

PART 4

RELATIONAL & MULTIENTITY ER

Outline

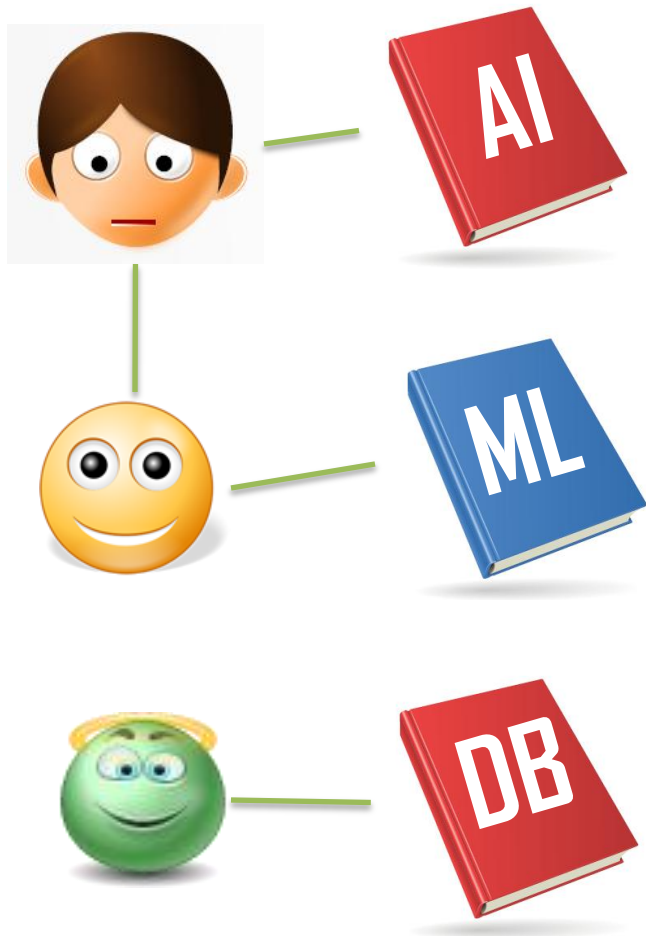
1. Introduction & Motivation
2. Classical Single Entity ER
3. Efficiency: Blocking/Canopy Generation
- 4. Relational & MultiEntity ER**
 - a) Problem Statement
 - b) Relational Features
 - c) Graph-based Approaches
 - d) Agglomerative Approaches
 - e) Generative Model Based Approaches
 - f) Declarative Approaches
5. Demo
6. Challenges & Future Directions

PART 4-a

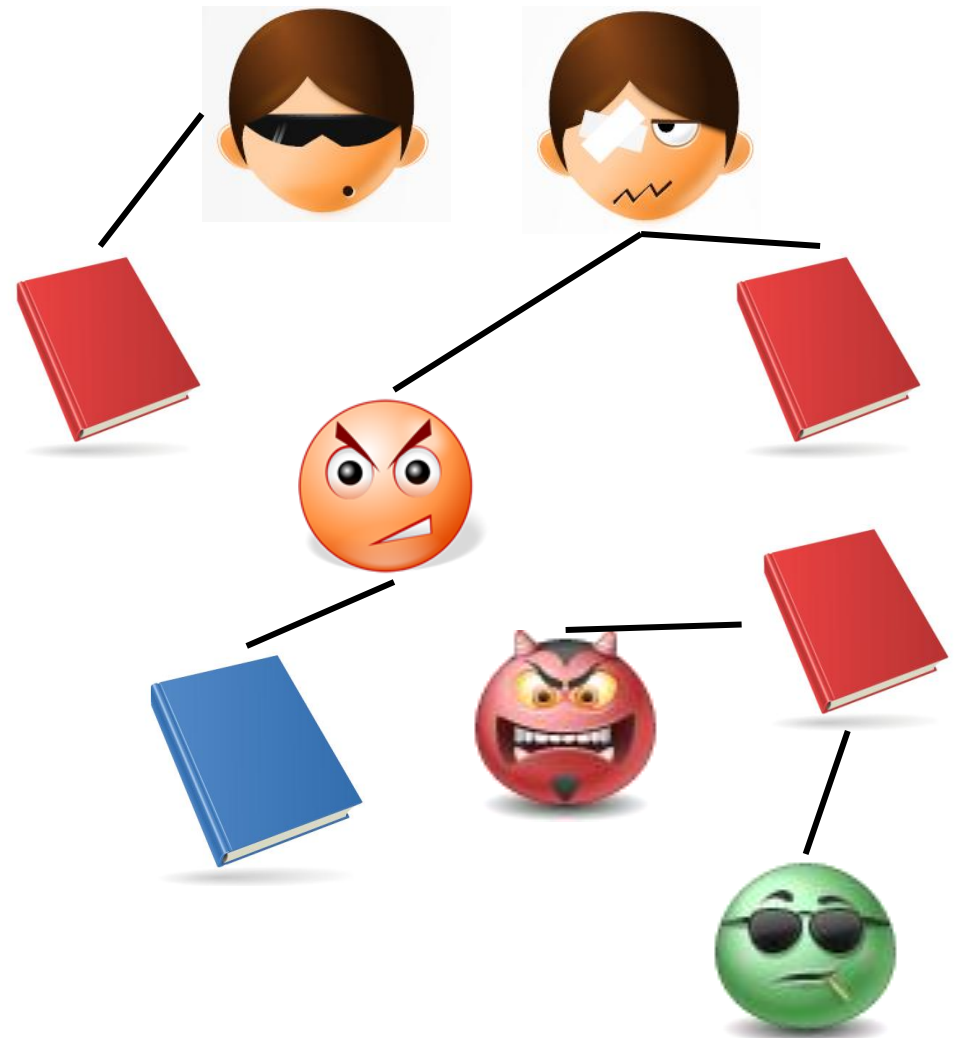
PROBLEM DEFINITION

Abstract Problem Statement

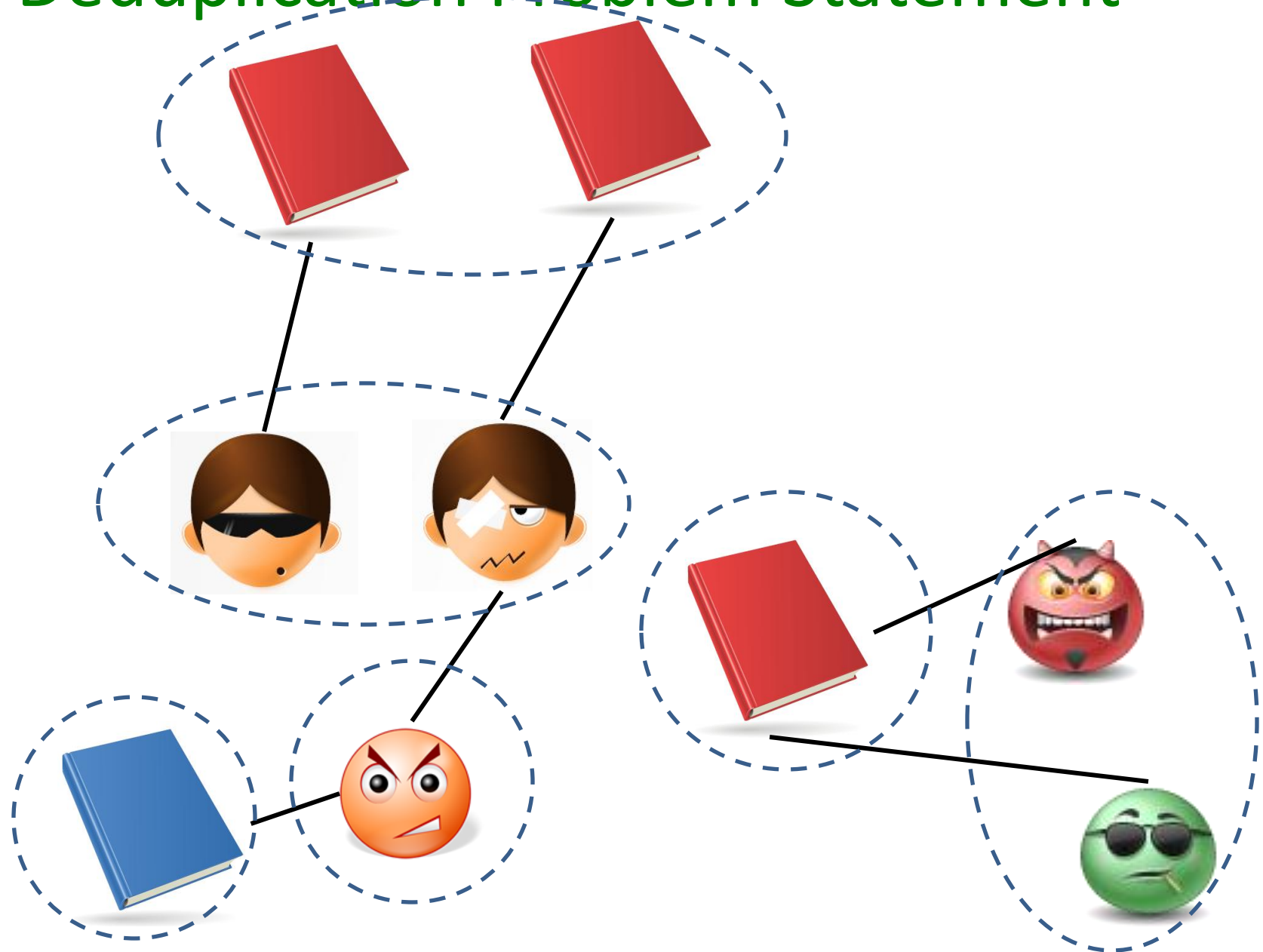
Real World



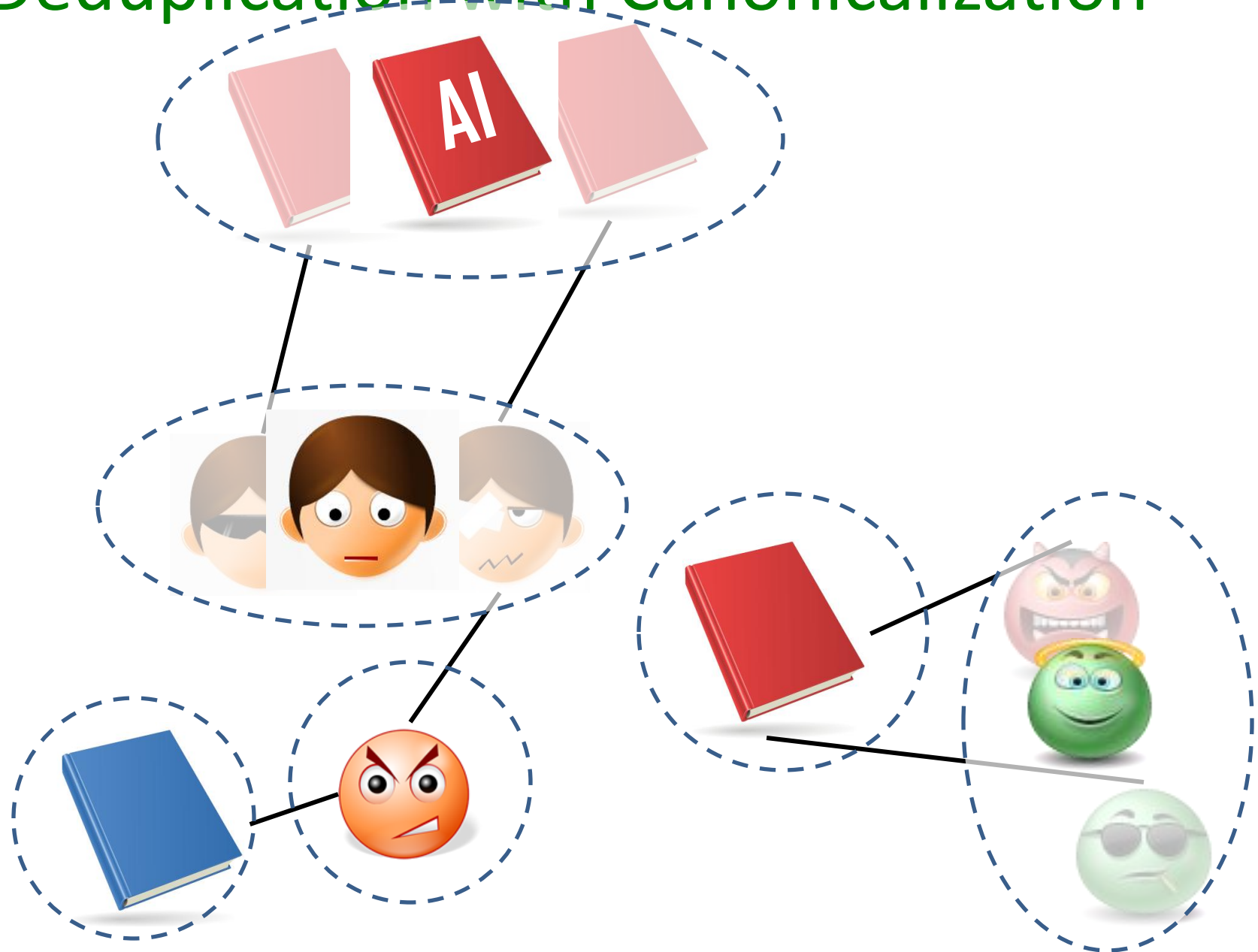
Digital World



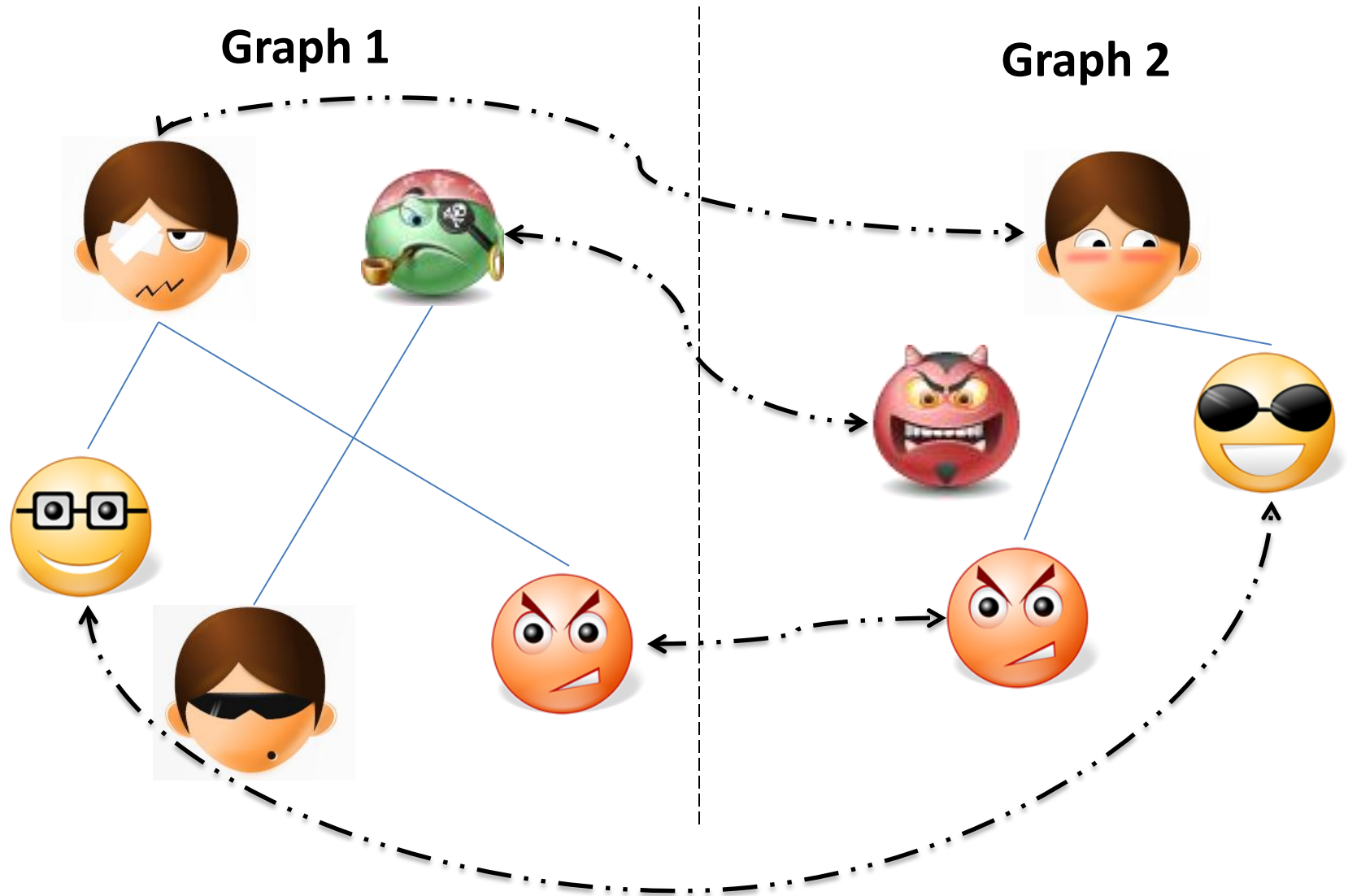
Deduplication Problem Statement



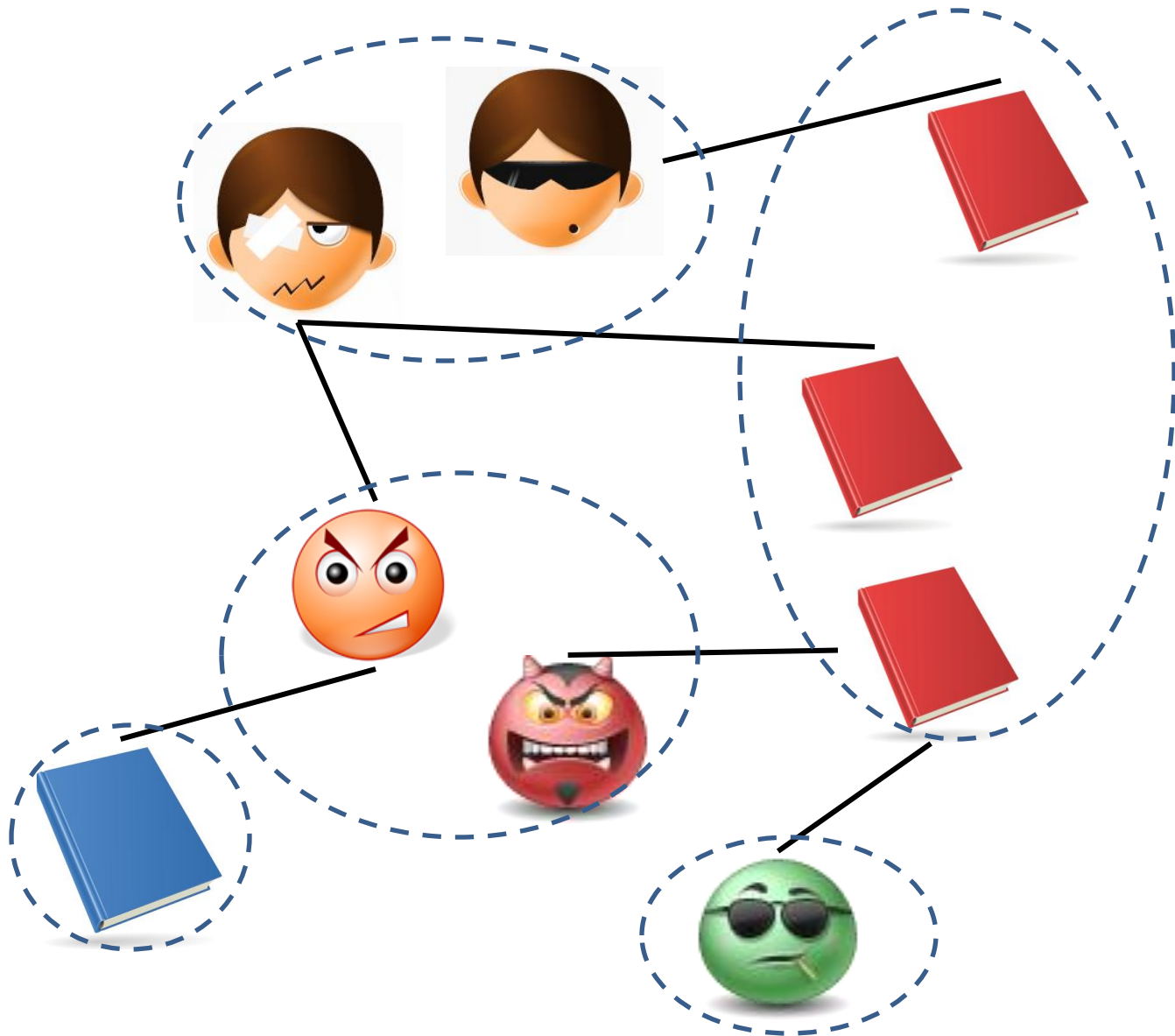
Deduplication with Canonicalization



Graph Alignment (& motif search)



Relationships are crucial



Notation & Assumptions

- R : set of records / mentions (typed)
- H : set of relations / hyperedges (typed)
- M : set of *matches* (record pairs that correspond to same entity)
- N : set of *non-matches* (record pairs corresponding to different entities)
- E : set of entities
- L : set of links

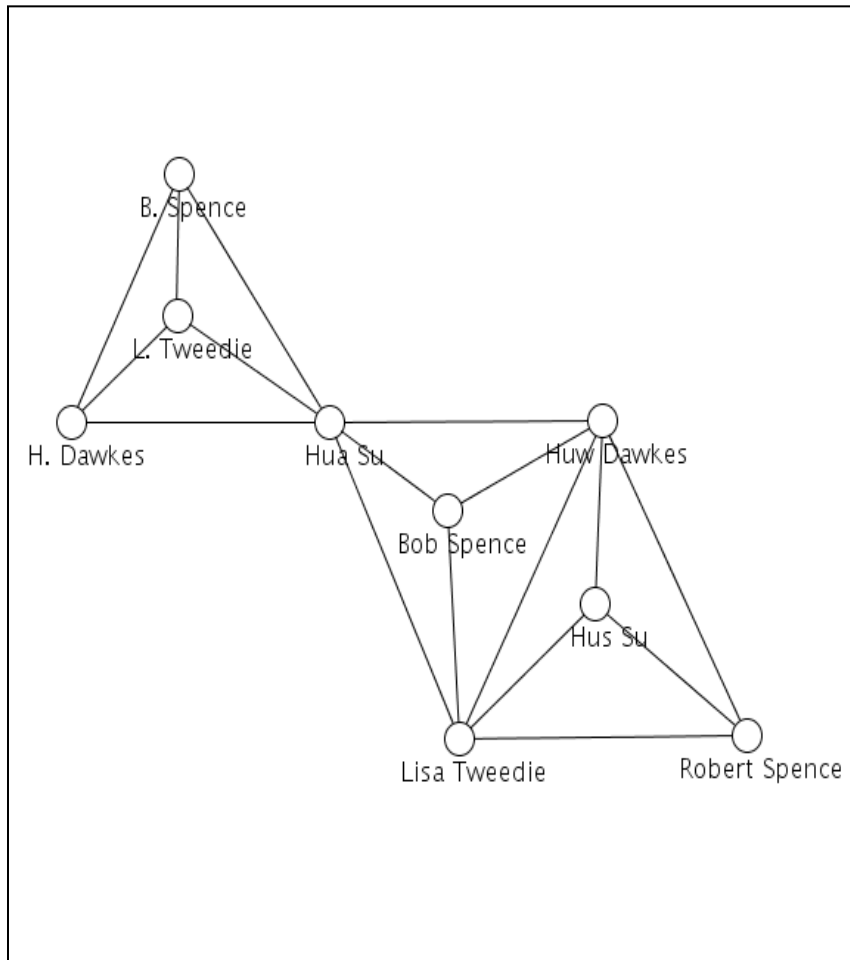
- True ($M_{true}, N_{true}, E_{true}, L_{true}$): according to real world
vs Predicted ($M_{pred}, N_{pred}, E_{pred}, L_{pred}$): by algorithm

Metrics

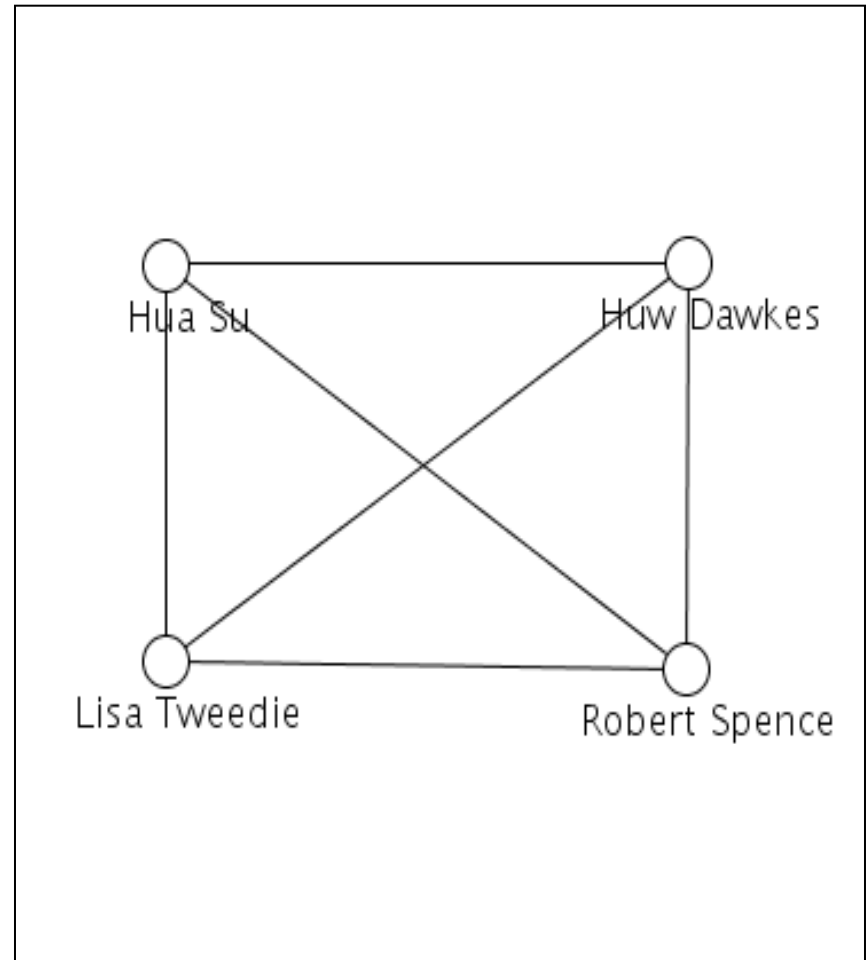
- Most algorithms use pairwise and cluster-based on each entity type
- Little work that evaluations correct prediction of links

MOTIVATING EXAMPLE: AUTHOR RESOLUTION

InfoVis Co-Author Network Fragment

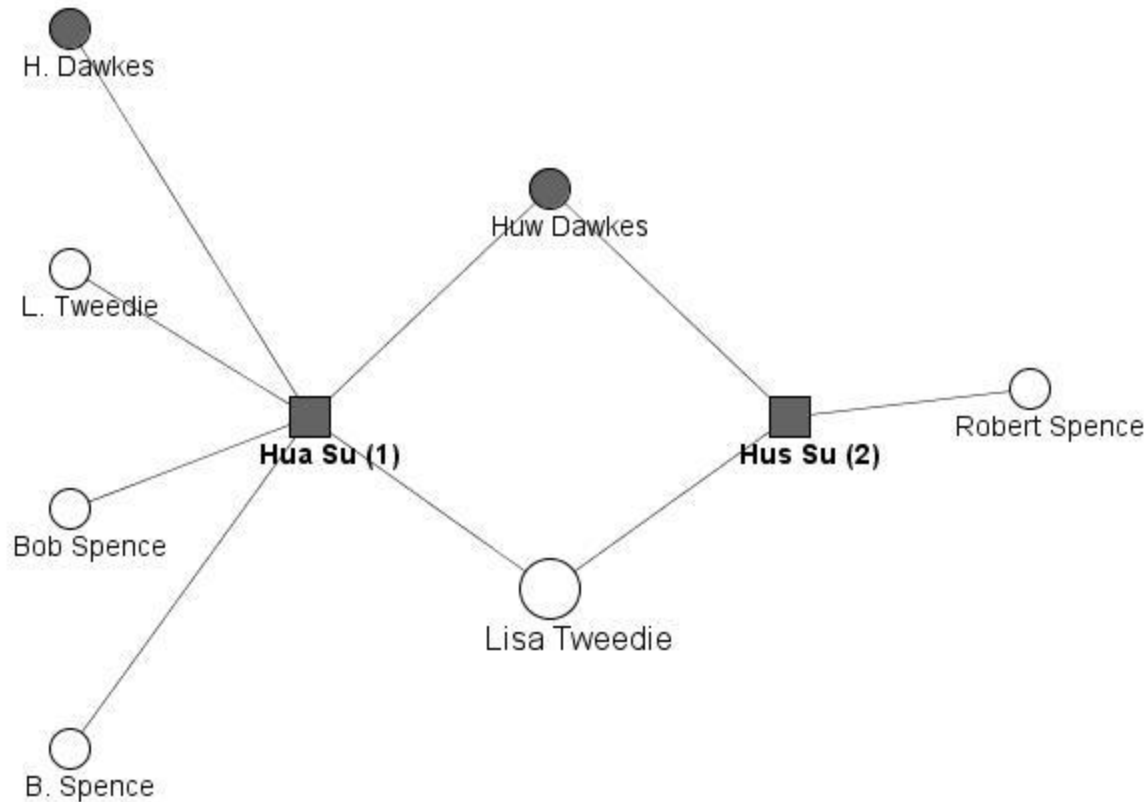


before



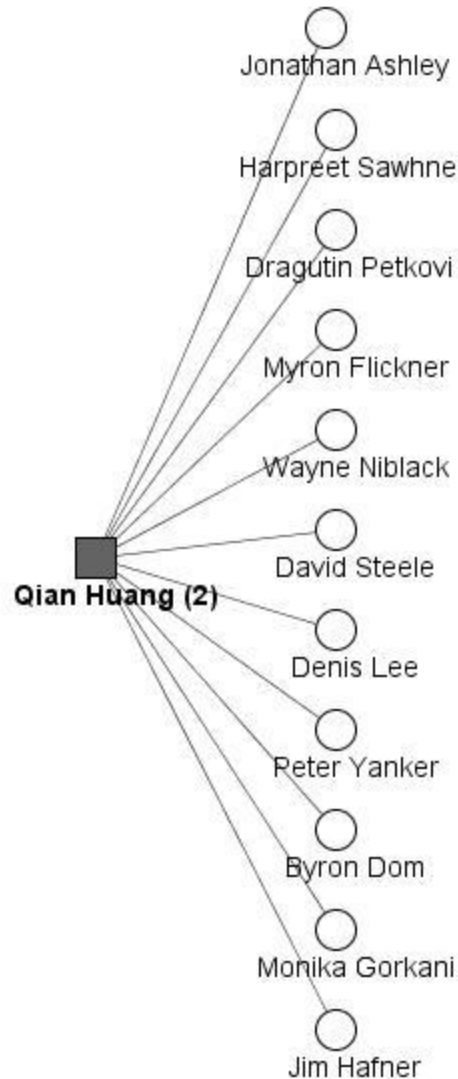
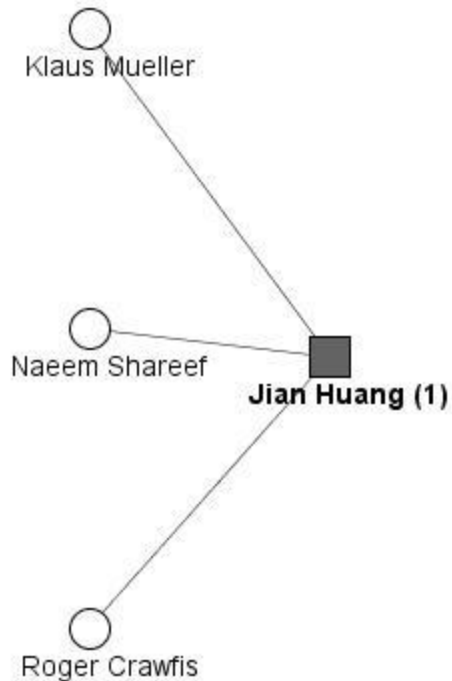
after

Relational Identification



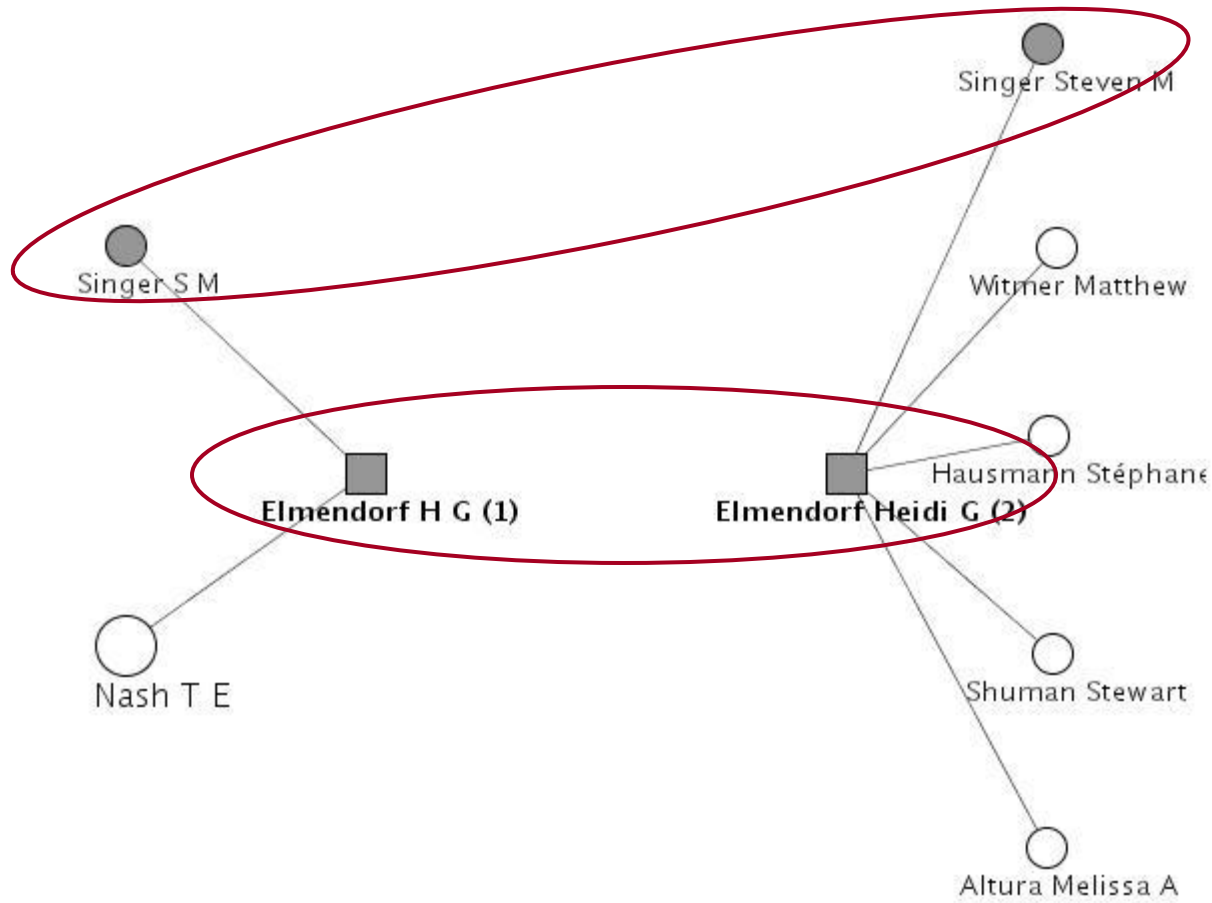
Very similar names.
Added evidence from
shared co-authors

Relational Disambiguation



Very similar names but
no shared collaborators

Collective Entity Resolution



One resolution provides evidence for another => joint resolution

PART 4-b

RELATIONAL FEATURES

Relational Features

- There are a variety of ways of improving ER performance when data is richer than a single table/entity type
- One of the simplest is to use additional information, to *enrich* model with *relational features* that will provide richer context for matching
 - This will often lead to increased precision
 - Relational information can help to distinguish references, add avoid false positives
 - It may also lead to increased recall
 - The best threshold will be different, and it may be, with the additional information, one can get increased recall as well.

Examples of relational features

- Value of edge or neighboring attribute (1-1)
- Aggregates (1-many)
 - Mode (sum, min, max) of related attribute
- Set similarity measures to compare nodes based on set of related nodes, e.g., compare neighborhoods
 - Overlap
 - Jaccard coefficient
 - Average similarity between set members
 - Others
 - Preferential Attachment
 - Adamic/Adar measure
 - SimRank
 - Katz score
- More sophisticated relational features can be easily constructed using FOL or relational language

Preferential Attachment Score

[Liben-Nowell & Kleinberg, JASIST07]

- Based on studies, e.g. [Newman, PRL01], showing that people with a larger number of existing relations are more likely to initiate new ones.

$$s(a, b) = |N_a| \cdot |N_b|$$

Set of a's neighbors

Adamic/Adar Measure

[Adamic & Adar, SN03]

- Two nodes are more similar if they share more items that are overall less frequent

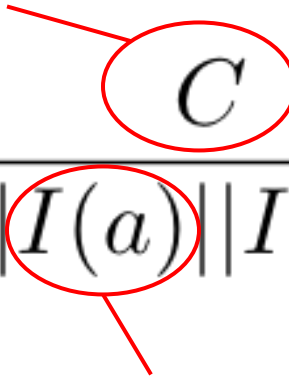
$$s(a, b) = \sum_{i \in \underbrace{\text{Shared items}}_{\substack{\text{Can be any kind of} \\ \text{shared attributes or} \\ \text{relationships to shared} \\ \text{entities}}}} \frac{1}{\underbrace{\log(\text{frequency}(i))}_{\substack{\text{Overall frequency} \\ \text{in the data}}}}$$

SimRank

[Jeh & Widom, KDD02]

- “Two objects are similar if they are related to similar objects”
- Defined as the unique solution to:

Decay factor between 0 and 1

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


Set of incoming edges into a

- Computed by iterating to convergence
- Initialization to $s(a, b) = 1$ if $a=b$ and 0 otherwise

Katz Score

- Two objects are similar if they are connected by shorter paths

$$s(a, b) = \sum_{l=1}^{\infty} \beta^l \cdot |\text{paths}^{\langle l \rangle}(a, b)|$$

Decay factor between 0 and 1

Set of paths between a and b of length exactly ℓ

- Since expensive to compute, often use approximate Katz, assuming some max path length of k

Using Hierarchical Structure

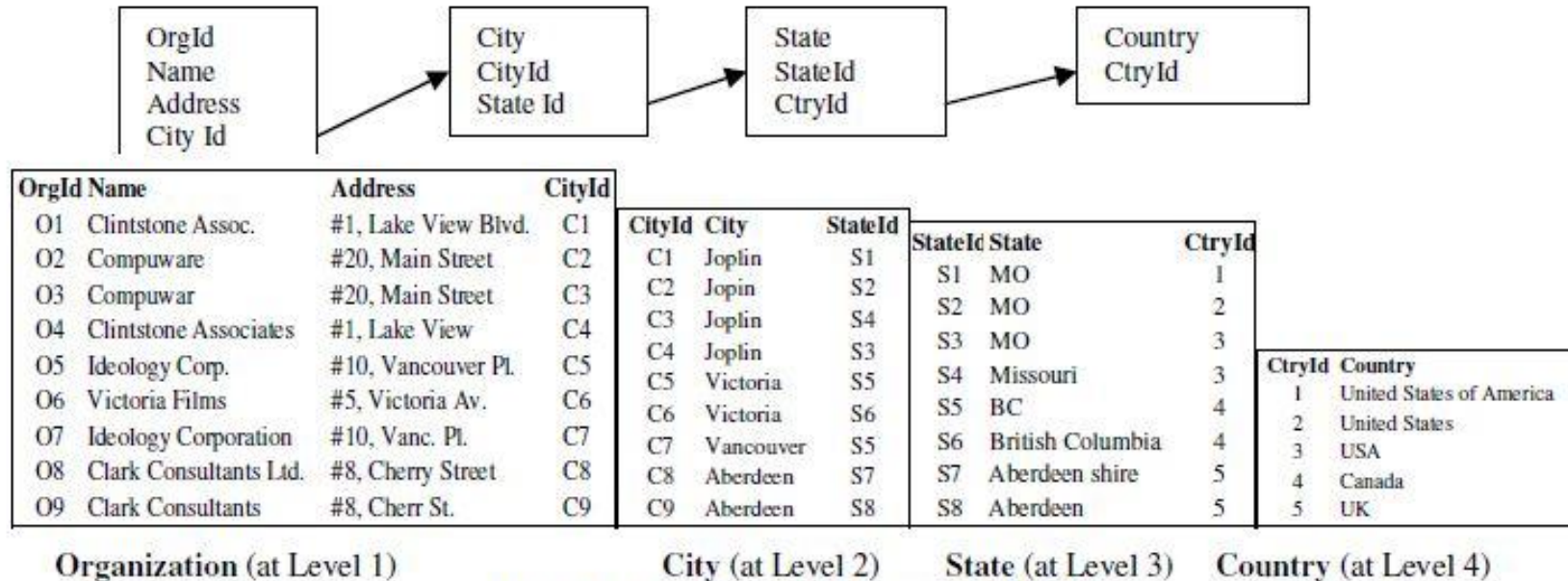


Figure 1: An Example Customer Database

- In dimensional hierarchy, [Ananthakrishna et al, VLDB02]
 - E.g., since same state maps to different countries, countries more likely to match
 - Since different countries (USA, UK), map to different states (child sets), unlikely to match

COLLECTIVE APPROACHES

Collective Approaches

- Decisions for cluster-membership depends on other clusters
 - Graph-based approaches
 - Agglomerative approaches
 - Generative Graphical Models
 - Declarative Approaches

PART 4-c

GRAPH-BASED APPROACHES

Graph-based Approaches

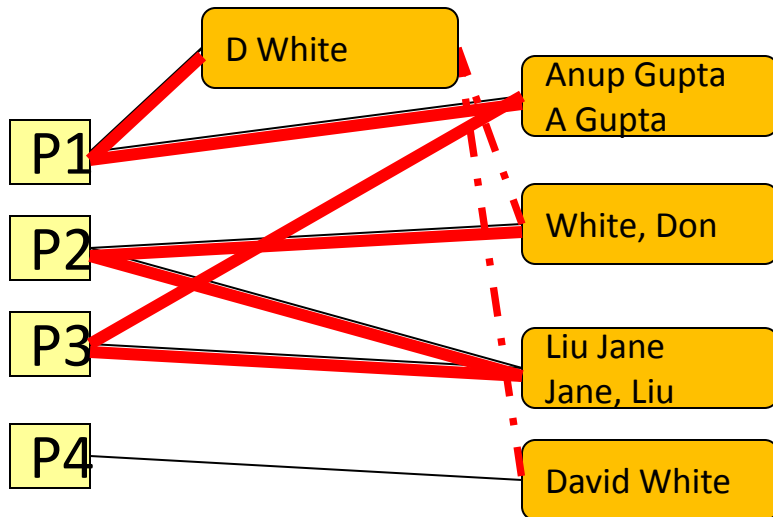
- **ReIDC [Kalashnikov et al, TODS06]**
- **Dependency Graph [Dong et al, SIGMOD05]**

Relational-based Data Cleaning (ReIDC)

- Reference Disambiguation [Kalashnikov et al, TODS06]
 - Ensuring that references (i.e., “foreign keys”₁) in a database point to the correct entities.
 - E.g., a database may store information about two distinct individuals ‘Donald L. White’ and ‘Donald E. White’, both of whom are referred to as ‘D. White’ in another database.
- Idea
 - Use both feature-based similarity and relational context to help determine references
 - Construct entity graph
 - Use a feature-based method to identify a set of candidate entities (choices)
 - Graph theoretic techniques are then used to discover and analyze similarity of relationships that exist between the entity containing the reference and the set of candidates.

ReIDC Example

P1	D White, A Gupta
P2	Liu, Jane & White, Don
P3	Anup Gupta and Liu Jane
P4	David White



Relate D White and Don White through the third paper

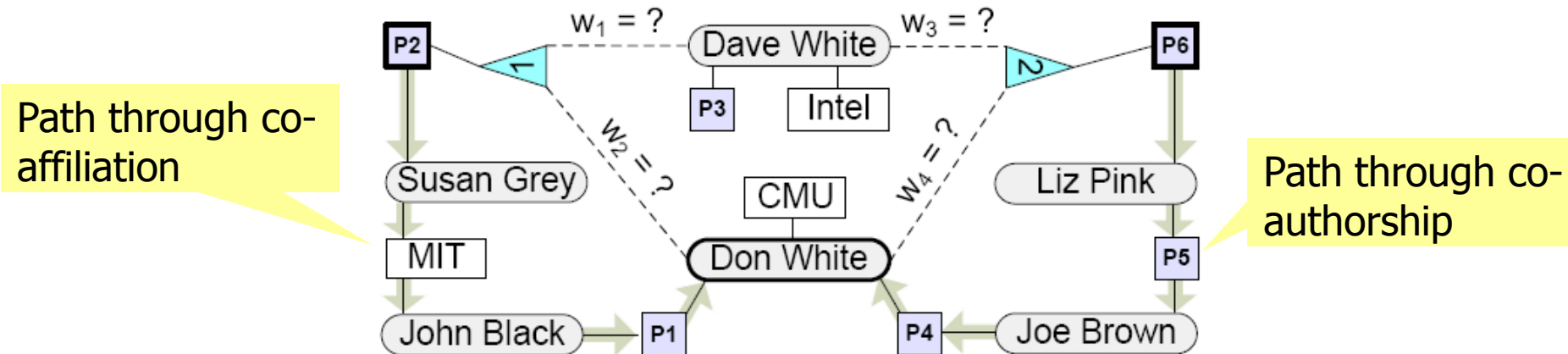
Path in graph makes D White more similar to Don White than David White

RelDC: Example with multiple entity types

$\langle A_1, \text{'Dave White'}, \text{'Intel'} \rangle$
 $\langle A_2, \text{'Don White'}, \text{'CMU'} \rangle$
 $\langle A_3, \text{'Susan Grey'}, \text{'MIT'} \rangle$
 $\langle A_4, \text{'John Black'}, \text{'MIT'} \rangle$
 $\langle A_5, \text{'Joe Brown'}, \text{unknown} \rangle$
 $\langle A_6, \text{'Liz Pink'}, \text{unknown} \rangle$

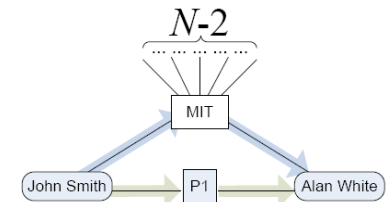
$\langle P_1, \text{'Databases ...'}, \text{'John Black'}, \text{'Don White'} \rangle$
 $\langle P_2, \text{'Multimedia ...'}, \text{'Sue Grey'}, \text{'D. White'} \rangle$
 $\langle P_3, \text{'Title3 ...'}, \text{'Dave White'} \rangle$
 $\langle P_4, \text{'Title5 ...'}, \text{'Don White'}, \text{'Joe Brown'} \rangle$
 $\langle P_5, \text{'Title6 ...'}, \text{'Joe Brown'}, \text{'Liz Pink'} \rangle$
 $\langle P_6, \text{'Title7 ...'}, \text{'Liz Pink'}, \text{'D. White'} \rangle$

Task: resolve author references in papers to author table



Computing Relational Similarities

- Graph G with edges denoting node similarity or some form of relationship, find connection strength between node u, v
- Methods
 - Simple methods: shortest path length or flow
 - Fails for high-degree nodes
 - Propose a probabilistic walk-based model
 - Treat edge weights as probability of transitioning out of node
 - Probability of reaching u from v via random walk
 - Extended to work for graphs with mutually exclusive choice nodes



ReIDC Algorithm

- Resolve whatever is possible via textual similarity alone
- Create relationship graph with unresolved references connected via choice nodes to options
- Find connection strength between each unresolved reference to options, resolve to strongest of these

Graph-based Approaches

- ReIDC, [Kalashnikov et al, TODS06]
- **Dependency Graph [Dong et al, SIGMOD05]**

Reference Reconciliation

- Focus on multi-relational data, specifically personal information management, where there few attribute values, multi-valued attributes, and missing values
- Highlevel:
 - Make use of relational context
 - given two references to persons, consider their co-authors and email contacts, to help decide whether to reconcile them.
 - Propagate resolution
 - When reconcile two papers, obtain additional evidence for reconciling the person references to their authors.
 - Perform reference enrichment (canonicalization?)
 - when we reconcile two person references, we gather the different representations of the person's name, email addresses, and enlarge co-authors list and email-contacts

References

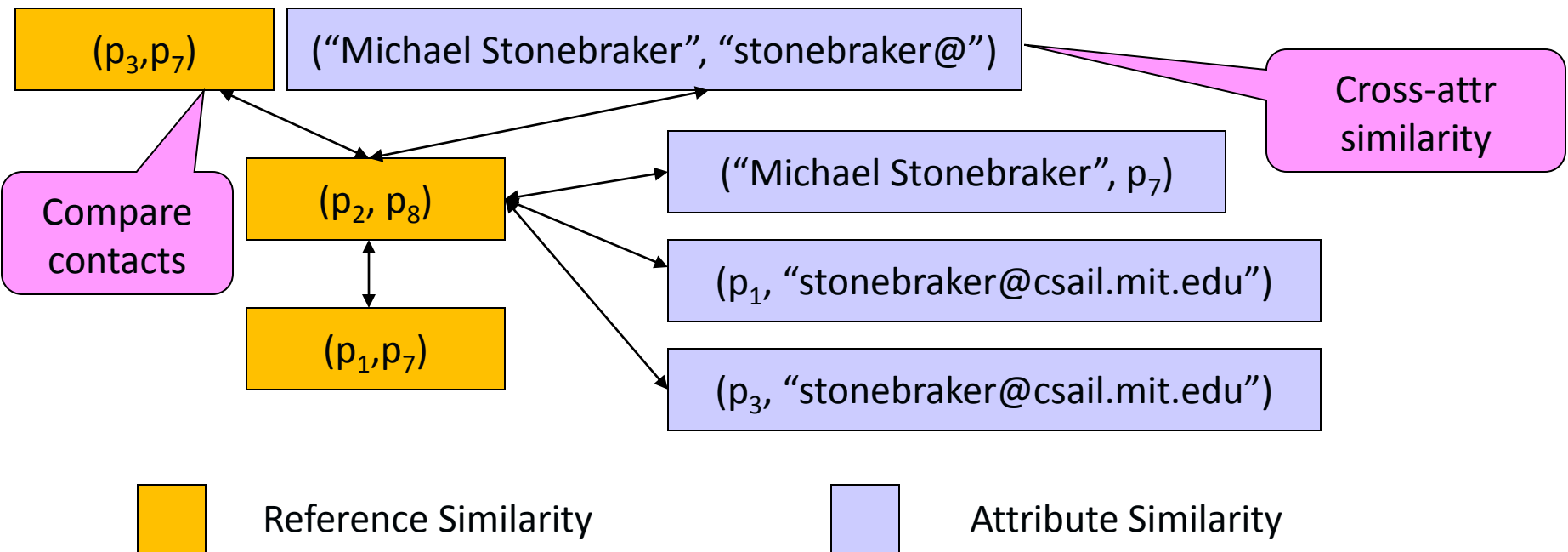
- **Article:**
 $a_1 = (\text{"Distributed Query Processing"}, \text{"169-180"}, \{p_1, p_2, p_3\}, c_1)$
 $a_2 = (\text{"Distributed query processing"}, \text{"169-180"}, \{p_4, p_5, p_6\}, c_2)$
- **Venue:**
 $c_1 = (\text{"ACM Conference on Management of Data"}, \text{"1978"}, \text{"Austin, Texas"})$
 $c_2 = (\text{"ACM SIGMOD"}, \text{"1978"}, \text{null})$
- **Person:**
 $p_1 = (\text{"Robert S. Epstein"}, \text{null})$
 $p_2 = (\text{"Michael Stonebraker"}, \text{null})$
 $p_3 = (\text{"Eugene Wong"}, \text{null})$
 $p_4 = (\text{"Epstein, R.S."}, \text{null})$
 $p_5 = (\text{"Stonebraker, M."}, \text{null})$
 $p_6 = (\text{"Wong, E."}, \text{null})$
")

Real-World Entities

- **Article:**
 - $a_1 = (\text{"Distributed Query Processing"}, \text{"169-180"}, \{p_1, p_2, p_3\}, c_1)$
 - $a_2 = (\text{"Distributed query processing"}, \text{"169-180"}, \{p_4, p_5, p_6\}, c_2)$
- **Venue:**
 - $c_1 = (\text{"ACM Conference on Management of Data"}, \text{"1978"}, \text{"Austin, Texas"})$
 - $c_2 = (\text{"ACM SIGMOD"}, \text{"1978"}, \text{null})$
- **Person:**
 - $p_1 = (\text{"Robert S. Epstein"}, \text{null})$
 - $p_2 = (\text{"Michael Stonebraker"}, \text{null})$
 - $p_3 = (\text{"Eugene Wong"}, \text{null})$
 - $p_4 = (\text{"Epstein, R.S."}, \text{null})$
 - $p_5 = (\text{"Stonebraker, M."}, \text{null})$
 - $p_6 = (\text{"Wong, E."}, \text{null})$
 - $p_7 = (\text{"Eugene Wong"}, \text{"eugene@berkeley.edu"})$
 - $p_8 = (\text{null}, \text{"stonebraker@csail.mit.edu"})$
 - $p_9 = (\text{"mike"}, \text{"stonebraker@csail.mit.edu"})$

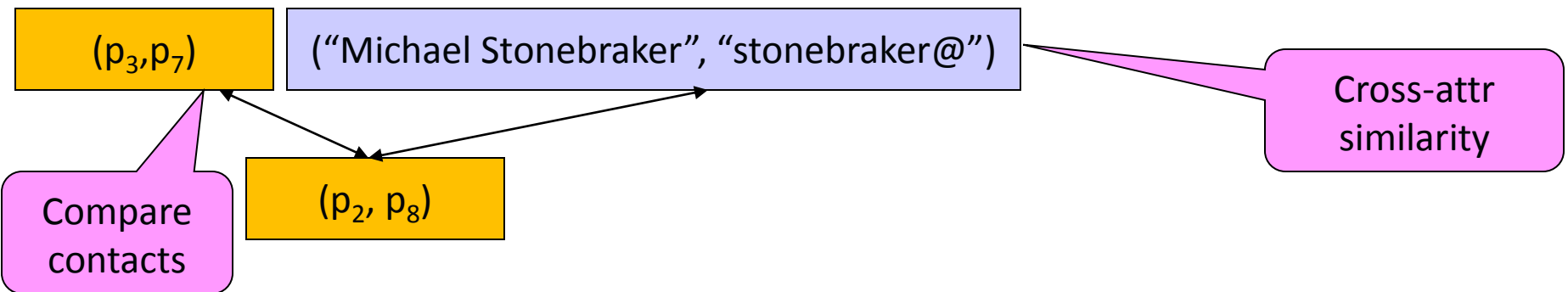
Framework: Dependency Graph

- $p_2 = (\text{"Michael Stonebraker"}, \text{null}, \{p_1, p_3\})$
 $p_3 = (\text{"Eugene Wong"}, \text{null}, \{p_1, p_2\})$
 $p_7 = (\text{"Eugene Wong"}, \text{"eugene@berkeley.edu"}, \{p_8\})$
 $p_8 = (\text{null}, \text{"stonebraker@csail.mit.edu"}, \{p_7\})$
 $p_9 = (\text{"mike"}, \text{"stonebraker@csail.mit.edu"}, \text{null})$



Framework: Dependency Graph

- $p_2 = (\text{"Michael Stonebraker"}, \text{null}, \{p_1, p_3\})$
 $p_3 = (\text{"Eugene Wong"}, \text{null}, \{p_1, p_2\})$
 $p_7 = (\text{"Eugene Wong"}, \text{"eugene@berkeley.edu"}, \{p_8\})$
 $p_8 = (\text{null}, \text{"stonebraker@csail.mit.edu"}, \{p_7\})$
 $p_9 = (\text{"mike"}, \text{"stonebraker@csail.mit.edu"}, \text{null})$



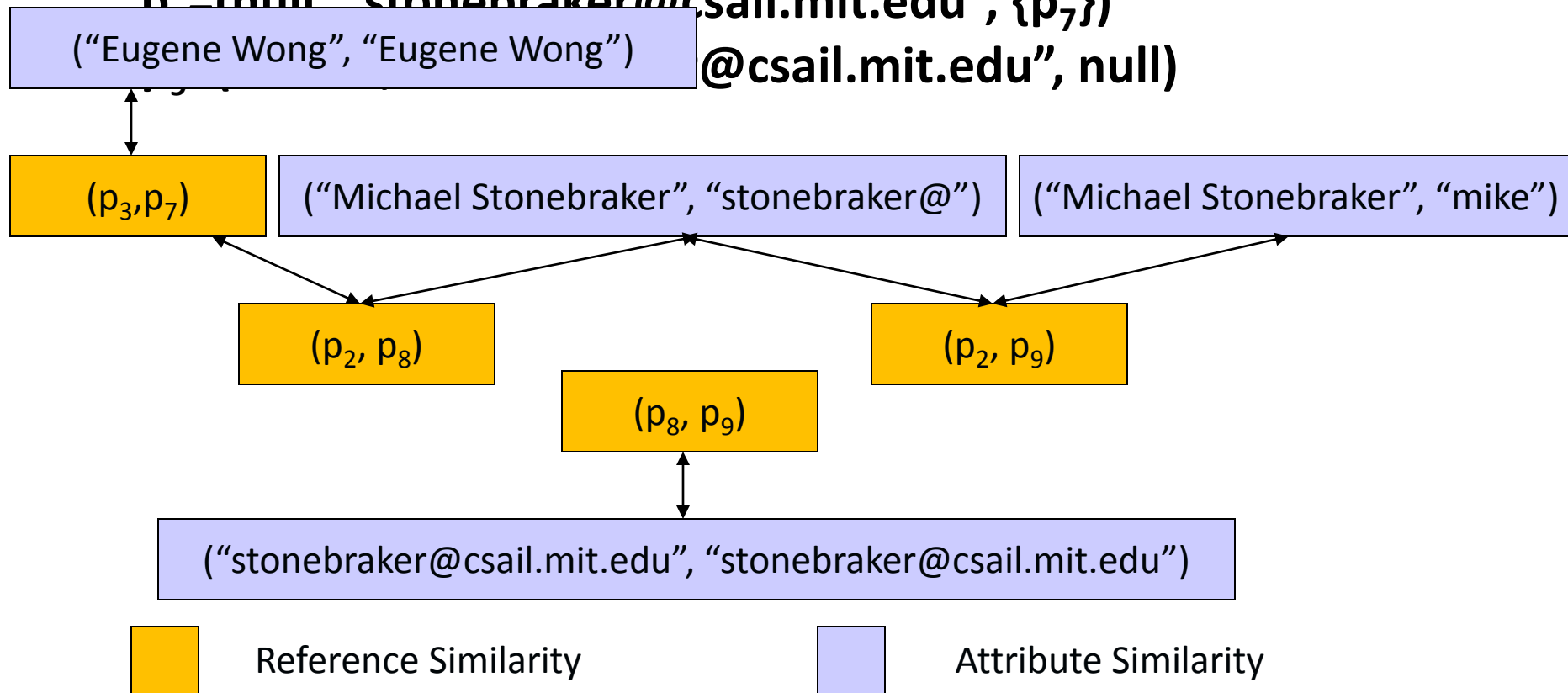
Reference Similarity



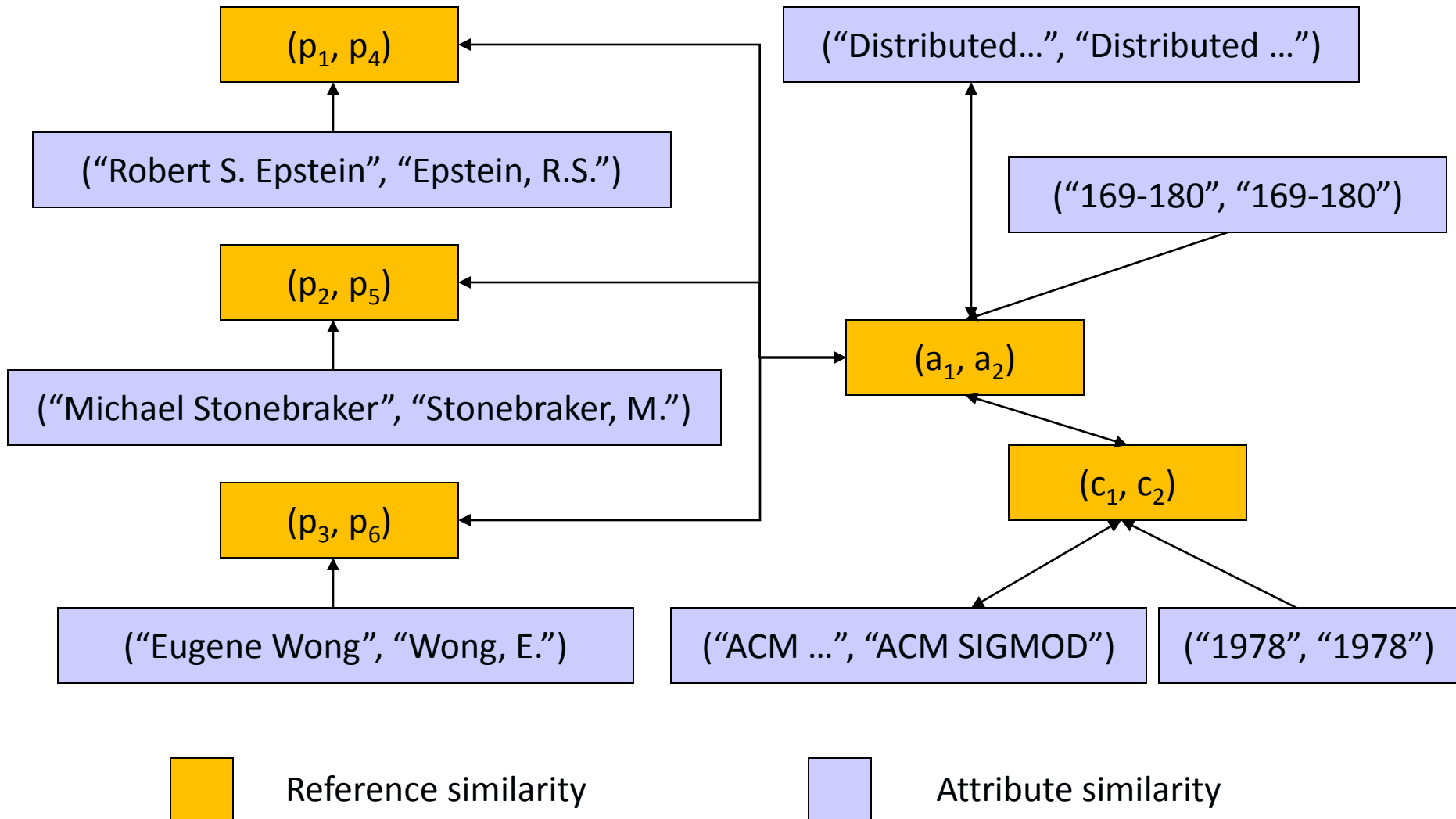
Attribute Similarity

Framework: Dependency Graph

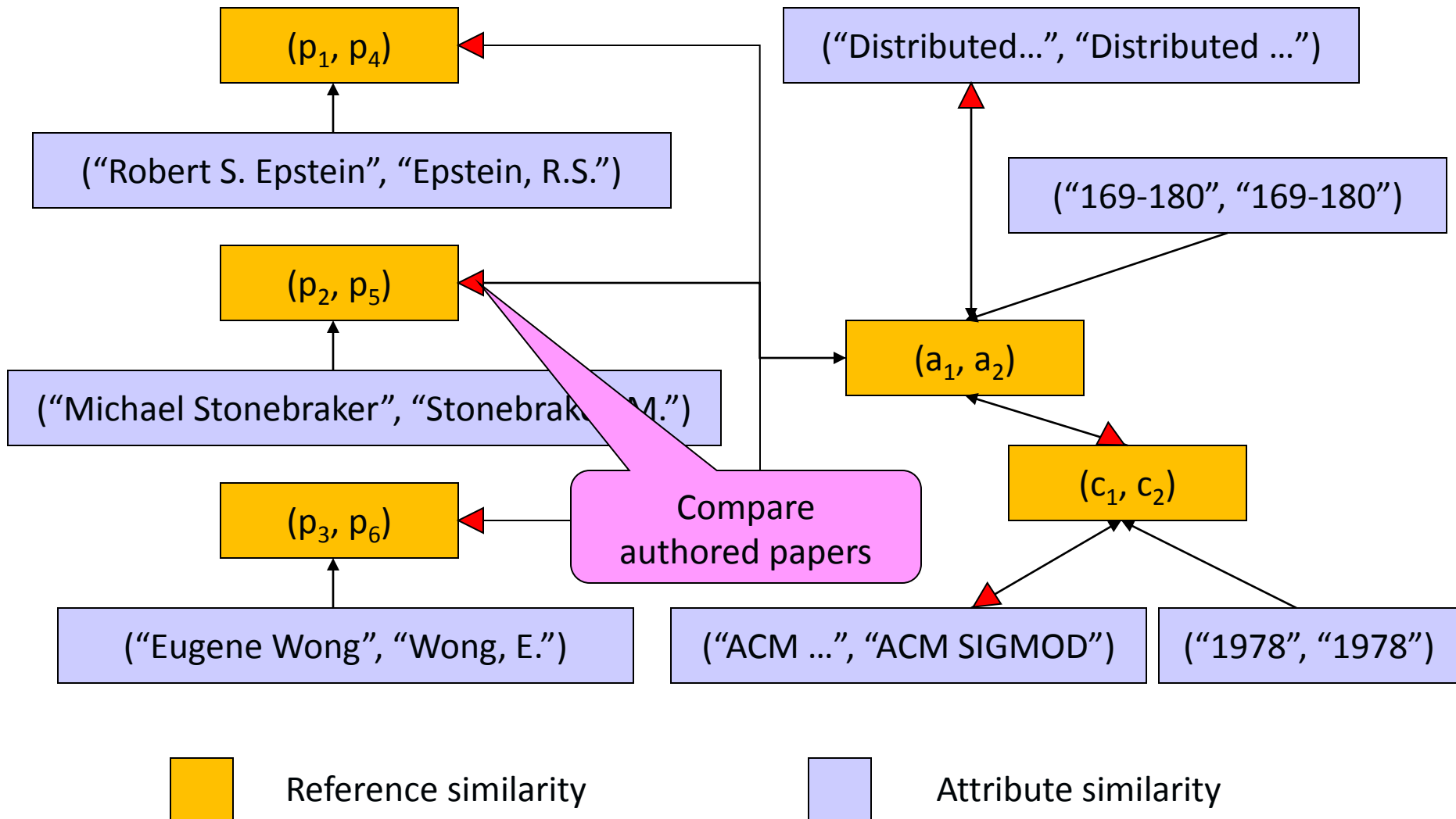
- $p_2 = (\text{"Michael Stonebraker"}, \text{null}, \{p_1, p_3\})$
 $p_3 = (\text{"Eugene Wong"}, \text{null}, \{p_1, p_2\})$
 $p_7 = (\text{"Eugene Wong"}, \text{"eugene@berkeley.edu"}, \{p_8\})$
 $p_8 = (\text{null}, \text{"stonebraker@csail.mit.edu"}, \{p_7\})$
 $p_9 = (\text{"Eugene Wong"}, \text{"Eugene Wong"}, \text{"@csail.mit.edu"}, \text{null})$



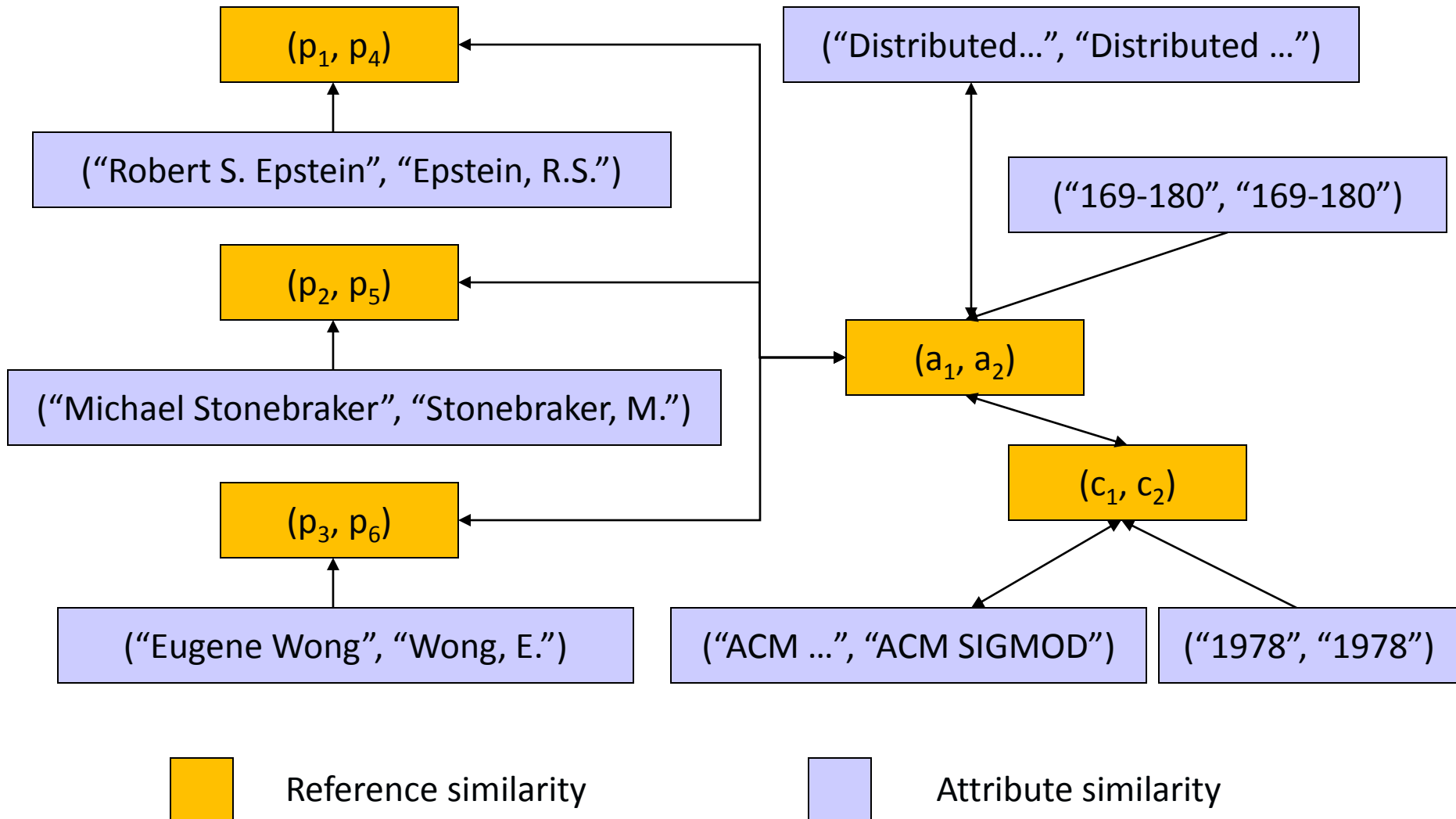
Exploit the Dependency Graph



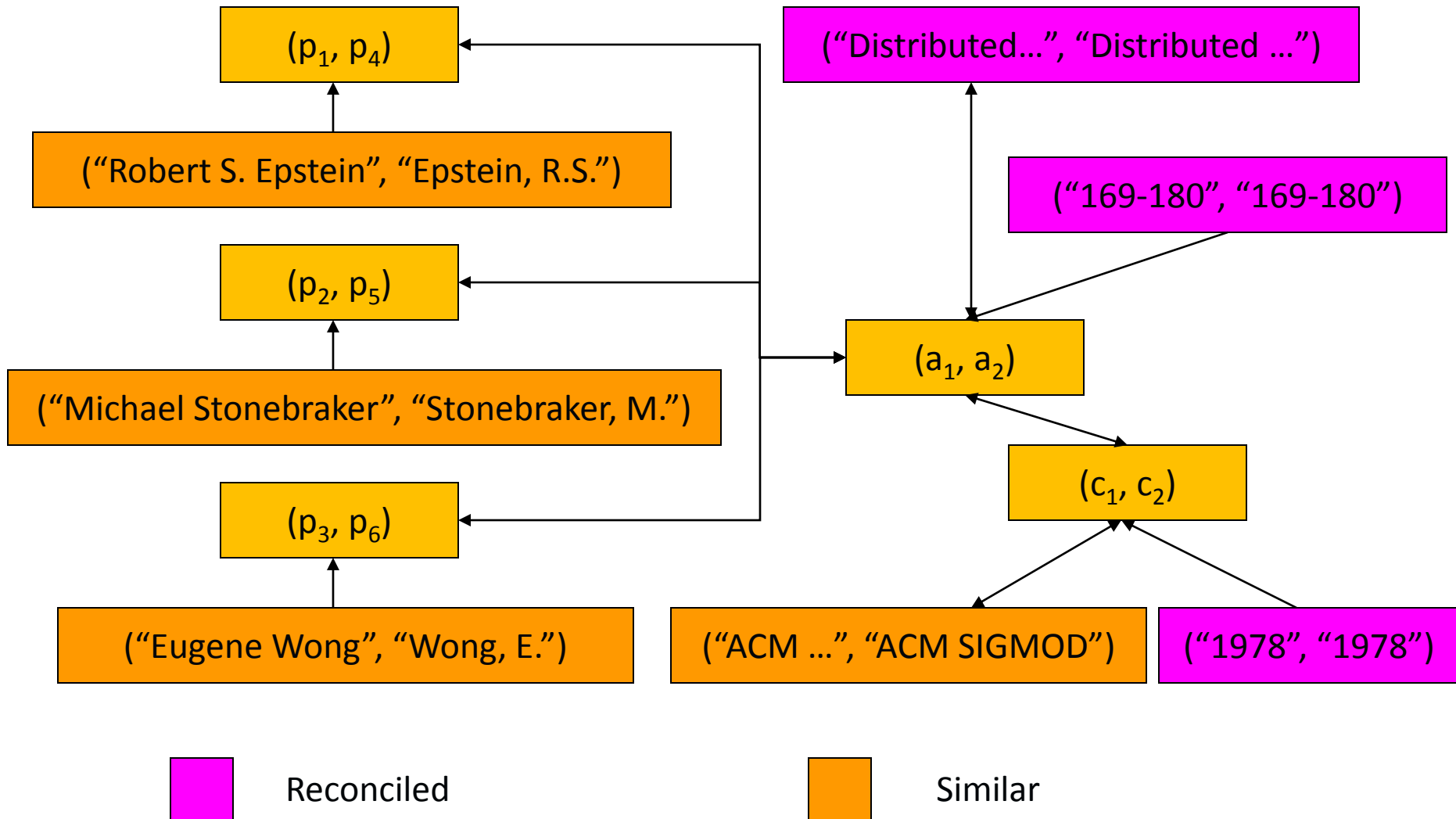
Dependency Graph Example II



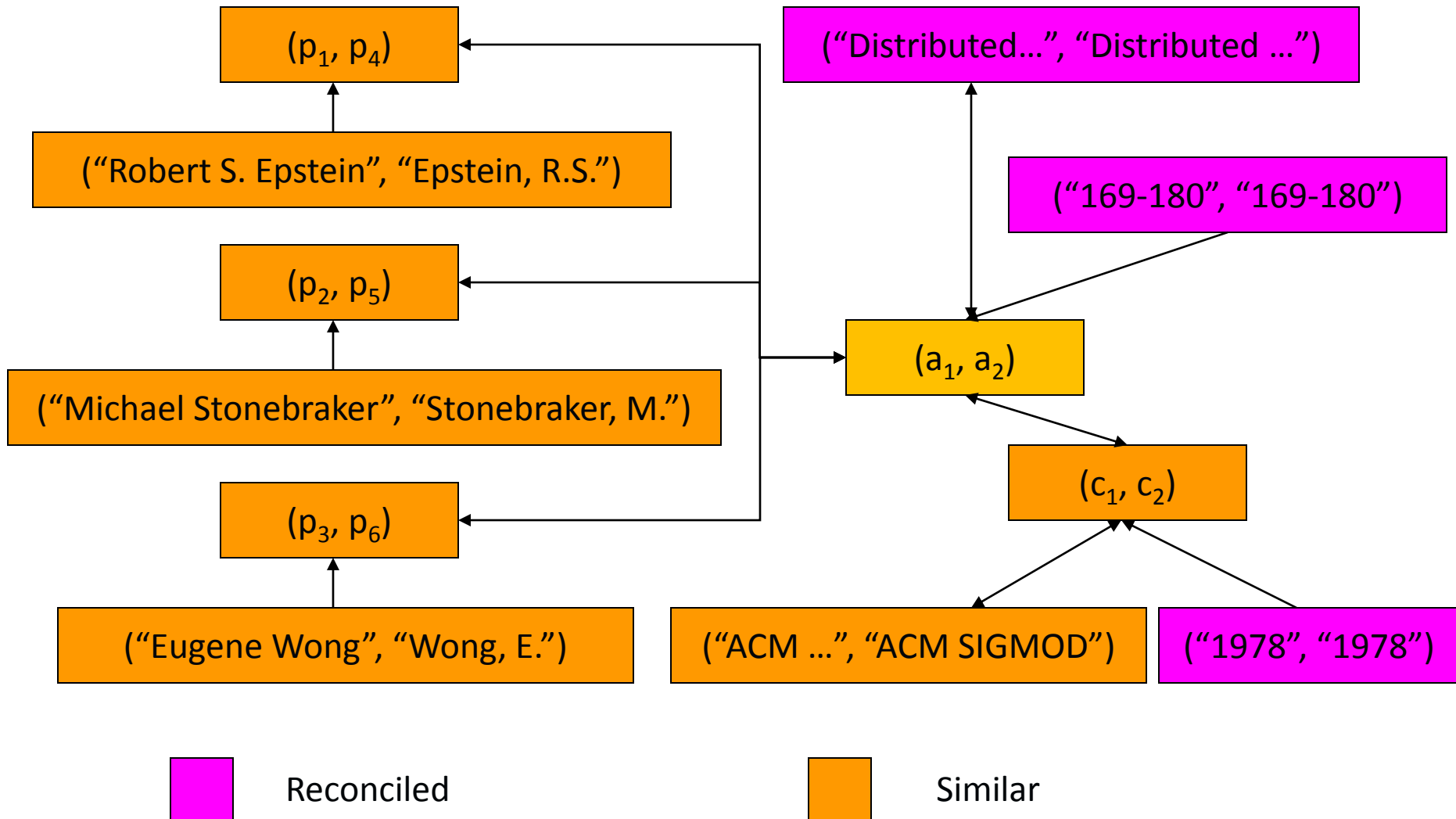
Exploit the Dependency Graph



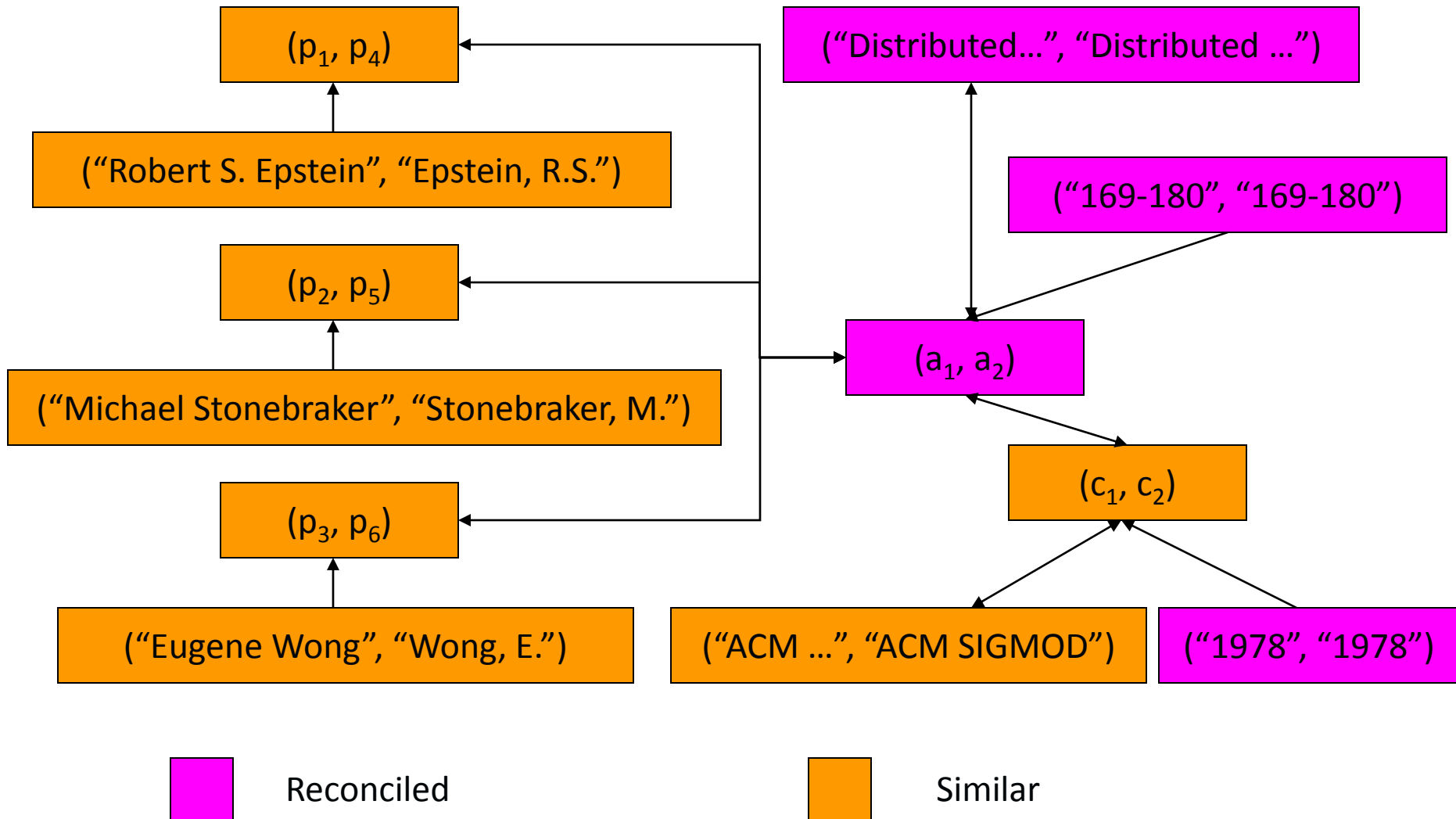
Exploit the Dependency Graph



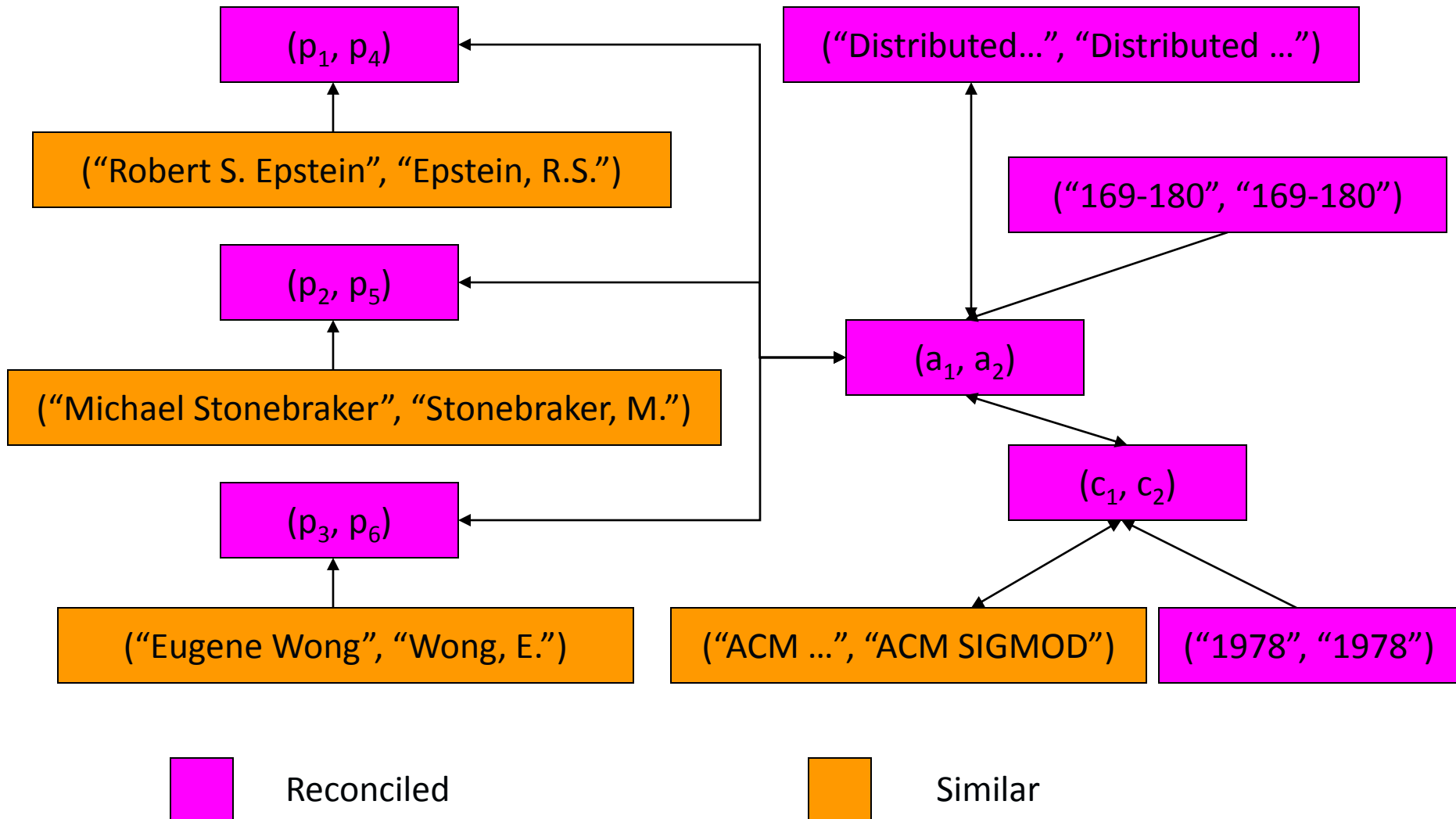
Exploit the Dependency Graph



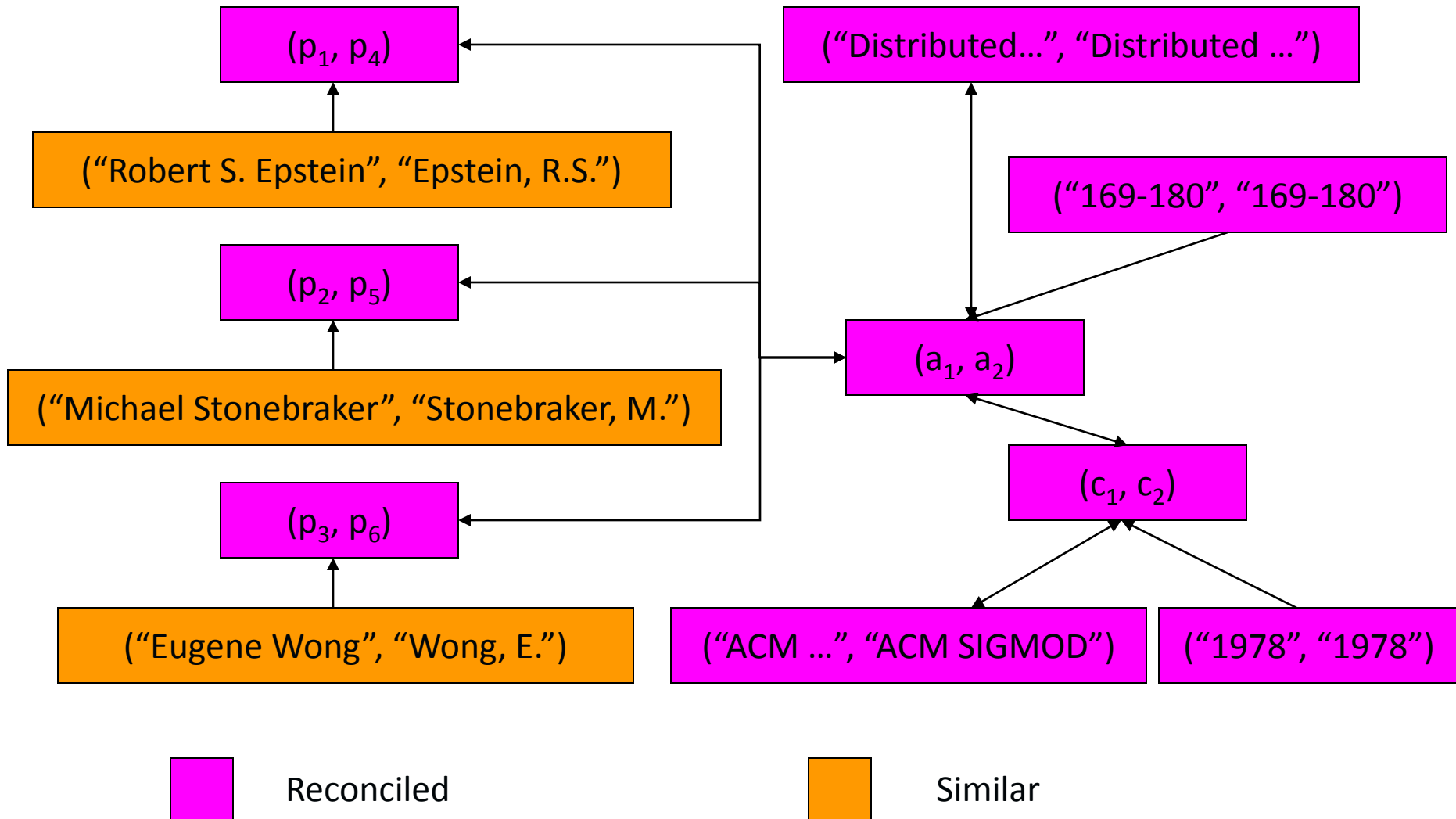
Exploit the Dependency Graph



Exploit the Dependency Graph

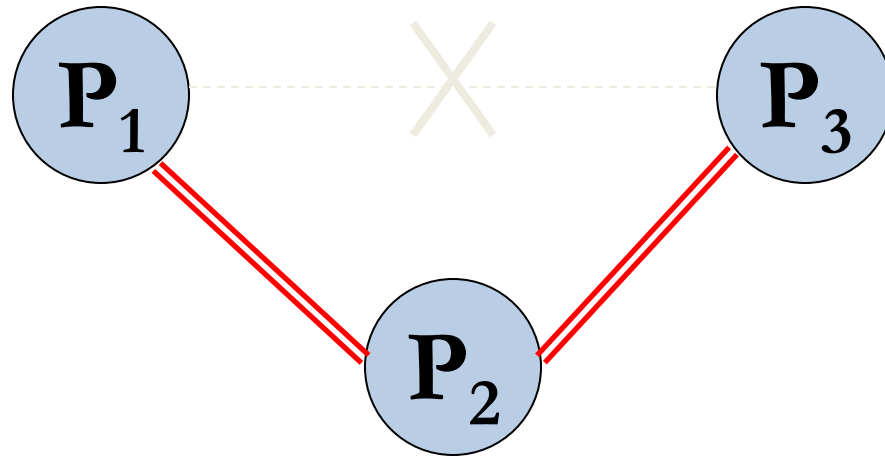


Exploit the Dependency Graph



Enforce Constraints

- **Problem:**



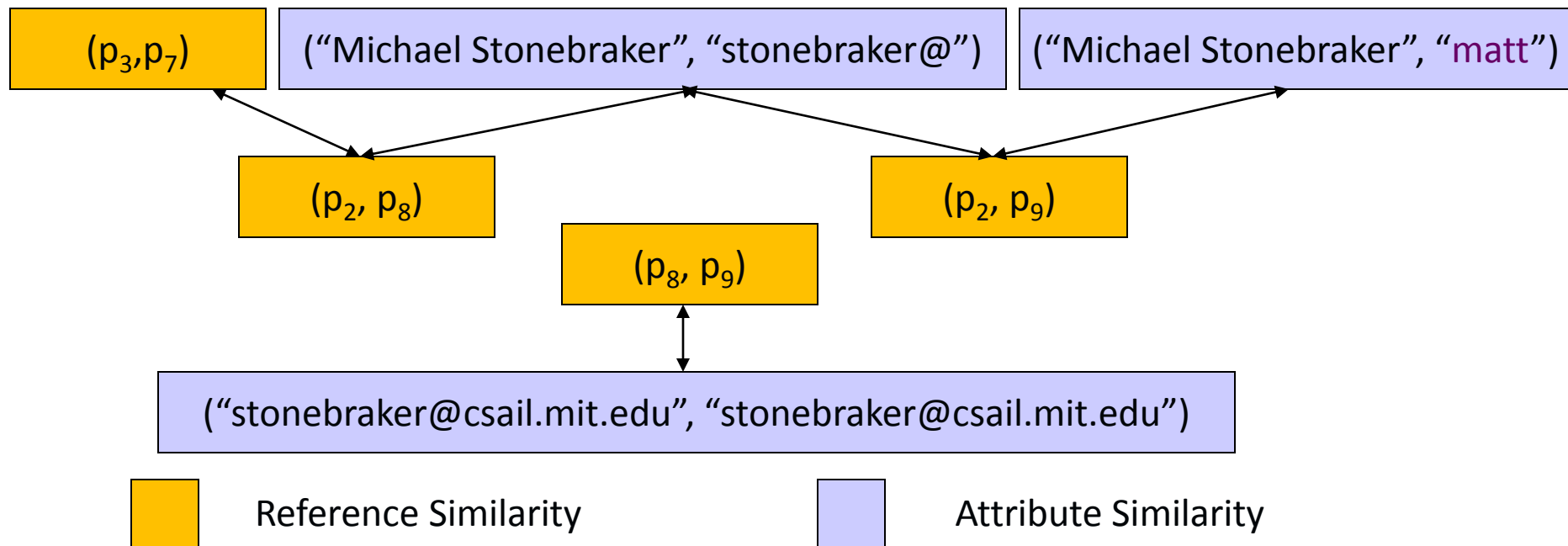
- **Solution:**

Propagate negative information—*Constraints*

- *Non-merge* node: the two elements are guaranteed to be different and should never be merged

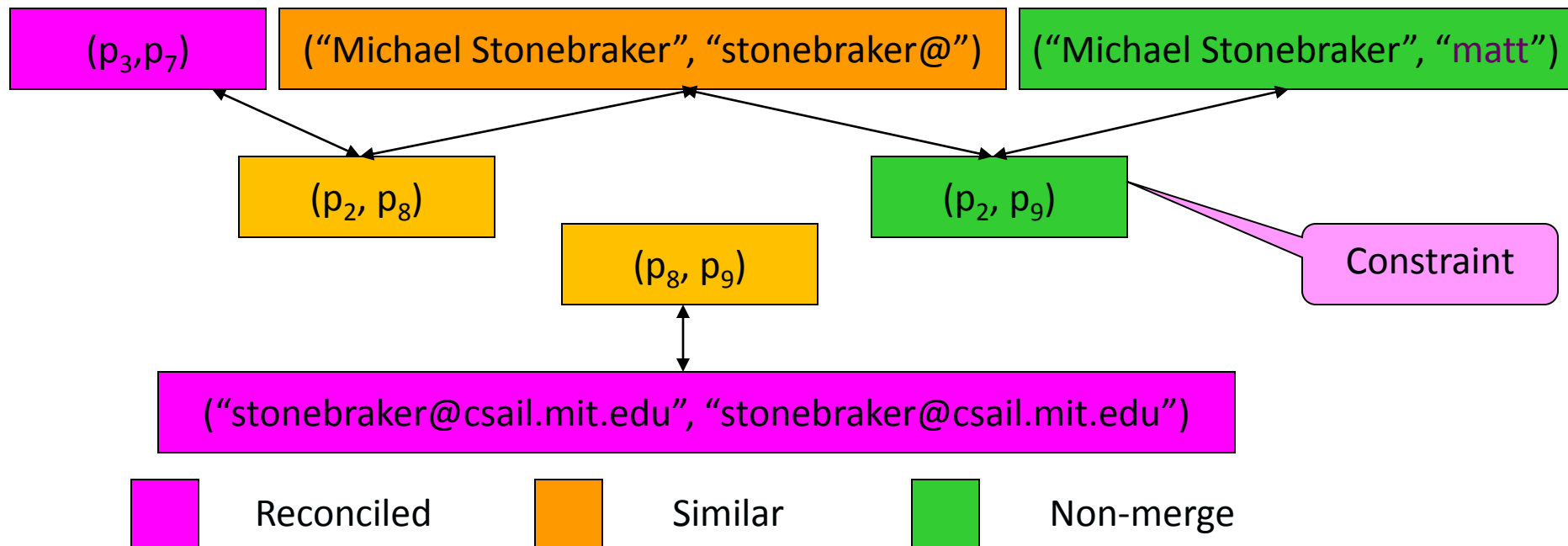
Enforce Constraints by Propagating Negative Information

- $p_2 = (\text{"Michael Stonebraker"}, \text{null}, \{p_1, p_3\})$
 $p_3 = (\text{"Eugene Wong"}, \text{null}, \{p_1, p_2\})$
 $p_7 = (\text{"Eugene Wong"}, \text{"eugene@berkeley.edu"}, \{p_8\})$
 $p_8 = (\text{null}, \text{"stonebraker@csail.mit.edu"}, \{p_7\})$
 $p_9 = (\text{"matt"}, \text{"stonebraker@csail.mit.edu"}, \text{null})$



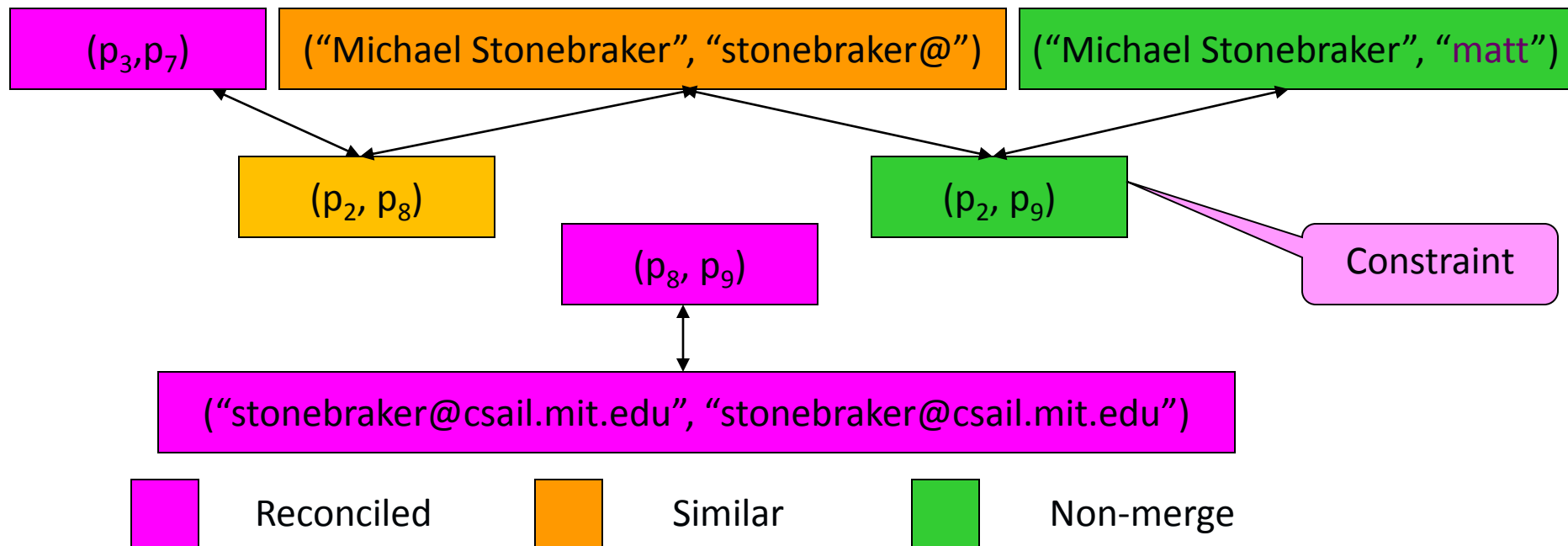
Enforce Constraints by Propagating Negative Information

- $p_2 = (\text{"Michael Stonebraker"}, \text{null}, \{p_1, p_3\})$
 $p_3 = (\text{"Eugene Wong"}, \text{null}, \{p_1, p_2\})$
 $p_7 = (\text{"Eugene Wong"}, \text{"eugene@berkeley.edu"}, \{p_8\})$
 $p_8 = (\text{null}, \text{"stonebraker@csail.mit.edu"}, \{p_7\})$
 $p_9 = (\text{"**matt**"}, \text{"stonebraker@csail.mit.edu"}, \text{null})$



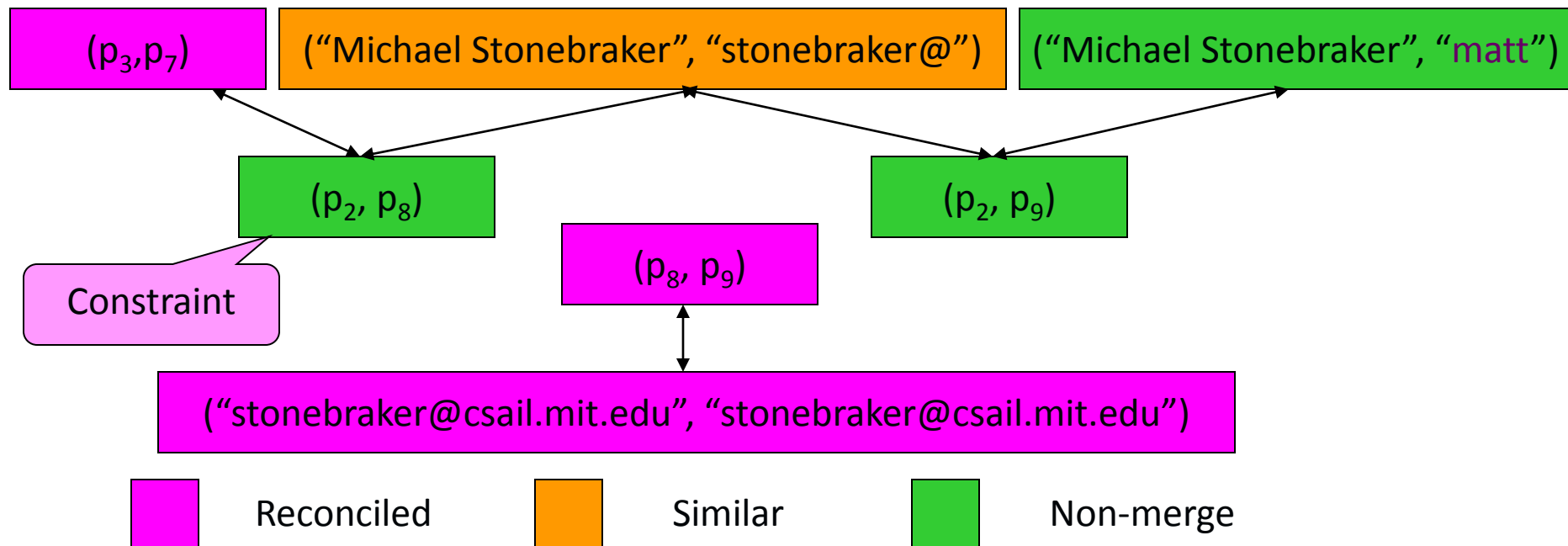
Enforce Constraints by Propagating Negative Information

- $p_2 = (\text{"Michael Stonebraker"}, \text{null}, \{p_1, p_3\})$
 $p_3 = (\text{"Eugene Wong"}, \text{null}, \{p_1, p_2\})$
 $p_7 = (\text{"Eugene Wong"}, \text{"eugene@berkeley.edu"}, \{p_8\})$
 $p_8 = (\text{null}, \text{"stonebraker@csail.mit.edu"}, \{p_7\})$
 $p_9 = (\text{"**matt**"}, \text{"stonebraker@csail.mit.edu"}, \text{null})$



Enforce Constraints by Propagating Negative Information

- $p_2 = (\text{"Michael Stonebraker"}, \text{null}, \{p_1, p_3\})$
 $p_3 = (\text{"Eugene Wong"}, \text{null}, \{p_1, p_2\})$
 $p_7 = (\text{"Eugene Wong"}, \text{"eugene@berkeley.edu"}, \{p_8\})$
 $p_8 = (\text{null}, \text{"stonebraker@csail.mit.edu"}, \{p_7\})$
 $p_9 = (\text{"matt"}, \text{"stonebraker@csail.mit.edu"}, \text{null})$



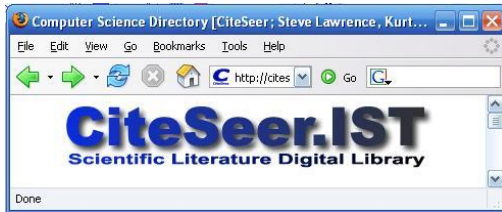
PART 4-d

AGGLOMERATIVE APPROACHES

Agglomerative Approach

- Collective Relational ER [Bhattacharya & Getoor, DMKD04, TKDD07]

A Motivating Example



P1: "*JOSTLE: Partitioning of Unstructured Meshes for Massively Parallel Machines*", C. Walshaw, M. Cross, M. G. Everett, S. Johnson

P2: "*Partitioning Mapping of Unstructured Meshes to Parallel Machine Topologies*", C. Walshaw, M. Cross, M. G. Everett, S. Johnson, K. McManus

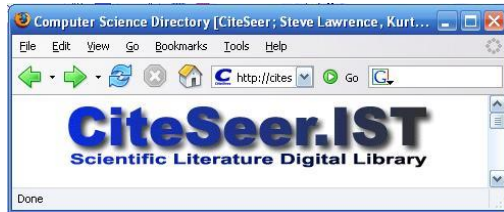
P3: "*Dynamic Mesh Partitioning: A Unied Optimisation and Load-Balancing Algorithm*", C. Walshaw, M. Cross, M. G. Everett

P4: "*Code Generation for Machines with Multiregister Operations*", Alfred V. Aho, Stephen C. Johnson, Jefferey D. Ullman

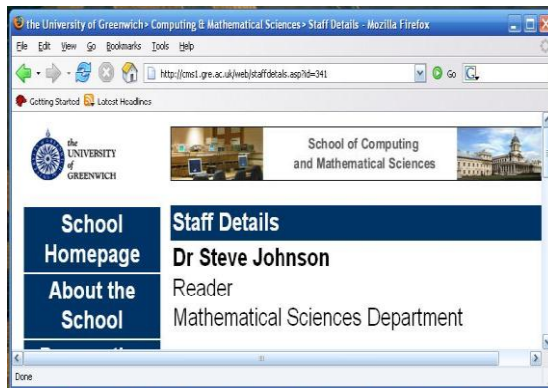
P5: "*Deterministic Parsing of Ambiguous Grammars*", A. Aho, S. Johnson, J. Ullman

P6: "*Compilers: Principles, Techniques, and Tools*", A. Aho, R. Sethi, J. Ullman

A Motivating Example

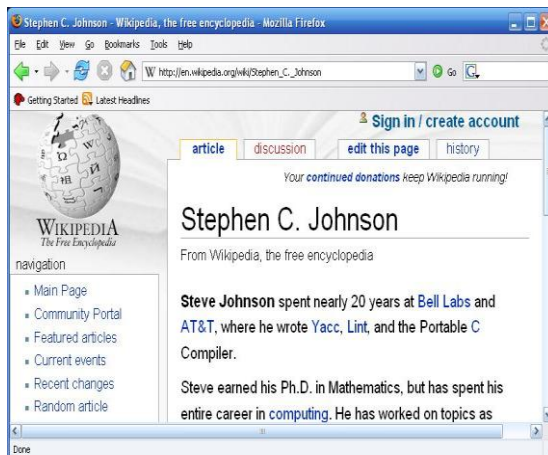


P1: "*JOSTLE: Partitioning of Unstructured Meshes for Massively Parallel Machines*", C. Walshaw, M. Cross, M. G. Everett, **S. Johnson**



P2: "*Partitioning Mapping of Unstructured Meshes to Parallel Machine Topologies*", C. Walshaw, M. Cross, M. G. Everett, **S. Johnson**, K. McManus

P3: "*Dynamic Mesh Partitioning: A Unied Optimisation and Load-Balancing Algorithm*", C. Walshaw, M. Cross, M. G. Everett



P4: "*Code Generation for Machines with Multiregister Operations*", Alfred V. Aho, **Stephen C. Johnson**, Jefferey D. Ullman

P5: "*Deterministic Parsing of Ambiguous Grammars*", A. Aho, **S. Johnson**, J. Ullman

P6: "*Compilers: Principles, Techniques, and Tools*", A. Aho, R. Sethi, J. Ullman

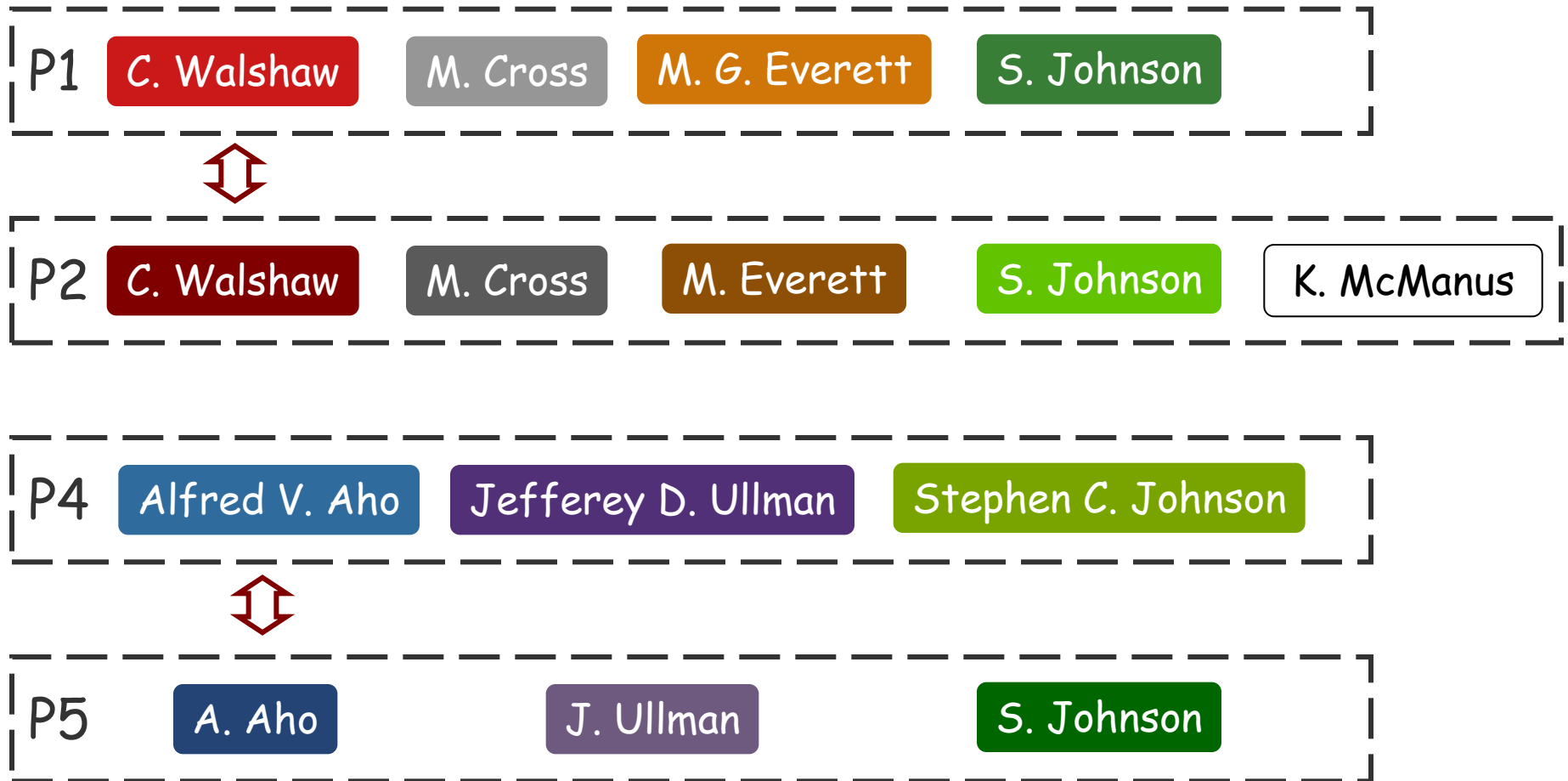
Motivation for Relational ER

- Relations between underlying entities
 - Researchers mostly co-author with colleagues
 - Friends exchange emails more frequently
- References for related entities co-occur
 - Co-author names in publications
 - Names/addresses in any email
- Use co-occurrence relations to resolve entities

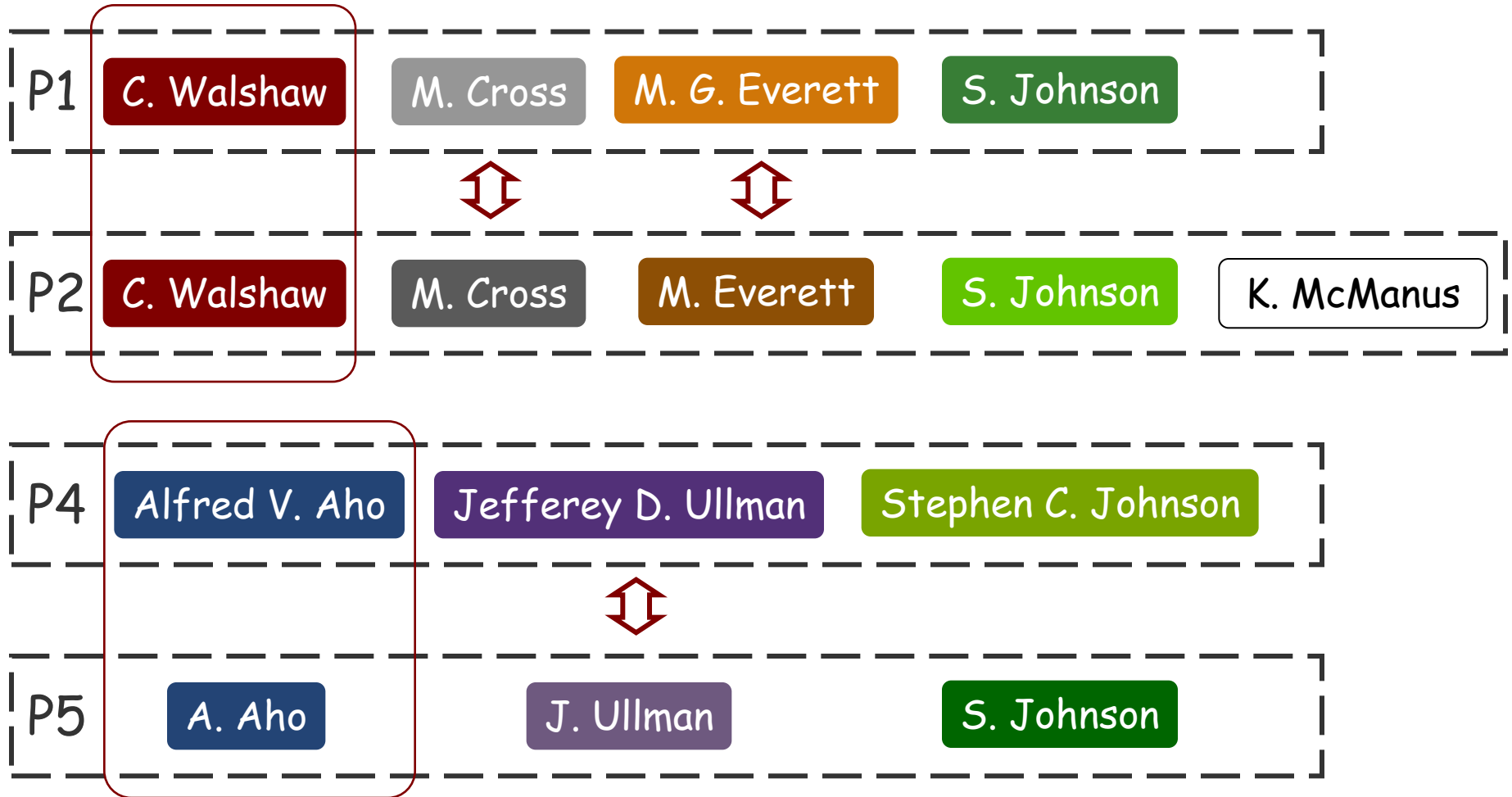
Entity Resolution With Relations

- Naïve Relational Entity Resolution
 - Also compare attributes of related references
 - Two ‘S Johnson’s have co-authors w/ similar names
 - May be inaccurate
- Collective Entity Resolution
 - Use **discovered entities** for related references
 - Entities cannot be identified independently
 - Harder problem to solve

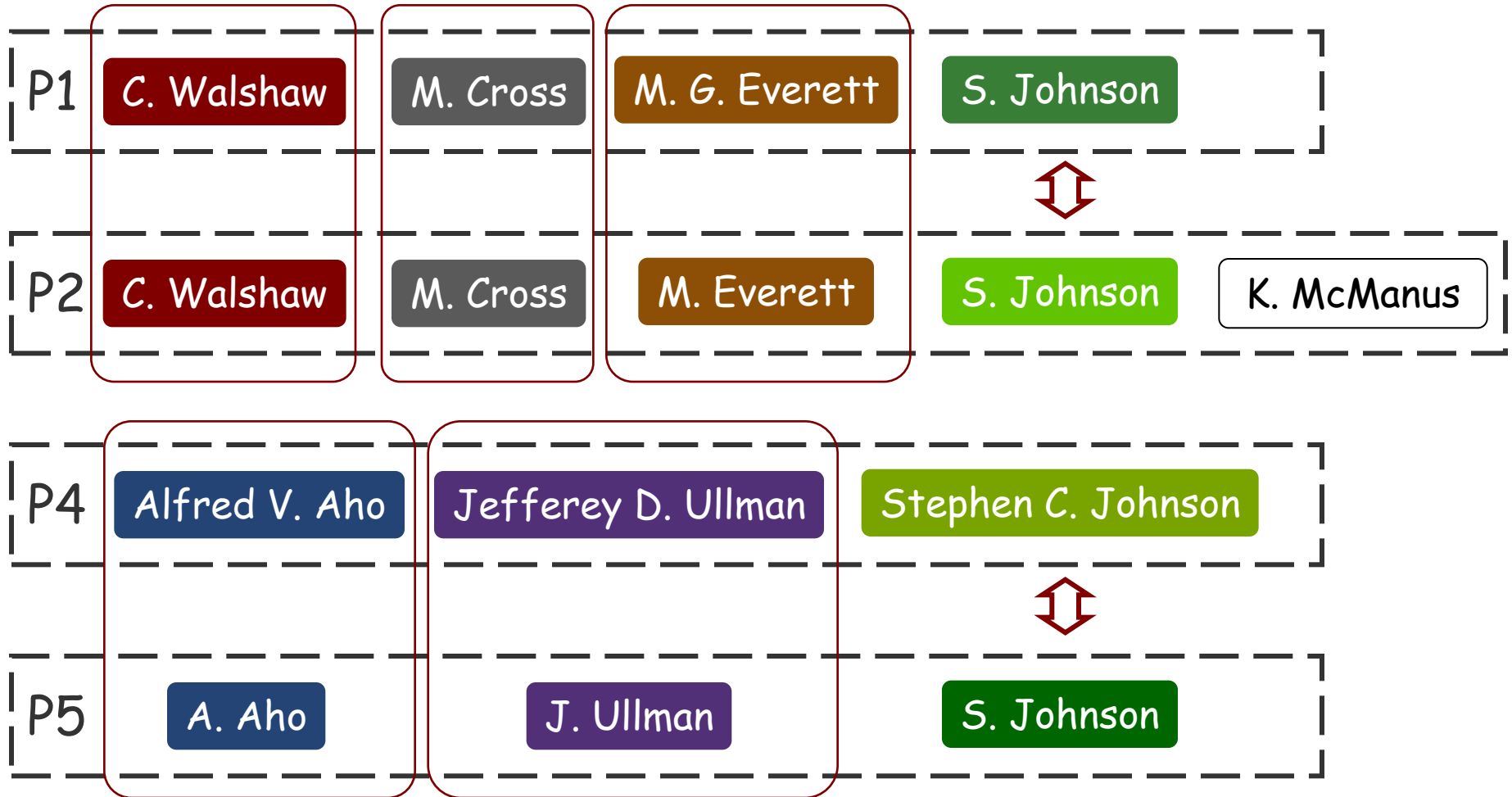
Relational Clustering for ER (RC-ER)



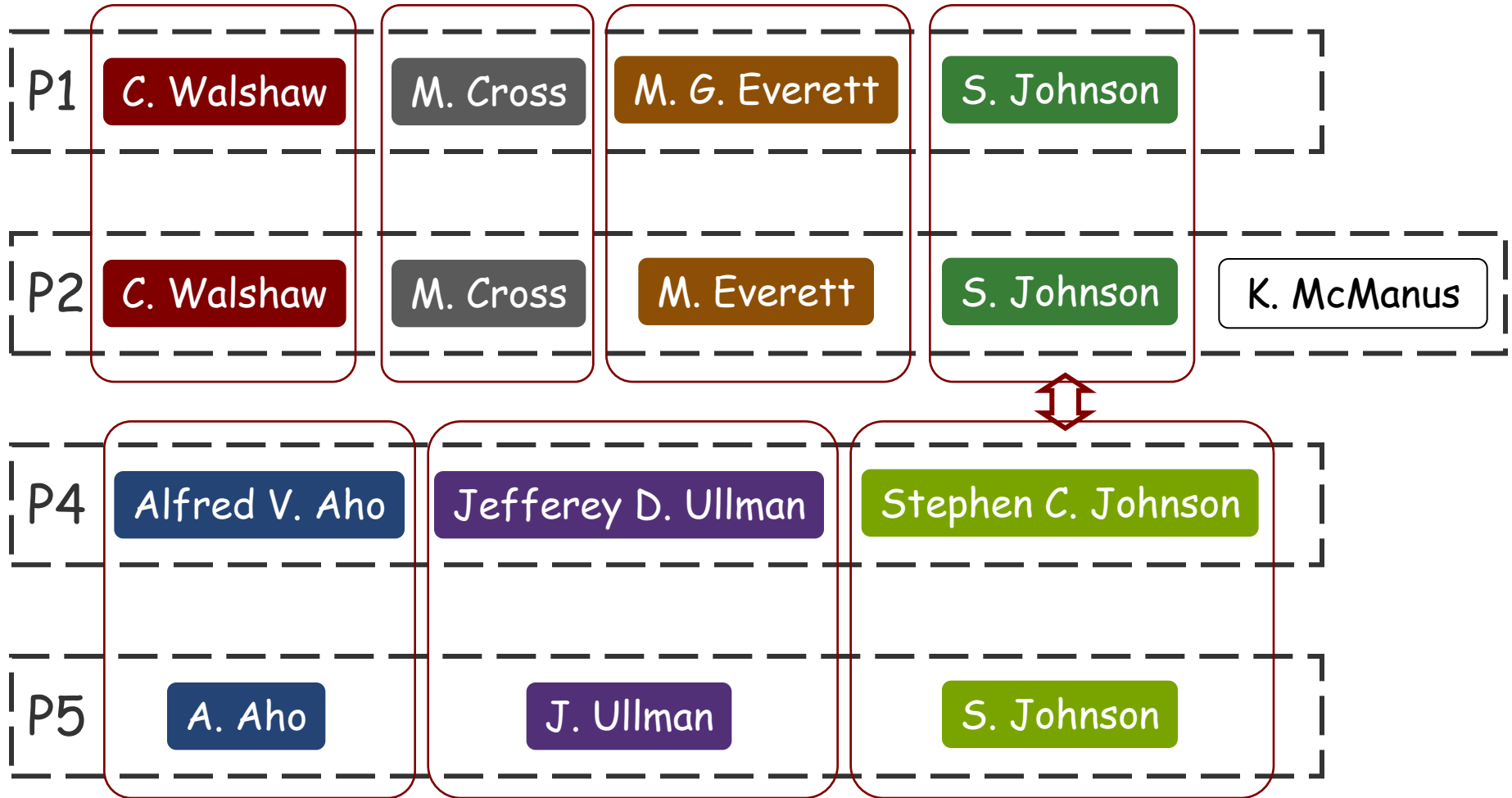
Relational Clustering for ER (RC-ER)



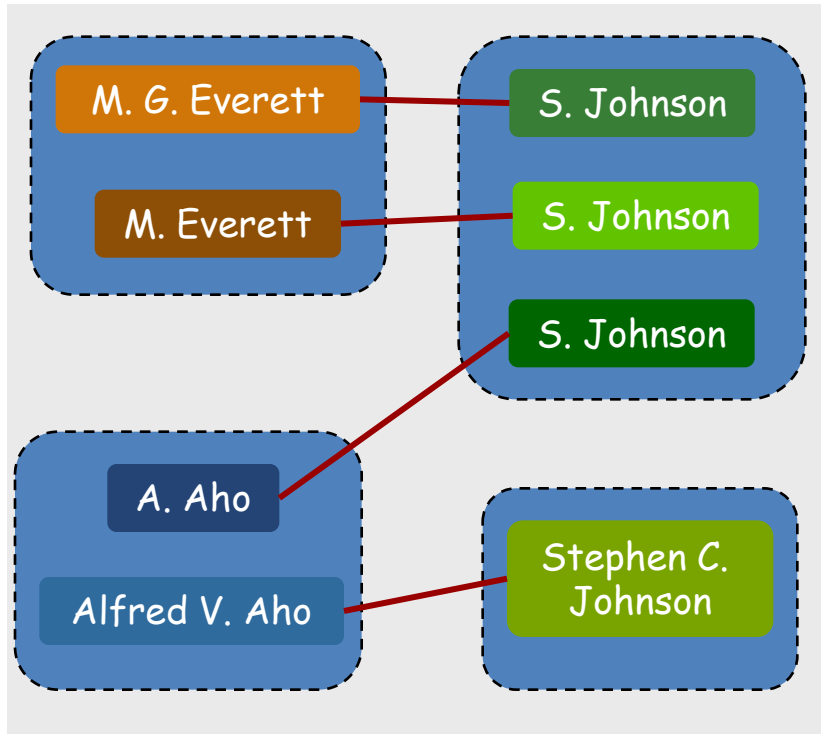
Relational Clustering for ER (RC-ER)



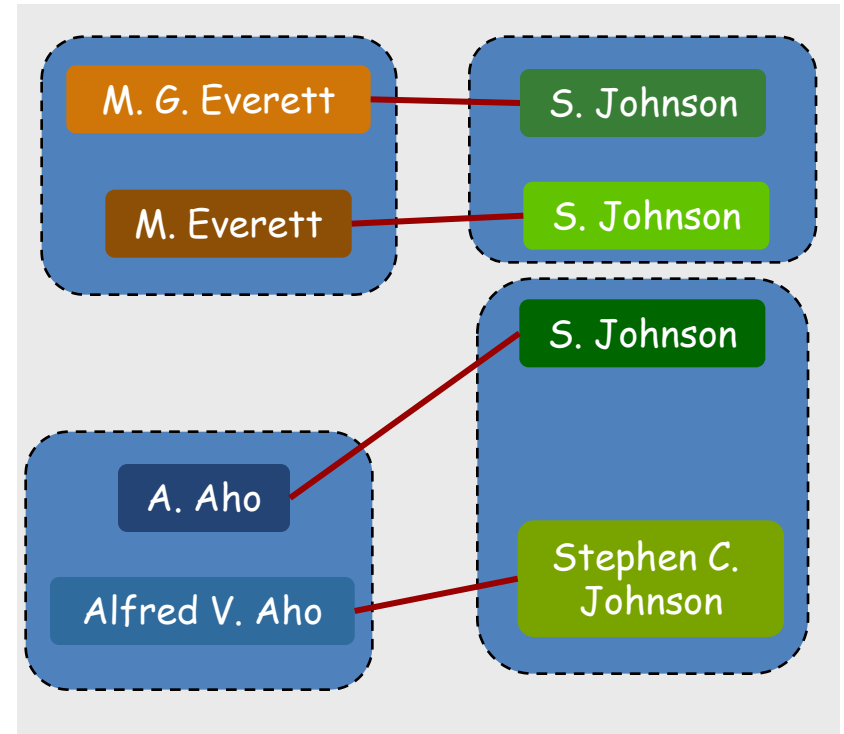
Relational Clustering for ER (RC-ER)



Cut-based Evaluation of Relational Clustering



Good separation of attributes
Many cluster-cluster relationships
➤ Aho-Johnson1, Aho-Johnson2,
Everett-Johnson1



Worse in terms of attributes
Fewer cluster-cluster relationships
➤ Aho-Johnson1, Everett-Johnson2

Objective Function

- Minimize:

$$\sum_i \sum_j w_A \text{sim}_A(c_i, c_j) + w_R \delta(c_i, c_j)$$

weight for
attributes

similarity of
attributes

weight for
relations

1 iff relational edge
exists between c_i and
 c_j

Objective Function

- Minimize:

$$\sum_i \sum_j w_A \text{sim}_A(c_i, c_j) + w_R \delta(c_i, c_j)$$

weight for
attributes

similarity of
attributes

weight for
relations

1 iff relational edge
exists between c_i and c_j

- Greedy clustering algorithm: merge cluster pair with max reduction in objective function

$$\Delta(c_i, c_j) = w_A \text{sim}_A(c_i, c_j) + w_R (|N(c_i) \cap N(c_j)|)$$

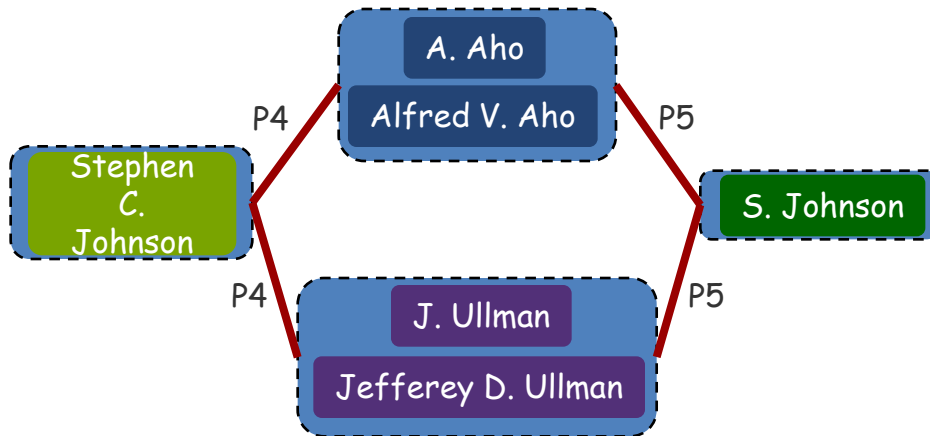
Similarity of
attributes

Common cluster
neighborhood

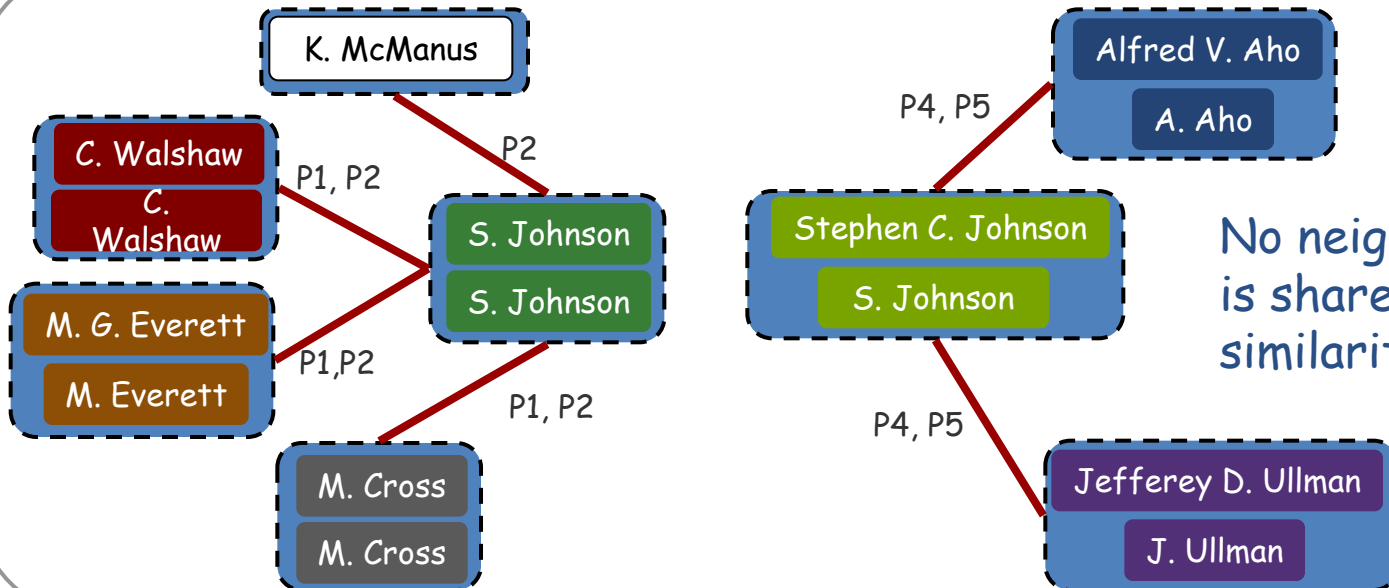
Measures for Attribute Similarity

- Use best available measure for each attribute
- Name Strings: *Soft TF-IDF, Levestein, Jaro*
- Textual Attributes: *TF-IDF*
- Aggregate to find similarity between clusters
 - Single link, Average link, Complete link
 - Cluster representative

Relational Similarity: Example



All neighborhood clusters are shared: high relational similarity



No neighborhood cluster is shared: no relational similarity

Comparing Cluster Neighborhoods

- Different measures of set similarity
 - Common Neighbors: Intersection size
 - Jaccard's Coefficient: Normalize by union size
 - Adar Coefficient: Weighted set similarity
 - Higher order similarity: Consider nbrs of nbrs
- Also consider neighborhood as multi-set

Relational Clustering Algorithm

1. Find similar references using 'blocking'
 2. Bootstrap clusters using attributes and relations
 3. Compute similarities for cluster pairs and insert into priority queue
 4. Repeat until priority queue is empty
 5. Find 'closest' cluster pair
 6. Stop if similarity below threshold
 7. Merge to create new cluster
 8. Update similarity for 'related' clusters
- $O(n k \log n)$ algorithm w/ efficient implementation

PART 4-e

GENERATIVE APPROACHES

Generative Approaches

- **Probabilistic Relational Models [Pasula et al., NIPS03]**
- Latent Dirichlet Allocation [Bhattacharya & Getoor, SDM06]

Identity Uncertainty [Pasula et al., NIPS03]

- Propose extensions to Probabilistic Relational Models (PRMs) [Koller, et al.; see ch. 5 of Intro to SRL, MIT Press]
- PRMs: compact representation for directed graphical model over relational schema
- Captures conditional probabilities of attributes given parents, where parents can be attributes in same relation, or attributes in another relation defined by a join or *slot chain*; Parameters tied, so probabilistic model is extremely compact
- Given a PRM and a database skeleton, defines a joint distribution over all of the unobserved attribute values using chain rule of Bayesian networks
- PRMs have been extended with relational uncertainty, class uncertainty, and *identity uncertainty*

RPM for Identity Uncertainty

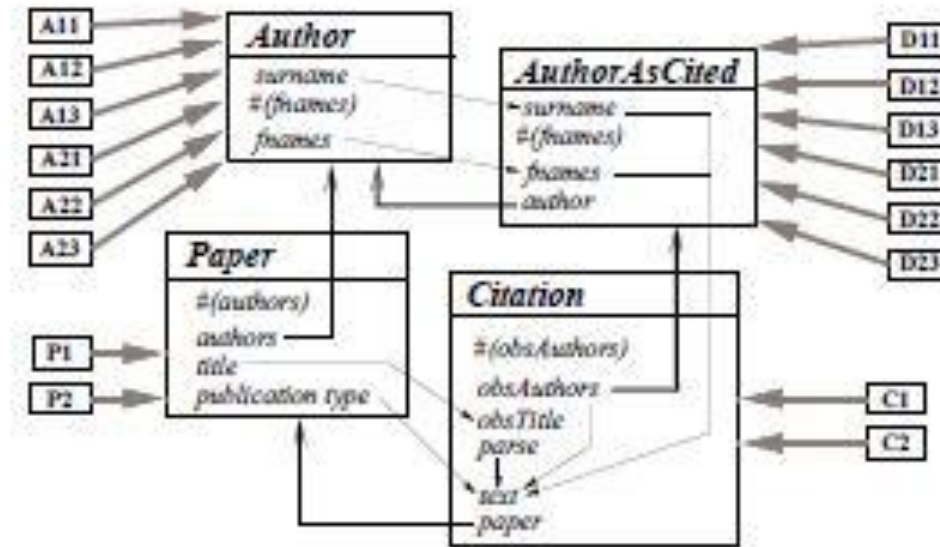


Figure 2: An RPM for our Citeseer example. The large rectangles represent classes: the dark arrows indicate the ranges of their complex attributes, and the light arrows lay out all the probabilistic dependencies of their basic attributes. The small rectangles represent instances, linked to their classes with thick grey arrows. We omit the instance statements which set many of the complex attributes.

Novelties of approach:

- Introduce the notion of an identity clustering, which captures the
- Handle variations in segmentation by modeling citation style
- Developed a MCMC approaches which is able to exploit canopies

Generative Approaches

- Probabilistic Relational Models [Pasula et al., NIPS03]
- **Latent Dirichlet Allocation [Bhattacharya & Getoor, SDM06]**

LDA-ER Probabilistic Generative Model

- Model how references co-occur in data
 1. Generation of references from entities
 2. Relationships between underlying entities
 - Groups of entities instead of pair-wise relations

Discovering Groups from Relations

Stephen P Johnson

Chris Walshaw

Kevin McManus

Mark Cross

Martin Everett

Parallel Processing Research Group



P1: C. Walshaw, M. Cross, M. G. Everett,
S. Johnson

P2: C. Walshaw, M. Cross, M. G. Everett,
S. Johnson, K. McManus

P3: C. Walshaw, M. Cross, M. G. Everett

Stephen C Johnson

Alfred V Aho

Ravi Sethi

Jeffrey D Ullman

Bell Labs Group

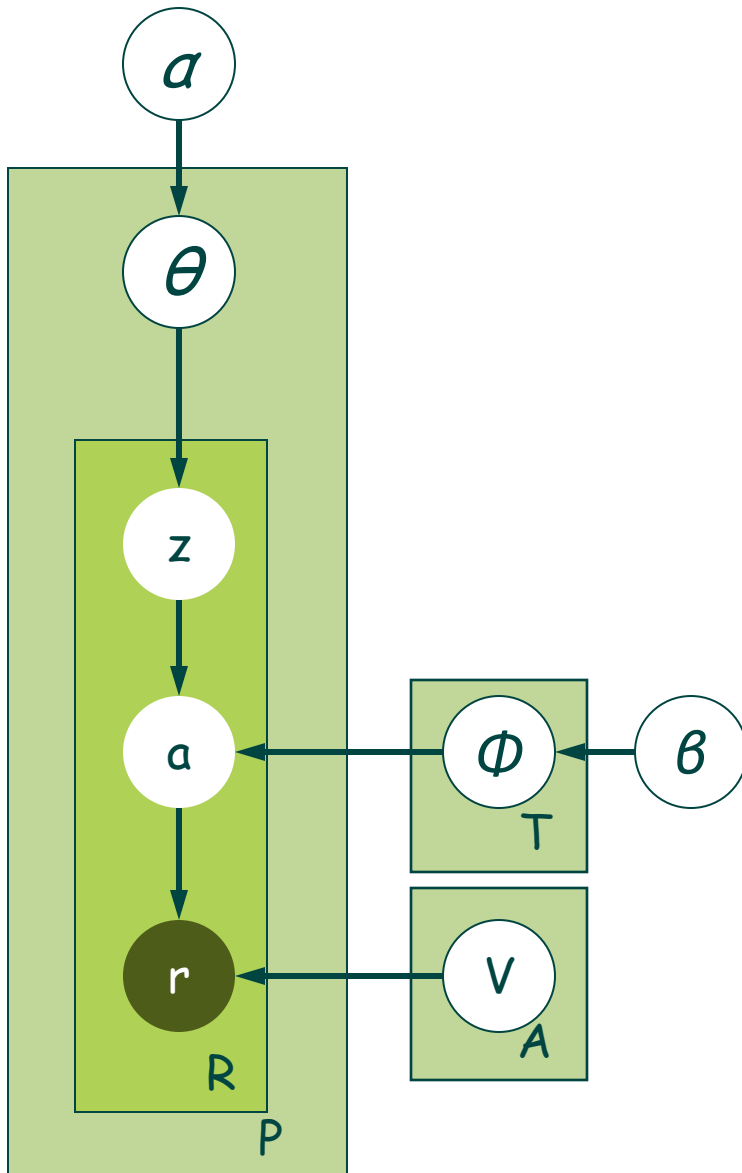


P4: Alfred V. Aho, Stephen C. Johnson,
Jefferey D. Ullman

P5: A. Aho, S. Johnson, J. Ullman

P6: A. Aho, R. Sethi, J. Ullman

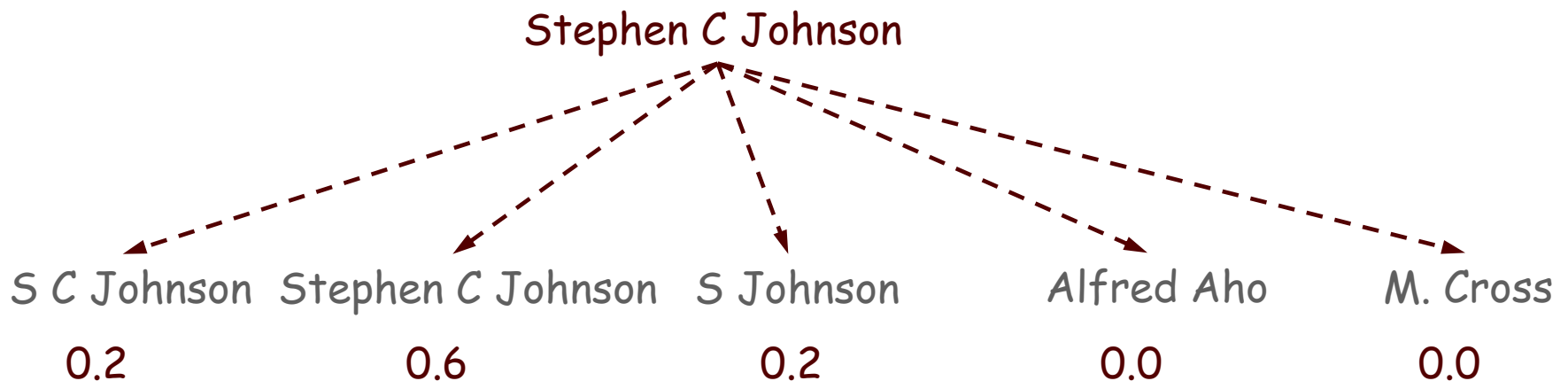
LDA-ER Model



- Entity label a and group label z for each reference r
- θ : 'mixture' of groups for each co-occurrence
- ϕ_z : multinomial for choosing entity a for each group z
- V_a : multinomial for choosing reference r from entity a
- Dirichlet priors with α and β

Generating References from Entities

- Entities are not directly observed
 1. Hidden attribute for each entity
 2. Similarity measure for pairs of attributes
- A distribution over attributes for each entity



Approx. Inference Using Gibbs Sampling

- Conditional distribution over labels for each ref.
- Sample next labels from conditional distribution
- Repeat over all references until convergence

$$P(z_i=t|\mathbf{z}_{-i},\mathbf{a},\mathbf{r}) \propto \frac{n_{d_i,t}^{DT} + \alpha/T}{n_{d_i,*}^{DT} + \alpha} \times \frac{n_{a_i,t}^{AT} + \beta/A}{n_{*,t}^{AT} + \beta}$$

$$P(a_i=a|\mathbf{z},\mathbf{a}_{-i},\mathbf{r}) \propto \frac{n_{a_i,t}^{AT} + \beta/A}{n_{*,t}^{AT} + \beta} \times \text{Sim}(r_i, v_a)$$

- Converges to most likely number of entities

Faster Inference: Split-Merge Sampling

- Naïve strategy reassigns references individually
- Alternative: allow entities to merge or split
- For entity a_i , find conditional probabilities for
 1. Merging with existing entity a_j
 2. Splitting back to last merged entities
 3. Remaining unchanged
- Sample next state for a_i from distribution
- $O(n g + e)$ time per iteration compared to $O(n g + n e)$

Comparison: RC-ER & LDA-ER

- **Attribute only:**
 - **A:** Pair-wise duplicate decisions w/ attributes only
 - **Names:** *Soft-TFIDF* with *Levenstein*, *Jaro*, *Jaro-Winkler*
 - **Other textual attributes:** *TF-IDF*
 - **A*:** Transitive closure over **A**
- **Naïve Relational:**
 - **A+N:** Add attribute similarity of co-occurring refs
 - **A+N*:** Transitive closure over **A+N**
- **Collective Relational:**
 - **RC-ER:** hierarchical agglomerative
 - **LDA-ER:** generative model
- Evaluate pair-wise decisions over references
- F1-measure (harmonic mean of precision and recall)

Evaluation Datasets

- CiteSeer
 - 1,504 citations to machine learning papers (Lawrence et al.)
 - 2,892 references to 1,165 author entities
- arXiv
 - 29,555 publications from High Energy Physics (KDD Cup'03)
 - 58,515 refs to 9,200 authors
- Elsevier BioBase
 - 156,156 Biology papers (IBM KDD Challenge '05)
 - 831,991 author refs
 - Keywords, topic classifications, language, country and affiliation of corresponding author, etc

ER Performance over Entire Dataset

	CiteSeer	arXiv	BioBase
A	0.980	0.976	0.568
A*	0.990	0.971	0.559
A+N	0.973	0.938	0.710
A+N*	0.984	0.934	0.753
RC-ER	0.995	0.985	0.818
LDA-ER	0.993	0.981	0.645

- RC-ER & LDA-ER outperform baselines in all datasets
- Collective resolution better than naïve relational resolution
- **BioBase:** Biggest improvement over baselines
- **arXiv:** 6,500 additional correct resolutions; 20% err. red.
- **CiteSeer:** Near perfect resolution; 22% error reduction

ER Performance over Entire Dataset

	CiteSeer	arXiv	BioBase
A	0.980	0.976	0.568
A*	0.990	0.971	0.559
A+N	0.973	0.938	0.710
A+N*	0.984	0.934	0.753
RC-ER	0.995	0.985	0.818
LDA-ER	0.993	0.981	0.645

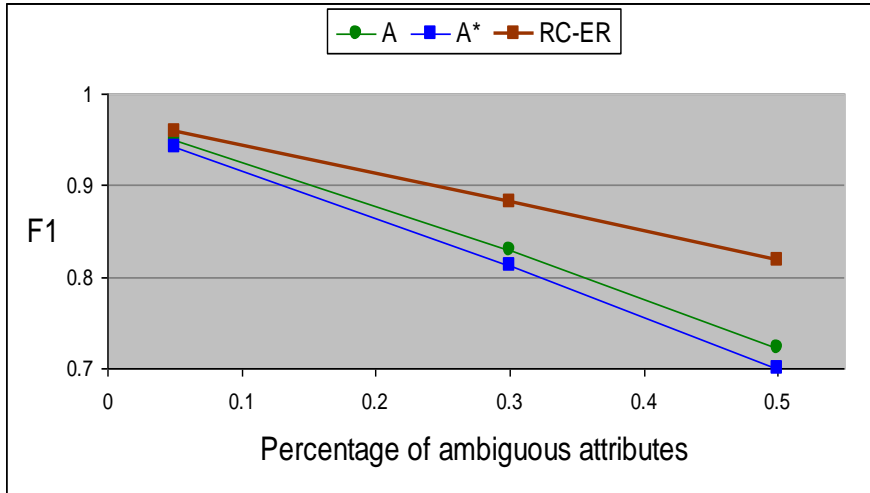
- RC-ER and baselines require threshold as parameter
 - Best achievable performance over all thresholds
- Best RC-ER performance better than LDA-ER
- LDA-ER does not require similarity threshold

Performance for Specific Names

Name	Best F1 for A/A*	F1 for LDA-ER
cho_h	0.80	1.00
davis_a	0.67	0.89
kim_s	0.93	0.99
kim_y	0.93	0.99
lee_h	0.88	0.99
lee_j	0.98	1.00
liu_j	0.95	0.97
sarkar_s	0.67	1.00
sato_h	0.82	0.97
sato_t	0.85	1.00
shin_h	0.69	1.00
veselov_a	0.78	1.00
yamamoto_k	0.29	1.00
yang_z	0.77	0.97
zhang_r	0.83	1.00
zhu_z	0.57	1.00

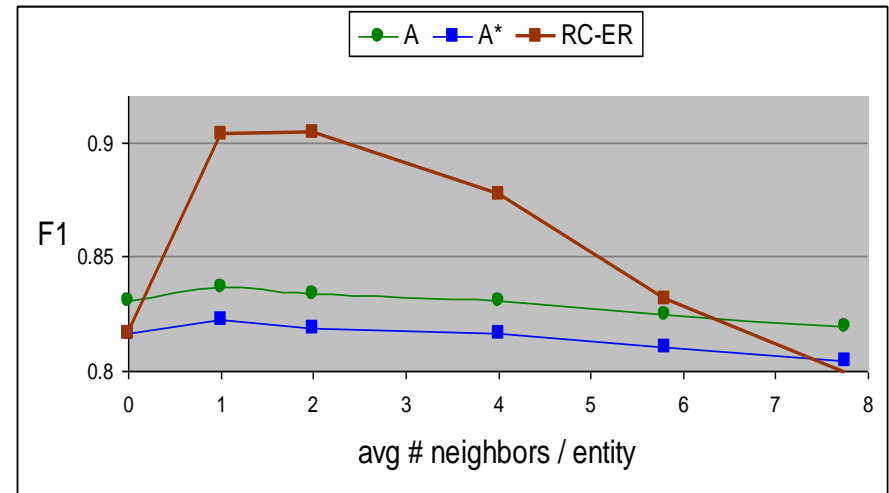
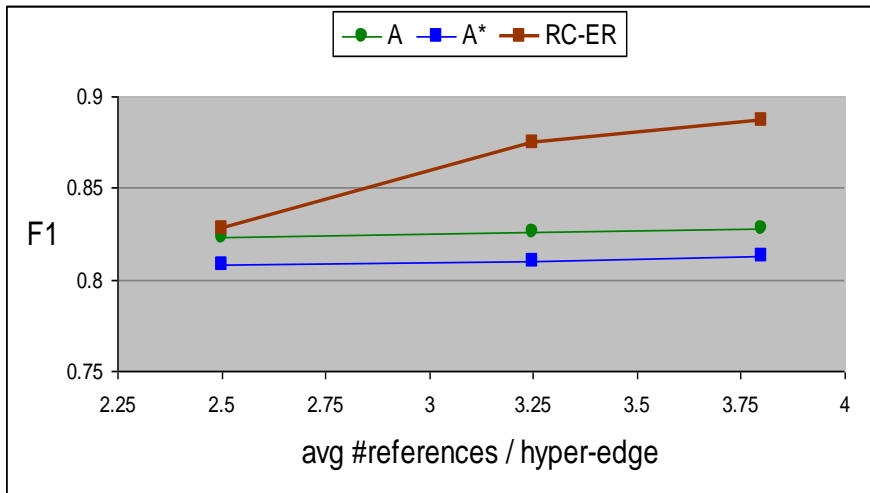
Significantly larger improvements for 'ambiguous names'

Trends in Synthetic Data



Bigger improvement with

- larger % of ambiguous refs
- more refs per co-occurrence
- more neighbors per entity



Data generator available: <http://www.cs.umd.edu/projects/linqs/projects/er/index.html>

PART 4-f

DECLARATIVE APPROACHES

Declarative Approaches

- **Markov Networks**
 - **Markov Logic Networks (MLNs)** [Singla & Domingos, ICDM06]
 - Probabilistic Similarity Logic [Broecheler & Getoor, UAI10]
- **Constraint-based**
 - Constraint-based Entity Matching [Shen, Li & Doan, AAAI05]
 - Dedupalog [Arasu, Re, Suciu, ICDE09]

Markov Logic

- A logical KB is a set of **hard constraints** on the set of possible worlds
- Let us make them **soft constraints**
- When a world violates a formula, it becomes less probable but not impossible
- Give each formula a **weight**
 - Higher weight \Rightarrow Stronger constraint

$$P(\text{world}) \propto \exp\left(\sum \text{weights of formulas it satisfies}\right)$$

Markov Logic

- A **Markov Logic Network (MLN)** is a set of pairs **(F, w)** where
 - **F** is a formula in first-order logic
 - **w** is a real number

$$P(X) = \frac{1}{Z} \exp \left(\sum_{i \in F} w_i n_i(x) \right)$$

The diagram illustrates the probability formula for a Markov Logic Network. The formula is $P(X) = \frac{1}{Z} \exp \left(\sum_{i \in F} w_i n_i(x) \right)$. Annotations include: a red box around Z with a red arrow pointing to a box labeled "Normalization Constant"; a blue box around $n_i(x)$ with a blue arrow pointing to a box labeled "# true groundings of *ith* clause"; and a purple box around the summation index $i \in F$ with a blue arrow pointing to a box labeled "Iterate over all first-order MLN formulas".

Problem Formulation

- **Given**

- A database of records representing entities in the real world e.g. citations
- A set of fields e.g. author, title, venue
- Each record represented as a set of typed predicates e.g. *HasAuthor(citation,author), HasVenue(citation,venue)*

- **Goal**

- To determine which of the records/fields refer to the same underlying entity

Example: Bibliography Database

Citation	Title	Author	Venue
C1	Entity Resolution	J. Cox	ICDM 06
C2	Entity Resolution and Logic	Cox J.	Sixth ICDM
C3	Learning Boolean Formulas	Jacob C.	ICDM 06
C4	Learning of Boolean Formulas	Jacob Coxe	Sixth ICDM

Problem Formulation

- Entities in the real world represented by one or more strings appearing in the DB e.g. *"J. Cox"*, *"Cox J."*
- String constant for each record e.g. *"C1"*, *"C2"*
- Goal: for each pair of string constants $\langle x_1, x_2 \rangle$ of the same type, is $x_1 = x_2$?

Handling Equality

- Introduce *Equals(x,y)* or $x = y$
- Introduce the axioms of equality
 - Reflexivity: $x = x$
 - Symmetry: $x = y \Rightarrow y = x$
 - Transitivity: $x = y \wedge y = z \Rightarrow z = x$
 - Predicate Equivalence:

$$x_1 = x_2 \wedge y_1 \wedge y_2 \Rightarrow (R(x_1, y_1) \Leftrightarrow R(x_2, y_2))$$

Handling Equality

- Introduce **reverse predicate equivalence**
- Same relation with the same entity gives evidence about two entities being same

$$R(x_1, y_1) \wedge R(x_2, y_2) \wedge x_1 = x_2 \Rightarrow y_1 = y_2$$

- Not true logically, but gives useful information
- Example

$$HasAuthor(C1, J. Cox) \wedge HasAuthor(C2, Cox J.) \wedge (J. Cox = Cox J.) \Rightarrow C1 = C2$$

Model for Entity Resolution

- Model is in the form of an MLN
- Query predicate is *Equality*
- Evidence predicates are relations which hold according to the DB
- Introduce axioms of equality
- First-order rules for field comparison, Fellegi-Sunter model, relational models

Field Comparison

- Each field is a string composed of tokens
- Introduce *HasWord(field, word)*
- Use reverse predicate equivalence

$$\textit{HasWord}(f_1, w_1) \wedge \textit{HasWord}(f_2, w_2) \wedge w_1 = w_2 \Rightarrow f_1 = f_2$$

- Example

$$\textit{HasWord}(\textit{J. Cox}, \textit{Cox}) \wedge \textit{HasWord}(\textit{Cox J.}, \textit{Cox}) \wedge (\textit{Cox} = \textit{Cox}) \Rightarrow (\textit{J. Cox} = \textit{Cox J.})$$

- Different weight for each word : learnable similarity measure of Bilenko & Mooney [2003]

Two-level Similarity

- Individual words as units: Can't deal with spelling mistakes
- Break each word into ngrams: Introduce *HasEngram(word, ngram)*
- Use reverse predicate equivalence for word comparisons
- Gives a two level similarity measure as proposed by Cohen et al. [2003]

Fellegi-Sunter Model

- Uses Naïve Bayes for match decisions with field comparisons used as predictors
- Simplest Version: Field similarities measured by presence/absence of words in common

$$\text{HasWord}(f_1, w_1) \wedge \text{HasWord}(f_2, w_2) \wedge \text{HasField}(r_1, f_1) \wedge \text{HasField}(r_2, f_2) \wedge w_1 = w_2 \Rightarrow r_1 = r_2$$

- Example

$$\text{HasWord}(J. Cox, Cox) \wedge \text{HasWord}(Cox J., Cox) \wedge \text{HasAuthor}(C1, J. Cox) \wedge \text{HasAuthor}(C2, Cox J.) \wedge (Cox = Cox) \Rightarrow (C1 = C2)$$

Relational Models

- Fellegi-Sunter + transitivity [McCallum & Wellner 2005]

$$(f_1 = f_2) \wedge (f_2 = f_3) \Rightarrow (f_3 = f_1)$$

- Fellegi-Sunter + reverse predicate equivalence for records/fields [Singla & Domingos 2005]

$$HasField(r_1, f_1) \wedge HasField(r_2, f_2) \wedge f_1 = f_2 \Rightarrow r_1 = r_2$$

$$HasAuthor(C1, J. Cox) \wedge HasAuthor(C2, Cox J.) \wedge (J. Cox = Cox J.) \Rightarrow C1 = C2$$

Relational Models

- Co-authorship relation for entity resolution [Bhattacharya & Getoor, DMKD04]

$$\textit{HasAuthor}(c, a_1) \wedge \textit{HasAuthor}(c, a_2) \Rightarrow \textit{Coauthor}(a_1, a_2)$$

$$\textit{Coauthor}(a_1, a_2) \wedge \textit{Coauthor}(a_3, a_4) \wedge a_1 = a_3 \Rightarrow a_2 = a_4$$

Declarative Approaches

- **Markov Networks**
 - Markov Logic Networks (MLNs) [Singla & Domingos, ICDM06]
 - **Probabilistic Similarity Logic [Broecheler & Getoor, UAI10]**
- **Constraint-based**
 - Constraint-based Entity Matching [Shen, Li & Doan, AAAI05]
 - Dedupalog [Arasu, Re, Suciu, ICDE09]

Probabilistic Soft Logic

- Declarative language for defining **constrained continuous Markov random field** (CCMRF) using first-order logic (FOL)
- Soft logic: truth values in $[0,1]$
- Logical operators relaxed using **Lukasiewicz t-norms**
- Mechanisms for incorporating **similarity functions**, and reasoning about **sets**
- MAP inference is a **convex optimization**
- Efficient sampling method for marginal inference

Predicates and Atoms

- Predicates
 - Describe relations
 - Combined with arguments to make atoms
- Atoms
 - Lifted: contains variables, e.g., `Friends(X, Y)`
 - Ground: no variables, e.g., `AuthorOf(author1, paper1)`
- Each ground atom can have a truth value in $[0,1]$
- PSL programs define distributions over the truth values of ground atoms

Weighted Rules

- A PSL program is a set of weighted, logical rules
- For example,

$$\text{authorName}(A1,N1) \wedge \text{authorName}(A2,N2) \wedge \text{similarString}(N1,N2) \\ \Rightarrow \text{sameAuthor}(A1,A2) : 1.0$$

- Variable substitution produces a set of weighted ground rules for a particular data set

Soft Logic Relaxation

- PSL uses the Lukasiewicz t-norm to relax hard logic operators to work on soft truth values

$$a \tilde{\wedge} b = \max\{0, a + b - 1\},$$

$$a \tilde{\vee} b = \min\{a + b, 1\},$$

$$\tilde{\neg} a = 1 - a,$$

- PSL converts rules to logical statements using above operators

$$X \tilde{\Rightarrow} Y \equiv \tilde{\neg} X \tilde{\vee} Y.$$

FOL to CCMRF

- PSL converts a weighted rule into potential functions by penalizing its **distance to satisfaction**, $d(g, x) = (1 - t_g(x))$,
- $t_g(x)$ is the truth value of ground rule g under interpretation x
- The distribution over truth values is

$$\Pr(x) = \frac{1}{Z} \exp \left(\sum_{r \in P} \sum_{g \in G(r)} w_r d(g, x) \right)$$

w_r : weight of rule r

$G(r)$: all groundings of rule r

P : PSL program

PSL Inference

- PSL finds the most likely state by solving

$$\operatorname{argmax}_x P(x) = \operatorname{argmax}_x \sum_{r \in P} \sum_{g \in G(r)} w_r d(g, x)$$

- The t-norms defining $t_g(x)$ form linear constraints on x , making inference a **linear program**
- PSL uses **lazy activation** to ground rules, thus reducing the number of active variables and increasing efficiency
- Other distance metrics (e.g., Euclidean) for distance to satisfaction produce other types of convex objectives (e.g., quadratic programs)

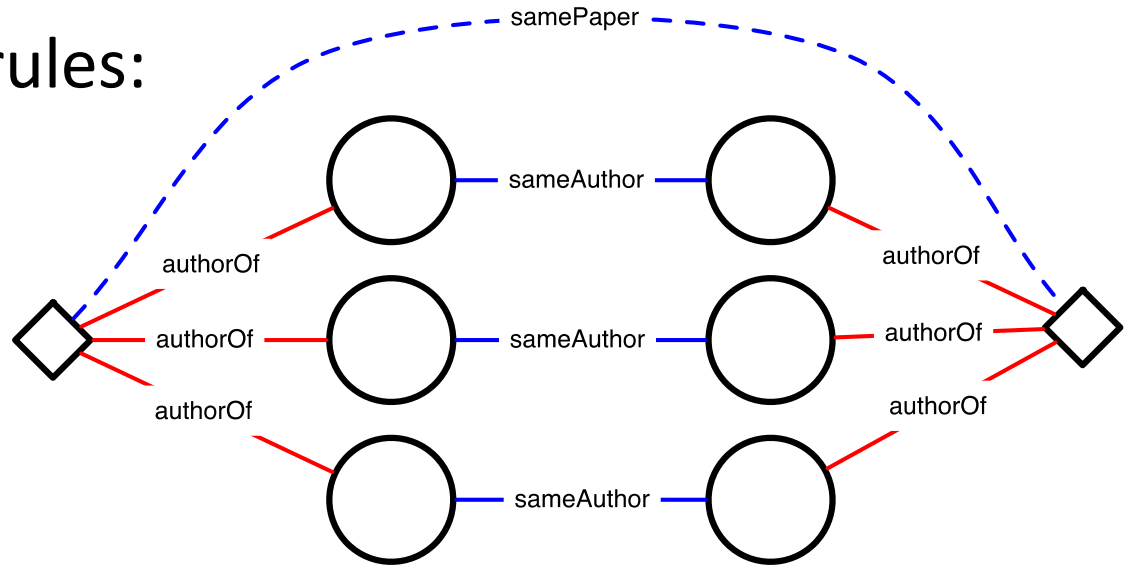
CiteSeer Example

- Citation listings collected from CiteSeer:
 - Pearl J. Probabilistic reasoning in intelligent systems.
Pearl, Judea. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.
- Duplicate authors and papers
- Base model: Levenstein string similarity
 - $\text{authorName}(A1, N1) \wedge \text{authorName}(A2, N2) \wedge \text{similarString}(N1, N2)$
 $\Rightarrow \text{sameAuthor}(A1, A2)$
 - $\text{paperTitle}(P1, T1) \wedge \text{paperTitle}(P2, T2) \wedge \text{similarString}(T1, T2)$
 $\Rightarrow \text{samePaper}(P1, P2)$
- Only activate rule on pairs with similarity > 0.5

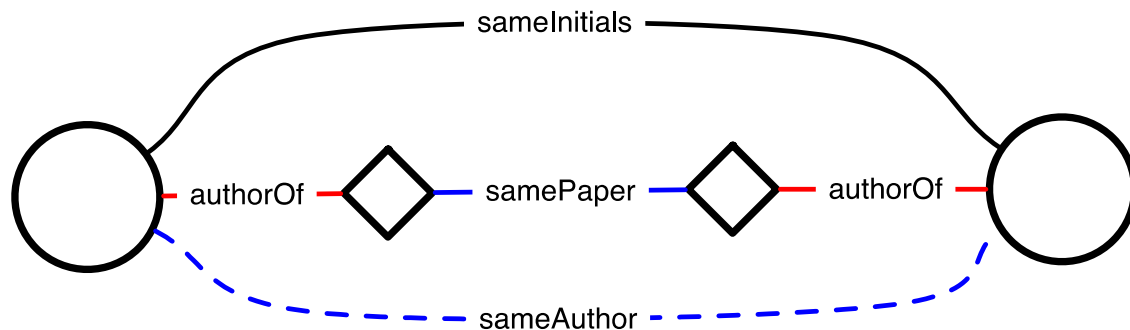
Relational Model A

- Multi-Relational rules:

- $\text{sameAuthorSet}(P1, P2)$
 $\Rightarrow \text{samePaper}(P1, P2)$



- $\text{samePaper}(P1, P2) \wedge \text{authorOf}(A1, P1) \wedge \text{authorOf}(A2, P2) \wedge \text{authorName}(A1, N1) \wedge \text{authorName}(A2, N2) \wedge \text{sameInitials}(N1, N2) \Rightarrow \text{sameAuthor}(A1, A2)$



Relational Model B

- Add coauthorship to multi-relational logic
- Coauthorship rules:
 - // define coauthorship
 $\text{authorOf}(A1,P) \wedge \text{authorOf}(A2,P) \Rightarrow \text{coauthor}(A1,A2)$
 - // coauthorship transfers through sameAuthor
 $\text{coauthor}(A1,A2) \wedge \text{sameAuthor}(A2,A3) \wedge \text{authorName}(A2,N2) \wedge \text{authorName}(A3,N3) \wedge \text{sameInitials}(N2,N3) \Rightarrow \text{coAuthor}(A1,A3)$
 - // a pair that shares a coauthor is likely to be coreferent
 $\text{coauthor}(A1,A2) \wedge \text{coauthor}(A2,A3) \wedge \text{authorName}(A2,N2) \wedge \text{authorName}(A3,N3) \wedge \text{sameInitials}(N2,N3) \Rightarrow \text{sameAuthor}(A1,A3)$
- sameInitials reduces number of active ground rules

Declarative Approaches

- Markov Networks
 - Markov Logic Networks (MLNs) [Singla & Domingos, ICDM06]
 - Probabilistic Similarity Logic [Broecheler & Getoor, UAI10]
- **Constraint-based**
 - Constraint-based Entity Matching [Shen, Li & Doan, AAAI05]
 - Dedupalog [Arasu, Re, Suciu, ICDE09]

Relational Constraints

- If two papers match, then their venues match
 - This constraint can be applied to all instances of venue strings
 - All occurrences of 'SIGMOD' can be matched to 'International Conference on Management of Data'
- If two papers match, then their authors match
 - This constraint can only be applied locally
 - Don't want to match all occurrences of 'J. Smith' with 'Jeff Smith', only in the context of the current paper

Declarative Approaches

- Markov Networks
 - Markov Logic Networks (MLNs) [Singla & Domingos, ICDM06]
 - Probabilistic Similarity Logic [Broecheler & Getoor, UAI10]
- **Constraint-based**
 - **Constraint-based Entity Matching [Shen, Li & Doan, AAAI05]**
 - Dedupalog [Arasu, Re, Suciu, ICDE09]

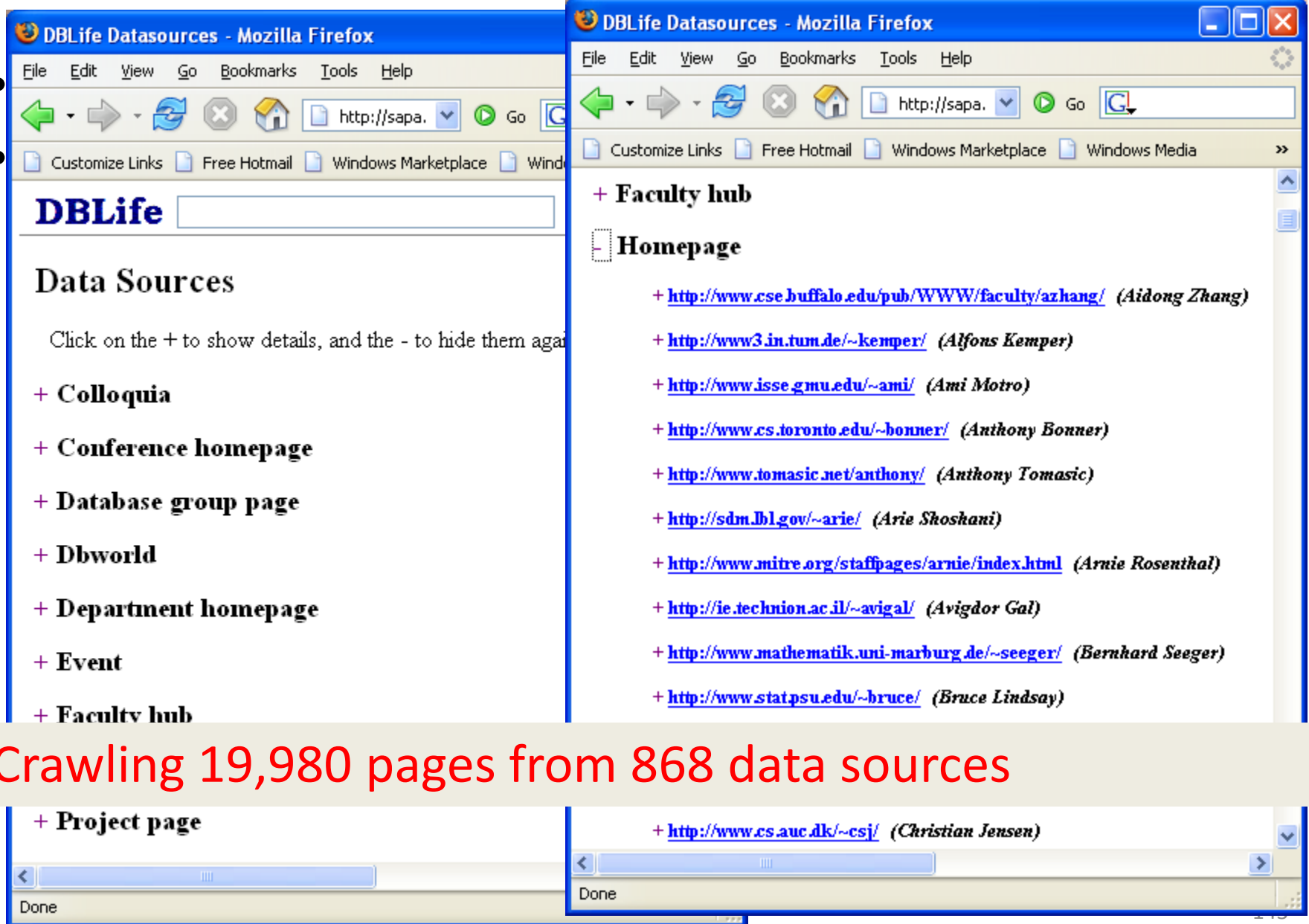
Semantic Integrity Constraints

Type	Example
Aggregate	C1 = No researcher has published more than five AAAI papers in a year
Subsumption	C2 = If a citation X from DBLP matches a citation Y in a homepage, then each author mentioned in Y matches some author mentioned in X
Neighborhood	C3 = If authors X and Y share similar names and some co-authors, they are likely to match
Incompatible	C4 = No researcher exists who has published in both HCI and numerical analysis
Layout	C5 = If two mentions in the same document share similar names, they are likely to match
Key/Uniqueness	C6 = Mentions in the PC listing of a conference is to different researchers
Ordering	C7 = If two citations match, then their authors will be matched in order
Individual	C8 = The researcher with the name “Mayssam Saria” has fewer than five mentions in DBLP (new graduate student)

CME

- Two layer model:
 - Layer 1: Generative model for data sets that satisfy constraints; builds on (Li, Morie, & Roth, AI Mag 2004).
 - Layer 2: EM algorithm and the relaxation labeling algorithm to perform matching. Matching process is carried out in multiple iterations. In each iteration, use EM to estimate parameters of the generative model and a matching assignment, then employs relaxation labeling to exploit the constraints
- First layer clusters mentions into groups (such that all matching mentions belong to the same group) and exploits constraints at the *group level*. *Once this is done*, the second layer exploits additional constraints at the level of *individual matching mention pair*.

Prototype System: DBLife



Crawling 19,980 pages from 868 data sources

Declarative Approaches

- Markov Networks
 - Markov Logic Networks (MLNs) [Singla & Domingos, ICDM06]
 - Probabilistic Similarity Logic [Broecheler & Getoor, UAI10]
- **Constraint-based**
 - Constraint-based Entity Matching [Shen, Li & Doan, AAAI05]
 - **Dedupalog [Arasu, Re, Suciu, ICDE09]**

Clustering with Dedupalog

PaperRef(id, title, conference, publisher, year)
Wrote(id, authorName, Position)

Data to be
deduplicated

TitleSimilar(title1,title2)
AuthorSimilar(author1,author2)

(Thresholded) Fuzzy-
Join Output

Step (0) Create Fuzzy Matches; this is input to Dedupalog.

Step (1) Declare the entities

“Cluster Papers, Publishers, & Authors”

Paper!(id) :- PaperRef(id,-,-,-)
Publisher!(p) :- PaperRef(-,-,-,p,-)
Author!(a) :- Wrote(-,a,-)

Dedupalog is *flexible*:
Unique Names Assumption (UNA)

Publishers (UNA) and Papers (NOT UNA)

Step (2) Declare Clusters

Input in the DB

PaperRef(id, title, conference, publisher, year)
Wrote(id, authorName, Position)

*"Cluster papers,
publishers, and authors"*

TitleSimilar(title1,title2)
AuthorSimilar(author1,author2)

Paper!(id) :- PaperRef(id,-,-,-)
Publisher!(p) :- PaperRef(-,-,-,p,-)
Author!(a) :- Wrote(-,a,-)

Clusters are *declared* using * (like IDBs or Views): These are output

Author*(a₁,a₂) <-> AuthorSimilar(a₁,a₂)

"Cluster authors with similar names"

Author1	Author2
AA	Arvind Arasu
Arvind A	Arvind Arasu

*IDBs are **equivalence relations**:
Symmetric, Reflexive , & Transitively-
Closed Relations: i.e., *Clusters*

A **Dedupalog program** is a
set of datalog-like rules

Simple Constraints

“Papers with similar titles should likely be clustered together”

Paper*(id₁,id₂) \leftrightarrow PaperRef(id₁,t₁,-), PaperRef(id₂,t₂,-), TitleSimilar(t₁,t₂)

Author*(a₁,a₂) \leftrightarrow AuthorSimilar(a₁,a₂)

(\leftrightarrow) Soft-constraints:
Pay a cost if violated.

Paper*(id₁,id₂) \leq PaperEq(id₁,id₂)

\neg **Paper***(id₁,id₂) \leq PaperNeq(id₁,id₂)

(\leq) Hard-constraints: *Any clustering must satisfy these*

*“Papers in PaperEQ **must** be clustered together, those in PaperNEQ **must not** be clustered together”*

Hard constraints
are challenging!

1. PaperEQ, PaperNEQ are relations (EDBS)
2. \neg denotes Negation here.

Advanced Constraints

“Clustering two papers, then must cluster their first authors”

Author*(a₁,a₂) ≤ **Paper***(id₁,id₂), Wrote(id₁,a₁,1), Wrote(id₂,a₂,1)

“Clustering two papers makes it likely we should cluster their publisher”

Publisher*(x,y) <- Publishes(x,p₁), Publishes(x,p₂), **Paper***(p₁,p₂)

[Bhattacharya, Getoor AAAI07]

“if two authors do not share coauthors, then do not cluster them”

¬ **Author*** (x, y) <- ¬ (Wrote(x, p₁, -), Wrote(y, p₂, -), Wrote(z, p₁, -),
Wrote(z, p₂, -), **Author***(x, y))

Bottomline: Dedupalog is powerful. How do we process it?


Dedupalog via CC

Semantics: Translate a Dedupalog Program to a set of graphs

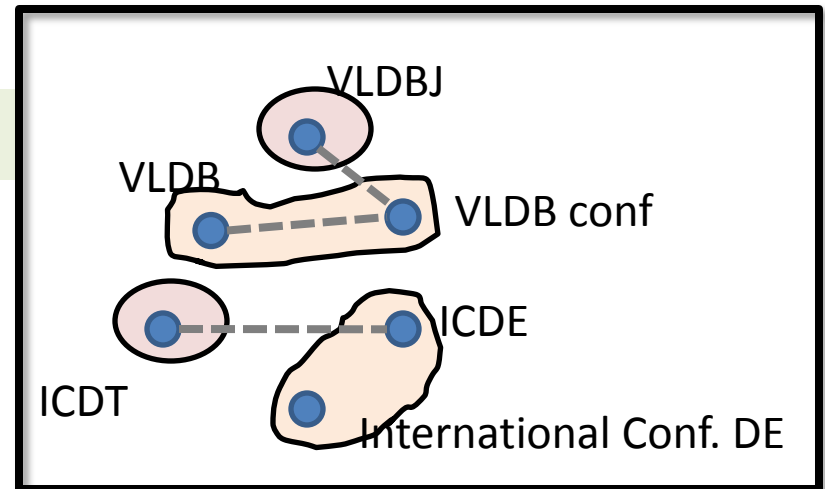
Nodes are references (in the ! Relation)

Entity References: Conference!(c)

Conference^{*}(c₁,c₂) <-> ConfSim(c₁,c₂)

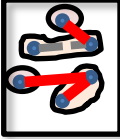
 Positive edges

 Negative edges are implicit



For a single graph w.o. hard constraints
we can reuse prior art for $O(1)$ apx.

Correlation Clustering




Conference $^*(c_1, c_2) \leftarrow \text{ConfSim}(c_1, c_2)$



Conference $^*(c_1, c_2) \leq \text{ConfEQ}(c_1, c_2)$

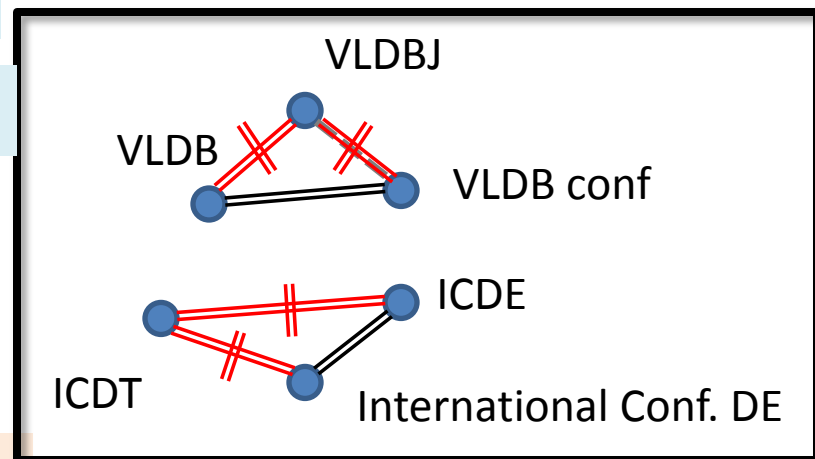
Conference $^*(c_1, c_2) \leq \text{ConfNEQ}(c_1, c_2)$

Soft

 Positive
[-] Negative




Hard

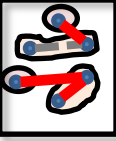
 Equal
 Not Equal



Simple, Combinatorial
algorithm is easy to scale!

Thm: This is a 3-apx!

1. Pick a random order of edges
2. **While** there is a soft edge **do**
 1. Pick first soft edge in order
 2. If  turn into 
 3. Else is [-] turn into 
 4. Deduce labels
3. **Return** Transitively closed subsets



Voting

Extend algorithm to **whole** language via *voting technique*.
Support many entities, recursive programs, etc.

Many dedupalog programs
have an $O(1)$ -apx

Thm: A recursive-hard
constraints no $O(1)$ apx!

Thm: All “soft” programs $O(1)$

Expert: multiway-cut hard

System properties:

- (1) Streaming algorithm
- (2) linear in # of matches (not n^2)
- (3) User interaction

Features: Support for weights, reference tables
(partially), and corresponding hardness results.

Summary: Collective Approaches

- Decisions for cluster-membership depends on other clusters
 - Graph-based approaches
 - Agglomerative approaches
 - Generative Graphical Models
 - Declarative Approaches
 - Markov Networks
 - Constraint-based Approaches

Summary

- Methods having increasing expressive power, allowing domain experts to customize to their domains
- However remaining challenges for scaling to big data
- Important Differences:
 - Single relation vs. multi-relational
 - Differences in inference algorithms
 - Inference algorithms leverage special structure of ER
 - Transitive closure
 - Sparsity and small clusters
 - Eg. Special MCMC algorithms, implementing inference as CC
 - Declarative vs. procedural