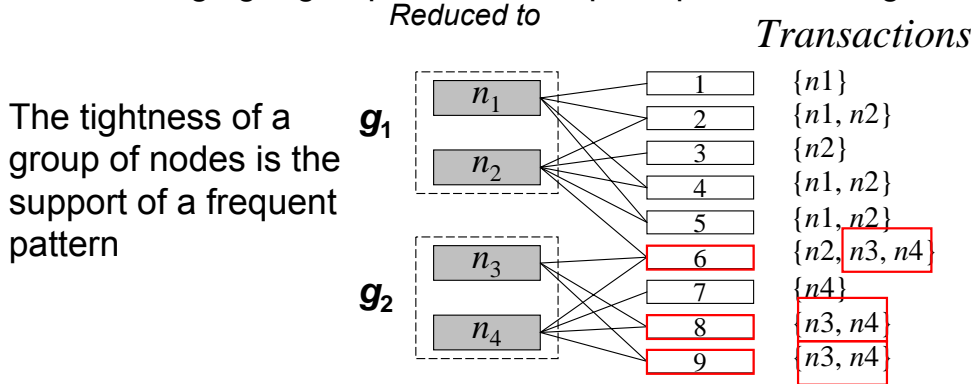


The tightness of  $\{n_1, n_2\}$  is 3

60

## (continued)

- Finding tight groups  $\longrightarrow$  Frequent pattern mining

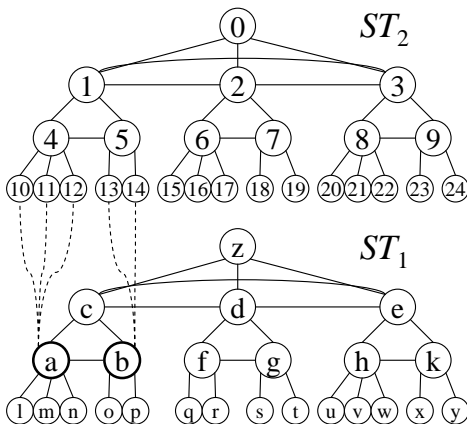


- Procedure of initializing a tree
  - Start from leaf nodes (level-0)
  - At each level  $l$ , find non-overlapping groups of similar nodes with frequent pattern mining

61

## Updating Similarities Between Nodes

- The initial similarities can seldom capture the relationships between objects
- Iteratively update similarities
  - Similarity between two nodes is the average similarity between objects linked with them

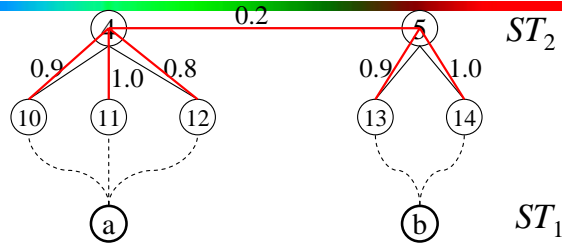


$$sim(n_a, n_b) = \text{average similarity between } \begin{matrix} \textcircled{10} \\ \textcircled{11} \\ \textcircled{12} \end{matrix} \text{ and } \begin{matrix} \textcircled{13} \\ \textcircled{14} \end{matrix}$$

takes  $O(3 \times 2)$  time

62

## Aggregation-Based Similarity Computation



For each node  $n_k \in \{n_{10}, n_{11}, n_{12}\}$  and  $n_l \in \{n_{13}, n_{14}\}$ , their path-based similarity  $sim_p(n_k, n_l) = s(n_k, n_4) \cdot s(n_4, n_5) \cdot s(n_5, n_l)$ .

$$sim(n_a, n_b) = \frac{\sum_{k=10}^{12} s(n_k, n_4)}{3} \cdot s(n_4, n_5) \cdot \frac{\sum_{l=13}^{14} s(n_l, n_5)}{2} = 0.171$$

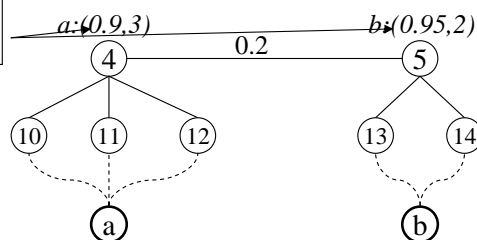
takes  $O(3+2)$  time

After aggregation, we reduce quadratic time computation to linear time computation.

63

## Computing Similarity with Aggregation

Average similarity and total weight



$sim(n_a, n_b)$  can be computed from aggregated similarities

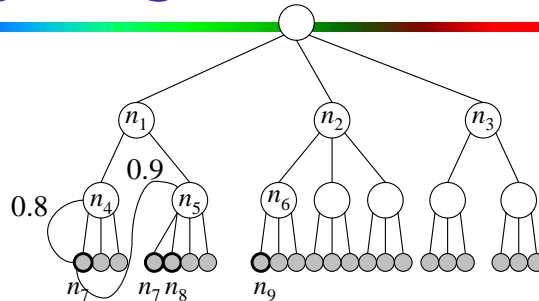
$$sim(n_a, n_b) = avg\_sim(n_a, n_4) \times s(n_4, n_5) \times avg\_sim(n_b, n_5) \\ = 0.9 \times 0.2 \times 0.95 = 0.171$$

To compute  $sim(n_a, n_b)$ :

- Find all pairs of sibling nodes  $n_i$  and  $n_j$ , so that  $n_a$  linked with  $n_i$  and  $n_b$  with  $n_j$ .
- Calculate similarity (and weight) between  $n_a$  and  $n_b$  w.r.t.  $n_i$  and  $n_j$ .
- Calculate weighted average similarity between  $n_a$  and  $n_b$  w.r.t. all such pairs.

64

# Adjusting SimTree Structures



- After similarity changes, the tree structure also needs to be changed
  - If a node is more similar to its parent's sibling, then move it to be a child of that sibling
  - Try to move each node to its parent's sibling that it is most similar to, under the constraint that each parent node can have at most  $c$  children

65

# Complexity

For two types of objects,  $N$  in each, and  $M$  linkages between them.

	Time	Space
Updating similarities	$O(M(\log N)^2)$	$O(M+N)$
Adjusting tree structures	$O(N)$	$O(N)$
<i>LinkClus</i>	$O(M(\log N)^2)$	$O(M+N)$
<i>SimRank</i>	$O(M^2)$	$O(N^2)$

66

## Empirical Study

---

- Generating clusters using a SimTree
  - Suppose  $K$  clusters are to be generated
  - Find a level in the SimTree that has number of nodes closest to  $K$
  - Merging most similar nodes or dividing largest nodes on that level to get  $K$  clusters
- Accuracy
  - Measured by manually labeled data
  - Accuracy of clustering: Percentage of pairs of objects in the same cluster that share common label
- Efficiency and scalability
  - Scalability w.r.t. number of objects, clusters, and linkages

67

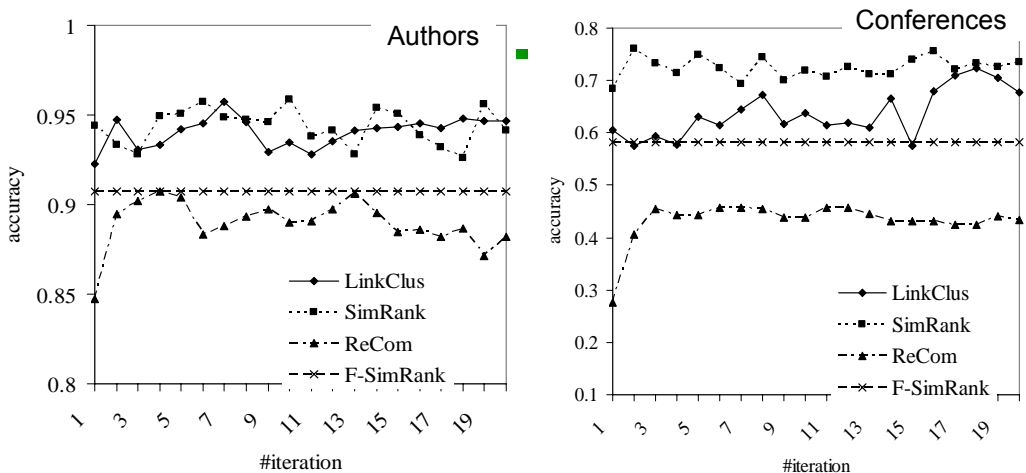
## Experiment Setup

---

- DBLP dataset: 4170 most productive authors, and 154 well-known conferences with most proceedings
  - Manually labeled research areas of 400 most productive authors according to their home pages (or publications)
  - Manually labeled areas of 154 conferences according to their call for papers
- Approaches Compared:
  - SimRank (Jeh & Widom, KDD 2002)
    - Computing pair-wise similarities
  - SimRank with FingerPrints (F-SimRank)
    - Fogaras & R´acz, WWW 2005
    - pre-computes a large sample of random paths from each object and uses samples of two objects to estimate SimRank similarity
  - ReCom (Wang et al. SIGIR 2003)
    - Iteratively clustering objects using cluster labels of linked objects

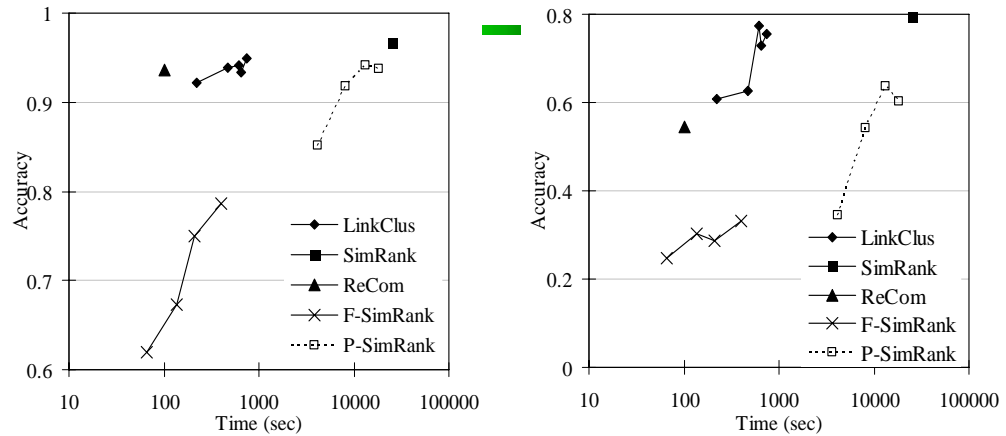
68

# Accuracy



Approaches	Accr-Author	Accr-Conf	average time
LinkClus	0.957	0.723	76.7
SimRank	0.958	0.760	1020
ReCom	0.907	0.457	43.1
F-SimRank	0.908	0.583	83.6

# (continued)



- Accuracy vs. Running time
  - LinkClus is almost as accurate as SimRank (most accurate), and is much more efficient

# Email Dataset

- F. Nielsen. Email dataset.  
<http://www.imm.dtu.dk/~rem/data/Email-1431.zip>
- 370 emails on conferences, 272 on jobs, and 789 spam emails

<i>Approach</i>	<i>Accuracy</i>	<i>Total time (sec)</i>
LinkClus	0.8026	1579.6
SimRank	0.7965	39160
ReCom	0.5711	74.6
F-SimRank	0.3688	479.7
CLARANS	0.4768	8.55

71

# Outline

Theme: “Knowledge is power, but knowledge is hidden in massive links”

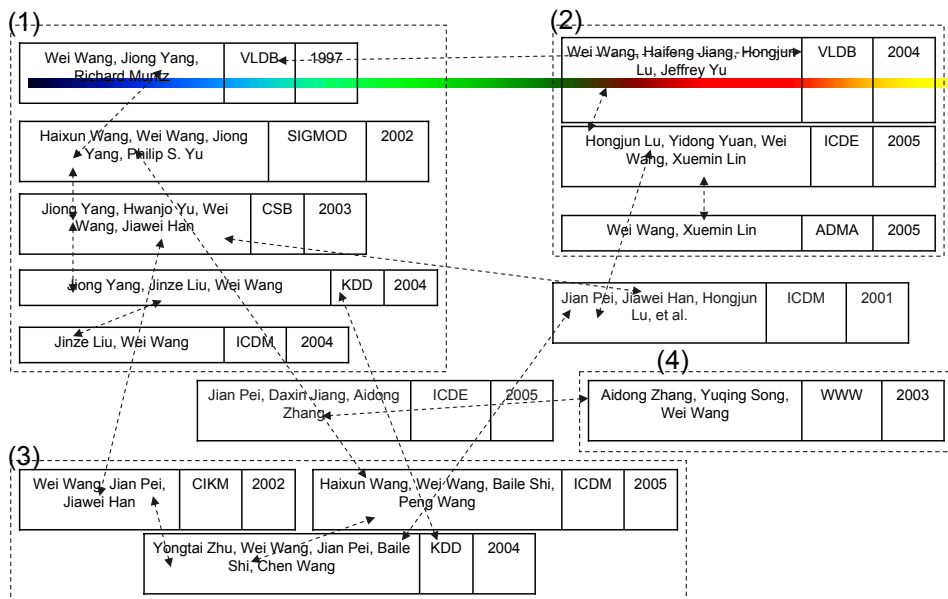
- Link Mining: A General Introduction
- CrossMine: Classification of Multi-relations by Link Analysis
- CrossClus: Clustering over Multi-relations by User-Guidance
- LinkClus: Efficient Clustering by Exploring the Power Law Distribution
- Distinct: Distinguishing Objects with Identical Names by Link Analysis
- TruthFinder: Conformity to Truth with Conflicting Information
- Conclusions and Future Work

72

# People/Objects Do Share Names

- Why distinguishing objects with identical names?
- Different objects may share the same name
  - In AllMusic.com, 72 songs and 3 albums named “Forgotten” or “The Forgotten”
  - In DBLP, 141 papers are written by at least 14 “Wei Wang”
  - How to distinguish the authors of the 141 papers?
- Our task: Object distinction
  - X. Yin, J. Han, and P. S. Yu, “Object Distinction: Distinguishing Objects with Identical Names by Link Analysis”, ICDE'07

73



(1) Wei Wang at UNC

(2) Wei Wang at UNSW, Australia

(3) Wei Wang at Fudan Univ., China

(4) Wei Wang at SUNY Buffalo

74

## Challenges of Object Distinction

---

- Related to duplicate detection, but
  - Textual similarity cannot be used
  - Different references appear in different contexts (e.g., different papers), and thus seldom share common attributes
  - Each reference is associated with limited information
- We need to carefully design an approach and use all information we have

75

## Overview of DISTINCT

---

- Measure similarity between references
  - Linkages between references
    - As shown by self-loop property, references to the same object are more likely to be connected
  - Neighbor tuples of each reference
    - Can indicate similarity between their contexts
- References clustering
  - Group references according to their similarities

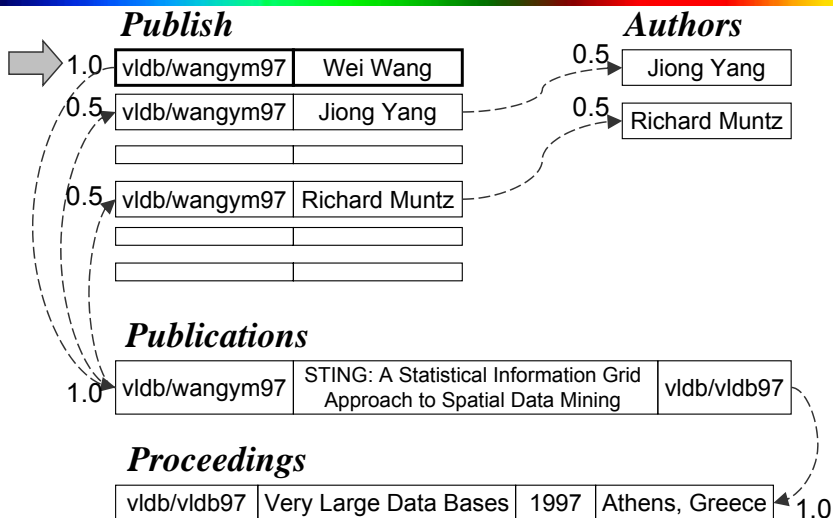
76

# Similarity 1: Link-based Similarity

- Indicate the overall strength of connections between two references
- We use *random walk probability* between the two tuples containing the references
- Random walk probabilities along *different join paths* are handled separately
  - Because different join paths have different semantic meanings
  - Only consider join paths of length at most  $2L$  ( $L$  is the number of steps of propagating probabilities)

77

## Example of Random Walk



78

## Similarity 2: Neighborhood Similarity

---

- Find the neighbor tuples of each reference
  - Neighbor tuples within  $L$  joins
- Weights of neighbor tuples
  - Different neighbor tuples have different connections to a reference
  - Assign each neighbor tuple a *weight*, which is the probability of walking from the reference to this tuple
- Similarity: Set resemblance between two sets of neighbor tuples

79

## Training with the “Same” Data Set

---

- Build a training set automatically
  - Select distinct names, e.g., Johannes Gehrke
  - The collaboration behavior within the same community share some similarity
  - Training parameters using a typical and large set of “unambiguous” examples
- Use SVM to learn a model for combining different join paths
  - Each join path is used as two attributes (with link-based similarity and neighborhood similarity)
  - The model is a weighted sum of all attributes

80

## Clustering References

---

- Why choose agglomerative hierarchical clustering methods?
  - We do not know number of clusters (real entities)
  - We only know similarity between references
  - Equivalent references can be merged into a cluster, which represents a single entity

81

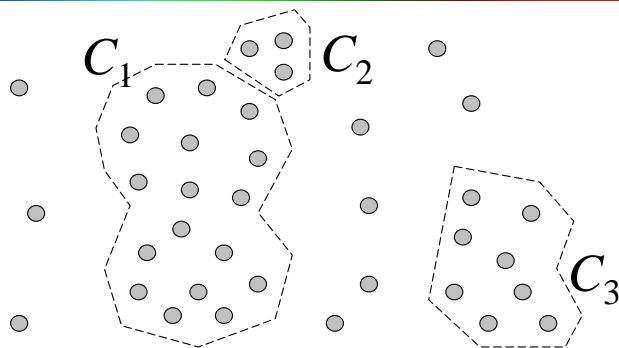
## How to Measure Similarity between Clusters?

---

- Single-link (highest similarity between points in two clusters) ?
  - No, because references to different objects can be connected.
- Complete-link (minimum similarity between them)?
  - No, because references to the same object may be weakly connected.
- Average-link (average similarity between points in two clusters)?
  - A better measure

82

## Problem with Average-link

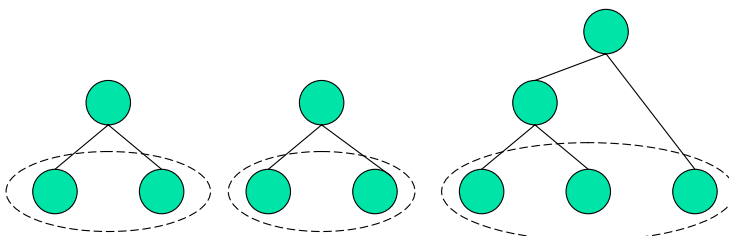


- $C_2$  is close to  $C_1$ , but their average similarity is low
- We use *collective random walk probability*: Probability of walking from one cluster to another
- Final measure:  
*Average neighborhood similarity and Collective random walk probability*

83

## Clustering Procedure

- Procedure
  - Initialization: Use each reference as a cluster
  - Keep finding and merging the most similar pair of clusters
  - Until no pair of clusters is similar enough



84

## Efficient Computation

- In agglomerative hierarchical clustering, one needs to repeatedly compute similarity between clusters
  - When merging clusters  $C_1$  and  $C_2$  into  $C_3$ , we need to compute the similarity between  $C_3$  and any other cluster
  - Very expensive when clusters are large
- We invent methods to compute similarity incrementally
  - Neighborhood similarity

$$Resem(C_3, C_i) = \frac{|C_1| \cdot Resem(C_1, C_i) + |C_2| \cdot Resem(C_2, C_i)}{|C_1| + |C_2|}.$$

- Random walk probability

$$WalkProb_Q(C_3 \rightarrow C_i) = \frac{|C_1| \cdot WalkProb_Q(C_1 \rightarrow C_i) + |C_2| \cdot WalkProb_Q(C_2 \rightarrow C_i)}{|C_1| + |C_2|}.$$

85

## Experimental Results

- Distinguishing references to authors in DBLP
- Accuracy of reference clustering
  - True positive: Number of pairs of references to same author in same cluster
  - False positive: Different authors, in same cluster
  - False negative: Same author, different clusters
  - True negative: Different authors, different clusters

- Measures

$$\text{Precision} = TP / (TP + FP)$$

$$\text{Recall} = TP / (TP + FN)$$

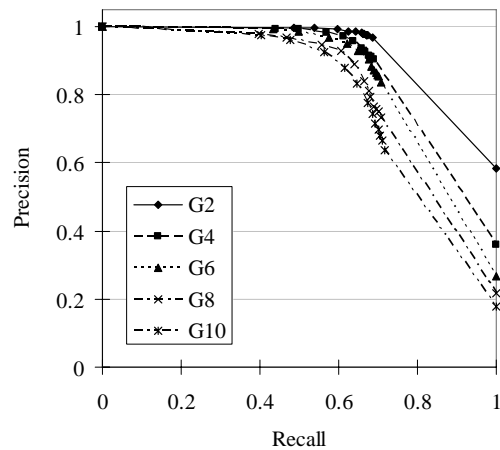
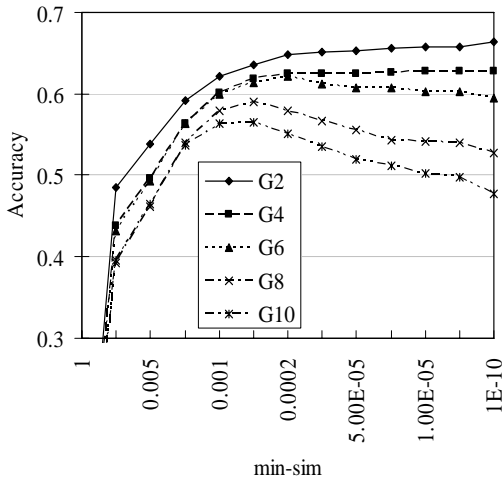
$$f\text{-measure} = 2 * \text{precision} * \text{recall} / (\text{precision} + \text{recall})$$

$$\text{Accuracy} = TP / (TP + FP + FN)$$

86

# Accuracy on Synthetic Tests

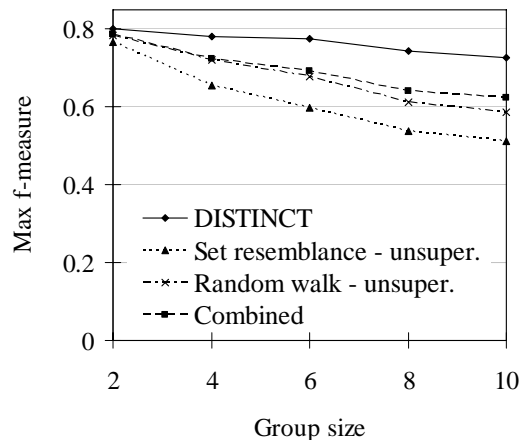
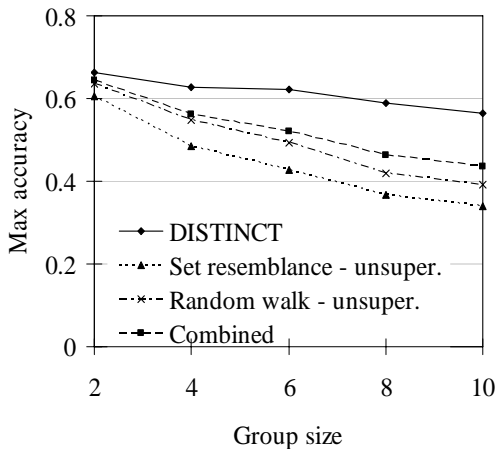
- Select 1000 authors with at least 5 papers
- Merge  $G$  ( $G=2$  to  $10$ ) authors into one group
- Use DISTINCT to distinguish each group of references



87

# Compare with “Existing Approaches”

- Random walk and neighborhood similarity have been used in duplicate detection
- We combine them with our clustering approaches for comparison

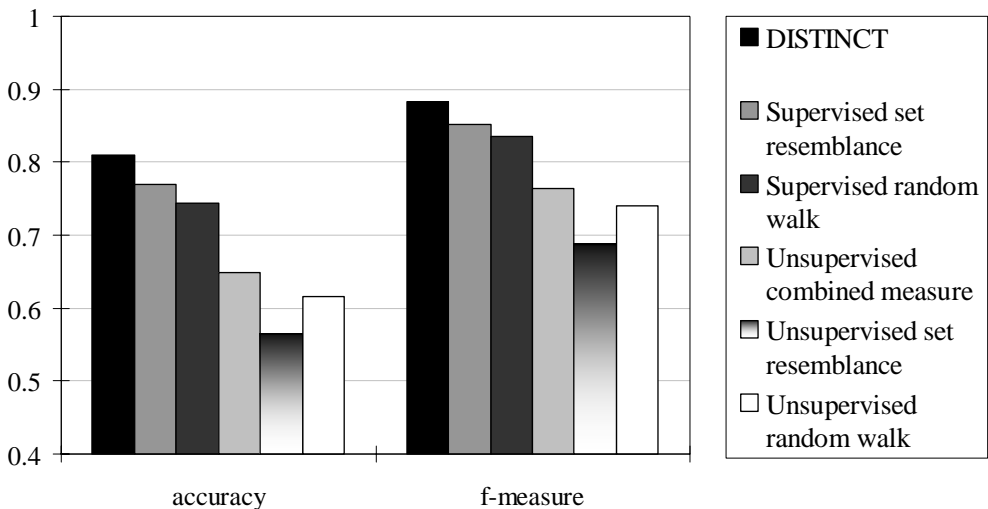


# Real Cases

Name	#author	#ref	accuracy	precision	recall	f-measure
Hui Fang	3	9	1.0	1.0	1.0	1.0
Ajay Gupta	4	16	1.0	1.0	1.0	1.0
Joseph Hellerstein	2	151	0.81	1.0	0.81	0.895
Rakesh Kumar	2	36	1.0	1.0	1.0	1.0
Michael Wagner	5	29	0.395	1.0	0.395	0.566
Bing Liu	6	89	0.825	1.0	0.825	0.904
Jim Smith	3	19	0.829	0.888	0.926	0.906
Lei Wang	13	55	0.863	0.92	0.932	0.926
Wei Wang	14	141	0.716	0.855	0.814	0.834
Bin Yu	5	44	0.658	1.0	0.658	0.794
average			0.81	0.966	0.836	0.883

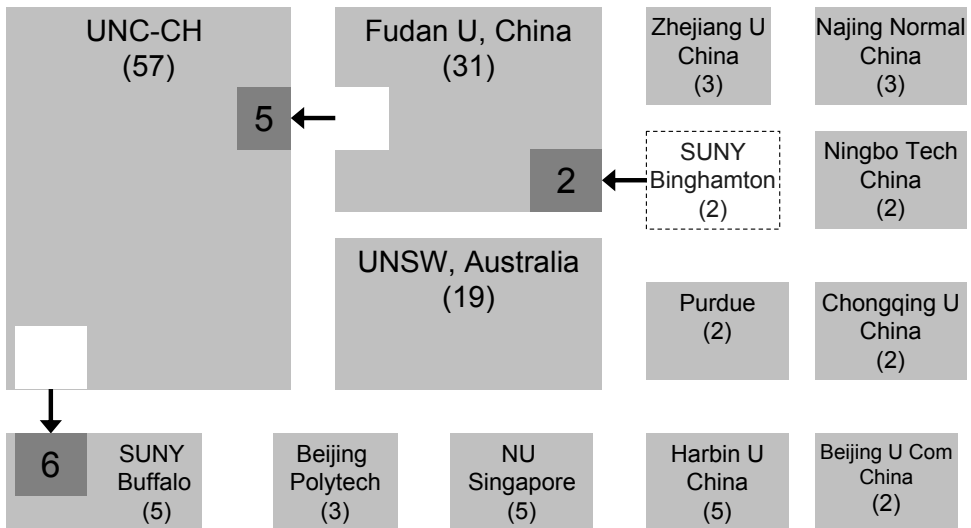
89

## Real Cases: Comparison



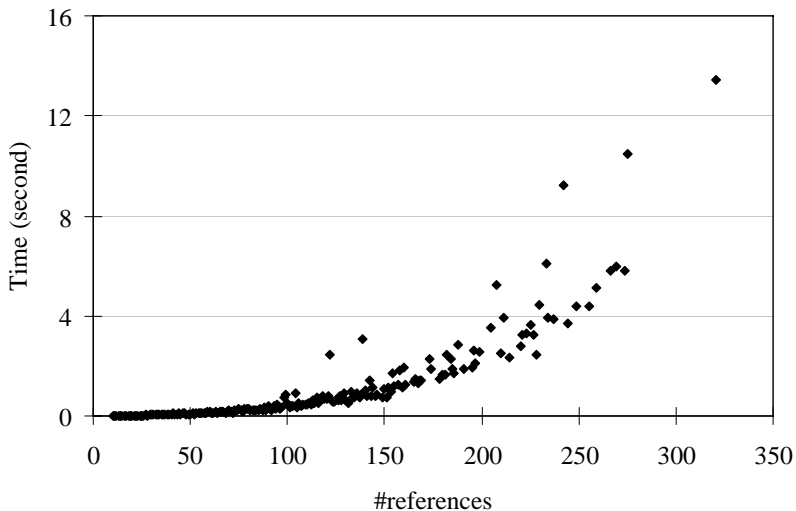
90

# Distinguishing Different “Wei Wang”s



# Scalability

- Agglomerative hierarchical clustering takes quadratic time
  - Because it requires computing pair-wise similarity



# Outline

---

Theme: "Knowledge is power, but knowledge is hidden in massive links"

- Link Mining: A General Introduction
- CrossMine: Classification of Multi-relations by Link Analysis
- CrossClus: Clustering over Multi-relations by User-Guidance
- LinkClus: Efficient Clustering by Exploring the Power Law Distribution
- Distinct: Distinguishing Objects with Identical Names by Link Analysis
- TruthFinder: Conformity to Truth with Conflicting Information
- Conclusions and Future Work

93

## Truth Discovery with Multiple Conflicting Information Providers [KDD'07]

---

- Xiaoxin Yin, Jiawei Han, Philip S. Yu, "Truth Discovery with Multiple Conflicting Information Providers on the Web", KDD'07
- The trustworthiness problem of the web (according to a survey):
  - 54% of Internet users trust news web sites most of time
  - 26% for web sites that sell products
  - 12% for blogs
- TruthFinder: Truth discovery on the Web by link analysis
  - Among multiple conflict results, can we automatically identify which one is likely the true fact?
- Veracity (conformity to truth):
  - Given a large amount of conflicting information about many objects, provided by multiple web sites (or other information providers)
  - How to discover the true fact about each object?

94

# Conflicting Information on the Web

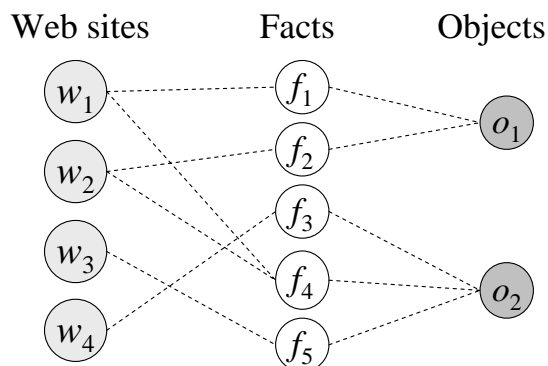
- Different websites often provide conflicting info. on a subject, e.g., Authors of “*Rapid Contextual Design*”

<i>Online Store</i>	<i>Authors</i>
Powell’s books	Holtzblatt, Karen
Barnes & Noble	Karen Holtzblatt, Jessamyn Wendell, Shelley Wood
A1 Books	Karen Holtzblatt, Jessamyn Burns Wendell, Shelley Wood
Cornwall books	Holtzblatt-Karen, Wendell-Jessamyn Burns, Wood
Mellon’s books	Wendell, Jessamyn
Lakeside books	WENDELL, JESSAMYNHOLTZBLATT, KARENWOOD, SHELLEY
Blackwell online	Wendell, Jessamyn, Holtzblatt, Karen, Wood, Shelley

95

# Our Problem Setting

- Each object has a set of **conflictive** facts
  - E.g., different author names for a book
- And each web site provides some facts
- How to find the true fact for each object?



96

# Basic Heuristics for Problem Solving

---

1. There is usually only one true fact for a property of an object
2. This true fact appears to be the same or similar on different web sites
  - E.g., “Jennifer Widom” vs. “J. Widom”
3. The false facts on different web sites are less likely to be the same or similar
  - False facts are often introduced by random factors
4. A web site that provides mostly true facts for many objects will likely provide true facts for other objects

97

# Overview of the TruthFinder Method

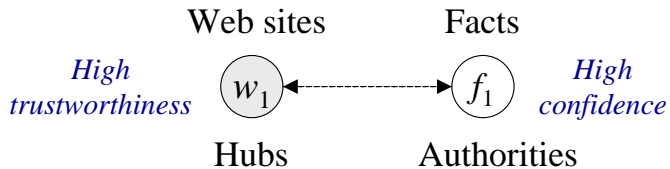
---

- Confidence of facts ↔ Trustworthiness of web sites
  - A fact has *high confidence* if it is provided by (many) trustworthy web sites
  - A web site is *trustworthy* if it provides many facts with high confidence
- Our method, TruthFinder, overview
  - Initially, each web site is equally trustworthy
  - Based on the above four heuristics, infer fact confidence from web site trustworthiness, and then backwards
  - Repeat until achieving stable state

98

# Analogy to Authority-Hub Analysis

- Facts ↔ Authorities, Web sites ↔ Hubs

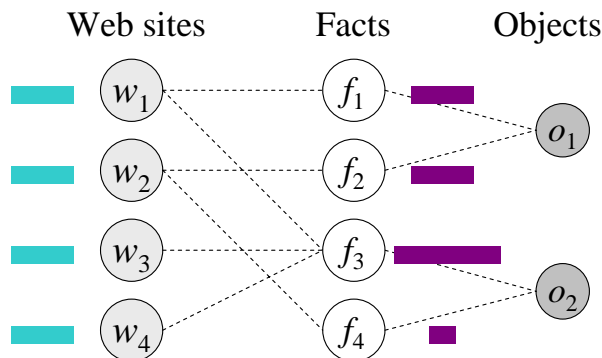


- Difference from authority-hub analysis
  - Linear summation cannot be used
    - A web site is trustable if it provides accurate facts, instead of many facts
    - Confidence is the probability of being true
  - Different facts of the same object influence each other

99

# Example: Inference on Trustworthiness

- Inference of web site trustworthiness & fact confidence



True facts and trustable web sites will become apparent after some iterations

100

## Computation Model (1): $t(w)$ and $s(f)$

- The trustworthiness of a web site  $w$ :  $t(w)$ 
  - Average confidence of facts it provides

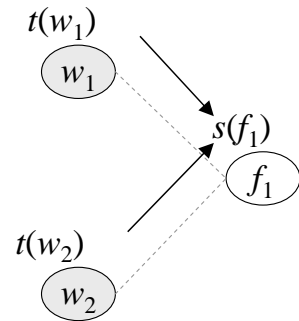
$$t(w) = \frac{\sum_{f \in F(w)} s(f)}{|F(w)|}$$

$\leftarrow$  Sum of fact confidence  
 $\leftarrow$  Set of facts provided by  $w$

- The confidence of a fact  $f$ :  $s(f)$ 
  - One minus the probability that all web sites providing  $f$  are wrong

$$s(f) = 1 - \prod_{w \in W(f)} (1 - t(w))$$

$\leftarrow$  Probability that  $w$  is wrong  
 $\leftarrow$  Set of websites providing  $f$



101

## Computation Model (2): Fact Influence

- Influence between related facts

Example: For a certain book  $B$

- $w_1$  :  $B$  is written by “Jennifer Widom” (fact  $f_1$ )
- $w_2$  :  $B$  is written by “J. Widom” (fact  $f_2$ )

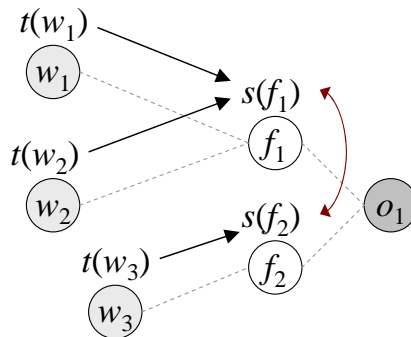
$f_1$  and  $f_2$  support each other

- If several other trustworthy web sites say this book is written by “Jeffrey Ullman”, then  $f_1$  and  $f_2$  are likely to be wrong
- Influence between related facts are very important

102

## Computation Model (3): Influence Function

- A user may provide “influence function” between related facts (e.g.,  $f_1$  and  $f_2$ )
  - E.g., Similarity between people’s names
- The confidence of related facts are adjusted according to the influence function



103

## Experiments: Finding Truth of Facts

- Determining authors of books
  - Dataset contains 1265 books listed on abebooks.com
  - We analyze 100 random books (using book images)

Case	Voting	TruthFinder	Barnes & Noble
Correct	71	85	64
Miss author(s)	12	2	4
Incomplete names	18	5	6
Wrong first/middle names	1	1	3
Has redundant names	0	2	23
Add incorrect names	1	5	5
No information	0	0	2

104

## Experiments: Trustable Info Providers

- Finding trustworthy information sources
  - Most trustworthy bookstores found by TruthFinder vs. Top ranked bookstores by Google (query “bookstore”)

### TruthFinder

Bookstore	<i>trustworthiness</i>	<i>#book</i>	<i>Accuracy</i>
TheSaintBookstore	0.971	28	0.959
MildredsBooks	0.969	10	1.0
Alphacraze.com	0.968	13	0.947

### Google

Bookstore	<i>Google rank</i>	<i>#book</i>	<i>Accuracy</i>
Barnes & Noble	1	97	0.865
Powell’s books	3	42	0.654

105

## Summary: TruthFinder


- Veracity: An important problem for Web search and analysis
- Resolving conflicting facts from multiple websites
- Our approach: Utilizing the inter-dependency between website trustworthiness and fact confidence to find (1) *trustable web sites*, and (2) *true facts*
- TruthFinder: A system based on this philosophy
  - Achieves high accuracy on finding both true facts and high quality web sites
- Future work: Put the framework into broader application scope
  - E.g., Mass collaboration and mass labeling

106

# Outline

---

Theme: "Knowledge is power, but knowledge is hidden in massive links"

- Link Mining: A General Introduction
- CrossMine: Classification of Multi-relations by Link Analysis
- CrossClus: Clustering over Multi-relations by User-Guidance
- LinkClus: Efficient Clustering by Exploring the Power Law Distribution
- Distinct: Distinguishing Objects with Identical Names by Link Analysis
- TruthFinder: Conformity to Truth with Conflicting Information
- Conclusions and Future Work 

107

# Conclusions

---

- Knowledge is power, but knowledge is hidden in massive links
- Exploring links in mining rather than Web page rank and search
- CrossMine: Classification of multi-relations by link analysis
- CrossClus: Clustering over multi-relations by user-guidance
- LinkClus: Efficient clustering by exploring the power law distribution
- Distinct: Distinguishing objects with identical names by link analysis
- Much more to be explored!

108

## References on Information Network Analysis

---

- S. Brin and L. Page, “The Anatomy of a Large-Scale Hypertextual Web Search Engine”, WWW'98
- D. Cai, X. He, J. Wen, and W. Ma, “Block-level Link Analysis”, SIGIR'04
- S. Chakrabarti, Mining the Web: Statistical Analysis of Hypertext and Semi-Structured Data, Morgan Kaufmann, 2002
- S. Chakrabarti, B. E. Dom, R. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins, D. Gibson, and J. M. Kleinberg, “Mining the Web's Link Structure”, COMPUTER, 32(8):60-67,1999
- L. Getoor, “Link mining: a new data mining challenge”, SIGKDD Explorations, 5(1):84-89, 2003
- G. Jeh and J. Widom, “Simrank: A measure of structural-context similarity”, KDD'02
- J. M. Kleinberg, “Authoritative Sources in a Hyperlinked Environment”, J. ACM, 46(5): 604-632, 1999.
- D. Kempe, J. Kleinberg and E. Tardos, “Maximizing the Spread of Influence through a Social Network”, KDD'03.
- B. Liu, Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data, Springer, 2006
- S. Wasserman and K. Faust, Social Network Analysis: Methods and Applications, Cambridge University Press, 1994

109

## References of Our Work in This Tutorial

---

- X. Yin, J. Han, J. Yang, and P. S. Yu, “CrossMine: Efficient Classification across Multiple Database Relations”, ICDE'04 (extended version in TKDE'06)
- X. Yin, J. Han, and P. S. Yu, “Cross-Relational Clustering with User's Guidance”, KDD'05
- X. Yin, J. Han, and P. S. Yu, “LinkClus: Efficient Clustering via Heterogeneous Semantic Links”, VLDB'06
- X. Yin, J. Han, and P. S. Yu, “Object Distinction: Distinguishing Objects with Identical Names by Link Analysis”, ICDE'07
- X. Yin, J. Han, and P. S. Yu, “Truth Discovery with Multiple Conflicting Information Providers on the Web”, KDD'07

Visit [www.cs.uiuc.edu/~hanj/pubs/index.htm](http://www.cs.uiuc.edu/~hanj/pubs/index.htm) to download these research papers

110