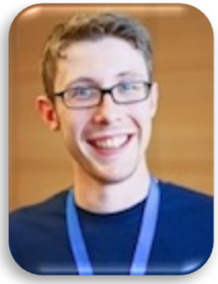
Hasso Plattner Institut
IT Systems Engineering | Universität Potsdam

Graph Exploration: Taking the User into the Loop

Davide Mottin, Anja Jentzsch, Emmanuel Müller
Hasso Plattner Institute, Potsdam, Germany

2016/10/24
CIKM2016, Indianapolis, US

Who we are



Davide Mottin

- graph mining, novel query paradigms, interactive methods
- https://hpi.de/en/mueller/team/davide-mottin.html

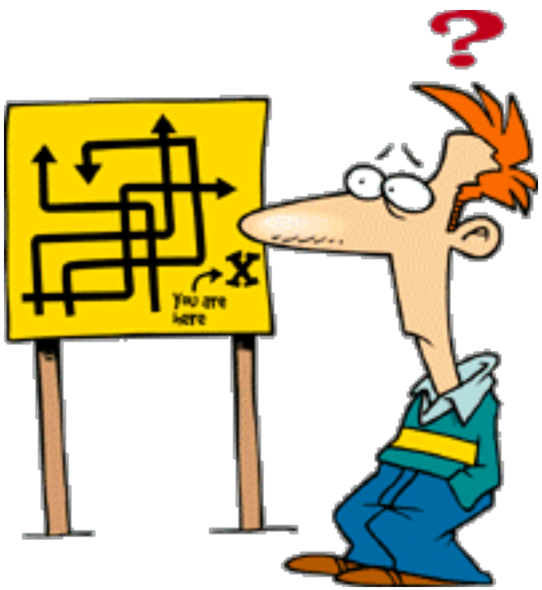
Anja Jentzsch

- Linked Open Data, graph exploration, data profiling
- http://hpi.de/naumann/people/anja-jentzsch.html

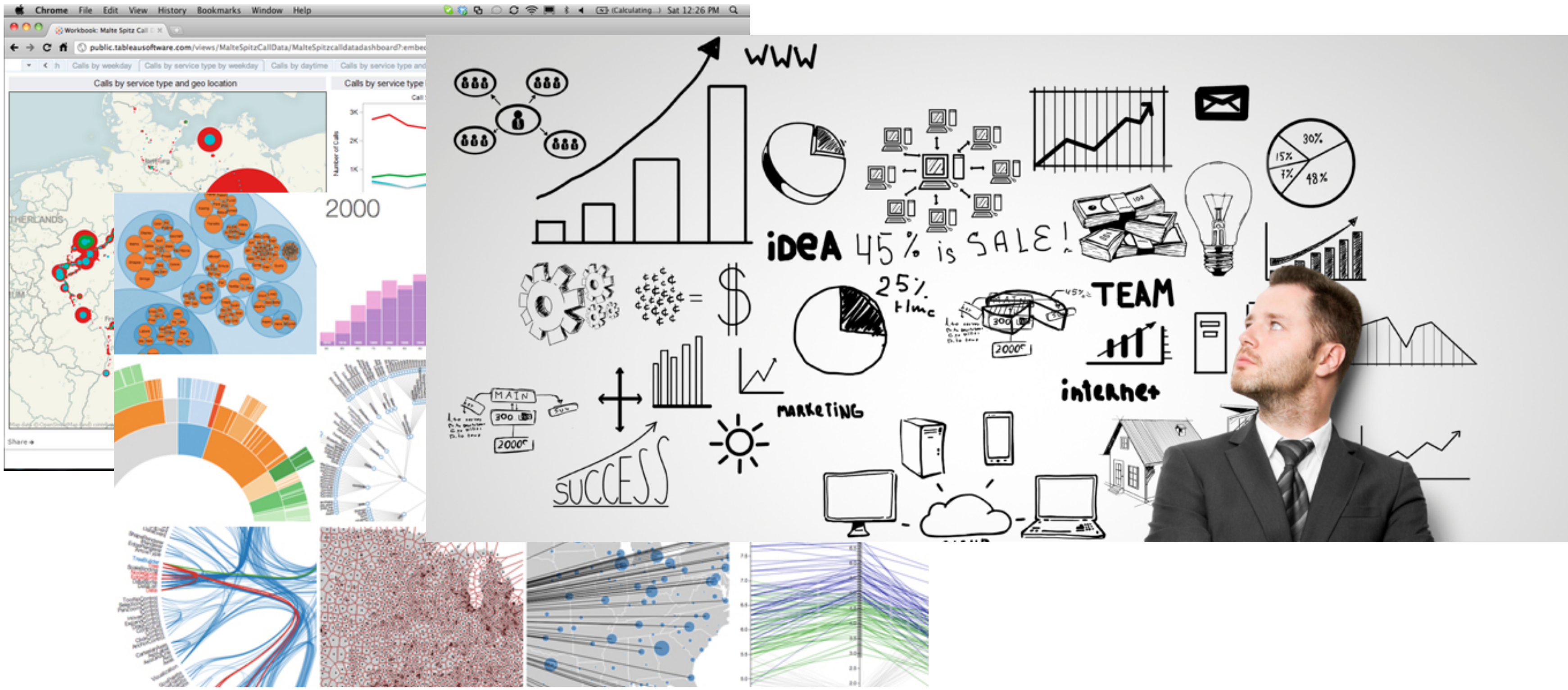
Emmanuel Müller

- graph mining, stream mining, clustering and outlier mining on graphs, streams, and traditional databases
- http://hpi.de/en/mueller/prof-dr-emmanuel-mueller.html

Big data and novice users



Data exploration



Efficiently extracting knowledge from data
even if we do not know exactly what we are looking for

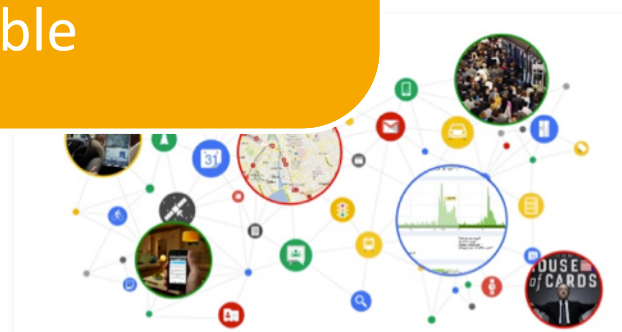
Idreos et al., Overview of Data Exploration Methods, SIGMOD 2015

The importance of graphs

Social Ne

s

Recommendation Graphs



Knowledge Graphs

Complex

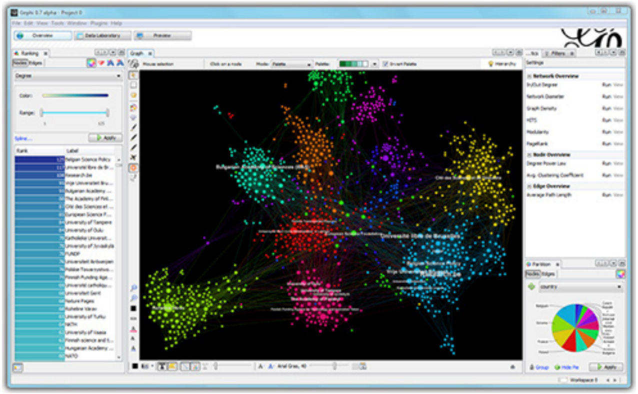
Ubiquitous

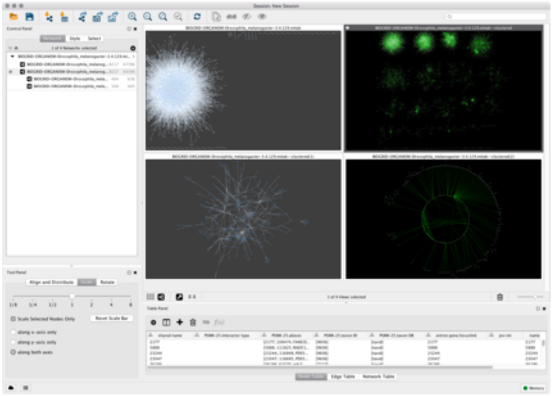
Large

Valuable

Lost in the graph?

Current: Visualization tools





Several visualization tools:
- General: Gephi, GraphViz, …
- Biological: Cytoscape, Network Workbench
- Social: EgoNet, NodeXL, …
- Relational: Tulip

but

- **No** Scalability to large networks!
- **No** for novice users
- Limited expressivity

Current: Query languages

```
SELECT ?name ?email
WHERE
  {
    ?person  a  foaf:Person .
    ?person  foaf:name ?name .
    ?person  foaf:mbox ?email .
  }
```

SPARQL

```
g.V().hasLabel('movie').as('a','b').
  where(inE('rated').count().is(gt(10))).
  select('a','b').
   by('name').
   by(inE('rated').values('stars').mean()).
  order().
   by(select('b'),decr). limit(10
```

GREMLIN

```
MATCH (node1:Label1)-->(node2:Label2)
  WHERE node1.propertyA = {value}
RETURN node2.propertyA, node2.propertyB
```

CYPHER

Query languages **ARE**:

- Expressive
- Powerful
- Scalable
- Compact

but

- **Not** user friendly
- **No** guided search
- **Not** interactive
- **Not** scalable

This tutorial is about …

- Algorithms for helping the user finding the wanted information
- Approximate search on graphs to assist the user in finding the information
- Interactive methods on graphs based on user feedback
- Automatically discovery of portions of graphs using examples

NOT about

- Visualization methods for graphs
- Query languages and semantics
- Efficient indexing methods
- Pure machine learning on graphs

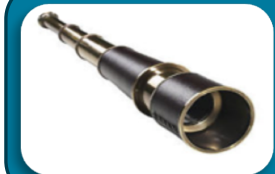
Our graph exploration taxonomy

Exploratory Graph Analysis

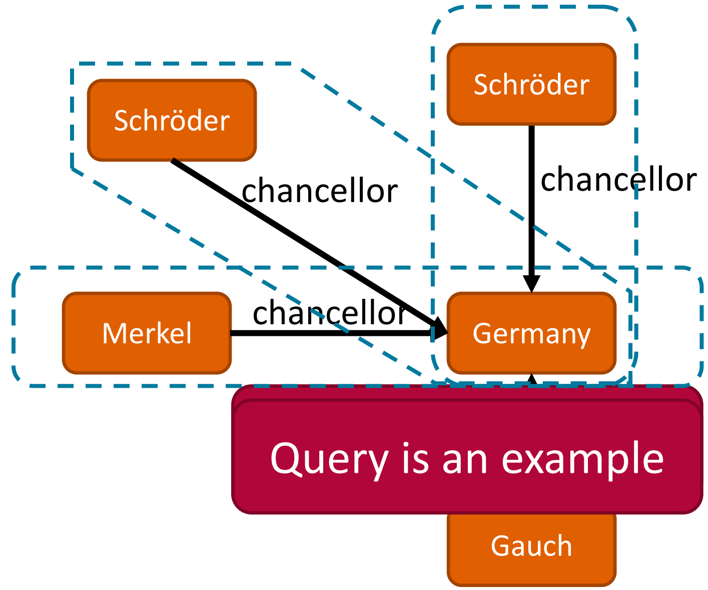
Focused Graph Mining

Refinement of Query Results

Graph exploration taxonomy



Exploratory Graph Analysis



Two exploratory options:

1. An imprecise query

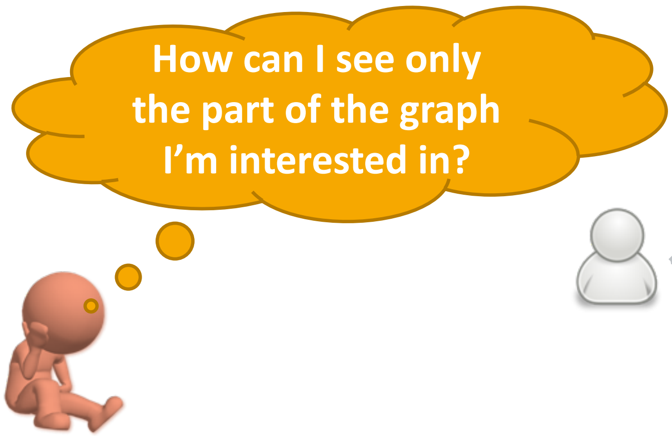
Merkel $\xrightarrow{\text{President of}}$?

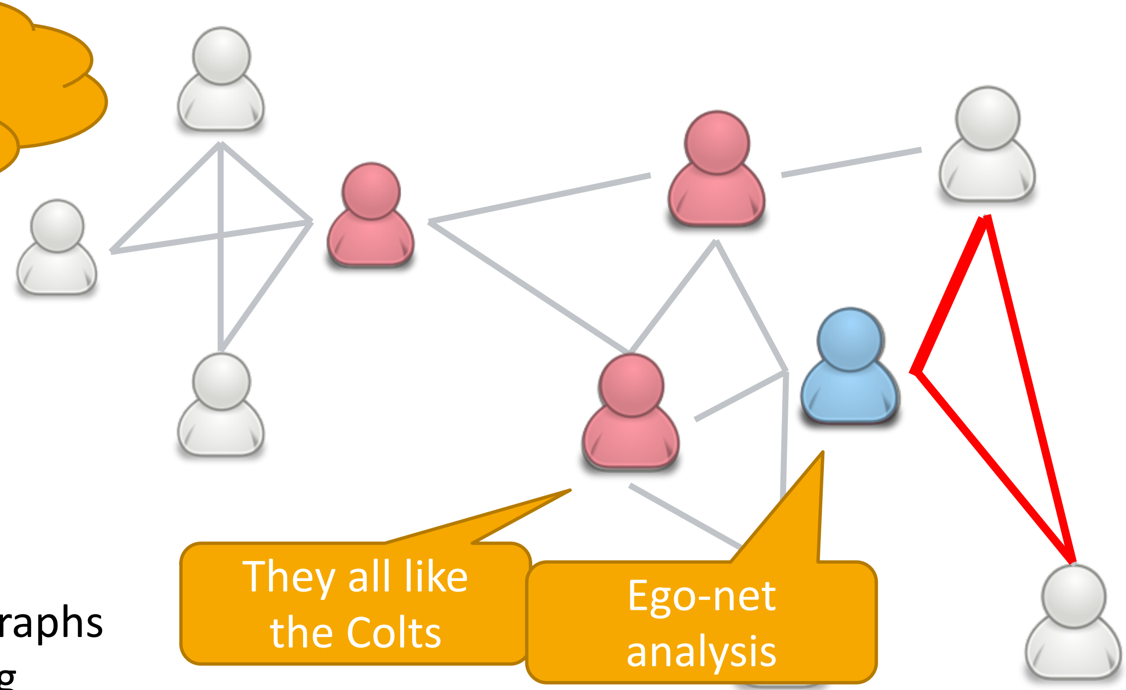
2. A by-example query

Merkel $\xrightarrow{\text{Chancellor}}$ Germany

Graph exploration taxonomy

Focused Graph Mining



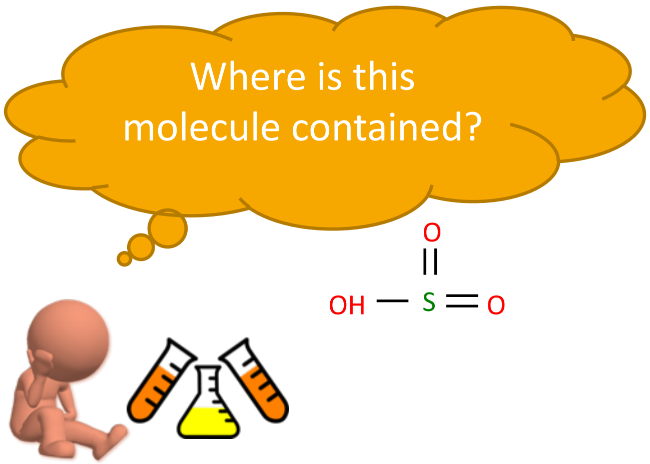


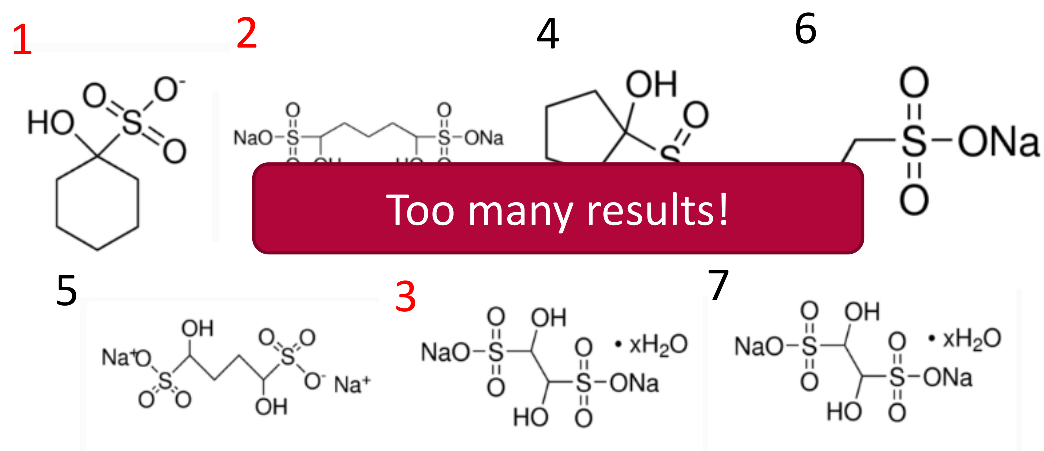
Targeted analysis on large graphs

1. Focused graph clustering
2. Space restriction methods
3. Graph Reweighting

Graph exploration taxonomy

Refinement of Query Results

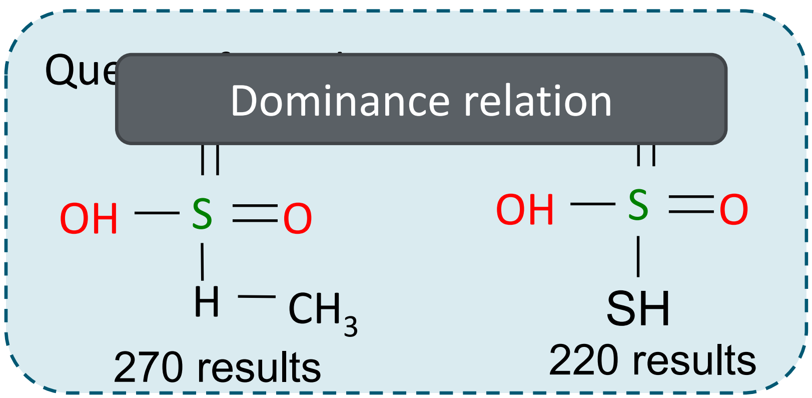




... + 500 results

Dealing with generic queries:

1. Reformulation and refinement
2. Top-k results
3. Skyline queries



Tutorial outline

Background (5 min)

Graph models, subgraph isomorphism, subgraph mining, graph clustering

Exploratory Graph Analysis (35 min)

Focused Graph Mining (35 min)

Refinement of Query Results (35 min)

Real World-Use Case (15min)

Linked Data graphs

Challenges and discussion

Where we are

Background (5 min)

Graph models, subgraph isomorphism, subgraph mining, graph clustering

Exploratory Graph Analysis (35 min)

Focused Graph Mining (35 min)

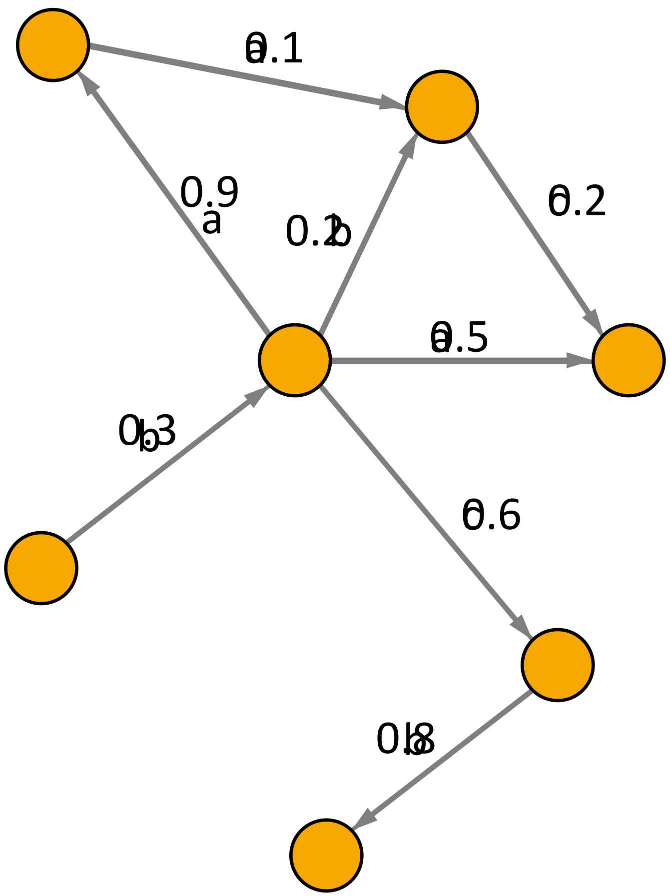
Refinement of Query Results (35 min)

Real World-Use Case (15min)

Linked Data graphs

Challenges and discussion

Graphs



$G = (V, E, l)$ / $G = (V, E, p)$

Vertices

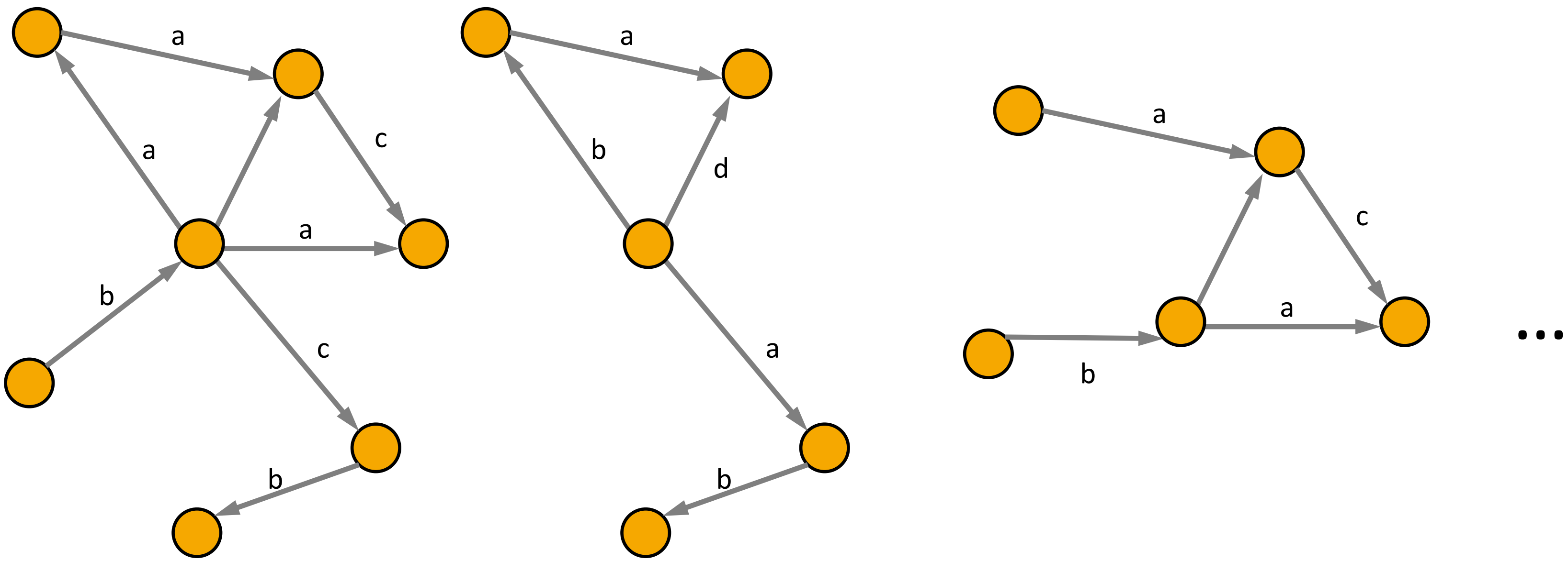
Edges

Labeling / Probability function

$l: V \cup E \to \Sigma$

- Undirected Graphs
  - Co-authorship, Roads, Biological
- Directed graphs
  - Follows, …
- Labeled Graphs
  - Knowledge graphs, …
- Probabilistic graphs
  - Causal graphs

Graph databases (set of graphs)

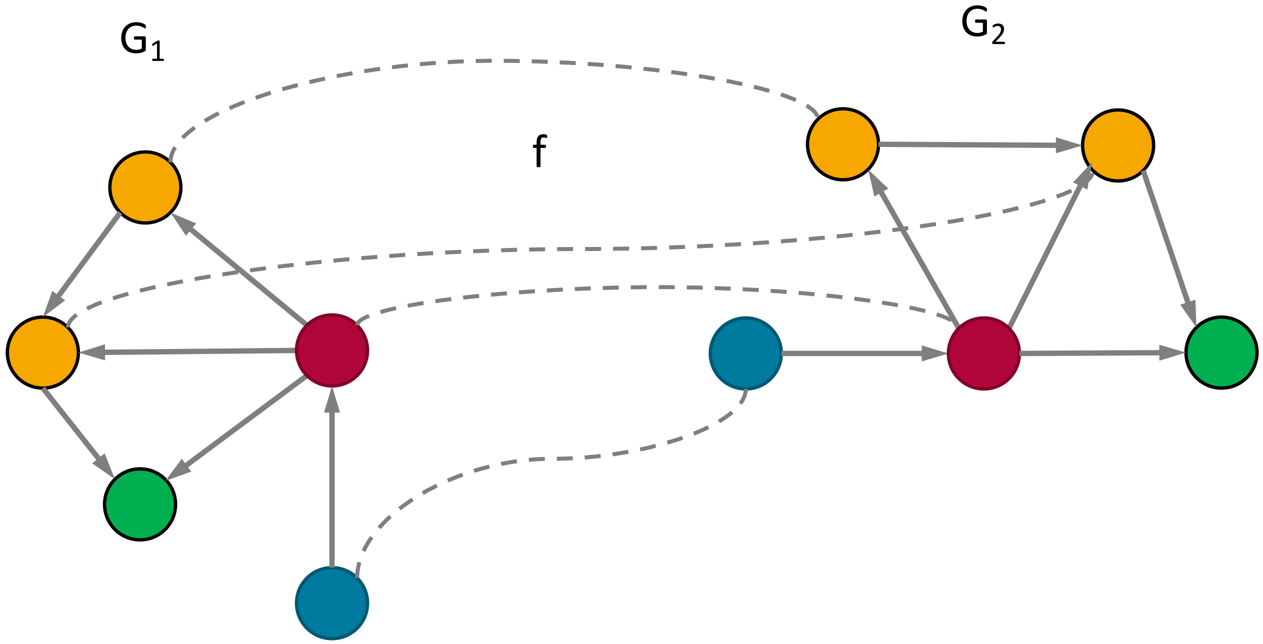


G1 G2 G3

$$D = \{G_1, G_2, \ldots, G_n\}, G_i = \langle V_i, E_i, l_i \rangle, l_i\colon E_i \cup V_i \rightarrow \Sigma$$

Set of small labeled graphs
Chemical compounds, Business models, 3D objects

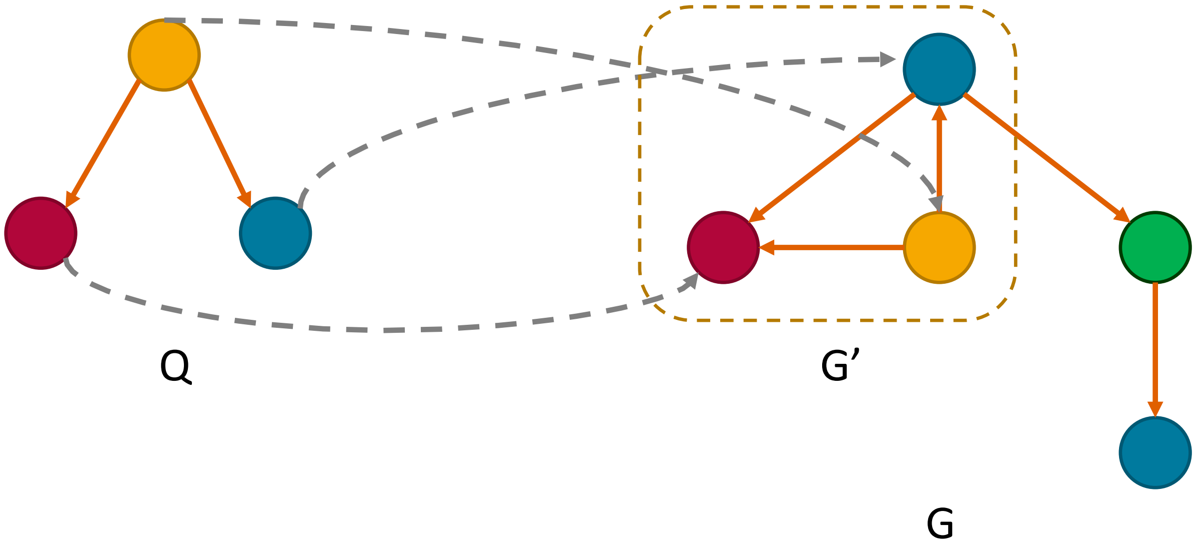
Graph Isomorphism



Given two graphs, $G_1: \langle V_1, E_1, l_1 \rangle$, $G_2: \langle V_2, E_2, l_2 \rangle$ G_1 is isomorphic G_2 iff exists a **bijective** function $f: V_1 \to V_2$ s.t.:

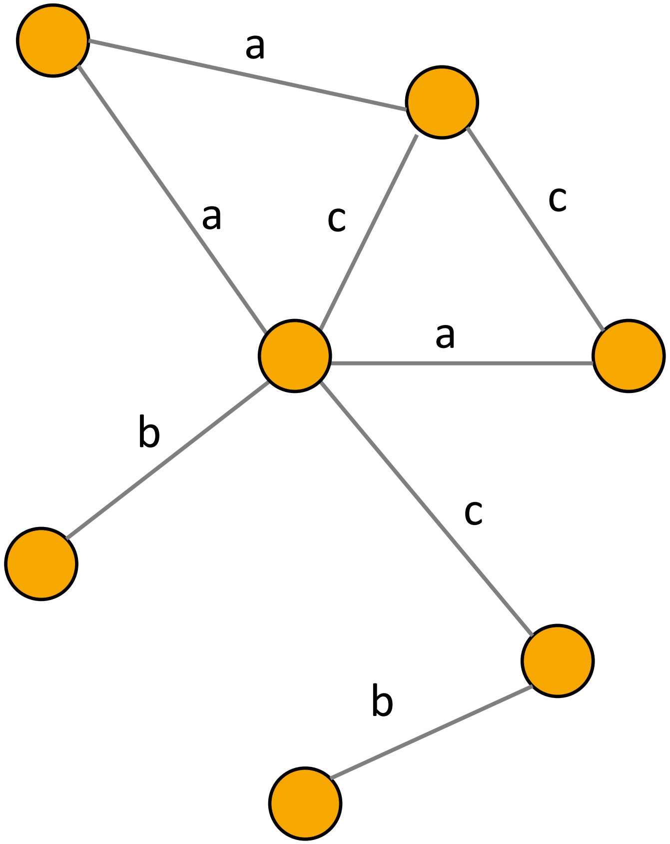
1. For each $v_1 \in V_1, l(v_1) = l(f(v_1))$
2. $(v_1, u_1) \in E_1$ iff $\left(f(v_1), f(u_1)\right) \in E_2$

Subgraph Isomorphism



A graph ,$Q\colon\langle V_Q, E_Q, l_Q\rangle$ is subgraph isomorphic to a graph $G\colon\langle V, E, l\rangle$ if exists a subgraph $G' \sqsubseteq G$, isomorphic to Q

Frequent Subgraph Mining



G

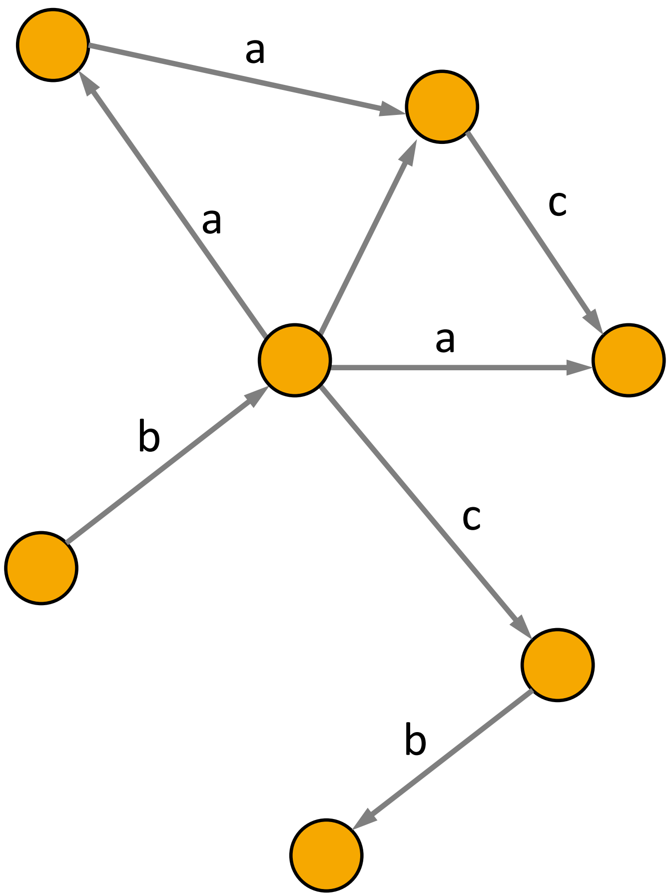
Problem
Find all subgraphs of G that appear at least σ times

Suppose $\sigma = 2$, the frequent subgraphs are (only edge labels)

- a, b, c
- a-a, a-c, b-c, c-c
- a-c-a …

Exponential number of patterns!!!

Graph Clustering and Community Detection



Given: graph with nodes, edges, labels

$$G = (V, E, l)$$

Vertices

Edges

Labeling function $l: V \cup E \rightarrow \Sigma$

Discover: a partitioning of communities

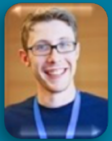
$$C = \{C_1, C_2, C_3, \ldots, C_k\}$$

- **Optimize a given quality criterion** Q(C), e.g. ***Modularity*** or other measures
- Is an **NP-hard problem** to find the optimal partitioning

Where we are

Background (5 min)

Graph models, subgraph isomorphism, subgraph mining, graph clustering

Exploratory Graph Analysis (35 min)

Focused Graph Mining (35 min)

Refinement of Query Results (35 min)

Real World-Use Case (15min)

Linked Data graphs

Challenges and discussion

Exploratory Search

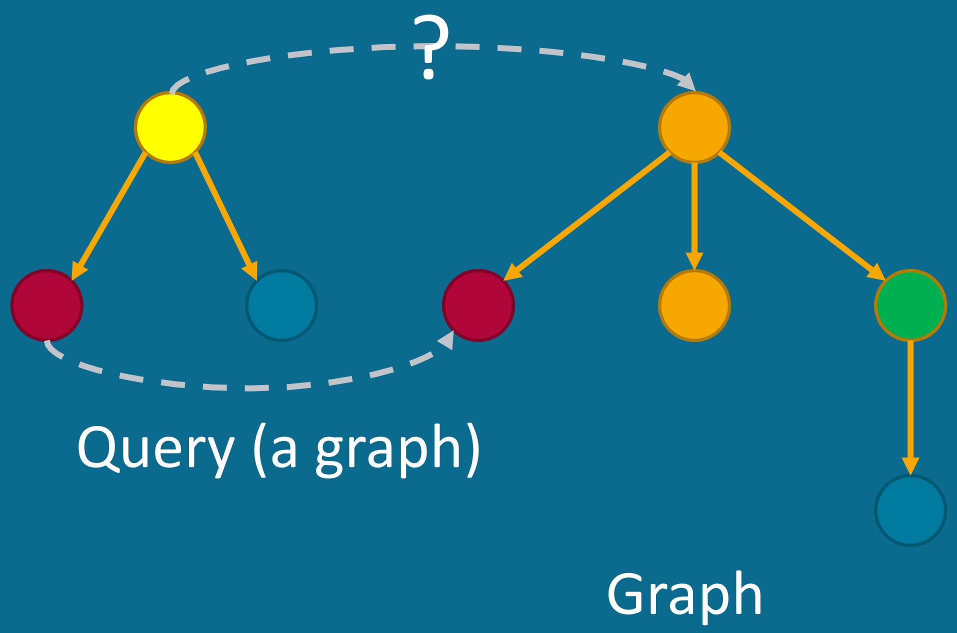
Approximate graph Search

- Given an imprecise query find the closest answers to the query
- User perspective: no need to know about the details of the data

Searching by Example

- Given a example results, find the other results of an unspecified query
- User perspective: it is not necessary to know how to describe the results

Approximate Graph Search



- The user might be imprecise in the search terms

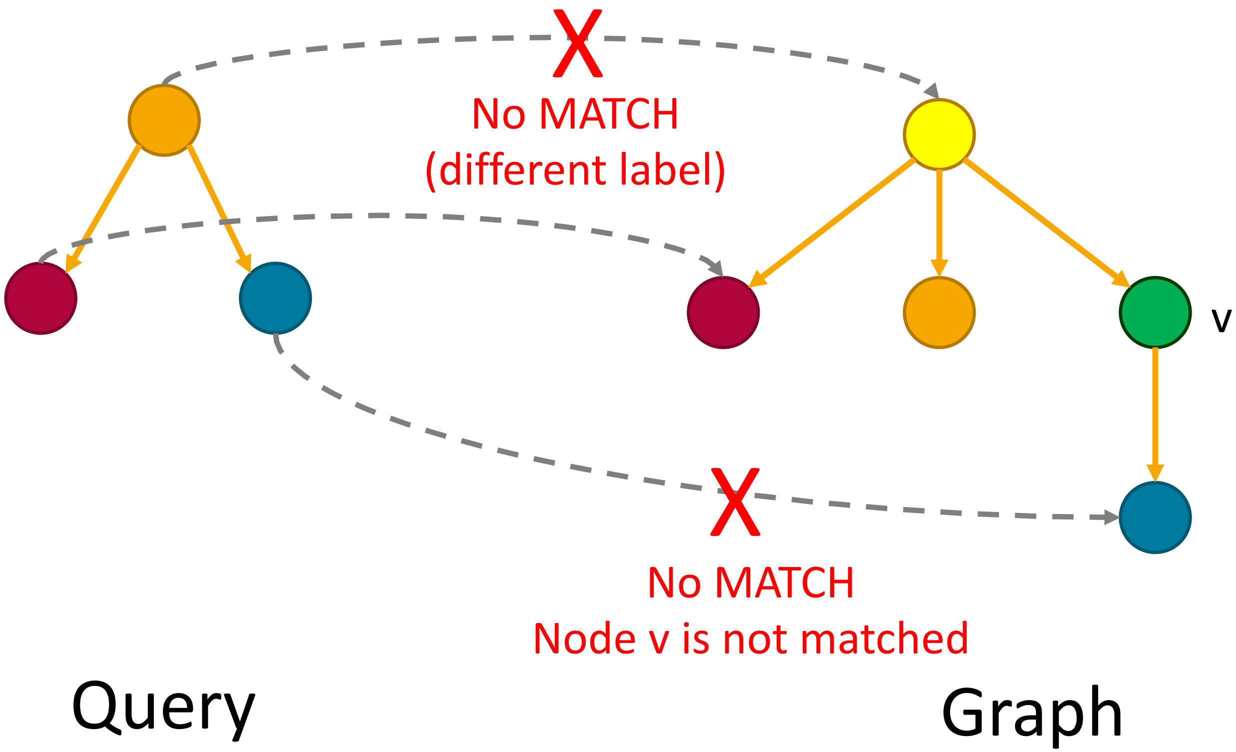
Solution

- Find (partial) correspondence from the query to the graph

- Structural mapping: Strong-simulation (Ma et al.)
- Node similarity approaches: P-homomorphism (Fan et al.), Nema (Khan et al.)
- Probabilistic approaches: SLQ (Yang et al.)

Subgraph isomorphism issues

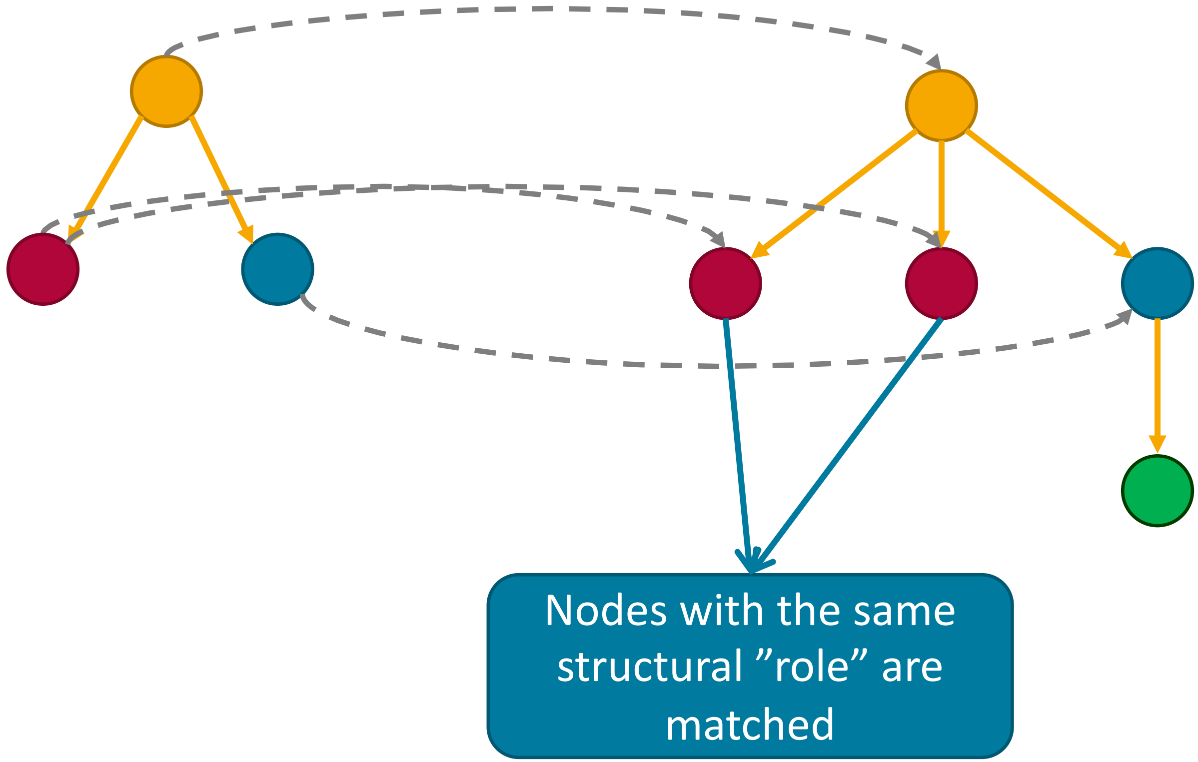
(Sub)Graph Isomorphism might be too restrictive



Fan, W., Li, J., Ma, S., Wang, H. and Wu, Y.. Graph homomorphism revisited for graph matching. PVLDB, 2010

Strong simulation

Revise subgraph isomorphism:
Instead of bijection, compute a binary relation between nodes



Ma, S., Cao, Y., Fan, W., Huai, J. and Wo, T. Strong simulation: Capturing topology in graph pattern matching. *TODS, 2014*

Strong simulation

Poly-time (cubic)

Given $Q: \langle V_q, E_q, l_q \rangle$ and data graph $G: \langle V, E, l \rangle$, a binary relation $S \subseteq V_q \times V$ is said to be a dual simulation if

Graph Simulation [Milner 1989]

- for each $(u, v) \in S$, $l(u) = l(v)$
- for each v $\in V_Q$ exists a node $u \in V$ $s.t. (v, u) \in S$
  - for each edge $(v, v') \in E_q$, there exists an edge $(u, u') \in E$ such that $(v', u') \in S$
  - for each edge $(v'', v) \in E_q$, there exists an edge $(u'', u) \in E$ such that $(v'', u'') \in S$

Parent-child relationship

Child-parent relationship

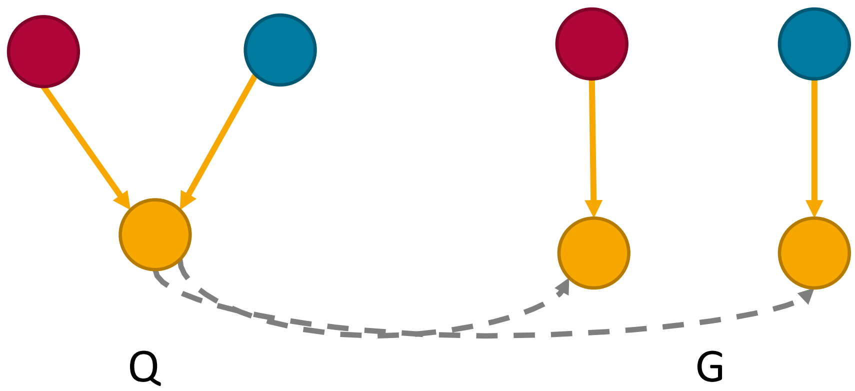
Duality

Locality

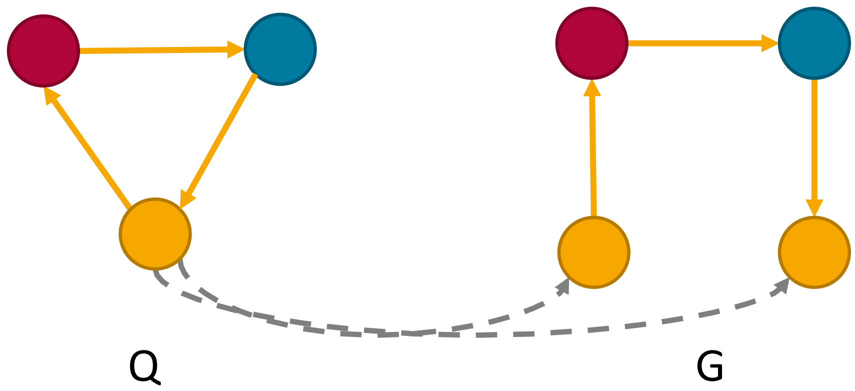
- The matching subgraph is:
  - connected graph
  - the diameter is not larger than twice the diameter of the query

Ma, S., Cao, Y., Fan, W., Huai, J. and Wo, T. Strong simulation: Capturing topology in graph pattern matching. *TODS, 2014*

Strong simulation



Without Locality: unconnected and unbounded graphs



Without Duality: Trees match cycles

Ma, S., Cao, Y., Fan, W., Huai, J. and Wo, T. Strong simulation: Capturing topology in graph pattern matching. *TODS, 2014*

Properties of Strong Simulation

If Q matches G, via subgraph isomorphism,
then Q matches G, via strong simulation

If Q matches G, via strong simulation,
then Q matches G, via dual simulation

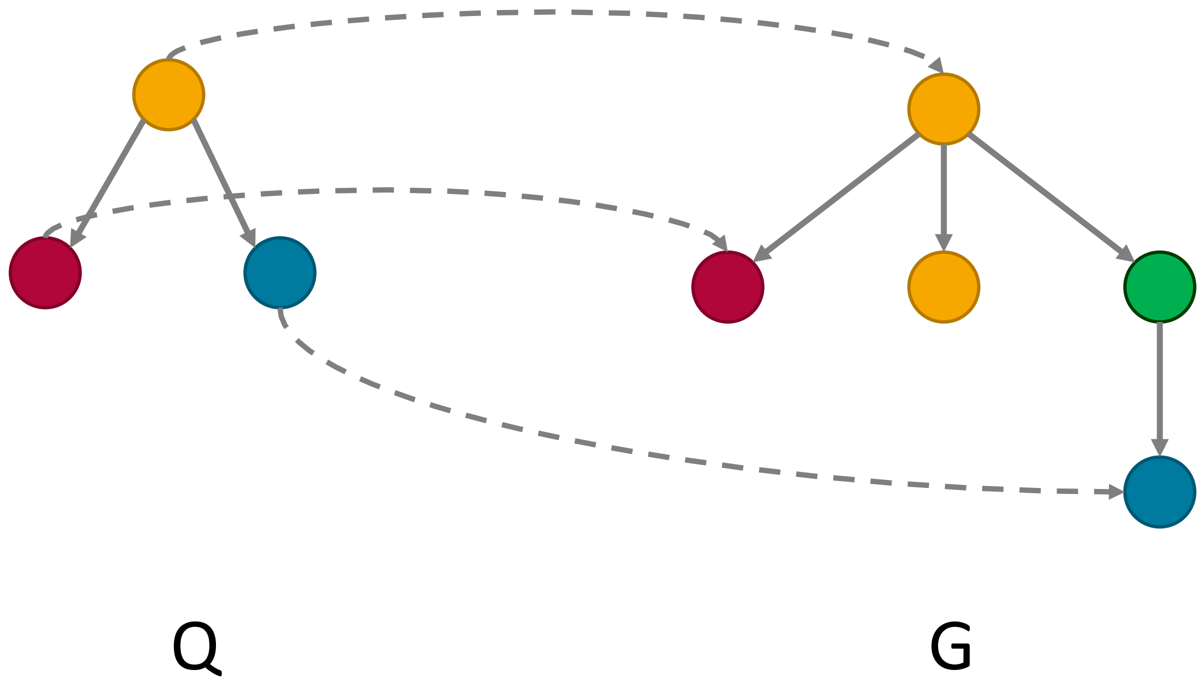
If Q matches G, via dual simulation,
then Q matches G, via graph simulation

Subgraph Isomorphism > Strong Simulation > Dual Simulation > Graph Simulation

Ma, S., Cao, Y., Fan, W., Huai, J. and Wo, T. Strong simulation: Capturing topology in graph pattern matching. *TODS, 2014*

Graph homomorphism

Revise graph homomorphism: match paths

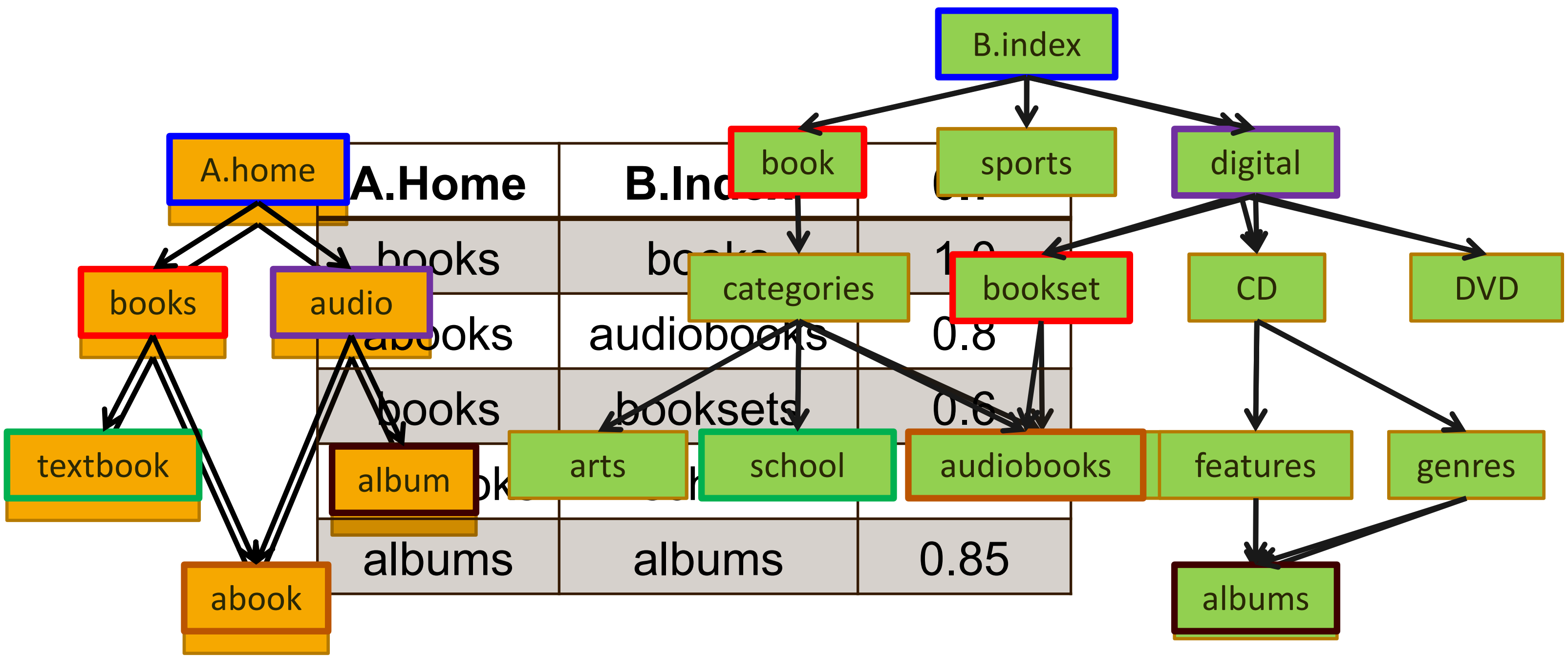


Fan, W., Li, J., Ma, S., Wang, H. and Wu, Y.. Graph homomorphism revisited for graph matching. PVLDB, 2010

P-Homomorphism

NP-complete

- Matches paths instead of single edges
- Similarity matrix between nodes M over Q and G,, M(u,v) similarity score of node u in Q and v in G.
- Similarity threshold ξ

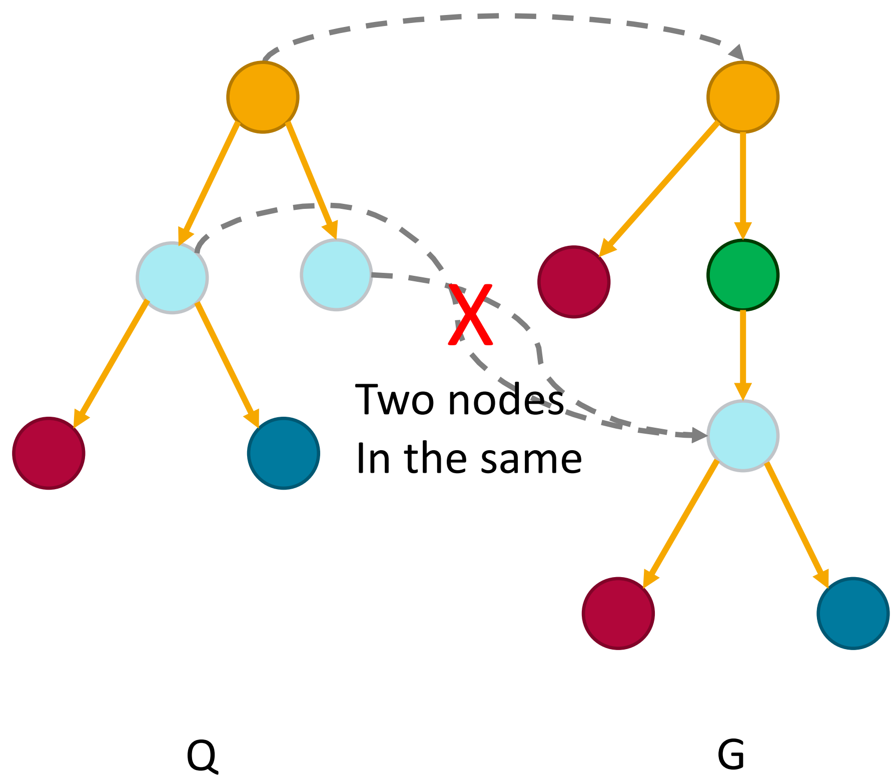


Fan, W., Li, J., Ma, S., Wang, H. and Wu, Y.. Graph homomorphism revisited for graph matching. PVLDB, 2010

1-1 P-Homomorphism

NP-complete

Injective P-Homomorphism mapping from Q and G



Fan, W., Li, J., Ma, S., Wang, H. and Wu, Y.. Graph homomorphism revisited for graph matching. PVLDB, 2010

Queries with (1-1)P-Homomorphism

Maximum cardinality problem (CPH)

- Return the (1-1)P-hom mapping ρ with maximum Card(ρ).
- The cardinality of p-hom mapping from a subgraph G' = (V', E',L') of Q to G:
  - $Card(\rho) = |V'|/|V_Q|$

Maximum Overall similarity (SPH)

- Return the (1-1)P-hom mapping ρ with maximum Sim(ρ) .
- The overall similarity of p-hom mapping from a subgraph G' of Q to G:

Decision problems
NP-hard for DAGs

Fan, W., Li, J., Ma, S., Wang, H. and Wu, Y.. Graph homomorphism revisited for graph matching. PVLDB, 2010

Approximation algorithm for CPH

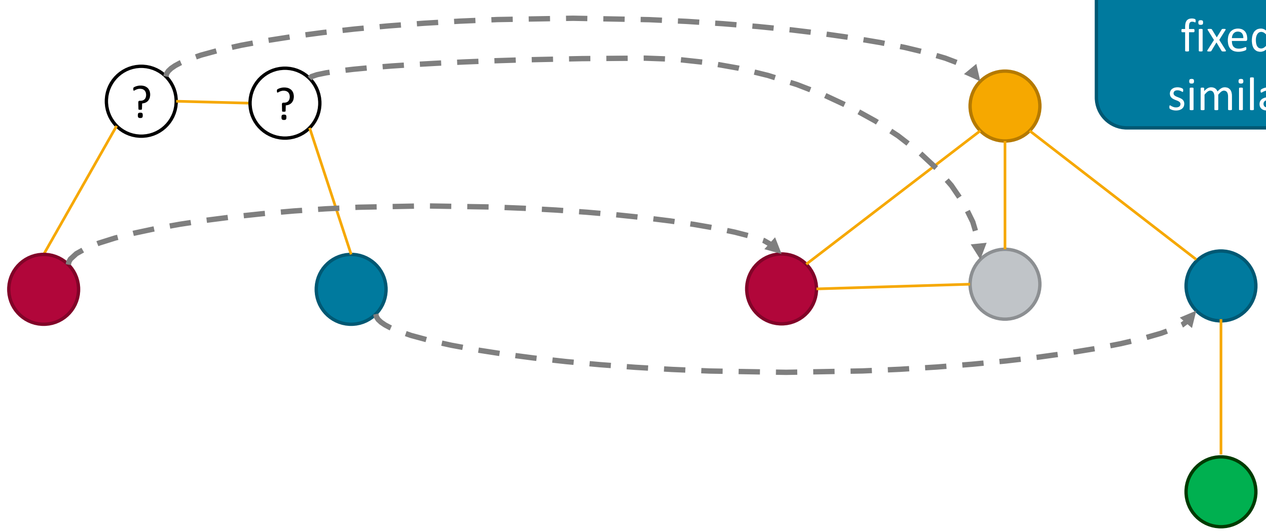
Algorithm compMaxCard(G_1,G_2,M, ξ)

- Input: $G_1 = (V_1, E_1, L_1)$, $G_2 = (V_2, E_2, L_2)$, similarity matrix M, similarity threshold ξ
- Output: a P-hom mapping from subgraph of G_1 to G_2
- Procedure
  - initialize matching list for each node in G_1
  - compute the transitive closure of G_2
  - starting from a match pair, recursively choose and include new matches to the match set until it can no longer be extended, via a greedy strategy.
- Complexity: $O(|V_1|^3 |V_2|^2 + |V_1||E_1||V_2|^3)$

P-Hom problems can be solved with a provable performance guarantee

NeMa

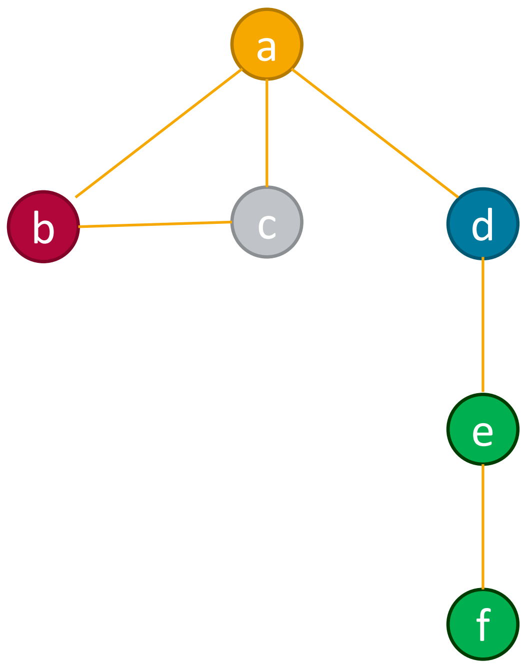
Relax **p-homomorphism**:
Structure and some labels are unknown, node closed in the query must be closed in the graph



The structure is not fixed anymore but similar to the query

Khan, A., Wu, Y., Aggarwal, C.C. and Yan, X. Nema: Fast graph search with label similarity. PVLDB, 2013

NeMa: compute node vectors



$$w_u(u') = \begin{cases} \alpha^{d(u,u')}, & d(u,u') \leq h \\ 0, & otherwise \end{cases}$$

Distance less than h (h-hop neighbor)

$h = 2, \alpha = 0.5$

$R_G(a) = \{(b, 0.5), (c, 0.5), (d, 0.5), (e, 0.25)\}$

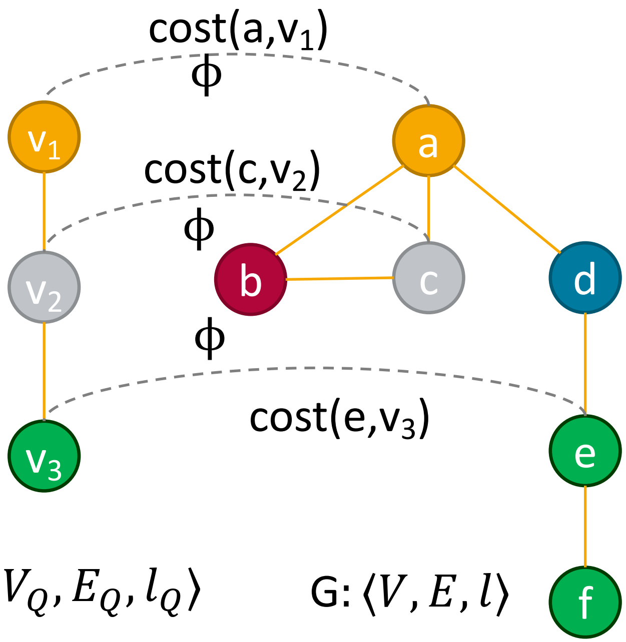
Vector of nodes at distance <= h from a

Khan, A., Wu, Y., Aggarwal, C.C. and Yan, X. Nema: Fast graph search with label similarity. PVLDB, 2013

NeMa

NP-hard

APX-hard



$Q{:}\,\langle V_Q, E_Q, l_Q\rangle$ G: $\langle V, E, l\rangle$

$$cost(v,u) = \Delta_L\big(l(v), l(u)\big) + \sum_{v' \in N(v)} \Delta_+\big(w_v(v'), w_u(u')\big)$$

Label comparison cost

Node vectors difference

$$C(\phi) = \sum_{v \in V_Q} cost(v, \phi(v))$$

Overall cost of mapping ϕ

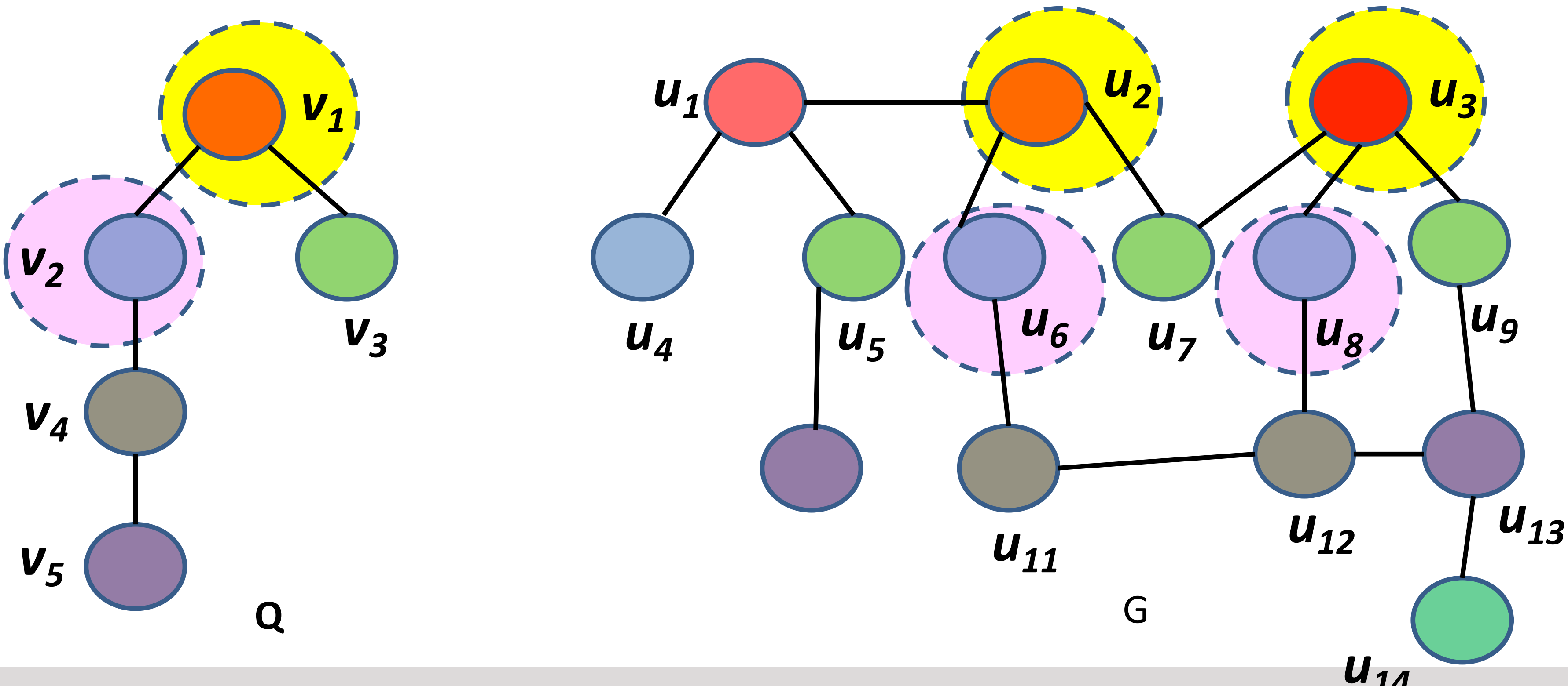
Problem
Given Q and G, find the mapping ϕ with the minimum cost $C(\phi)$

Khan, A., Wu, Y., Aggarwal, C.C. and Yan, X. Nema: Fast graph search with label similarity. PVLDB, 2013

NemaInfer algorithm

If a node has "good" neighbors, more likely it is a "good" match.

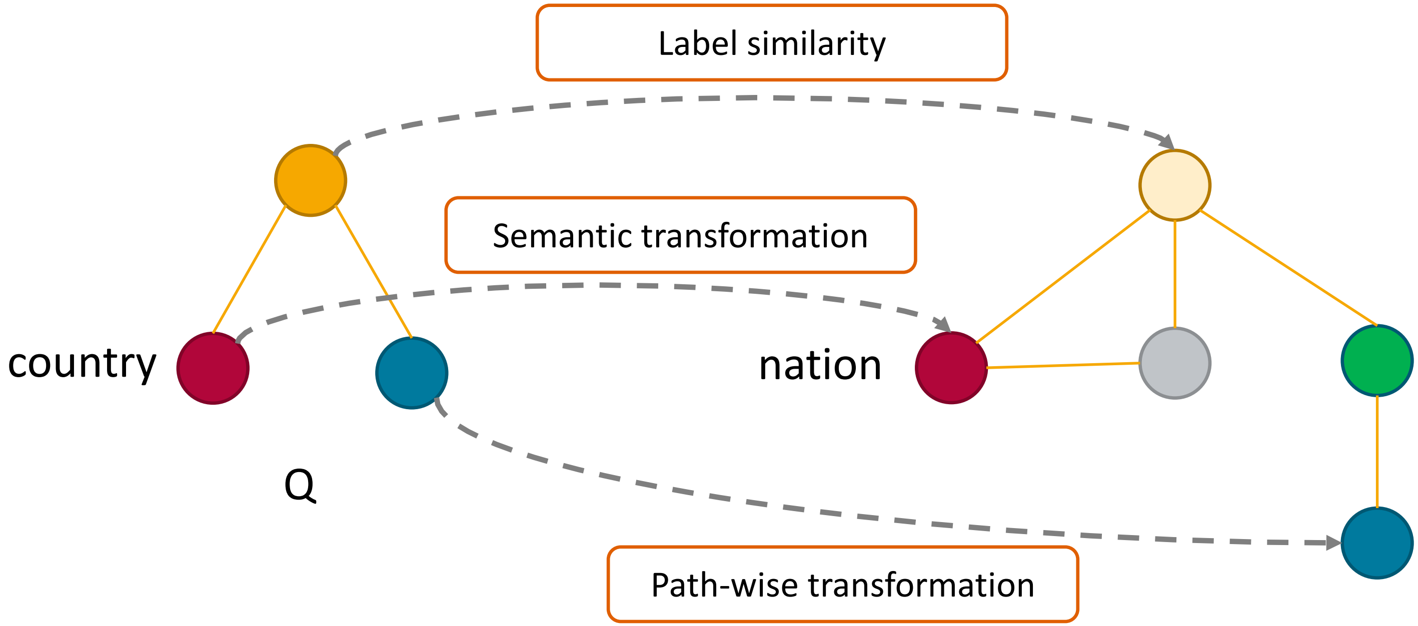
$$U_i(v,u) = \min_{\{\phi:\phi(v)=u\}} \left[F_\phi(v,u) + \sum_{v' \in \mathbb{N}(v)} U_{i-1}(v',u')\right]$$



Khan, A., Wu, Y., Aggarwal, C.C. and Yan, X. Nema: Fast graph search with label similarity. PVLDB, 2013

SLQ

Similar to **NEMA**
Assume that a match is obtained by a sequence of transformations of the query nodes into the graph



Yang, S., Wu, Y., Sun, H. and Yan, X. Schemaless and structureless graph querying. *PVLDB, 2014*.

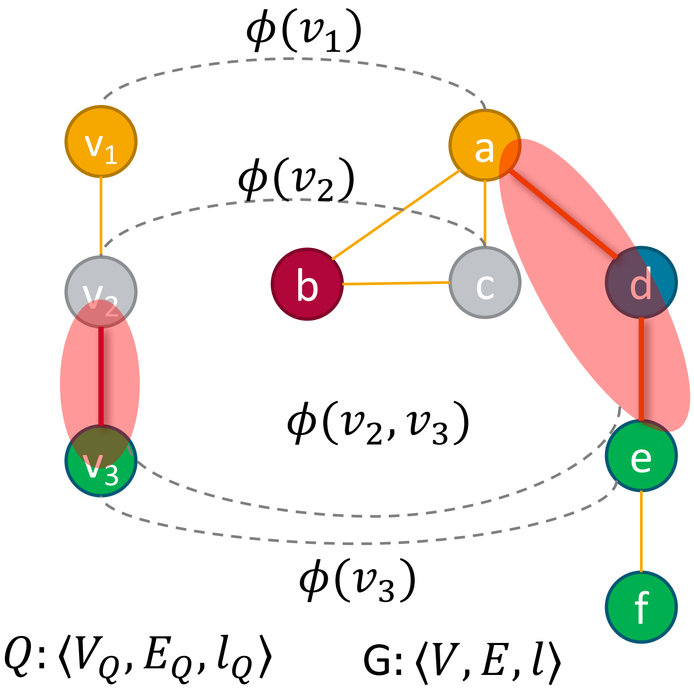
Transformations

Transformation	Category	Example
First/last token	String	Barack Obama -> Obama
Abbreviation	String	Jeffrey Jacob Abrams -> J. J. Abrams
Prefix	String	Engineer-> Eng.
Acronym	String	Microsoft -> MS
Synonym	Semantic	Country -> Nation
Ontology	Semantic	Table -> Furniture
Range	Numeric	~30 -> 33
Distance	Topology	Dallas – USA -> Dallas – Texas - USA

They can be expanded arbitrarily

Yang, S., Wu, Y., Sun, H. and Yan, X. Schemaless and structureless graph querying. *PVLDB, 2014*.

Model on transformations



$$F_V(v, \phi(v)) = \sum_i \alpha_i f_i(v, \phi(v))$$ Node matching score

$$F_E(e, \phi(e)) = \sum_i \beta_i f_i(e, \phi(e))$$ Edge matching score

$$P(\phi|Q) \propto \exp\Big(\sum_{v \in V_Q} F_V(v, \phi(v)) + \sum_{e \in E_Q} F_E(e, \phi(e))\Big)$$

Overall score for matching ϕ

Problem

- How to learn the parameters α_i, β_i ?
- How to find the matching with the highest score?

Yang, S., Wu, Y., Sun, H. and Yan, X. Schemaless and structureless graph querying. *PVLDB, 2014*.

Querying with SLQ

Learning the parameters (offline)

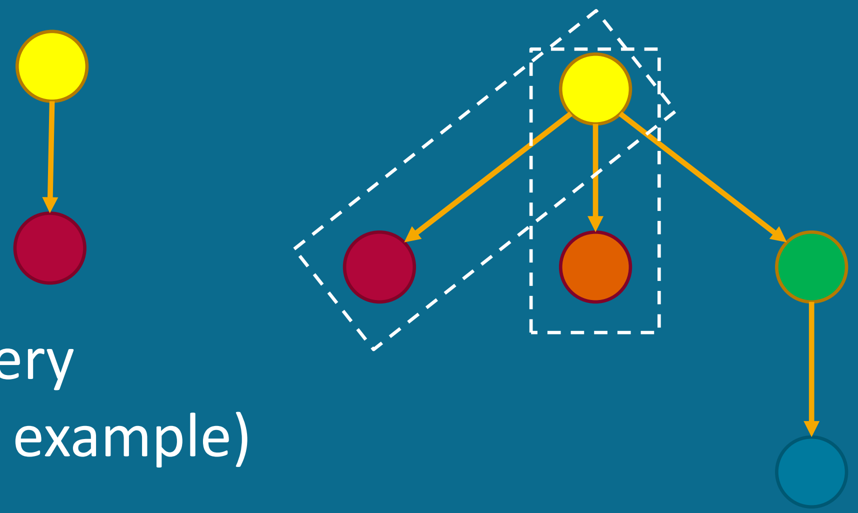
1. Random sample a structure from the graph
2. Apply random transformations on the found structure
3. Search the generated queries on the graphs
4. Label the results as positive or negative
5. Train a Conditional Random Field on the examples

Query phase

1. Construct a CRF model on the query and matching candidates
2. Use Loopy Belief Propagation to find the most likely (top-1) assignment
   - $m_{ji}^{(t)}(u_i) = \max_{u_j} F_V(v_j, u_j) F_E((v_j, v_i), (u_j, u_i)) \prod_{v_k \in N(v_j) \setminus v_i} m_{kj}^{(t-1)}(u_j)$

Yang, S., Wu, Y., Sun, H. and Yan, X. Schemaless and structureless graph querying. *PVLDB, 2014*.

Querying by Example



Query
(an example)

Graph

- The user query is an example result

Solution

- Find results that are similar to the one in input

Exemplar Queries (Mottin et al.), GQBE (Jayaram et al.)

NOT approximate queries:
a result to an approximate query is the closest possible to the query itself

Exemplar Queries

Input: Q_e, an example element of interest

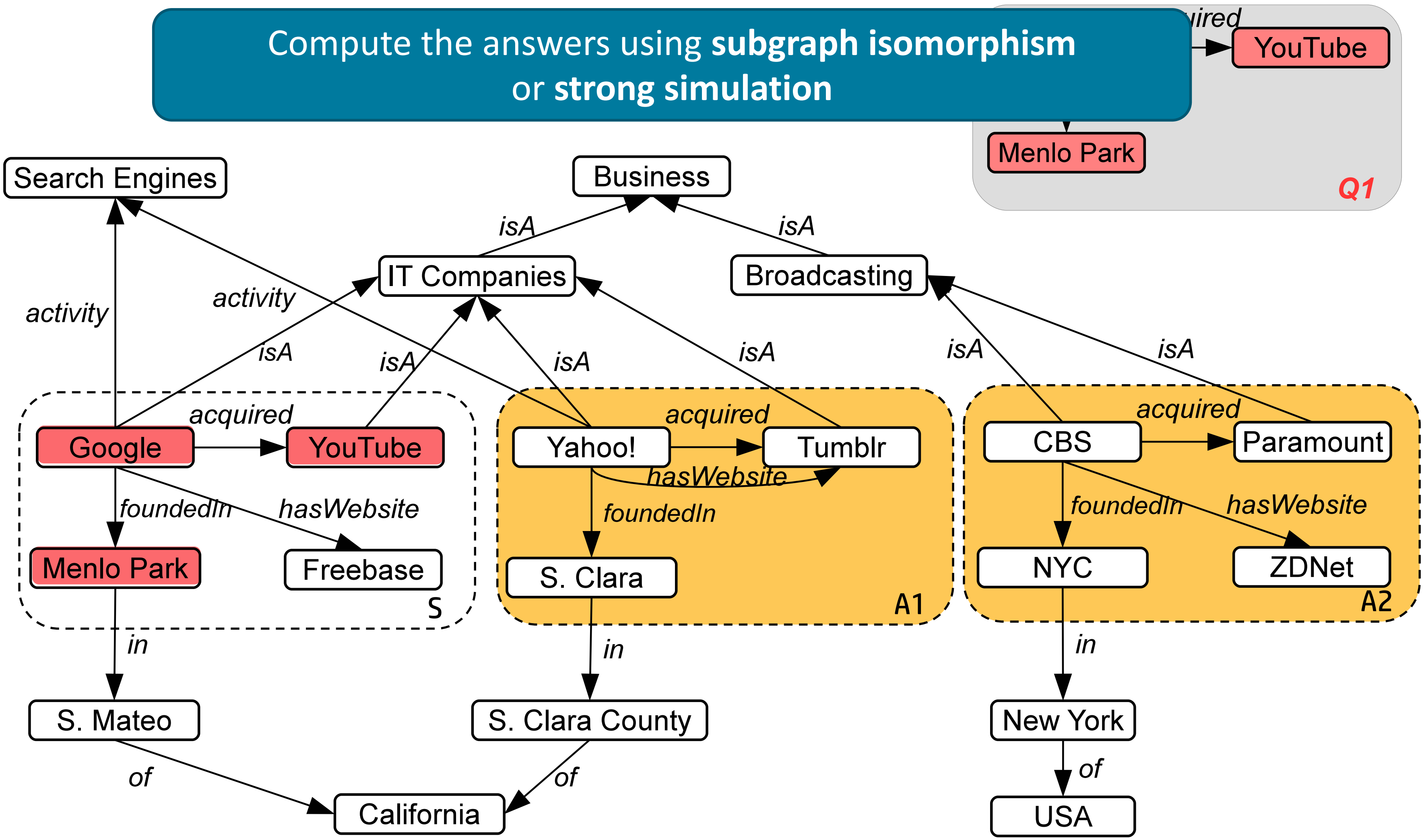
Output: set of elements in the desired result set

Exemplar Query Evaluation

- evaluate Q_e in a database D, finding a sample *s*
- find the set of elements *a* similar to *s* given a *similarity relation*

Mottin, D., Lissandrini, M., Velegrakis, Y. and Palpanas, T. Exemplar queries: Give me an example of what you need. *PVLDB 2014*

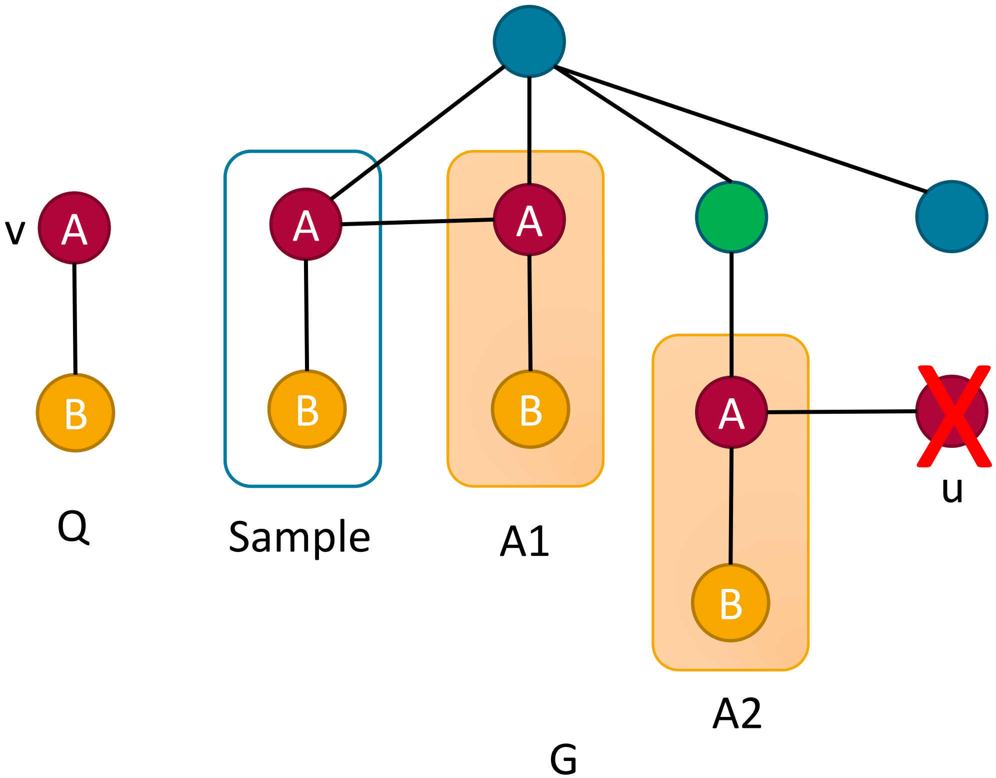
Exemplar Queries



Computing exemplar queries

NP-complete (subgraph isomorphism)

$O(|V|^4)$ (simulation)



Pruning technique:

- Compute the neighbor labels of each node

$$W_{n,a,i} = \{n_1 | l(n_1, n_2) = a \vee \in N_{i-1}(n)\}$$

- Prune nodes not matching query nodes neighborhood labels
- Apply the technique iteratively on the query nodes

Labels at distance 1

v neighborhood = {(B,1)}

$\not\subseteq$

u neighborhood = {(A,1)}

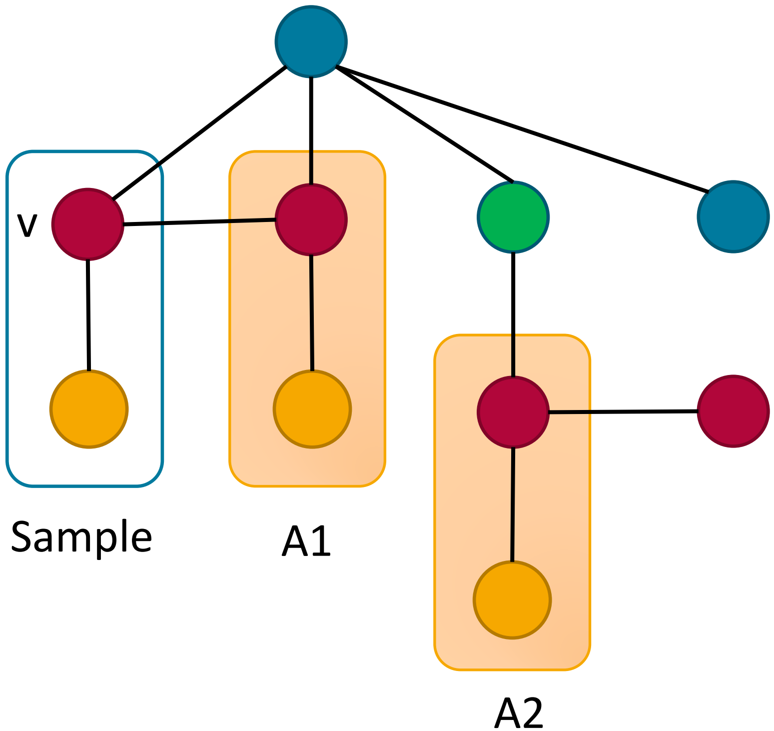
No Match

Mottin, D., Lissandrini, M., Velegrakis, Y. and Palpanas, T. Exemplar queries: Give me an example of what you need. *PVLDB 2014*

Computing exemplar queries

NP-complete
(subgraph isomorphism)

$O(|V|^4)$ (simulation)



Approximation:

- Nodes closed to the sample are more important
- Use Personalized PageRank with a weighted matrix

$$\boldsymbol{v} = (1-c)A\boldsymbol{v} + c\boldsymbol{p}$$

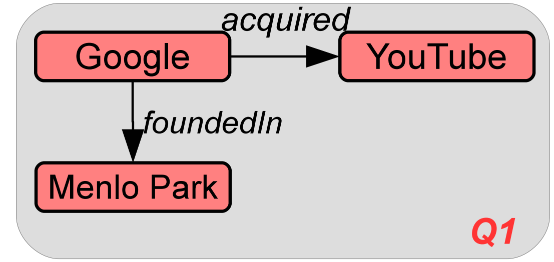
- Weight edges using the frequency of the edge-label

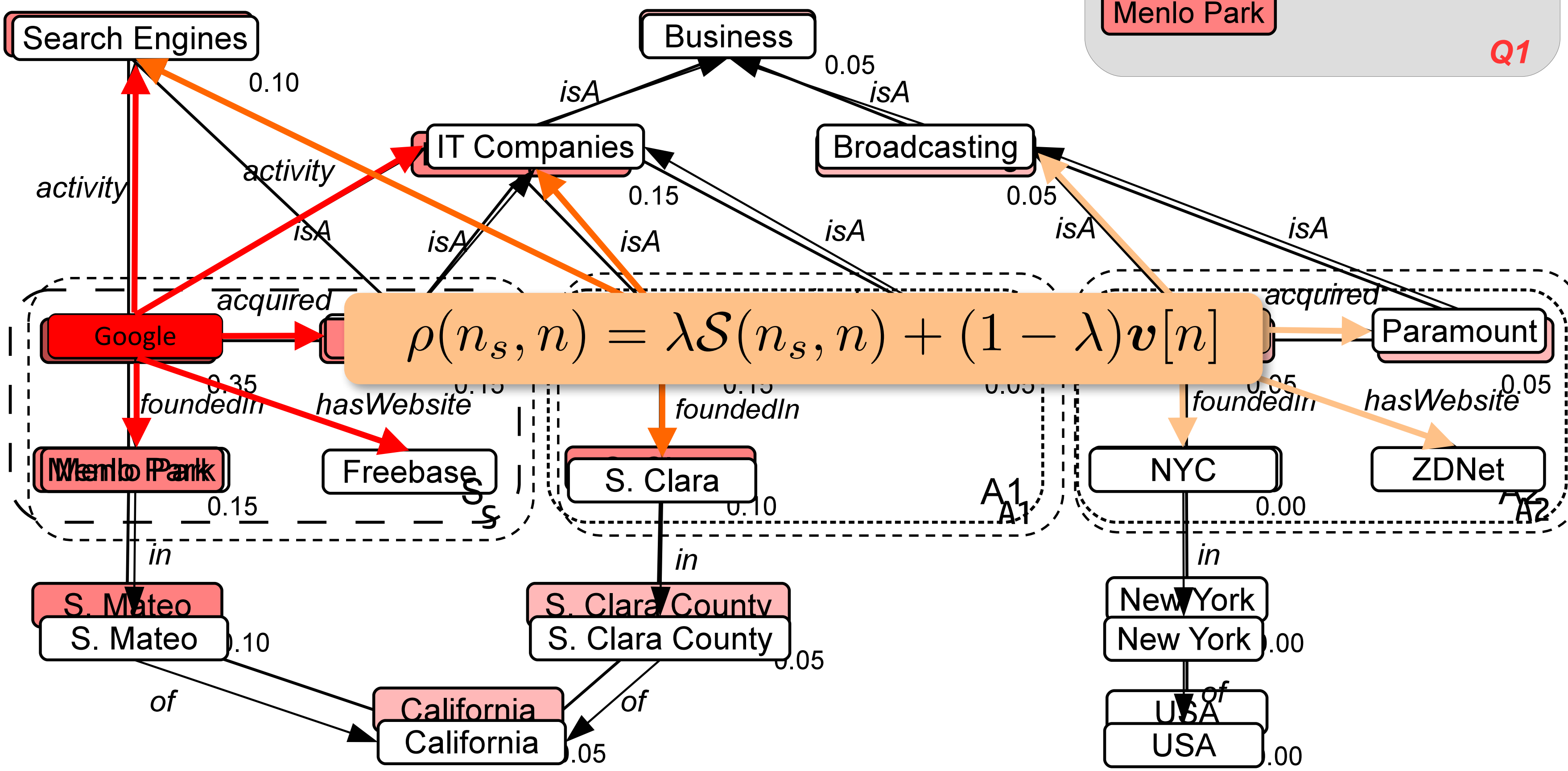
$$I(e_{ij}^{\ell}) = I(\ell) = \log \frac{1}{P(\ell)} = -\log P(\ell)$$

$$P(\ell) = \frac{|E^{\ell}|}{|E|}$$

Mottin, D., Lissandrini, M., Velegrakis, Y. and Palpanas, T. Exemplar queries: Give me an example of what you need. *PVLDB 2014*

Ranking results





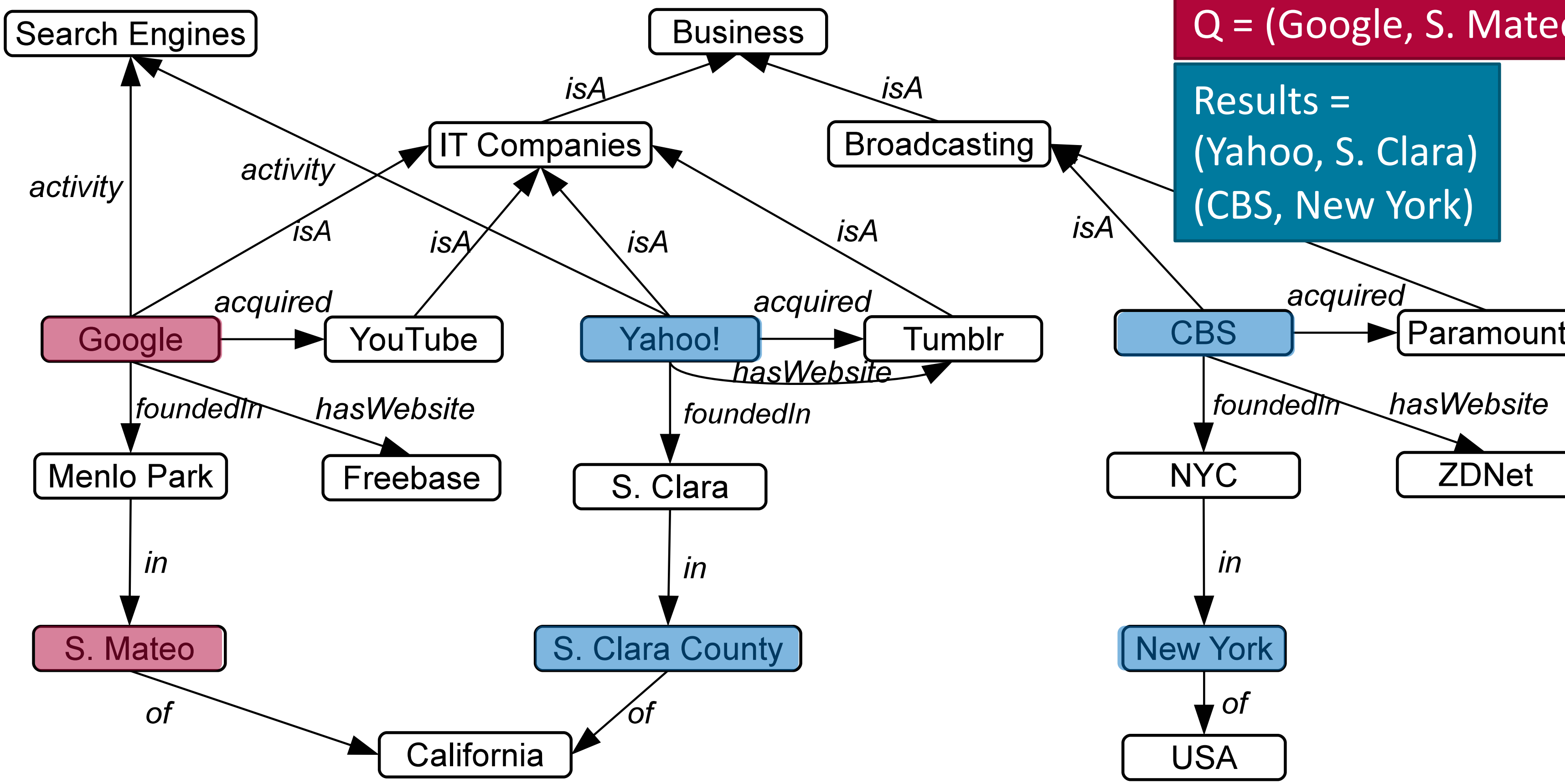
$$\rho(n_s, n) = \lambda \mathcal{S}(n_s, n) + (1 - \lambda)\boldsymbol{v}[n]$$

Combination of two factors

1. **Structural**: similarity of two nodes in terms of neighbor relationships
2. **Distance-based**: the PageRank already computed

Graph query by example (GQBE)

In GQBE Input is a set of (disconnected) entity mention tuples



Q = (Google, S. Mateo)

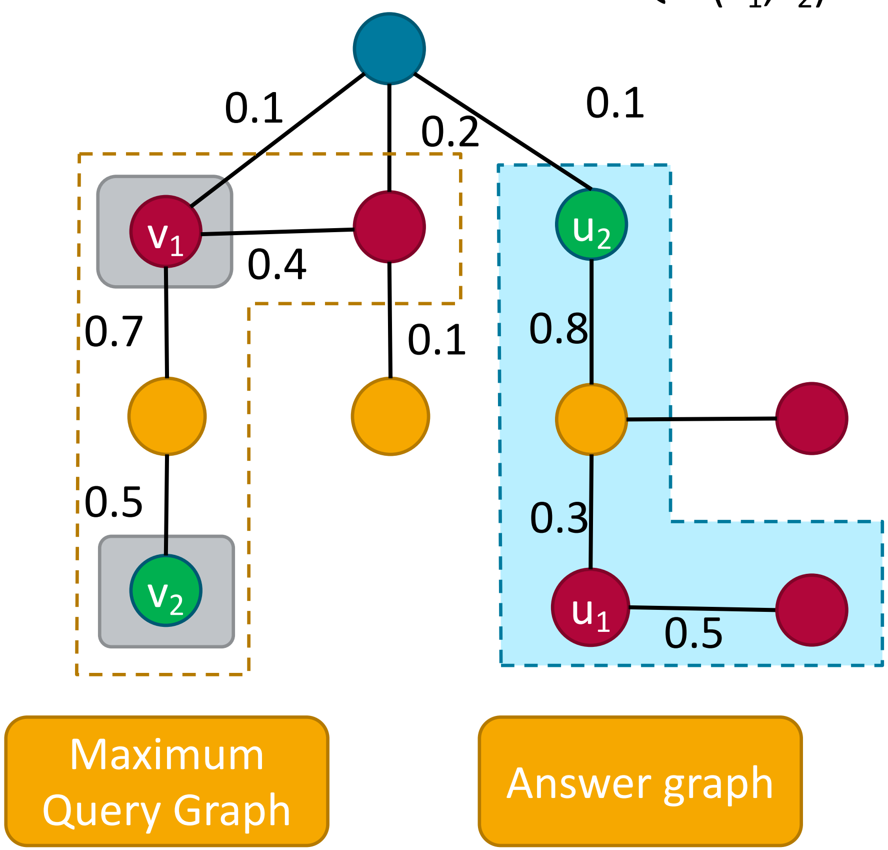
Results =
(Yahoo, S. Clara)
(CBS, New York)

Jayaram, N., Khan, A., Li, C., Yan, X. and Elmasri, R. Querying knowledge graphs by example entity tuples. *TKDE, 2015*

GQBE

NP-hard

$Q = (v_1, v_2)$



1. Find the maximum query graph
   - Graph with m edges having the maximum weight
2. Find all the answers subgraph isomorphic to the query graph
3. Rank the answers and return the top-k tuples

Answer score:

- Sum of query graph weights
- Similarity match between edges in the answer and the query

$$\text{match}(e, e') = \begin{cases} \frac{\text{w}(e)}{|E(u)|} & \text{if } u{=}f(u) \\ \frac{\text{w}(e)}{|E(v)|} & \text{if } v{=}f(v) \\ \frac{\text{w}(e)}{min(|E(u)|,|E(v)|)} & \text{if } u{=}f(u), v{=}f(v) \\ 0 & \text{otherwise} \end{cases}$$

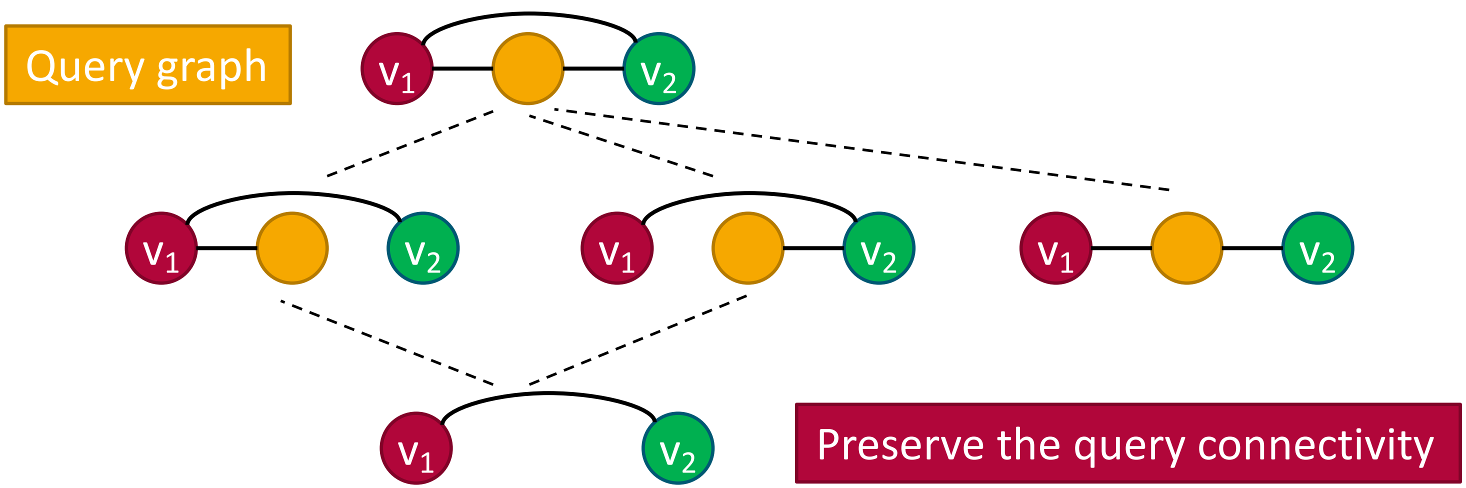
Bonifati, A., Ciucanu, R. and Lemay, A. Learning path queries on graph databases. EDBT 2015.

Multiple query tuples

NP-hard

GQBE finds answers for multiple query tuples

1. Compute a re-weighted union graph of the individual query graphs
2. Find answers using a lattice obtained removing edges from the union graph



Bonifati, A., Ciucanu, R. and Lemay, A. Learning path queries on graph databases. EDBT 2015.

Where we are

Background (5 min)

Graph models, subgraph isomorphism, subgraph mining, graph clustering

Exploratory Graph Analysis (35 min)

Focused Graph Mining (35 min)

Refinement of Query Results (35 min)

Real World-Use Case (15min)

Linked Data graphs

Challenges and discussion

Graph Mining – a very broad topic

Link Prediction

Community Detection

Anomaly Detection

Frequent Subgraph Mining

Graph Partitioning

… many more …

Graph Mining Focused on User Interest

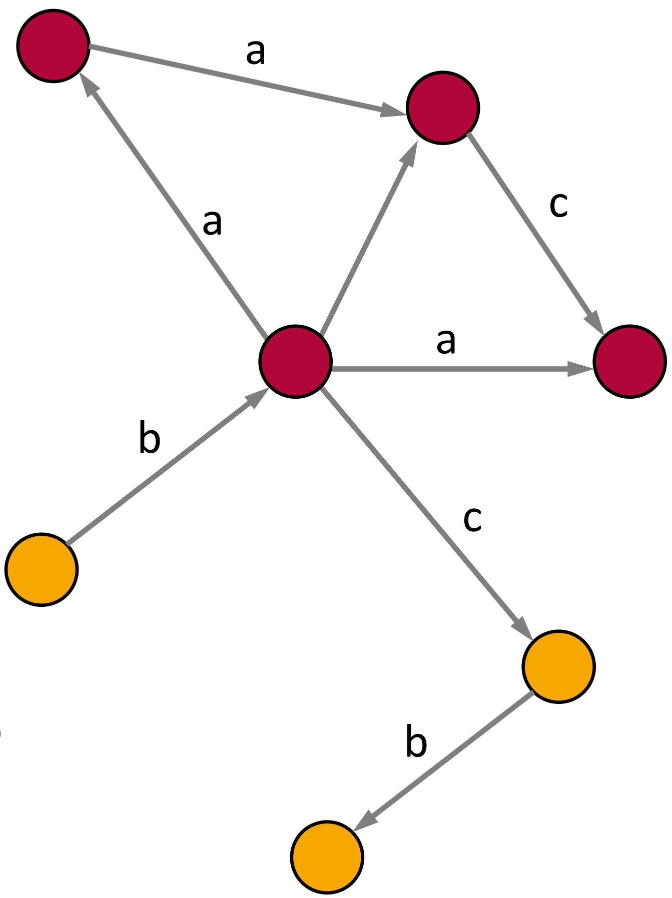
We consider "user interest" at a major tool for adaptive graph mining

- In contrast to **raw analysis of graphs** (i.e. with no or very little user interaction)
- Example (modularity based clustering):

Given a graph
discover best partitioning of the nodes

Optimize a given quality criterion Q(C),
e.g. ***Modularity*** or other measures

- Where is the user interest in such definitions?
- How to include the user into the loop?
- How do we need to change the algorithmic search?



Focus: Given a Set of Query Nodes

Given Q nodes (by the user)

How can we **find the center-piece node**
that has direct or indirect connections
to all or most of these nodes?

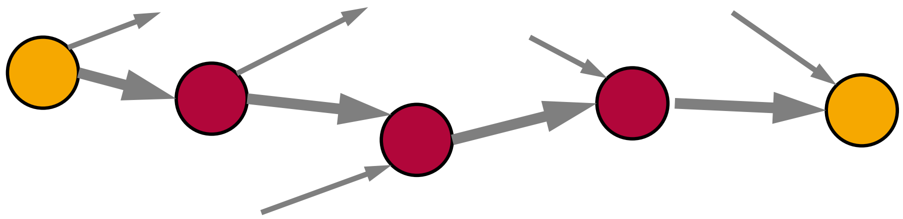
- Neither a clustering of nodes
- Nor the shortest path between pairs of nodes
- Nor any other graph mining method (with lack of user input)

H. Tong & C. Faloutsos: Center-Piece Subgraphs: Problem Definition and Fast Solutions. (KDD 2006)

CEPS “Center-Piece Subgraph”

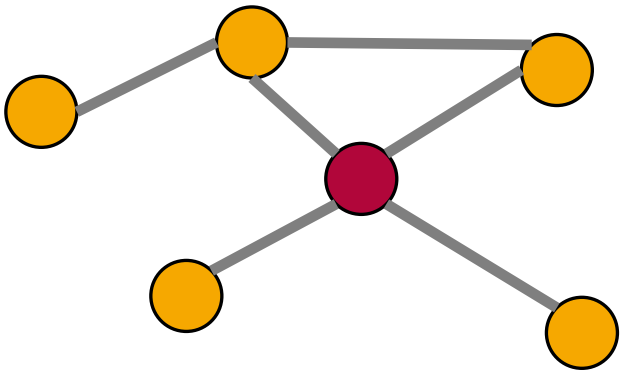
|Q| = 2

Only a pair of query nodes



|Q| >> 2

Arbitrary number of query nodes



Relaxation of constraints:

- Must be connected to all nodes in Q
- Connected to at least k nodes out of Q
- …

Definition CEPS

Given an edge-weighted undirected graph
and **Q nodes as source queries**

Find a suitably connected **subgraph H** that
contains **all query nodes**,
at most some number of other vertices,
and **maximizes** a goodness **function g(H)**

$$\mathbf{g(H)} = \sum_{j \in H} \boldsymbol{r(Q, j)}$$

- How to define reasonable scores r(Q,j)
- How to quickly find a connected subgraph H that maximizes g(H)

Focused Communities:
Given a Set of Seed Nodes

Traditional detection of **communities**
as **internally dense subgraphs**
(e.g. measured by modularity or conductance)

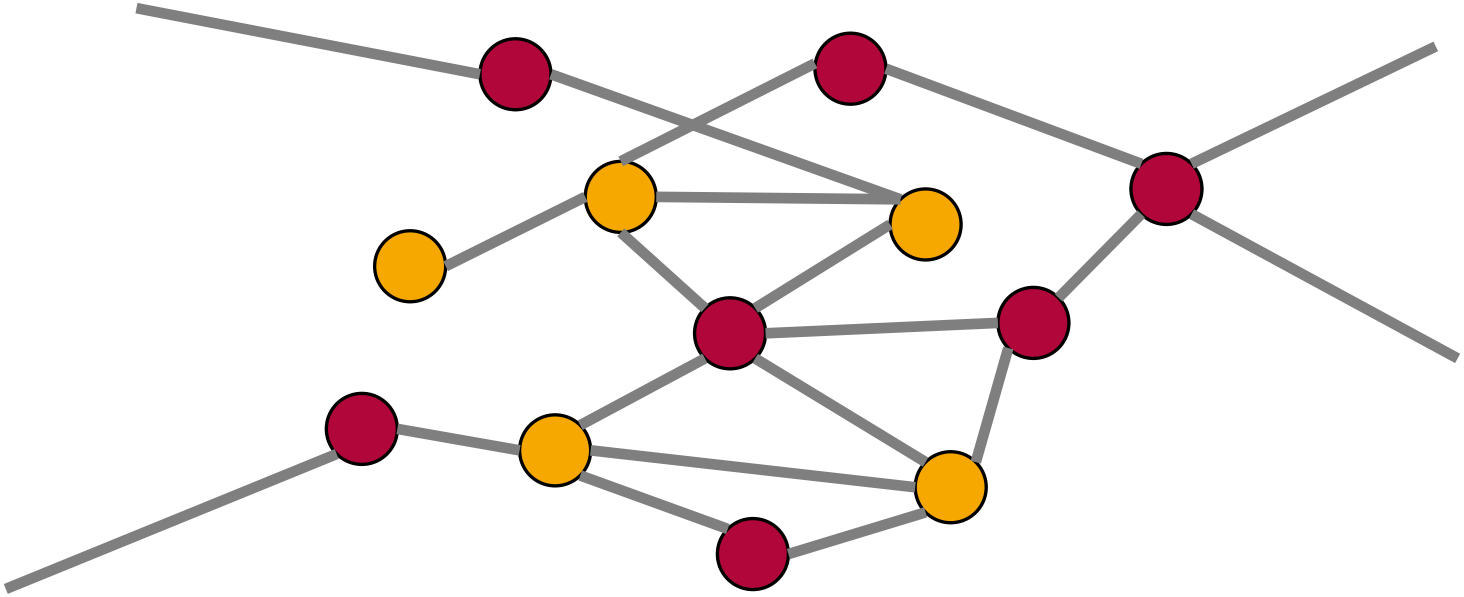
Given seed nodes (by the user)

Perform **selective search** for communities
local community detection
seed set expansion

- Global search is not appropriate for such local/selective models
- Communities may overlap or coincide

C. Staudt, Y. Marrakchi, H. Meyerhenke: Detecting Communities Around Seed Nodes in Complex Networks (BigData 2014)

SCD: Selective Community Detection



Scalability achieved by **greedy community expansion algorithm**.
Flexible framework allows instantiations with state-of-the-art algorithms such as PageRank-Nibble [1]

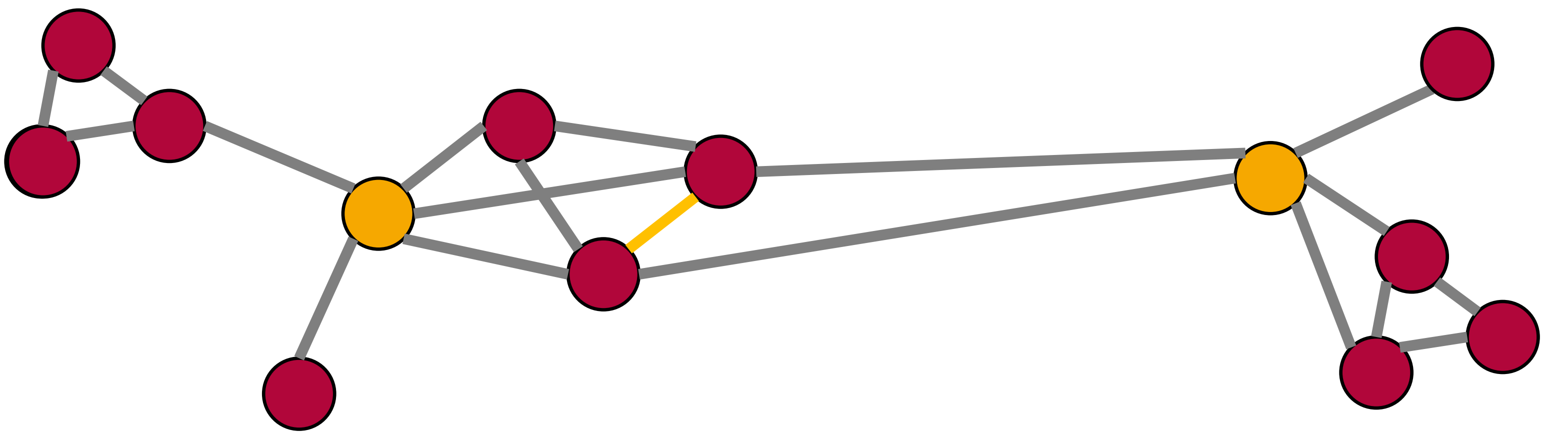
[1] Andersen, Chung, Lang: Local graph partitioning using pagerank vectors. (FOCS 2006)

Egoistic Focus on Yourself: Ego-Nets

For a given node
consider their neighbors and
the connections among these neighbors

Compute ego-nets for each given node that is of interest.

Useful for link prediction, community detection, anomaly detection, and many more, as pre-processing (feature extraction).

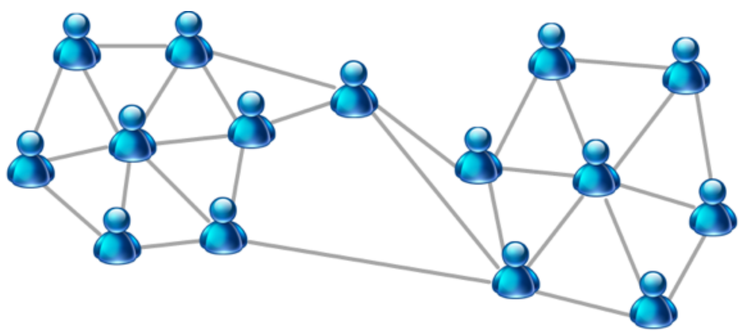


Epasto et al. Ego-Net Community Mining Applied to Fried Suggestion. (VLDB 2015)

Attributed and Weighted Graphs

Several application domains

- Communication networks, co-purchased networks, social networks



graph structure

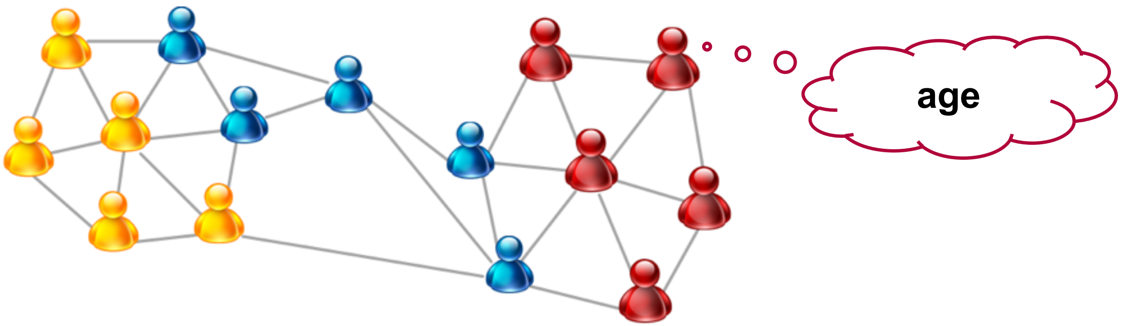
attributes

Novel challenges and opportunities on attributed graphs

Should user interest be modeled as one additional attribute?

Homophily: Commonly Used Assumption

Homophily: *„birds of a feather flock together"*

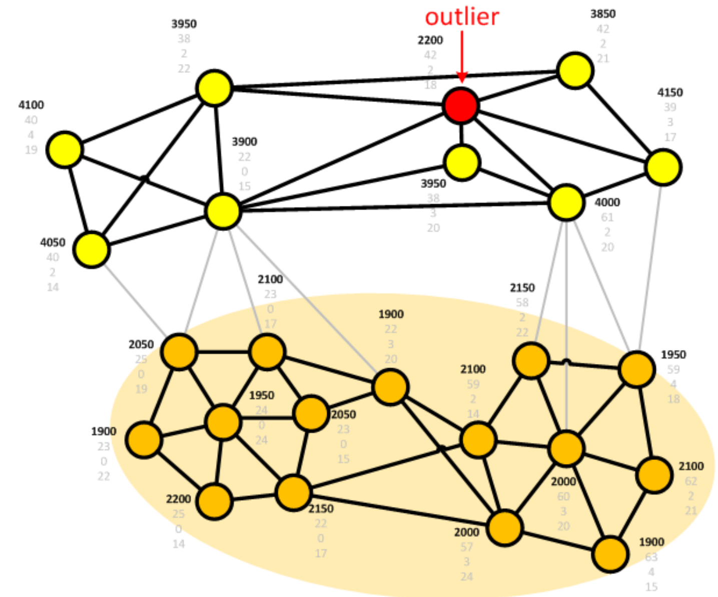


Homophily: not fullfilled for all attributes

Mining Attributed Graphs

Different graph mining techniques

- Clustering / graph partitioning / …
- **Community detection and anomaly detection**



Used assumption: **Homophily**
has to be fulfilled for **all** the attributes

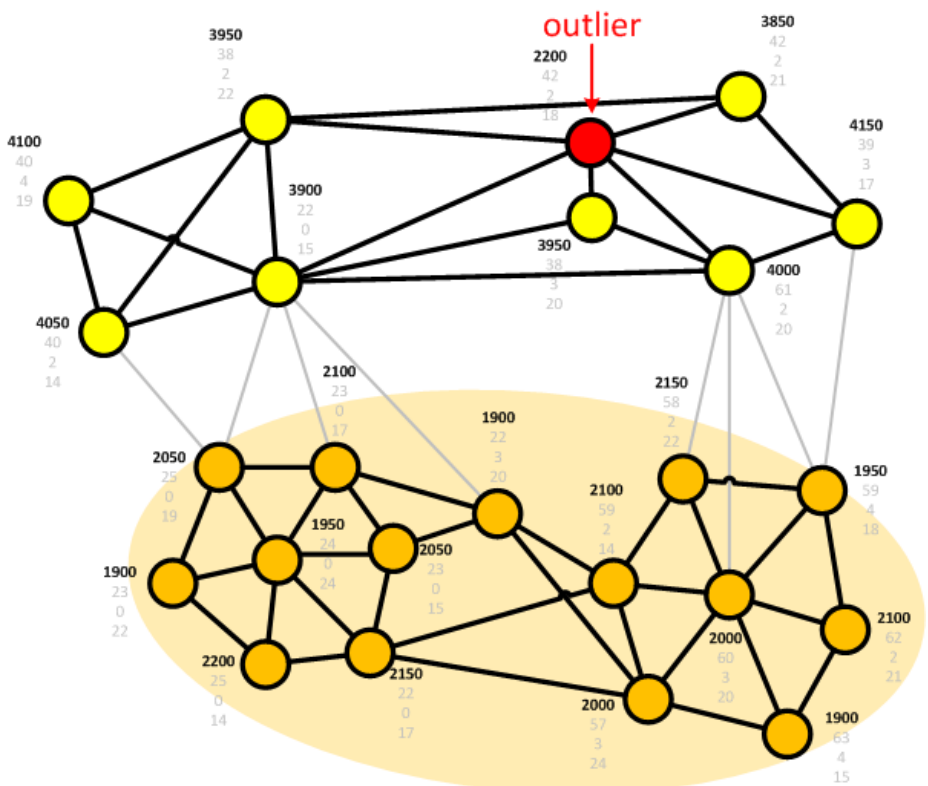
Problem: **disassortative mixing** [Newman 2003]
hinders the detection of communities
(i.e. similarity assessment of nodes)

Solution: Selection of relevant views ensuring homophily

Newman. Mixing patterns in networks. Physical Review, 2003

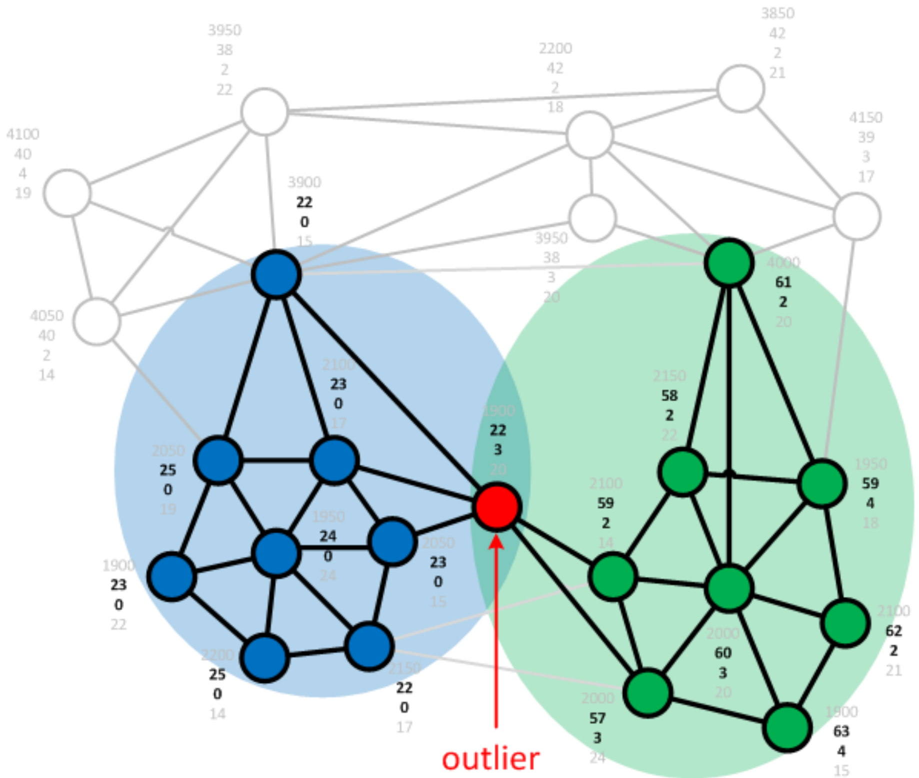
Multiple Views in Attributed Graphs

Different structures depending on the subset of attributes



Multiple Views in Attributed Graphs

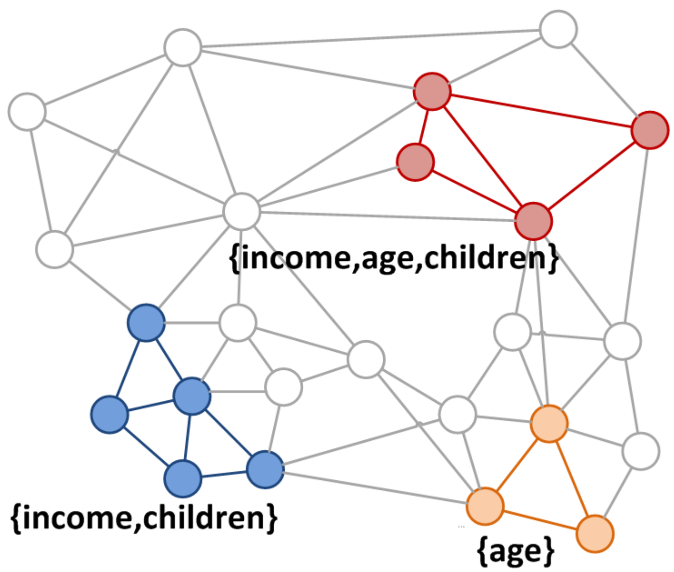
Different structures depending on the subset of attributes

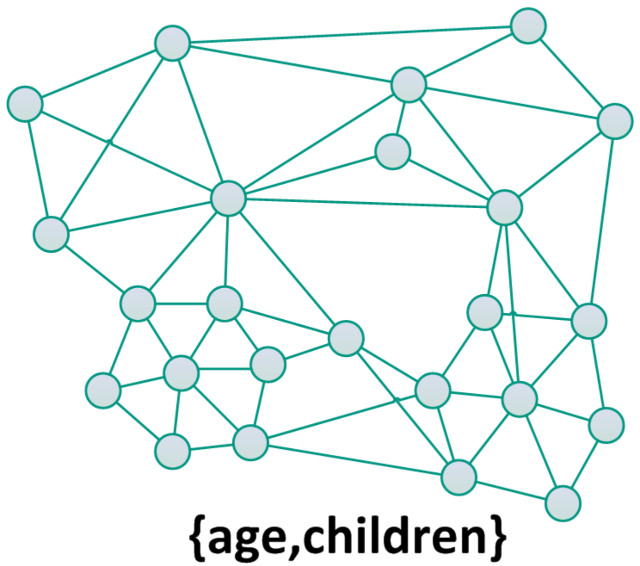


Specialized Approaches

Frequent subgraph mining, subspace clustering ...

- Local selection of the attributes
- Individual subgraphs



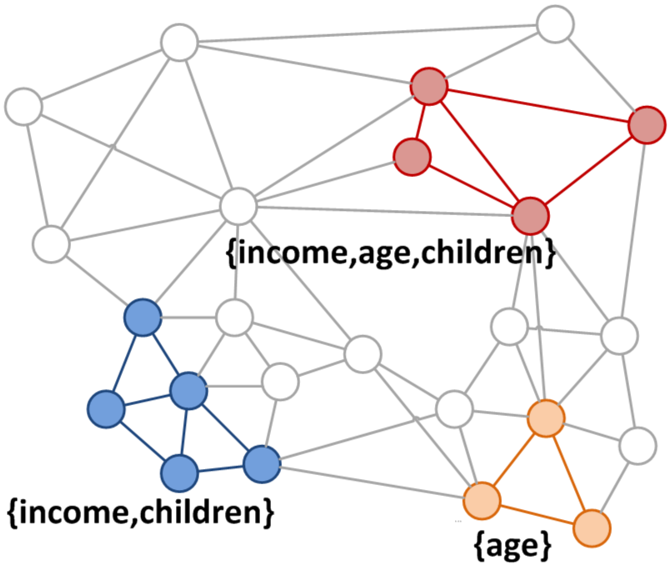


First Idea: Local Context Selection

Local Context:
- Subset of relevant attributes
- Selection w.r.t. a subgraph

How to **define a local context** for each node?

How to **efficiently** select only the **relevant attributes**?



Model dependent solution for community outlier mining
- Statistical test of attribute value distribution for each local context
- Measure deviation of each node w.r.t. its local context only

Iglesias et al. Local Context Selection for Outlier Ranking in Graphs with Multiple Numeric Node Attributes (SSDBM 2014)

Selection of Congruent Subspaces (ConSub)

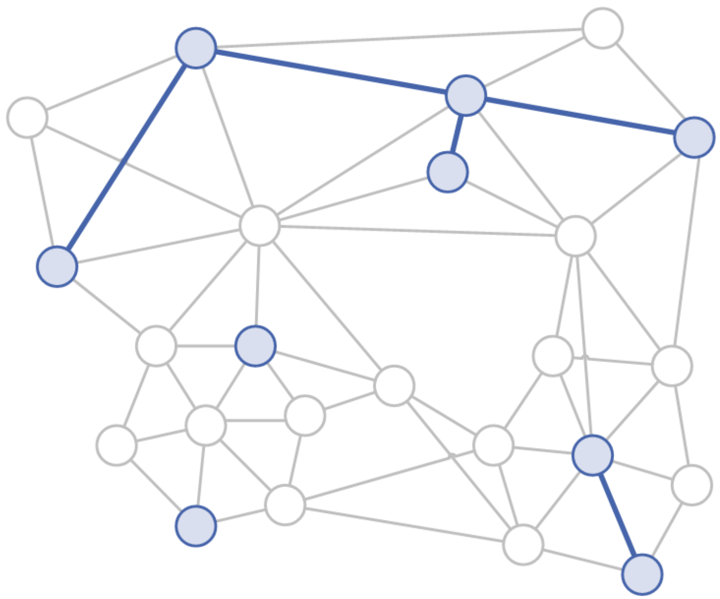
Definition: Congruent subspaces

- **Mutual similarity** between attribute values in subspace *S*
- **Significantly more edges** than expected by a random distribution

Constraint Subgraph $G_{C,S}$

- Set of constraints formed by all the pairs $(\boldsymbol{I_j} = [low_j, high_j],\ \boldsymbol{A_j} \in S)$

S = {shoe size}
nodes with **8 ≤ *shoe size* ≤ 9**



small number of edges

Iglesias et al. Statistical Selection of Congruent Subspaces for Mining Attributed Graphs (ICDM 2013)

Selection of Congruent Subspaces (ConSub)

Definition: Congruent subspaces

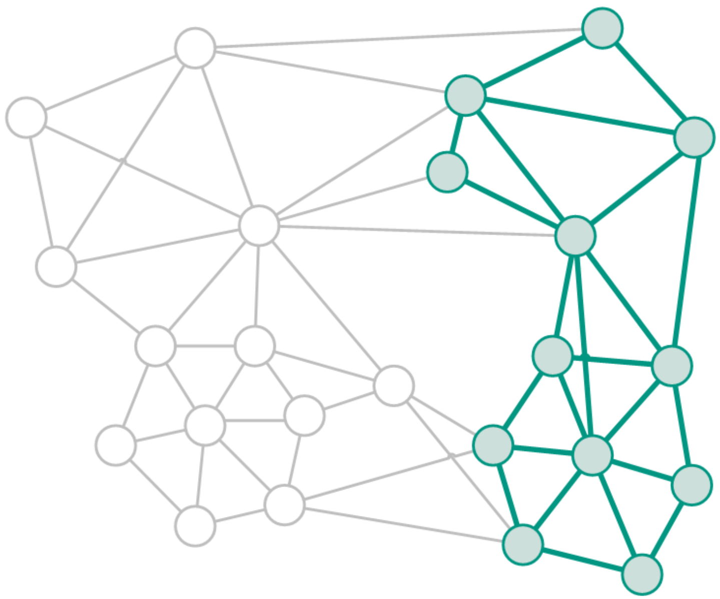
- **Mutual similarity** between attribute values in subspace *S*
- **Significantly more edges** than expected by a random distribution

Constraint Subgraph $G_{C,S}$

- Set of constraints formed by all the pairs $(\boldsymbol{I_j} = [low_j, high_j],\ \boldsymbol{A_j} \in S)$

S ={age,income}
nodes with **45 ≤ *age* ≤ 60** and
1900 ≤ *income* ≤ 4500

→ **high number of edges**

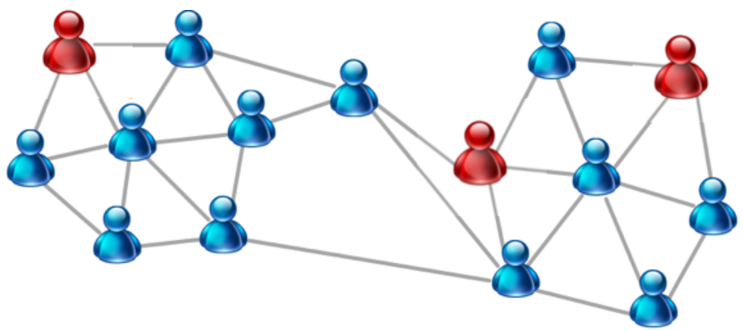


Iglesias et al. Statistical Selection of Congruent Subspaces for Mining Attributed Graphs (ICDM 2013)

Focus on User Preference

Examples for user preference:

- attribute weighting
- examples of similar nodes
- some notion of similarity



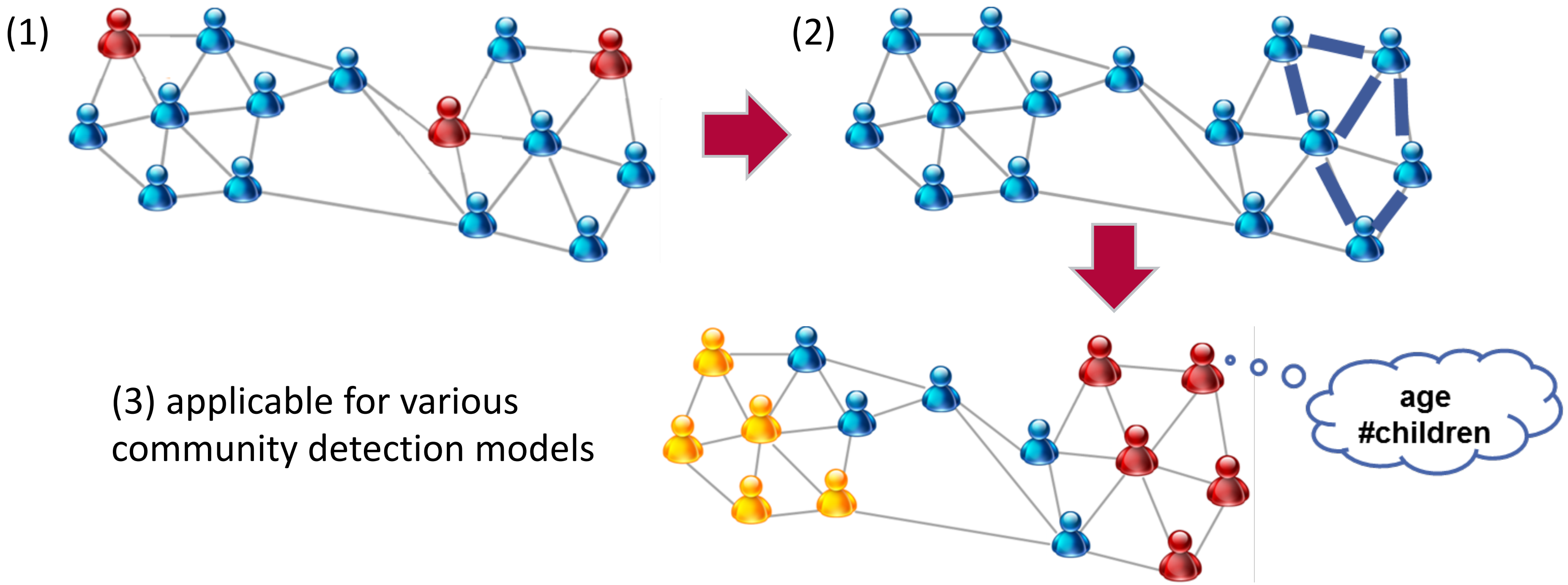
examples of similar nodes

attribute weighting

Focused Selection of Subsaces (FocusCO)

Decoupled mining for given user preference

1. Infer similarity measure
2. Re-weighting of graph edges
3. Community detection & community outlier mining



(3) applicable for various community detection models

Perozzi et al. Focused Clustering and Outlier Detection in Large Attributed Graphs (KDD 2014)

Knowledge Discovery by Focused Graph Mining

Example Sociology:

hypothesis testing vs. hypothesis generation

Let's break!

Tutorial outline

Background (5 min)

Graph models, subgraph isomorphism, subgraph mining, graph clustering

Exploratory Graph Analysis (35 min)

Focused Graph Mining (35 min)



Refinement of Query Results (35 min)

Real World-Use Case (15min)

Linked Data graphs

Challenges and discussion

Refinement of Graph Query Results

Reformulation and Refinement

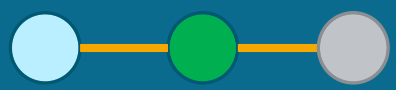
- Generate reformulations (explanations) for query with too-many too few results
- Explain results by providing summaries
- User perspective: even if the query is imprecise the system provides assistance

Top-k results

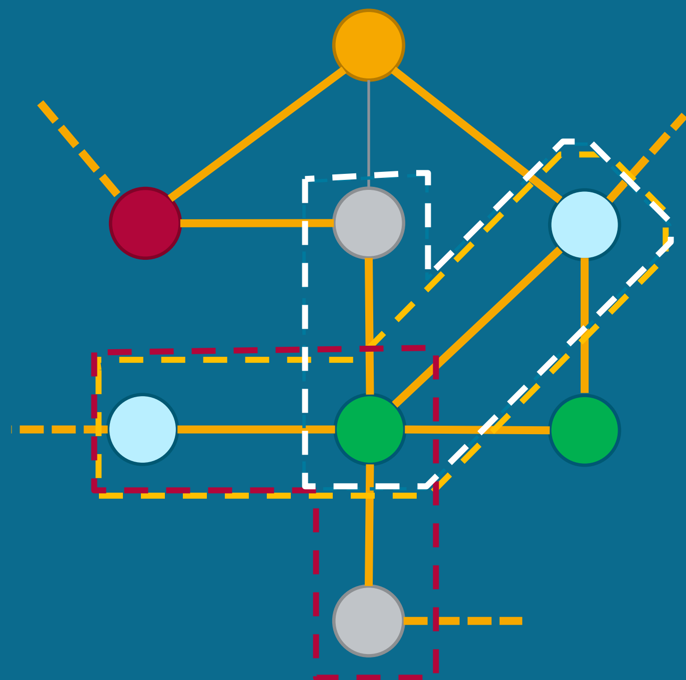
- Use user feedback to find the k results with the highest score
- User perspective: the results are potentially the most preferred items

Skyline queries

Reformulation and Refinement



Query (a graph)



- The user query is too restrictive (few results) or too generic (many results)

Solution

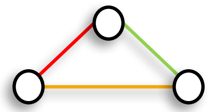
- Change the query to include more/less results

  OR
- Summarize the results

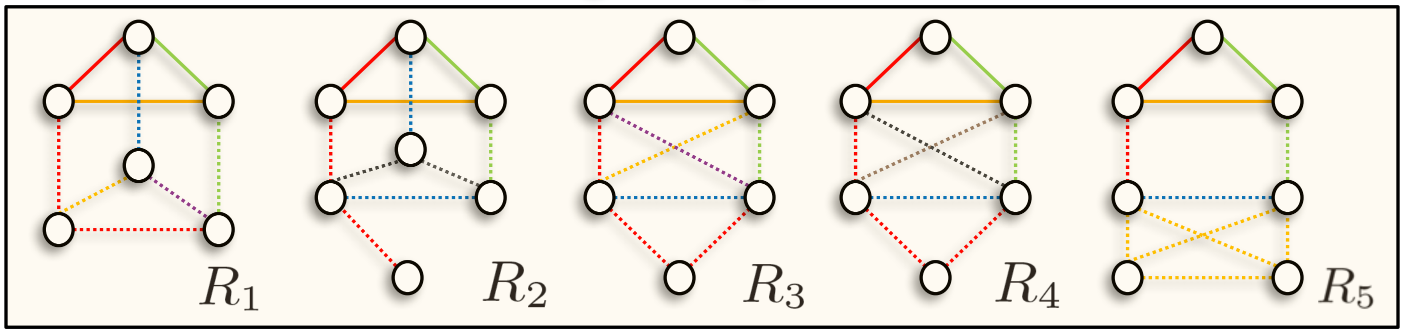
- Query Reformulation approaches: in Graph Databases (Mottin et al.), in connected networks (Vasilyeva et al.)
- Result summarization approaches: top-k representative (Ranu et al.), keyword induced result summarization (Wu et al.)

Graph Query Reformulation

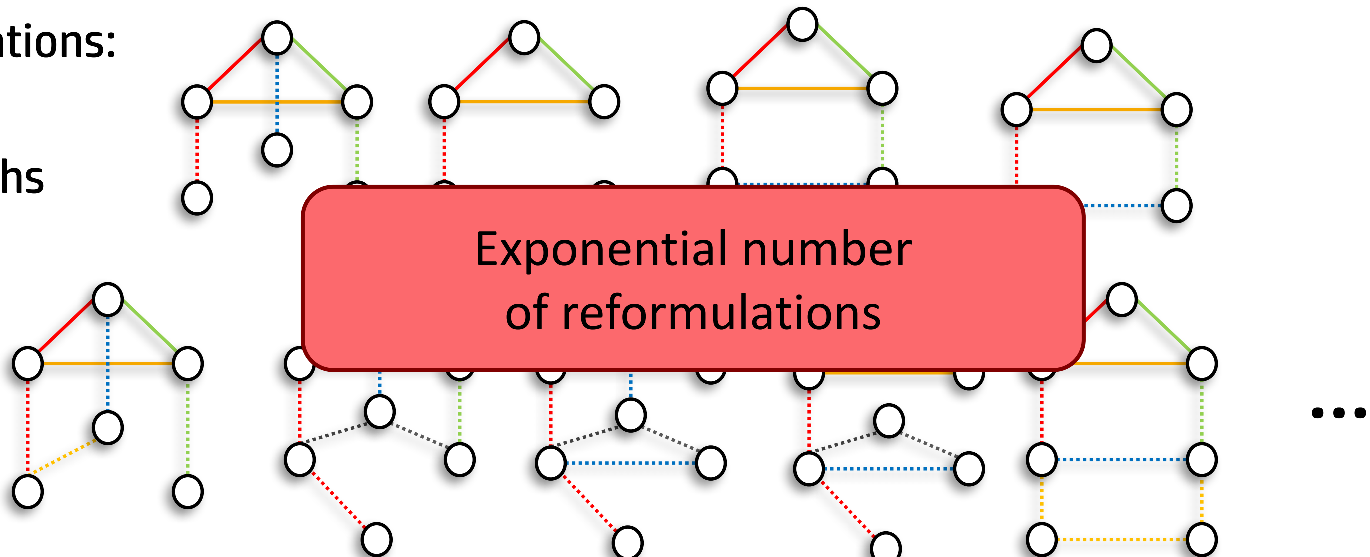
Query



Results



Reformulations:
query
supergraphs



Mottin, D., Bonchi, F. and Gullo, F. Graph Query Reformulation with Diversity. KDD, 2015

Graph Query Reformulation

Find **k meaningful** reformulations:

1. Span **all** the results
2. Present **different** aspects of the results

?

Query

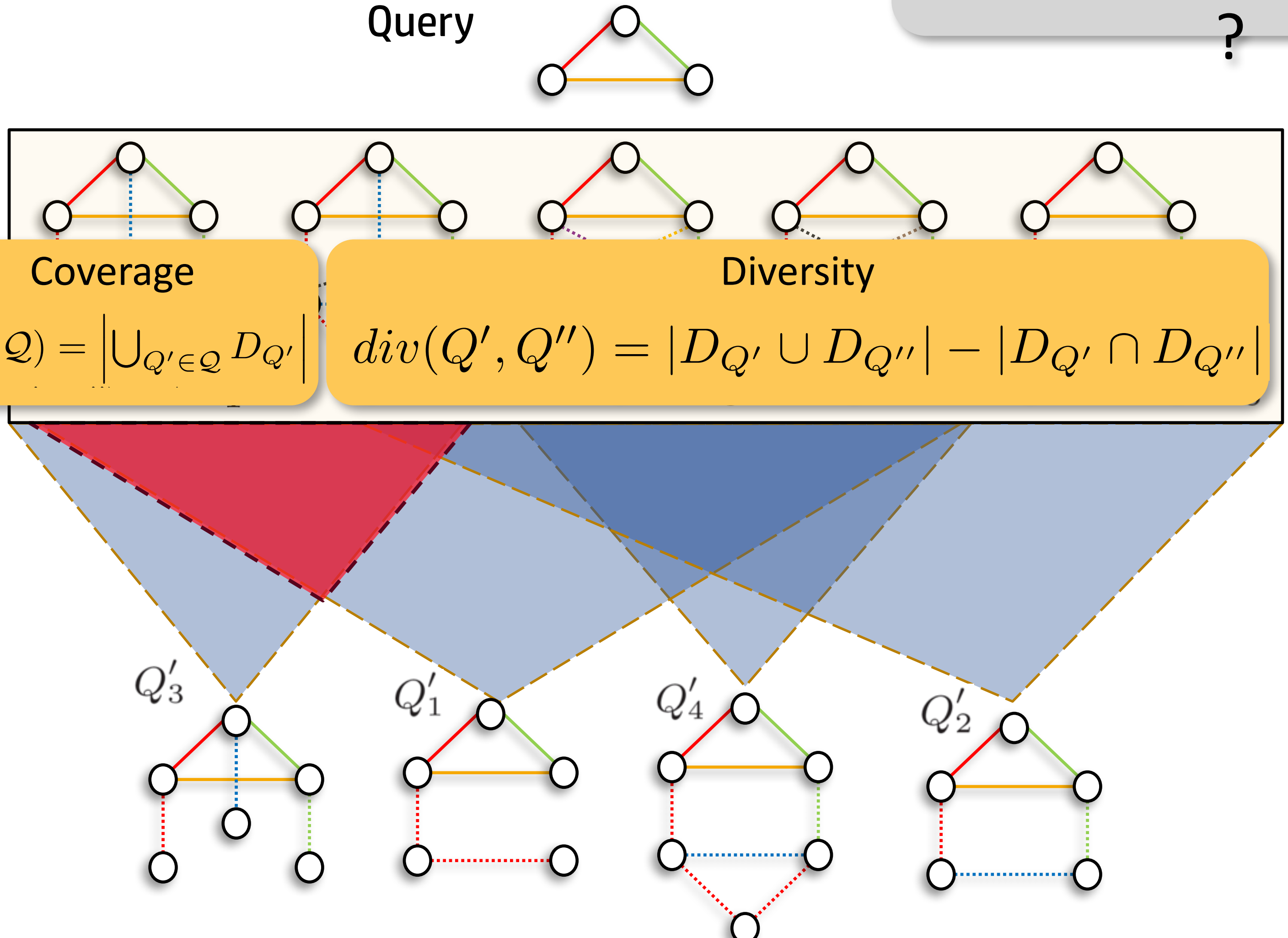
Results

Coverage

$$cov(\mathcal{Q}) = \left|\bigcup_{Q' \in \mathcal{Q}} D_{Q'}\right|$$

Diversity

$$div(Q', Q'') = |D_{Q'} \cup D_{Q''}| - |D_{Q'} \cap D_{Q''}|$$



Mottin, D., Bonchi, F. and Gullo, F. Graph Query Reformulation with Diversity. KDD, 2015

Graph Query Reformulation

#P-complete

Problem

Find a set $\mathcal{Q}$ of k reformulations that maximize a combination of **coverage** and **diversity**

$$f(\mathcal{Q}) = cov(\mathcal{Q}) + \lambda \sum_{Q',Q'' \in \mathcal{Q}} div(Q', Q'')$$

$$\mathcal{Q}^* = \underset{\mathcal{Q} \subseteq \mathbb{S}_Q}{\arg\max} \quad f(\mathcal{Q})$$

$$\text{subject to} \qquad |\mathcal{Q}| = k.$$

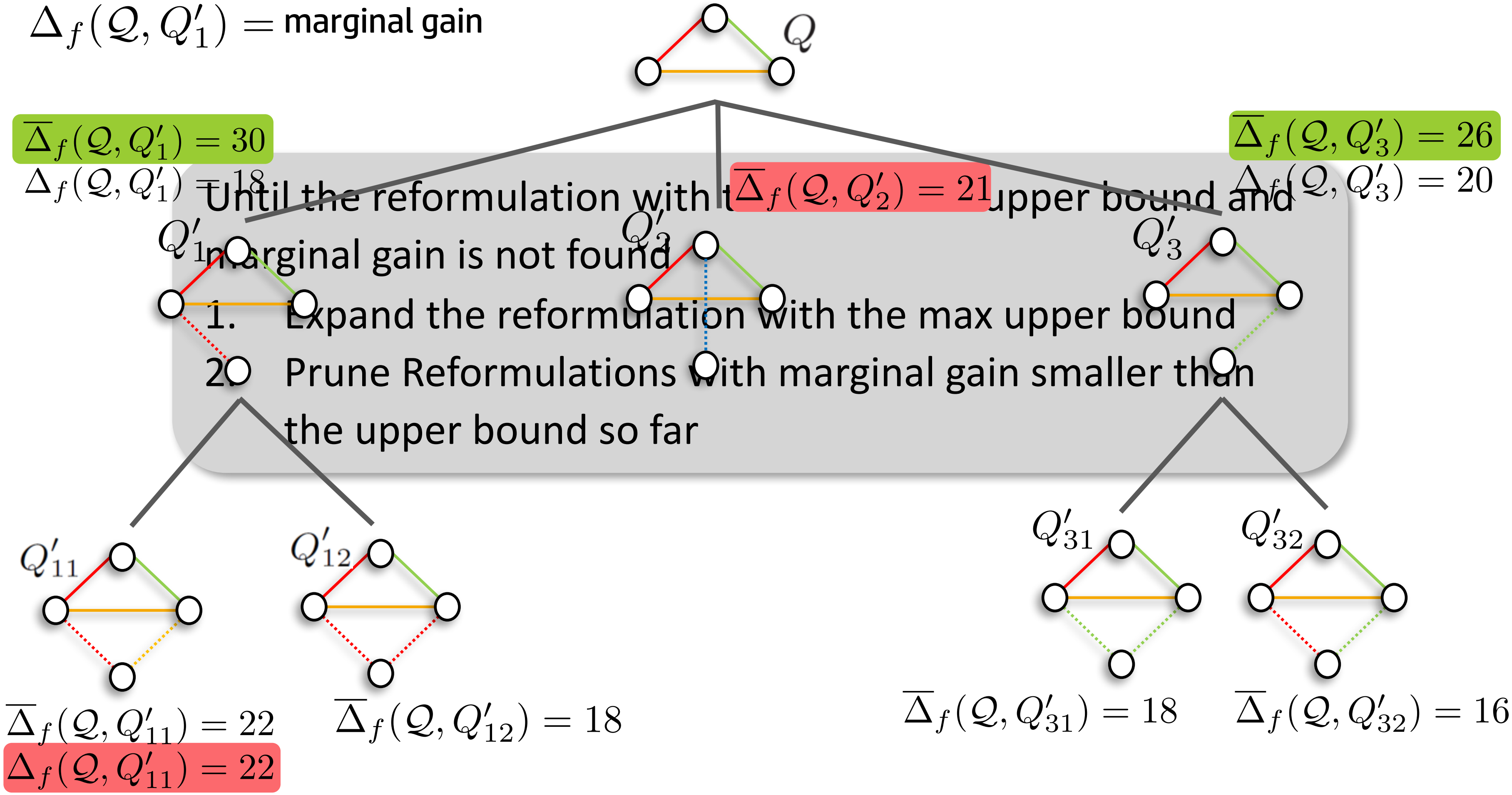
Theorem (NP-hardness)
The problem reduces to **MAX-SUM Diversification** Problem, so it is NP-hard

Mottin, D., Bonchi, F. and Gullo, F. Graph Query Reformulation with Diversity. KDD, 2015

The Fast_MMPG Algorithm

$\overline{\Delta}_f(\mathcal{Q}, Q'_1) =$ upper bound

$\Delta_f(\mathcal{Q}, Q'_1) =$ marginal gain

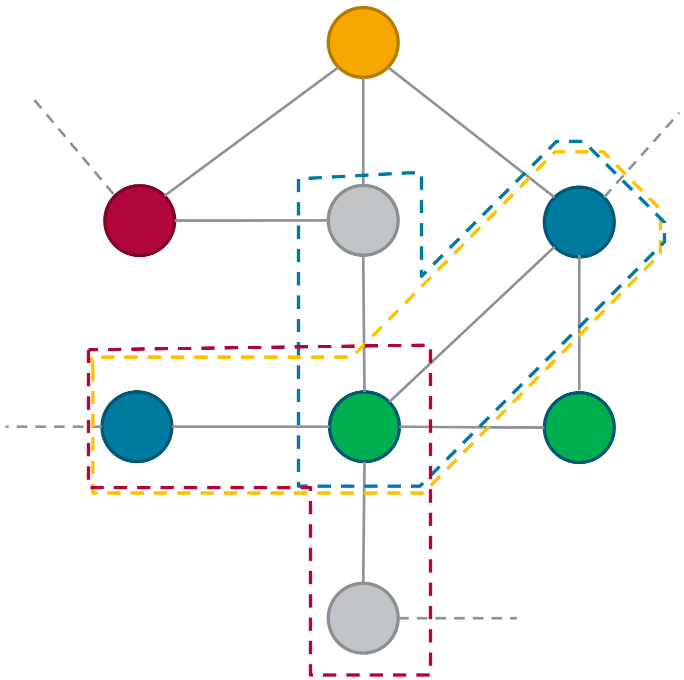


Until the reformulation with the upper bound and marginal gain is not found

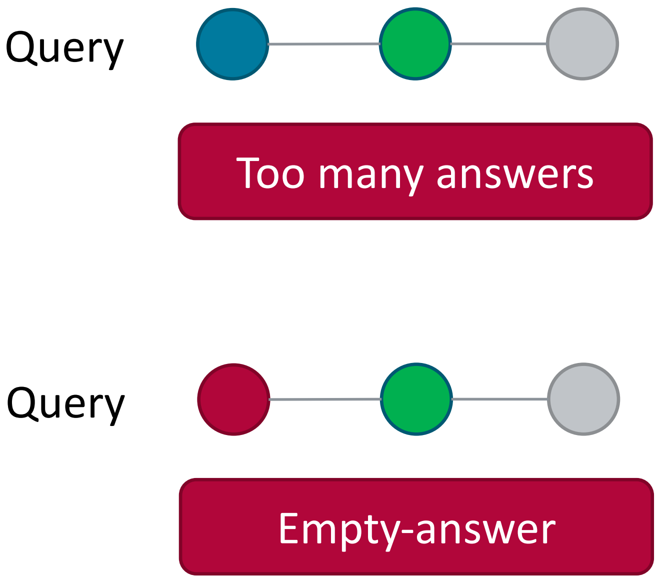
1. Expand the reformulation with the max upper bound
2. Prune Reformulations with marginal gain smaller than the upper bound so far

Mottin, D., Bonchi, F. and Gullo, F. Graph Query Reformulation with Diversity. KDD, 2015

Why empty, Why so-many answers in graphs



Large graph



Problem
Given a query Q and a graph G, restrict/enlarge the result set with minimal changes in the query.

Vasilyeva, E., Thiele, M., Bornhövd, C. and Lehner, W.. Answering “Why Empty?” and “Why So Many?” queries in graph databases. *JCSS, 2016*

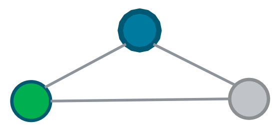
Why empty, Why so-many answers in graphs

Why?
Empty/Too Many

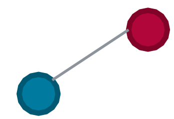
Change the query

Exponential variations!

Explanations



+

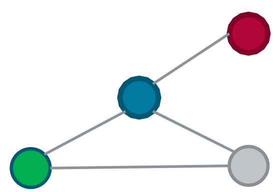


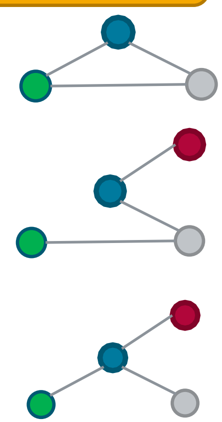
Maximum Common Subgraph

Differential graph

Graphs and unexpected subgraphs

Modifications

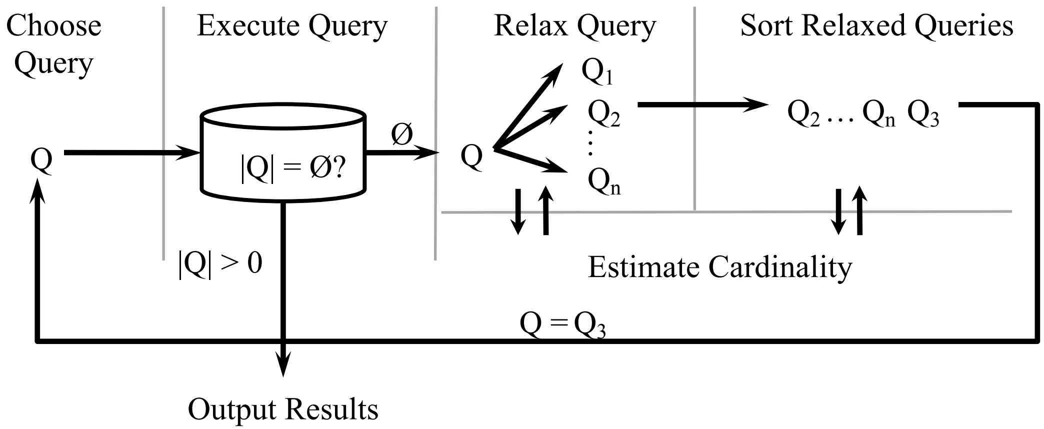




Answers to the new queries

Vasilyeva, E., Thiele, M., Bornhövd, C. and Lehner, W.. Answering "Why Empty?" and "Why So Many?" queries in graph databases. *JCSS, 2016*

Why empty, Why so-many answers in graphs



Cardinality estimation:

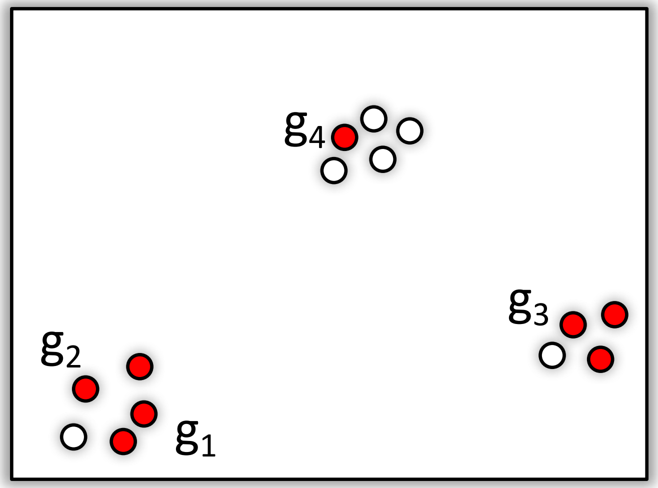
- Frequency of single edges
- Entropy

Generate candidates based on minimal modifications

Vasilyeva, E., Thiele, M., Bornhövd, C. and Lehner, W.. Answering “Why Empty?” and “Why So Many?” queries in graph databases. *JCSS, 2016*

Top-k representative queries

Graphs in a multidimensional space

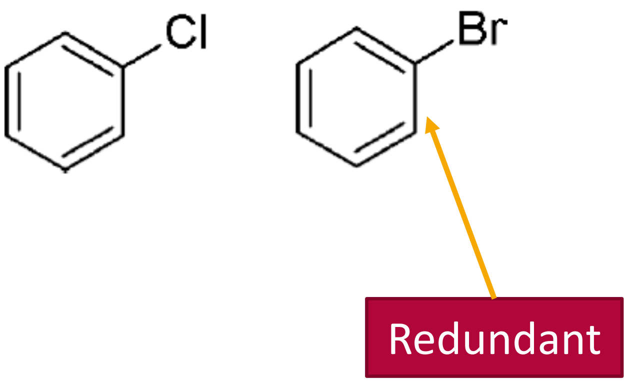


- Object is *relevant*
- Object is non-relevant

Two objects are close if they are similar

Select k=2 relevant objects

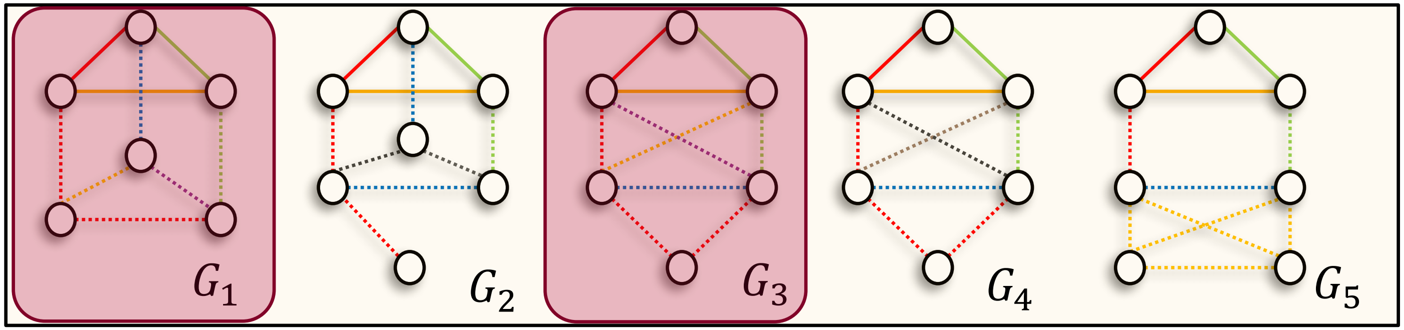
Top-*2* answer: g_1, g_2



Ranu, S., Hoang, M. and Singh, A. Answering top-k representative queries on graph databases. SIGMOD, 2014

Top-k representative queries

Result of a query



Vector graph $\vec{g}_i$: vectorial representation of G_i

Example: Binding compatibility with m proteins, frequent subgraphs, belonged communities

Query: function from $\vec{g}$ to [-1,1], $q: \vec{g} \rightarrow [-1,1]$

Example: Molecules with some properties, graphs with some structure, some community

Top-k Representative queries:

$$A = \arg\max_{S}\{\pi_\theta(S) | S \subseteq R(q), |S| = k\}$$

where $R(q)$ = results of q, $\pi_\theta(S)$=**representative power** of S, given threshold θ

Ranu, S., Hoang, M. and Singh, A. Answering top-k representative queries on graph databases. SIGMOD, 2014

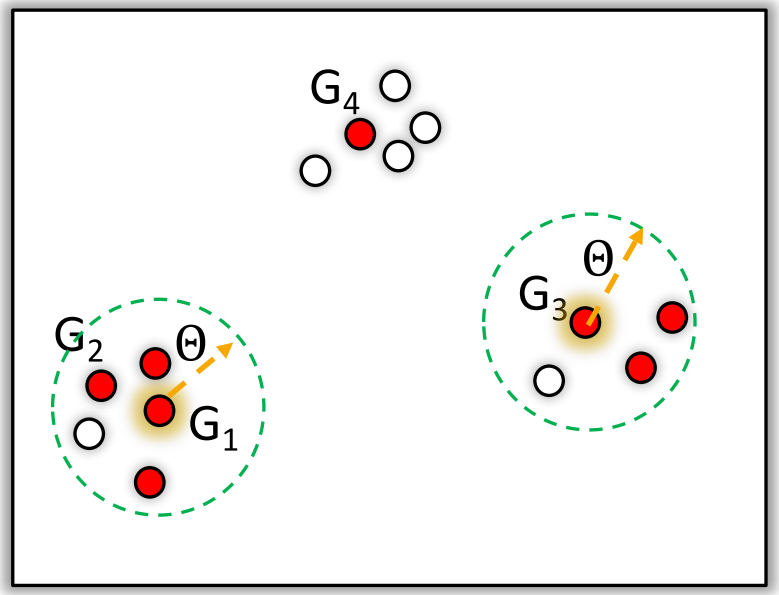
Representative power

R(q) = answers to the query

- q : query

θ-neighborhood

- $N_\theta(G) = \{G' \in R(q) | d(G, G') \leq \theta\}$
- θ: distance threshold
- $d(G, G')$: graph edit distance



Given a set of graphs S

- Representative power of S
- $\pi_\theta(S) = \frac{|\bigcup_{G \in S} N_\theta(G)|}{R(q)}$

Represent the coverage of a graph neighborhood

$$\pi(\{G_1, G_3\}) = \frac{7}{8}$$

$$\pi(\{G_1, G_2\}) = \frac{4}{8}$$

Ranu, S., Hoang, M. and Singh, A. Answering top-k representative queries on graph databases. SIGMOD, 2014

Greedy algorithm

1-1/e-Approximation

A=∅

$$G^* = \arg\max_G \{Score(A \cup \{G\}) - Score(A)\}$$

Repeat untill |A|=*k*

A= A ∪ {G*}

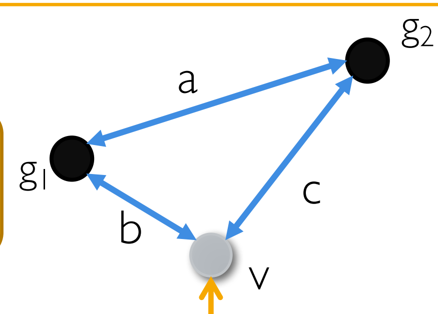
NP-hard (graph edit dist)

Update representative powers of ALL graphs

$$N_\theta(G') = N_\theta(G') \backslash N_\theta(G^*)$$

PTIME

Indexed with vantage points and clustering



Vantage point

$$|b - c| \leq a \Rightarrow$$

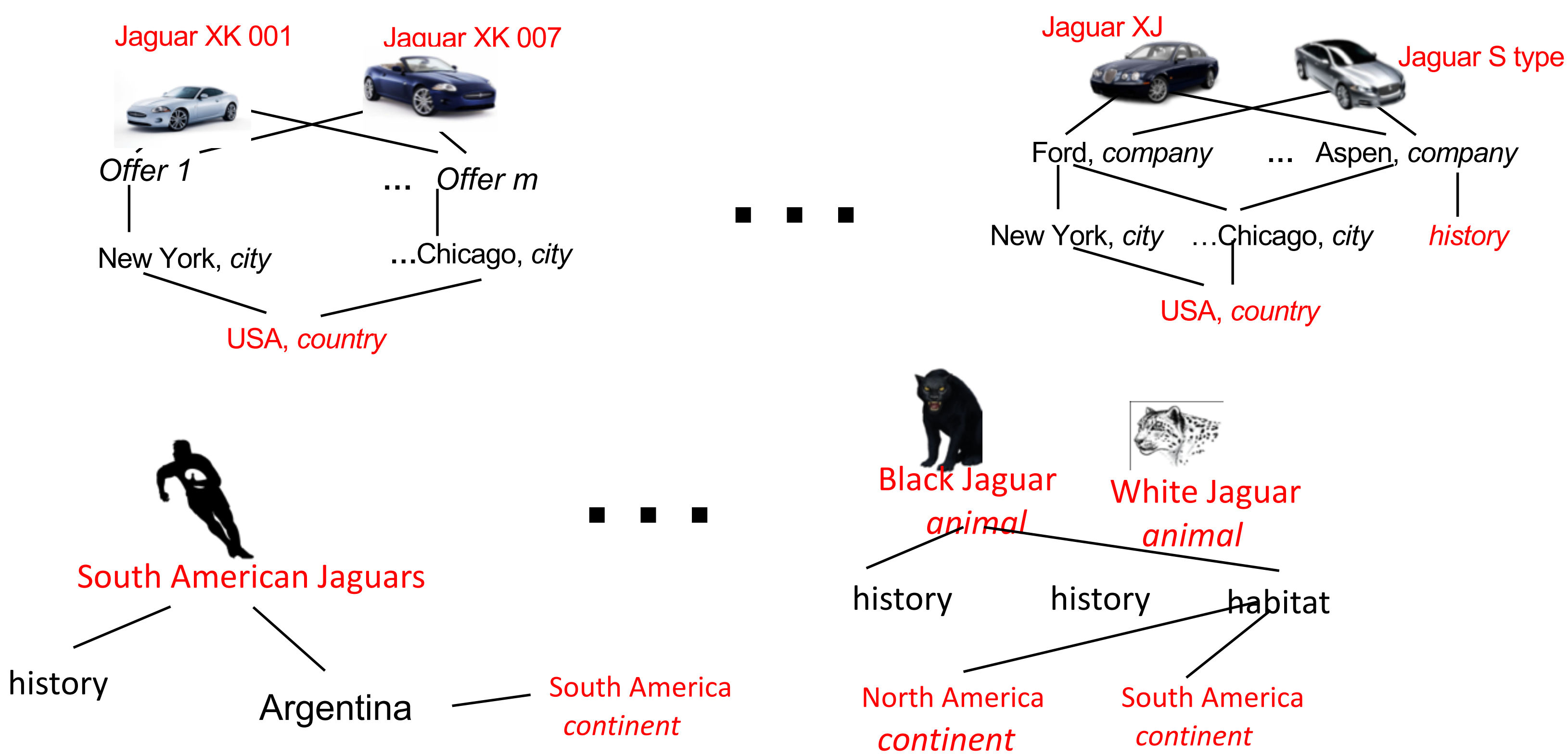
$$d_v(g_1, g_2) = |d(v, g_1) - d(v, g_2)|$$

If $d_v(g_1, g_2) > \theta, g_2 \notin N_\theta(g_1)$

Ranu, S., Hoang, M. and Singh, A. Answering top-k representative queries on graph databases. SIGMOD, 2014

Summarizing graph results

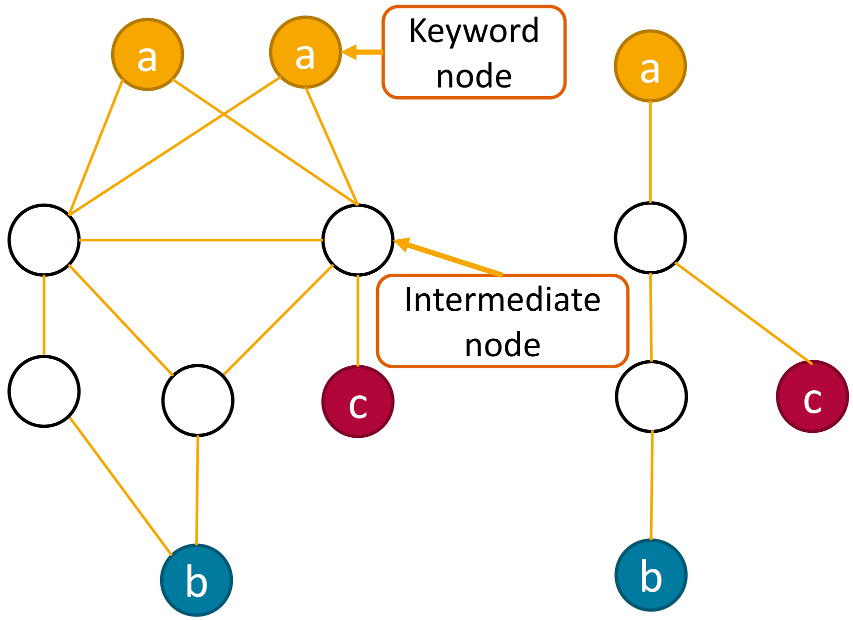
Query: keyword query on graph e.g., Jaguar, America, History



Wu, Y., Yang, S., Srivatsa, M., Iyengar, A. and Yan, X. Summarizing answer graphs induced by keyword queries. *PVLDB, 2013*

Summarizing graph results

$Q = \{a,b,c\}$



Answer graph

Summary graph

Answer graph: keyword nodes and intermediate nodes

Summary graph Gs:

- Preserve connections between keyword nodes
- Each node is a hypernode
- For any path in Gs there is a path in the union of answer graphs with the same label

Quality of a summary (coverage)
$\alpha = 2 * M/(|Q|(|Q|-1),$
M = number of covered keyword pairs

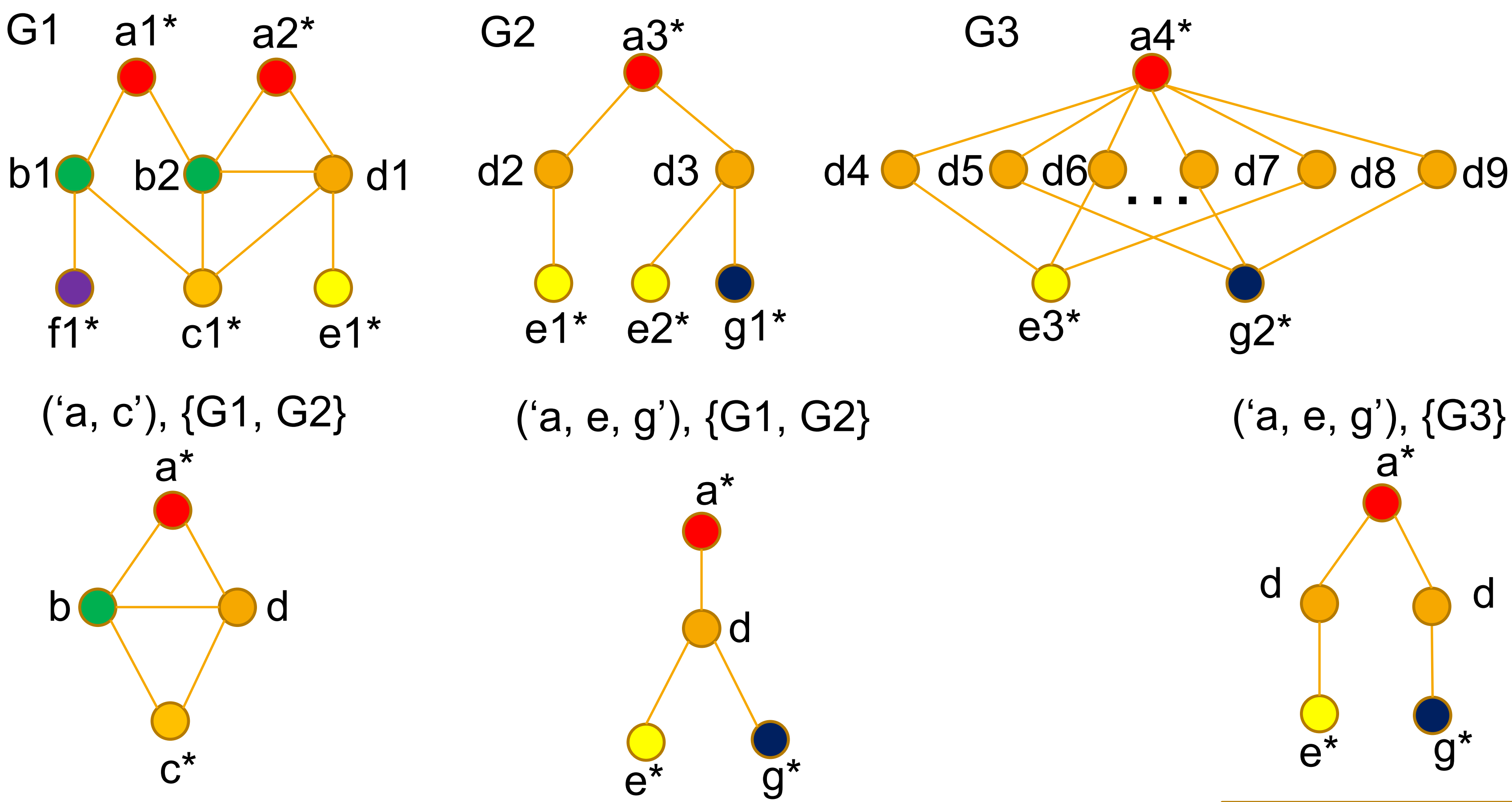
Two problems

1. Minimum α-summarization: find the **minimum size** summary which covers at least α
2. K-summarization: find K 1-summaries with minimum total size that form a K-partition on the answer graph sets (no repeated answers)

Wu, Y., Yang, S., Srivatsa, M., Iyengar, A. and Yan, X. Summarizing answer graphs induced by keyword queries. *PVLDB, 2013*

Summarizing graph results

Q = {a,c,e,f,g}



0.1-summary Gs1

0.3-summary Gs2

1-summary Gs3

Summarizing graph results algorithms

PTIME

1-summarization

1. Based on dominance relation: a node n1 dominates n2 if they have the same label and each path from a keyword pair that contains n2 also contains n1
2. Discover dominance relation and remove dominated nodes until no change

NP-complete

α-summarization

1. Greedy heuristic: compute 1-summaries for all keyword paths
2. Merge summaries with the minimum merge cost (extra edges added)
3. Repeat until the desired α is reached

NP-complete

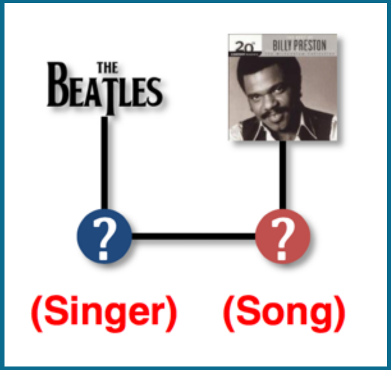
K-summarization

1. Select K answer graphs as centers
2. Refine the clusters merging answer graphs with minimum merge cost until convergence
3. Compute 1-summary graphs for each cluster

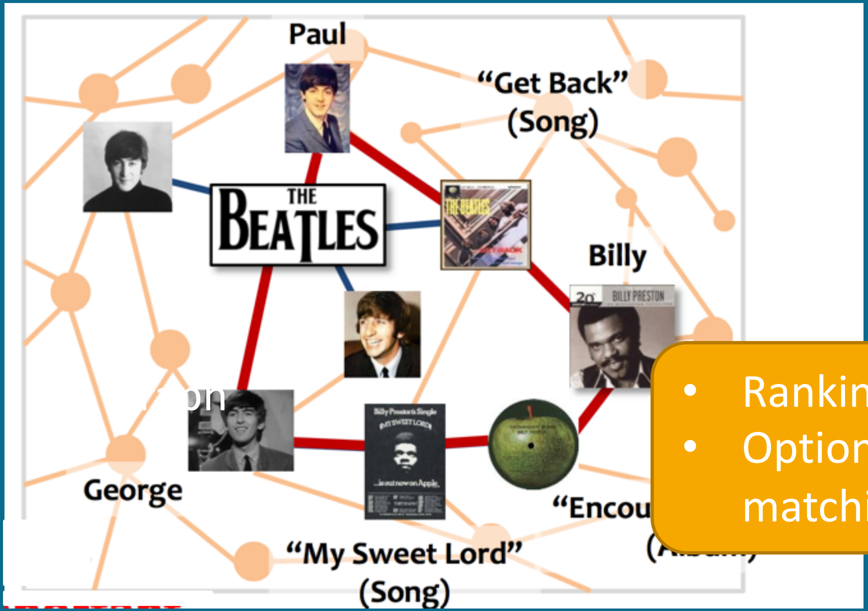
Wu, Y., Yang, S., Srivatsa, M., Iyengar, A. and Yan, X. Summarizing answer graphs induced by keyword queries. *PVLDB, 2013*

Top-k Results

- Large query results
- Find interesting exact and similar matches



Query



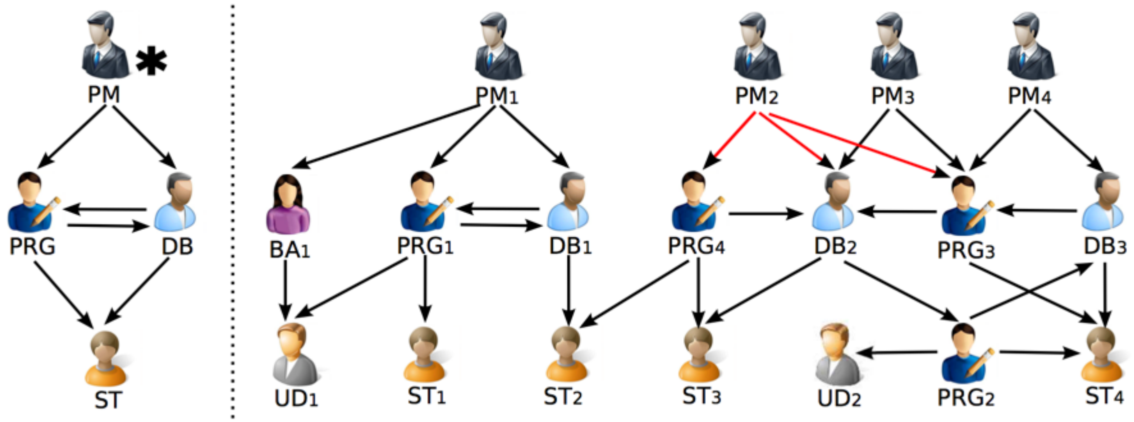
Solution

- Ranking the results
- Optionally diversifying the matching

- Diversified top-k graph pattern matching (Fan et al.)
- Exploiting relevance feedback in knowledge graph search (Su et al.)
- Top-k interesting subgraph discovery in information networks (Gupta et al.)
- Querying web-scale information networks through bounding matching scores (Jin et al.)

Diversified top-k graph pattern matching

Query:
Find good PM (project manager) candidates collaborated with PRG (programmer), DB (database developer) and ST (software tester).



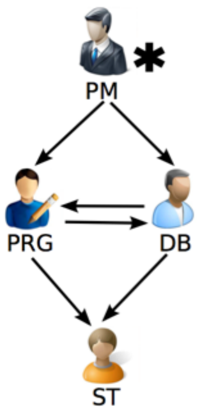
Pattern Q

Graph G

Find matches using graph simulation, which computes a binary relation on the pattern nodes in Q and their matches in G

Fan, W., Wang, X. and Wu, Y. Diversified top-k graph pattern matching. *VLDB*, 2013

Diversified top-k graph pattern matching



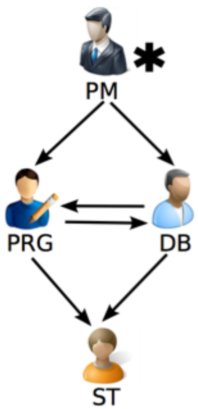
Pattern Q

- Graph pattern matching revised
  - extend a pattern with a designated output node u_0
  - matches Q(G): the matches of u_0
  - readily extends to multiple output nodes

- Problem:
  - Find (diversified) top-K matches for graph pattern matching with a designated output node.

Fan, W., Wang, X. and Wu, Y. Diversified top-k graph pattern matching. *VLDB*, 2013

Diversified top-k graph pattern matching



Pattern Q

- Relevance
  - Relevant set R(u,v) for a match v of a query node u:
  - all descendants of v as matches of descendants of u
  - a unique, maximum relevance set
  - Relevance function
    - The more reachable matches, the better

$$\delta_r(u, v) = |R_{(u,v)}|$$

- Top-k matching:
  - find top-k match set that maximizes total relevance

$$\delta_r(S) = \underset{S' \subseteq M_u(Q,G,u_o), |S'|=k}{\arg\max} \sum_{v_i \in S'} \delta_r(u_o, v_i)$$

Fan, W., Wang, X. and Wu, Y. Diversified top-k graph pattern matching. *VLDB*, 2013

Match Diversification

Match diversity

- Diversity function: set difference of the relevant set

$$\delta_d(v_1, v_2) = 1 - \frac{|R_{(u,v_1)} \cap R_{(u,v_2)}|}{|R_{(u,v_1)} \cup R_{(u,v_2)}|}$$

Diversification: a bi-criteria combination of both relevance and diversity

$$F(S) = (1-\lambda) \sum_{v_i \in S} \delta'_r(u_o, v_i) + \frac{2 \cdot \lambda}{k-1} \sum_{v_i \in S, v_j \in S, i<j} \delta_d(v_i, v_j)$$

- relevance: common neighbors, Jaccard coefficient…
- diversity: neighborhood diversity, distance-based diversity

Diversified Top-k Matching: find a set S of matches for output node

$$F(S) = \underset{S' \subseteq M_u(Q,G,u_o)}{\arg\max} F(S')$$

Finding Top-k Diversified Matches

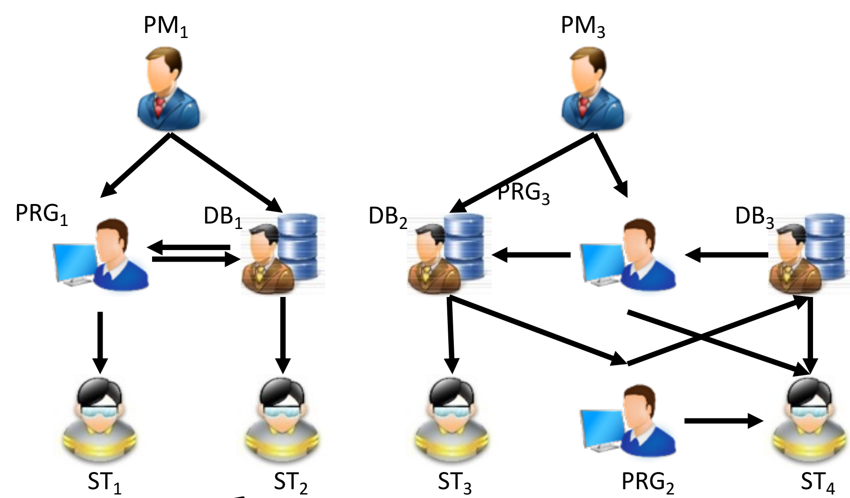
V	R(u_o, v)	δ_r ()
PM_1	{PRG_1, DB_1, ST_1, ST_2}	4
PM_2	{PRG_4, PRG_3, PRG_2, DB_2, DB_3, ST_2, ST_3, ST_4}	8
PM_3	{PRG_3, PRG_2, DB_2, DB_3, ST_3, ST_4}	6
PM_4	{PRG_3, PRG_2, DB_2, DB_3, ST_3, ST_4}	6

δ_d ()	PM_1	PM_2	PM_3	PM_4
PM_1	0	10/11	1	1
PM_2	10/11	0	1/4	1/4
PM_3	1	1/4	0	0
PM_4	1	1/4	0	0

PM_1 and PM_3 are picked by TopKDiv as top-2 diversified matches.

F()	PM_1	PM_2	PM_3	PM_4
PM_1		1.45	1.45	1.45
PM_2	1.45		0.89	0.89
PM_3	1.45	0.89		0.55
PM_4	1.45	0.89	0.55	

F'(PM1, PM3)=0.5*(4/11+6/11) + 1 = 1.45

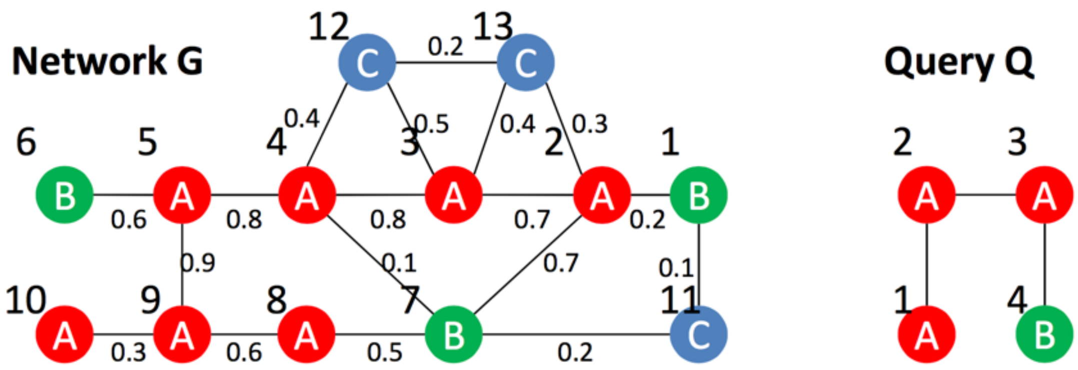


PM1 and PM3 have no descendant matches in common, and influence a large part of the matches.

Top-k interesting subgraph discovery in information networks

- Given
  - Typed unweighted query
  - A heterogeneous edge-weighted information network
  - Edge interestingness measure

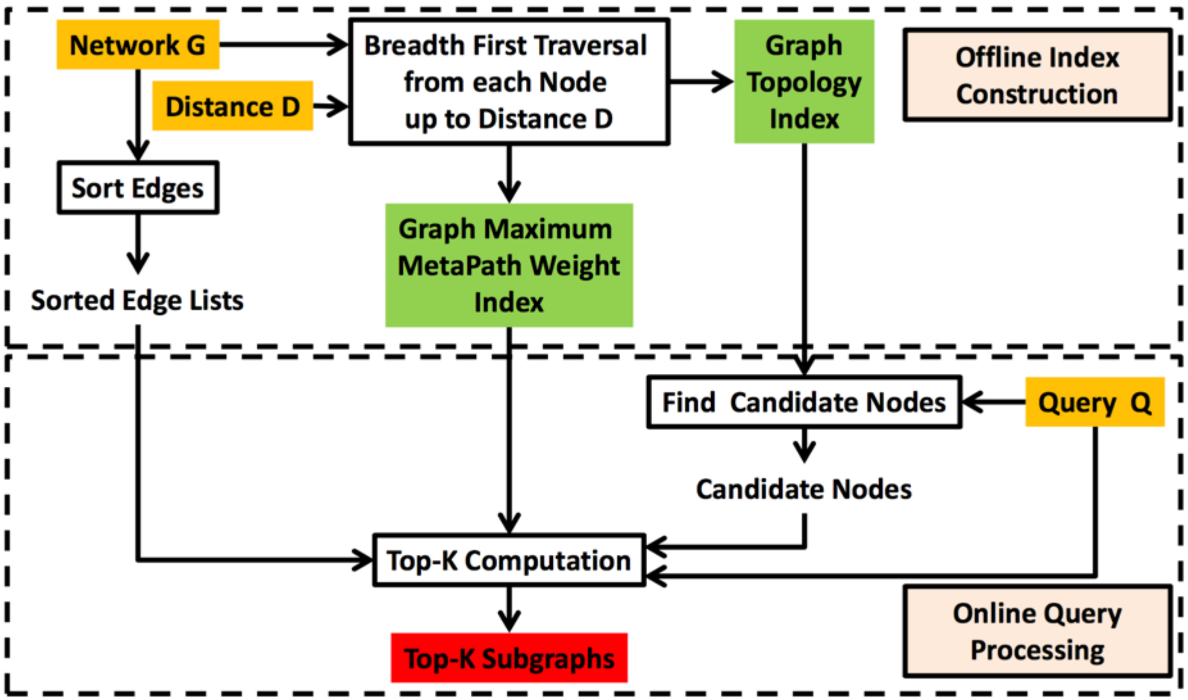
- Find
  - Top-k interesting subgraphs



Gupta, M., Gao, J., Yan, X., Cam, H. and Han, J. Top-k interesting subgraph discovery in information networks. ICDE, 2014

Top-k interesting subgraph discovery in information networks

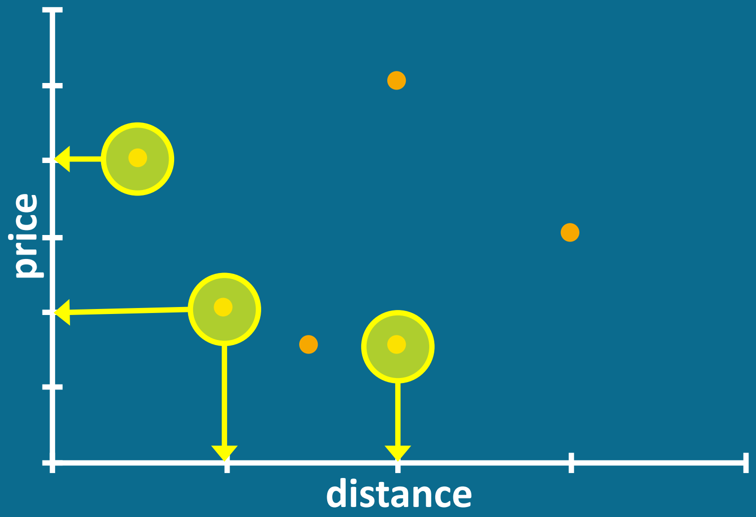
- 3 new graph indexes for building a top-k solution
  - Graph topology index
  - Sorted edge lists
  - Graph maximum metapath weight index



Gupta, M., Gao, J., Yan, X., Cam, H. and Han, J. Top-k interesting subgraph discovery in information networks. ICDE, 2014

Skyline Queries

- Prune a search space of large numbers of multi-dimensional data items to a small set of interesting items



Solution

- Eliminating items that are dominated by others

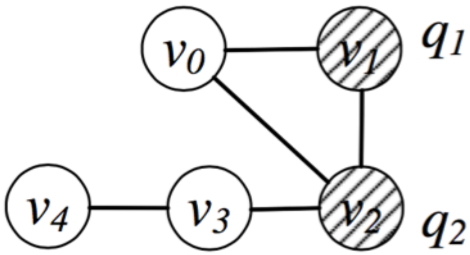
- Dynamic skyline queries in large graphs (Zou et al.)
- Efficient subgraph skyline search over large graphs (Zheng et al.)

Dynamic skyline queries

- Users can specify different sets of query points
  - ➢Offer users more flexibility in specifying their search criteria
- Skylines are dynamically updated

- Naïve approach:
  - Computing all new vectors according to the query points and then searching the skylines over the generated vectors

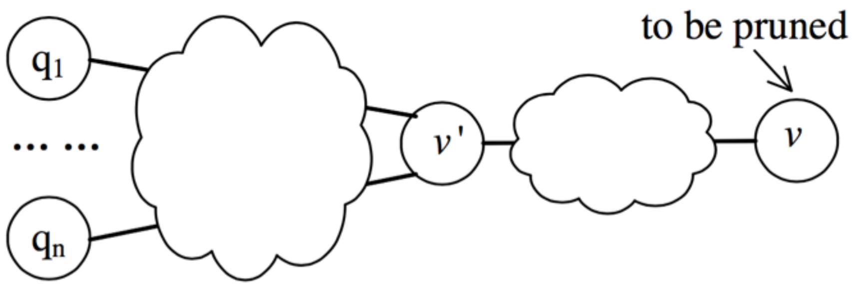
Dynamic skyline queries in large graphs



ID	$Dist(v_i, q_1)$	$Dist(v_i, q_2)$
v_0	1	1
v_3	2	1
v_4	3	2

Shared Shortest Path (SSP) pruning

- if there exists at least one joint (common) vertex v′ among all shortest paths between v and q_l, v can be pruned safely



Zou, L., Chen, L., Özsu, M.T. and Zhao, D. Dynamic skyline queries in large graphs. DASFAA, 2010

Where we are

Background (5 min)
Graph models, subgraph isomorphism, subgraph mining, graph clustering

Exploratory Graph Analysis (35 min)

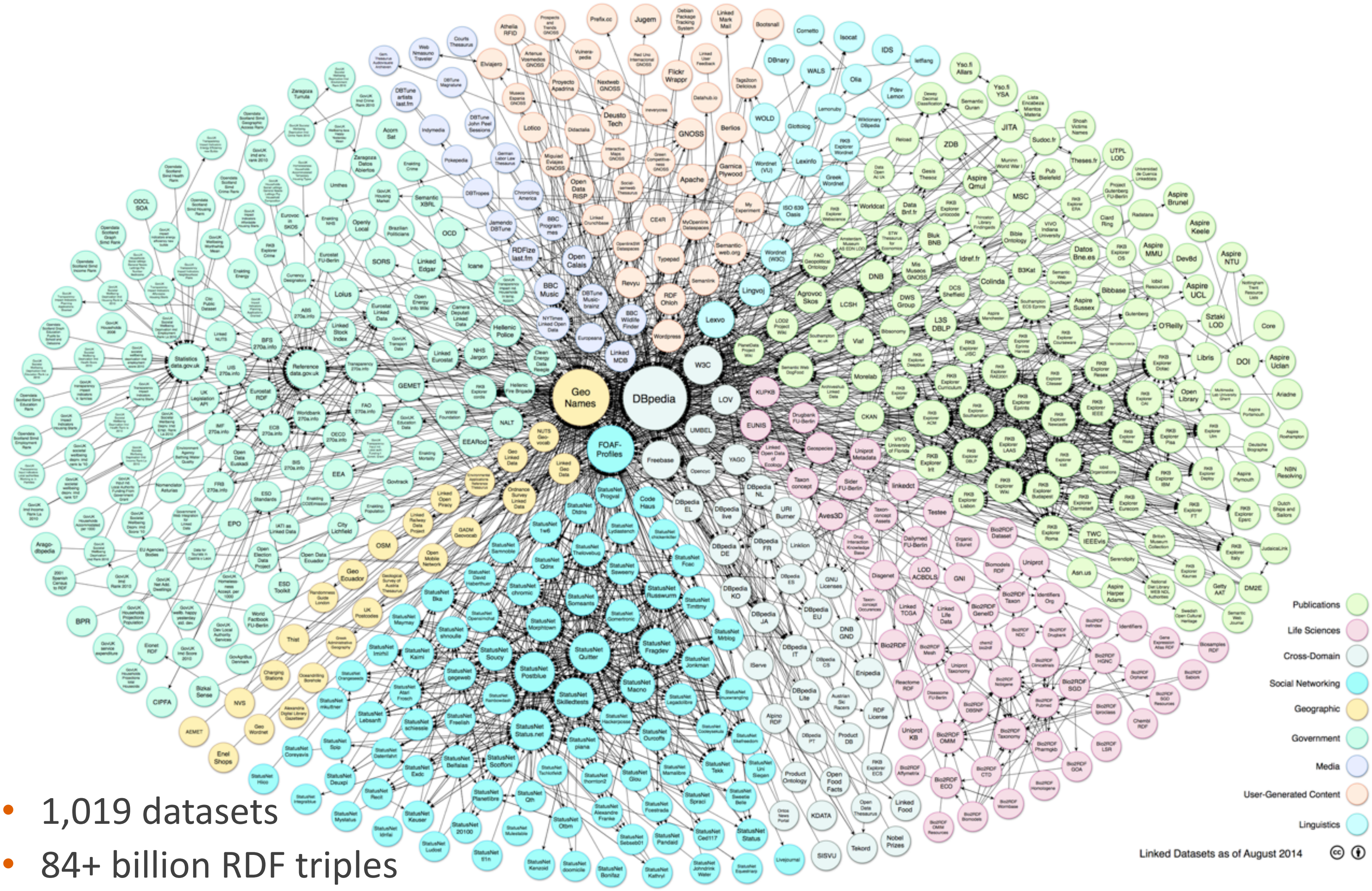
Focused Graph Mining (35 min)

Refinement of Query Results (35 min)

Real World-Use Case (15min)
Linked Data graphs

Challenges and discussion

The Web of Data

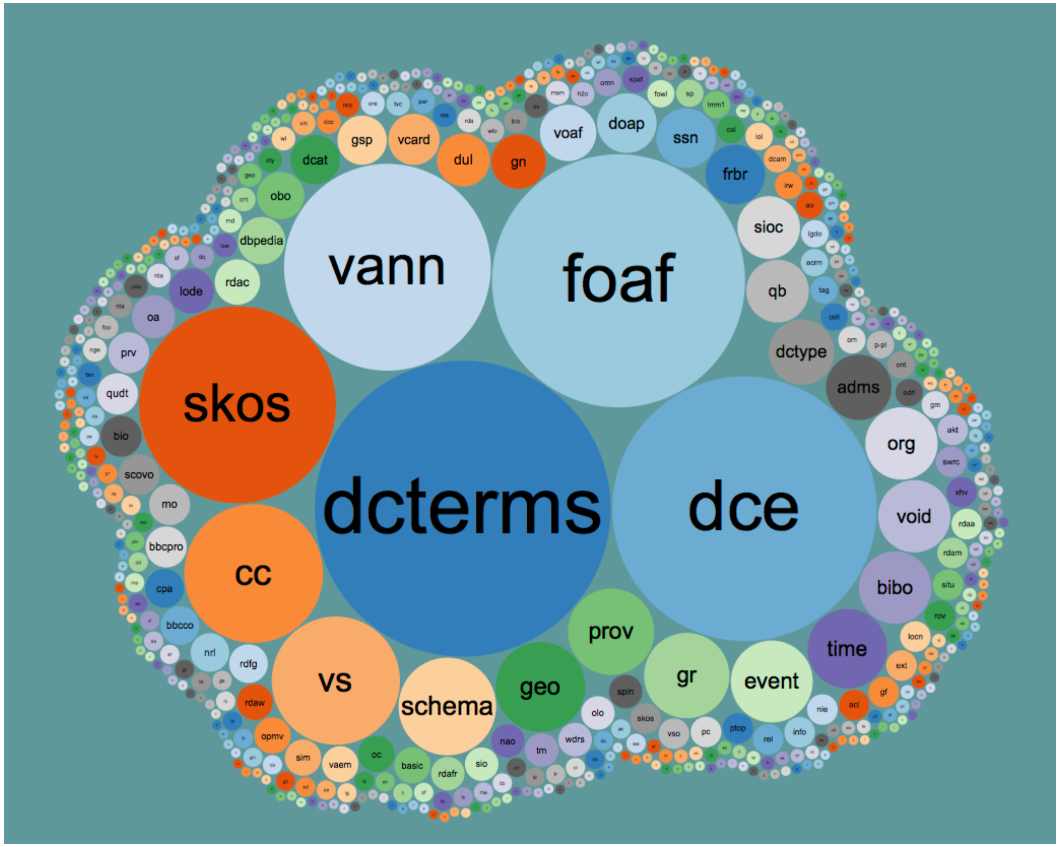


- 1,019 datasets
- 84+ billion RDF triples
- 808+ million RDF links between datasets

http://lod-cloud.net

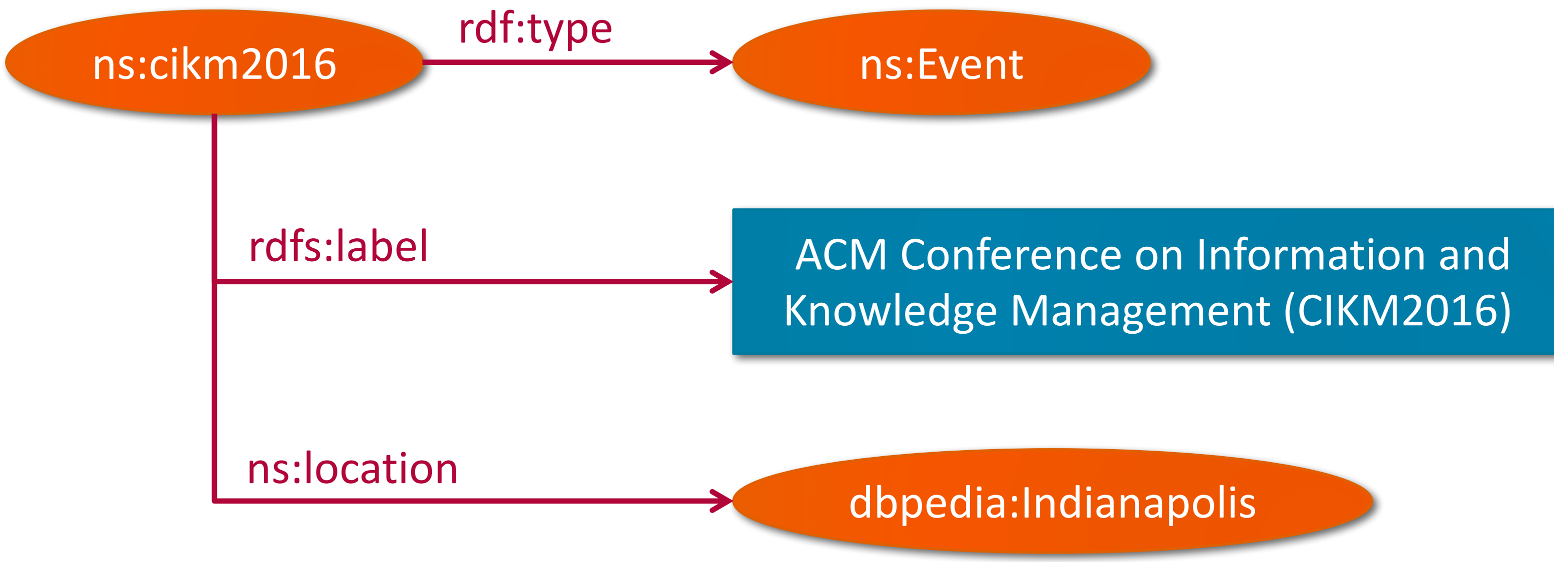
Vocabularies on the Web of Data

- The Web of Data is heterogeneous
  - Many vocabularies are in use (576 as of October 2016)
  - Many different ways to represent the same information

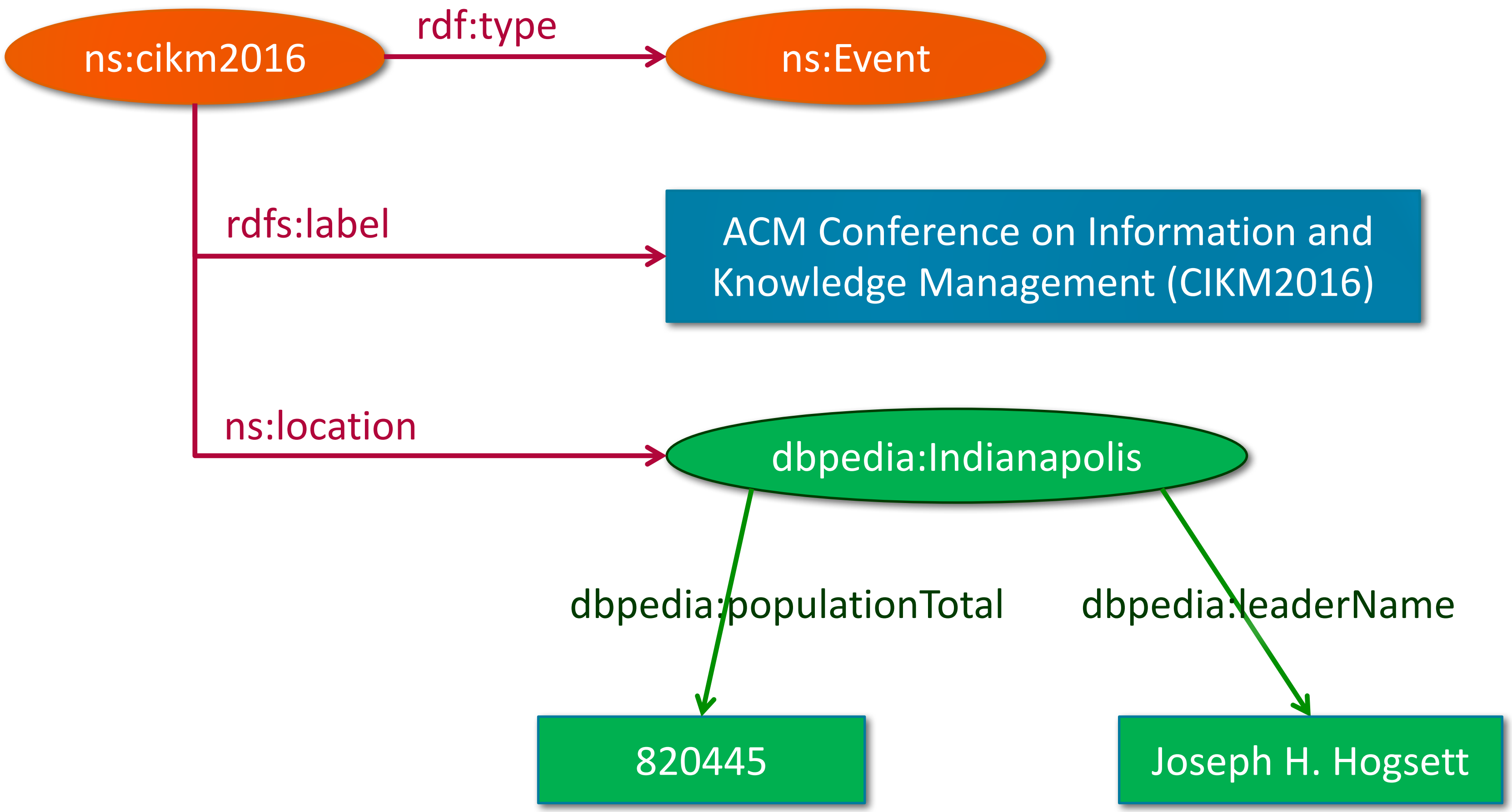


https://lov.okfn.org

RDF Data Model



RDF Data Model



Linked Data exploration use cases

- Dataset exploration
- Graph mining
- Query formulation and refinement

! But Linked Data is messy

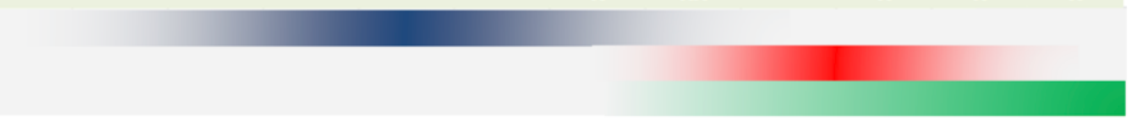
Linked Data graph exploration challenges

- Nested graphs → Makes reasoning difficult
- Loose structure → Things have different property sets
- Incomplete → Missing property definitions
- Poorly formatted → Property types used inconsistently
- Inconsistent → Multiple representations claim opposite things

Linked Data exploration systems timeline

System category / year		2001...	...2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014
Browsers	Textual			x		xx							
	Visual		x	x			xx						
	Faceted			xx	x		x						
	Other paradigms							xx	x	x			
Recommenders	Type/domain specific								xxx				
	Cross-type/domain									xx			
	Undisclosed/commercial										x	xx	
ESS	View-based										x	x	
	Algorithm-based							x	xx		x	x	x

Research focus



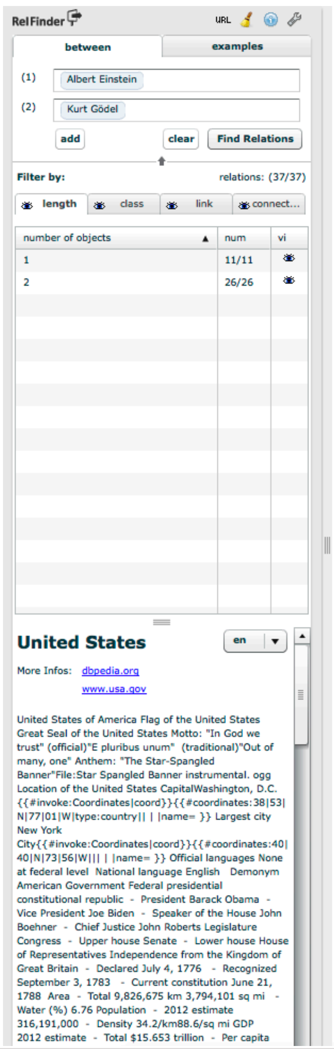
DBpedia Mobile

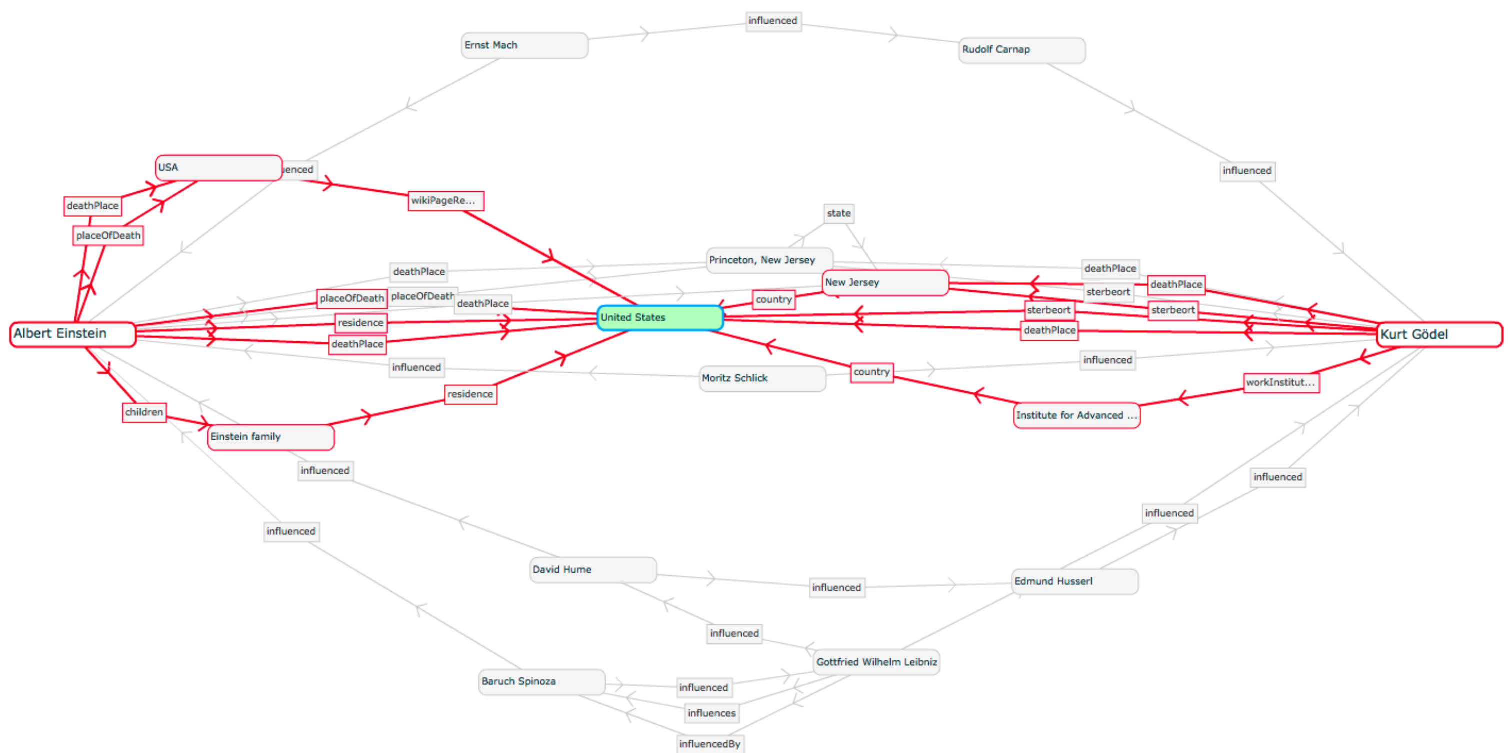
- displays Wikipedia data on map
- aggregates different data sources

C. Becker and C. Bizer. DBpedia mobile: A location-enabled linked data browser. LDOW 2008.

RelFinder

- visualization of paths between any 2 entities
- path identification on instance level

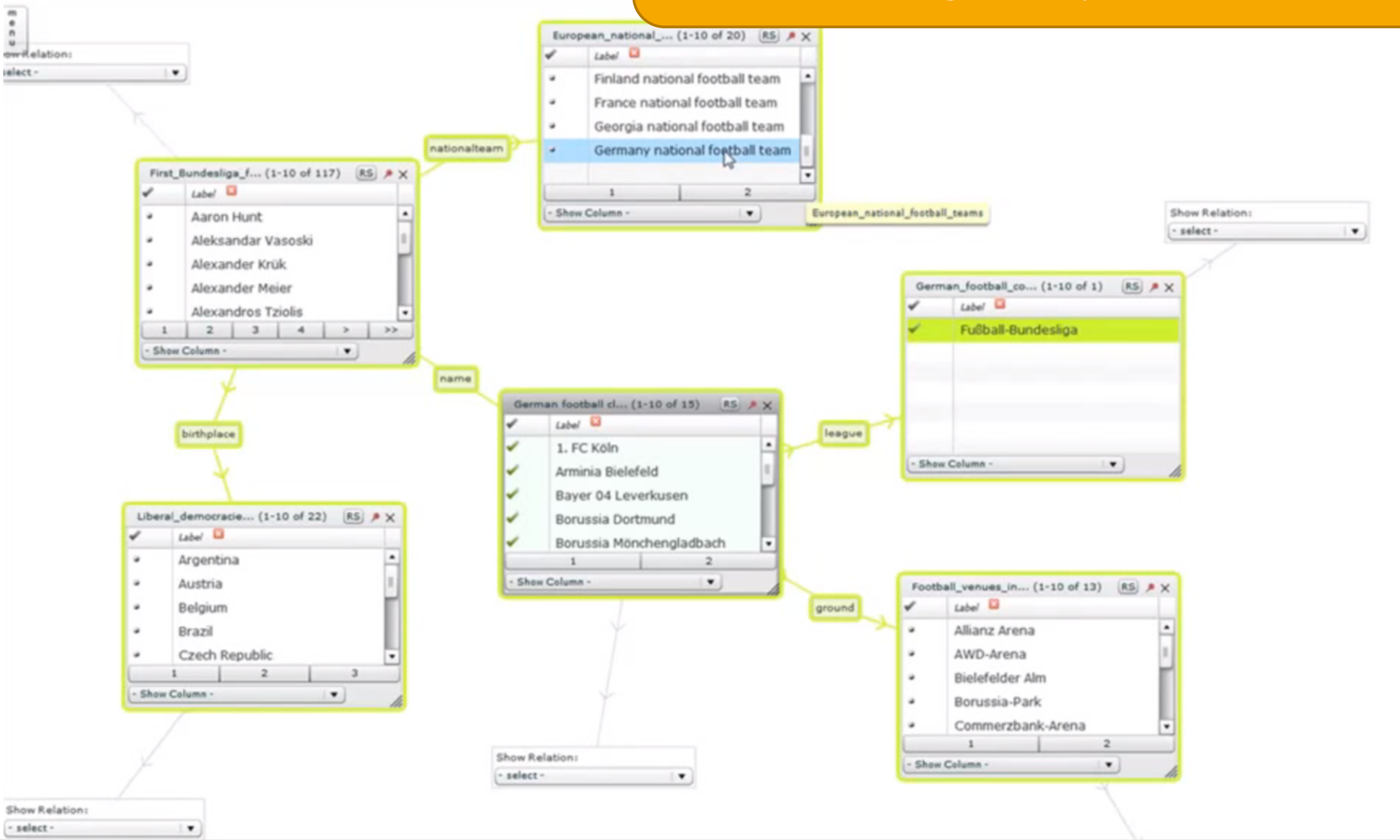




Heim, P., Hellmann, S., Lehmann, J., Lohmann, S., and Stegemann, T. RelFinder: Revealing Relationships in RDF Knowledge Bases. SAMT 2009.

gFacet

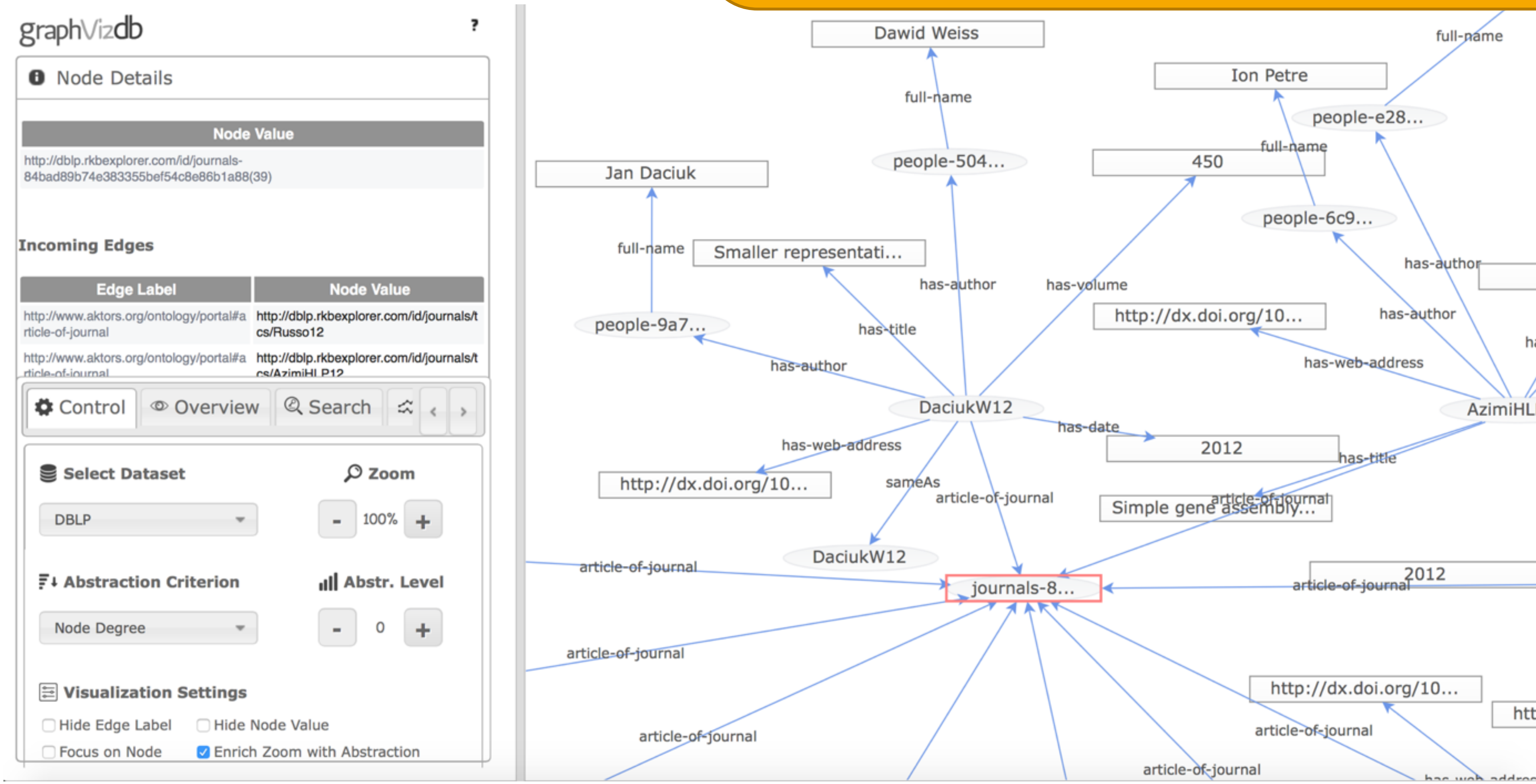
- Schema exploration
- combines graph-based visualization and faceted filtering techniques



P. Heim, T. Ertl, and J. Ziegler. Facet graphs: Complex semantic querying made easy. The Semantic Web: Research and Applications. Springer, 2010.

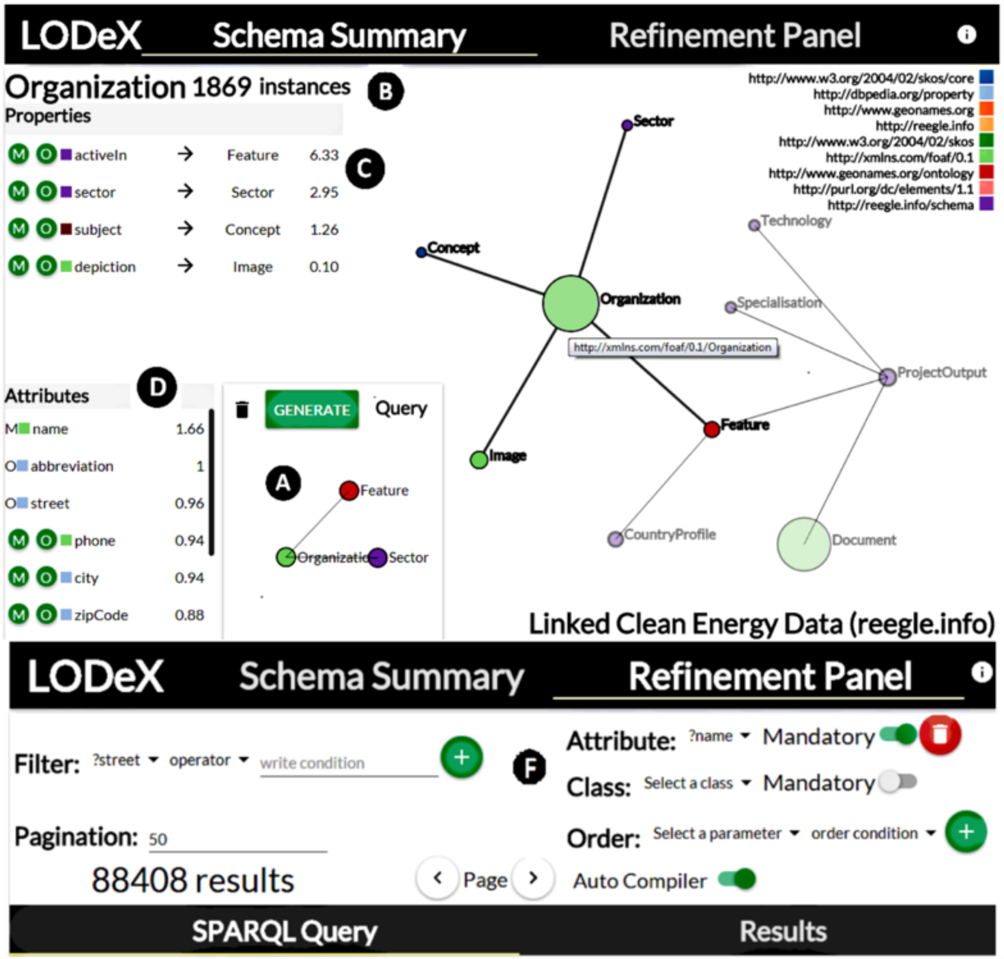
graphVizdb

- Graph layout is indexed with a spatial data structure, i.e., an R-tree, and stored in a database
- In runtime, user operations are translated into efficient spatial operations (i.e., window queries) in the backend



Bikakis, N., Liagouris, J., Krommyda, M., Papastefanatos, G. and Sellis, T. graphVizdb: A scalable platform for interactive large graph visualization. ICDE, 2016

LODeX

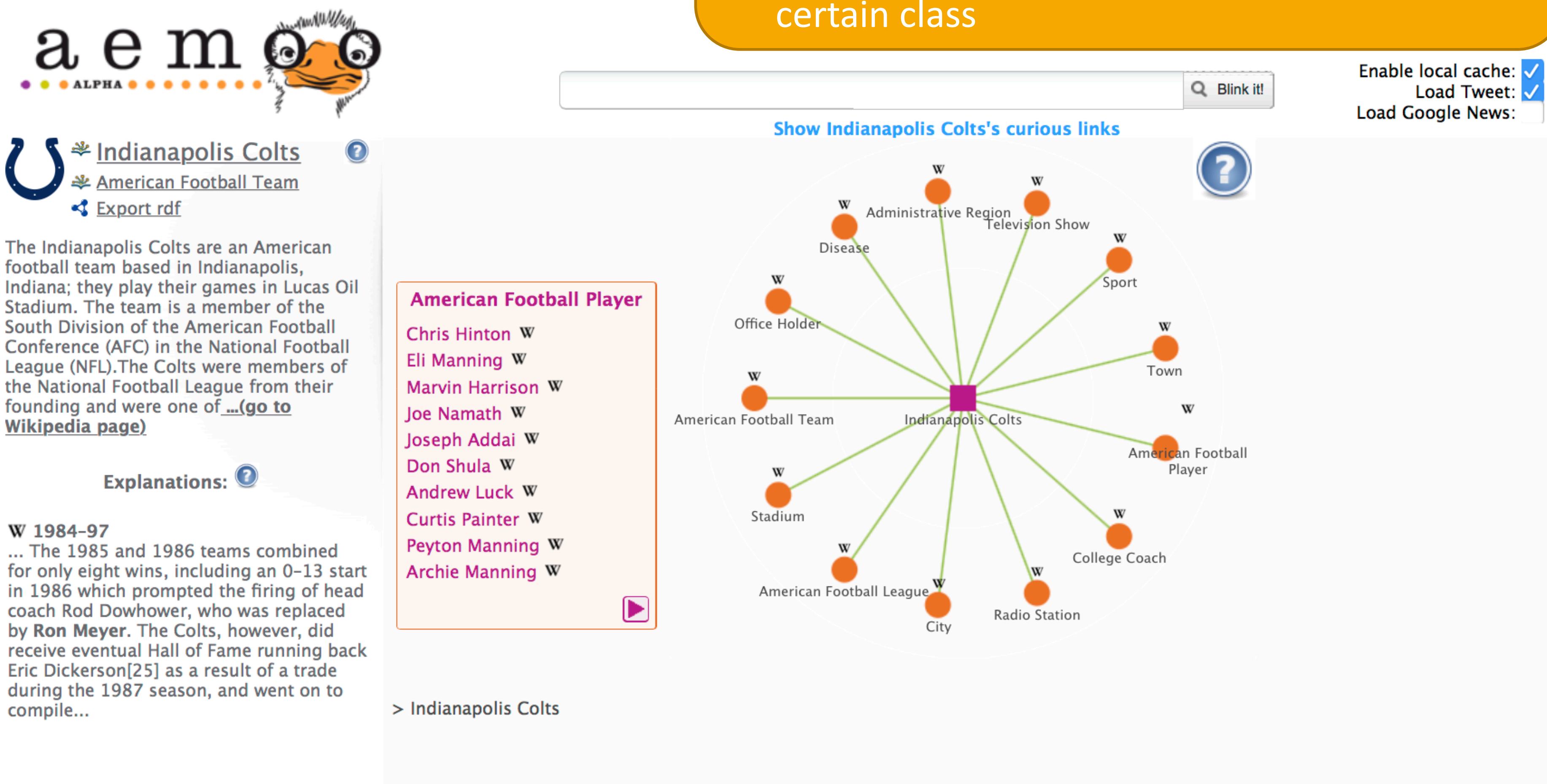


- Explore a Linked Dataset using a schema summary
- Pick graphical elements from it to create a visual query
- Browse the results
- Refine the query

Benedetti, F., Bergamaschi, S. and Po, L. Lodex: A tool for visual querying linked open data. ISWC, 2015

Aemoo

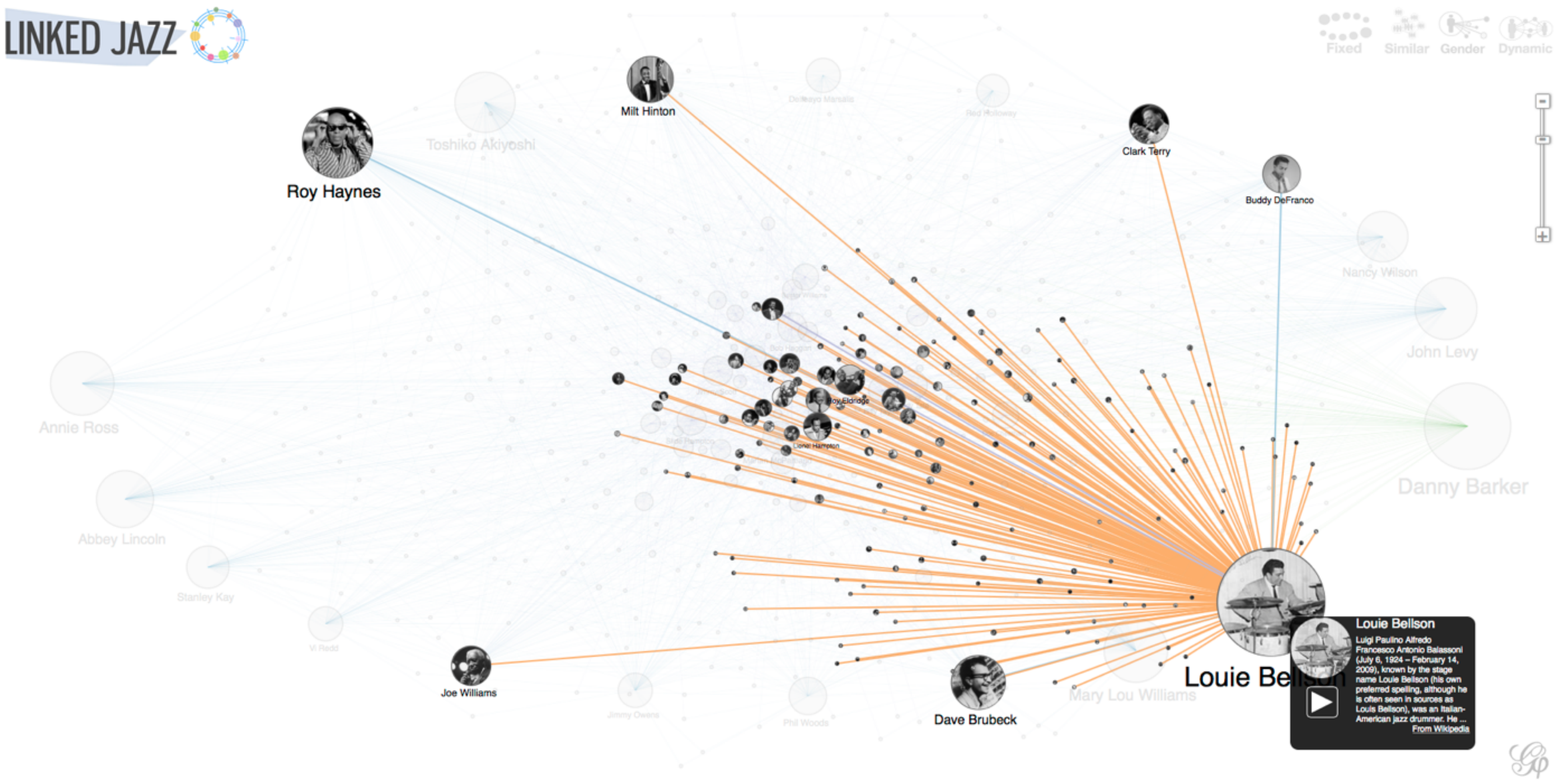
- Exploratory search system based on Encyclopedic Knowledge Patterns
- EKP are knowledge patterns that define the typical classes used to describe entities of a certain class



A. Musetti, A. G. Nuzzolese, F. Draicchio, V. Presutti, E. Blomqvist, A. Gangemi, and P. Ciancarini. Aemoo: Exploratory search based on knowledge patterns over the semantic web. Semantic Web Challenge, 2012.

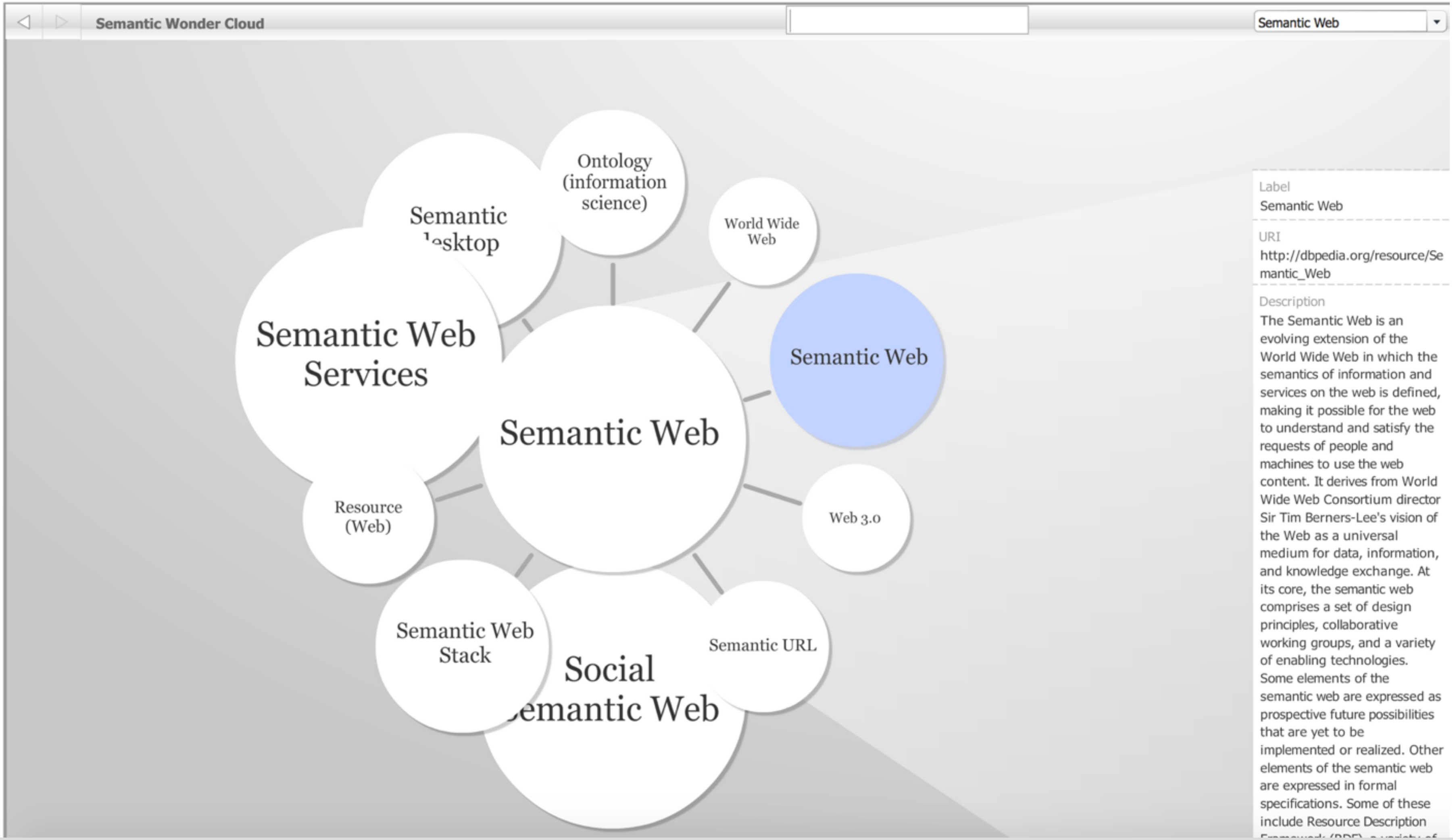
Linked Jazz

- reveals the network of the social and professional relations within the American jazz community



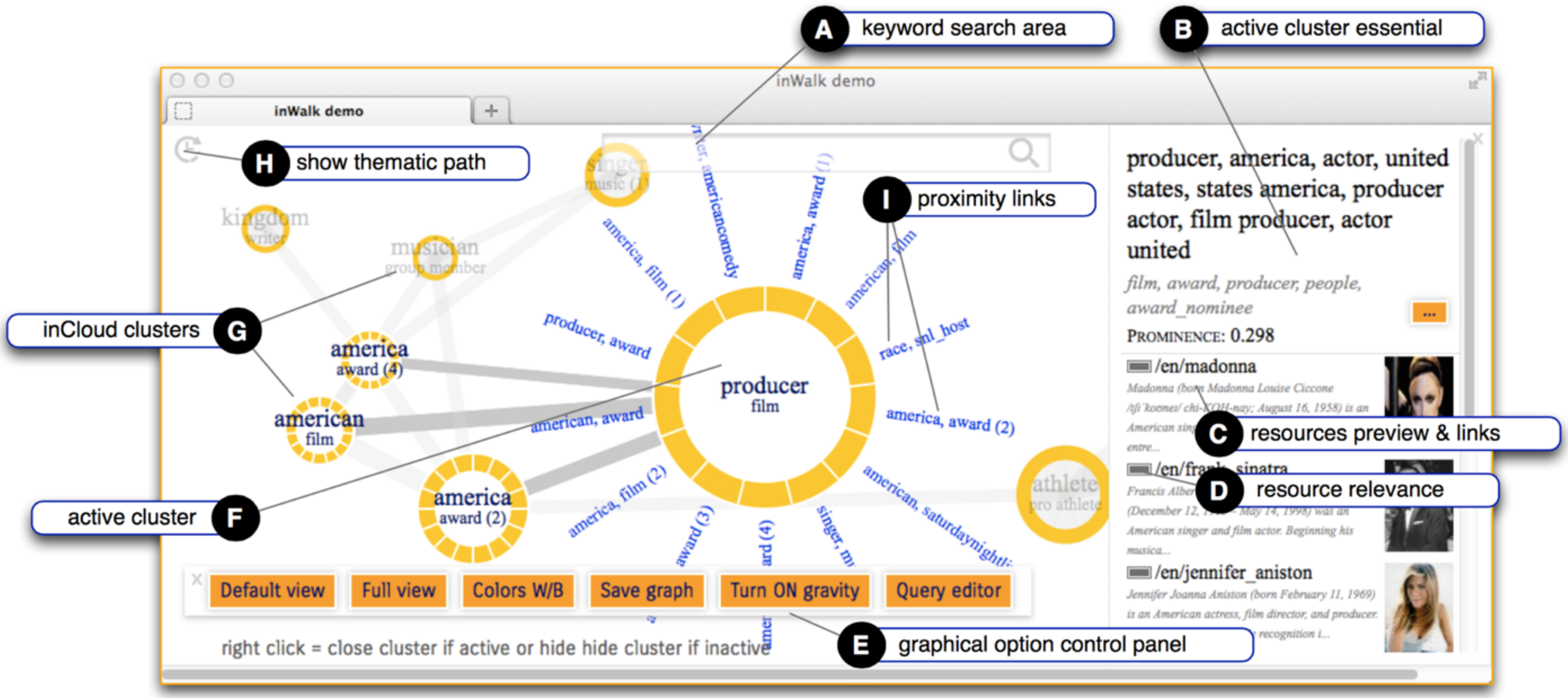
M. C. Pattuelli, M. Miller, L. Lange, S. Fitzell, and C. Li-Madeo. Crafting linked open data for cultural heritage: Mapping and curation tools for the linked jazz project. Code4Lib Journal, 2013.

Semantic Wonder Cloud



http://sisinflab.poliba.it/semantic-wonder-cloud/index/

inWalk



Castano, S., Ferrara, A. and Montanelli, S. inWalk: Interactive and Thematic Walks inside the Web of Data. EDBT, 2014

ProLOD++
Mining Graph Patterns on the Web of Data

ProLOD++

- Web framework for various data profiling and mining tasks on Linked Datasets

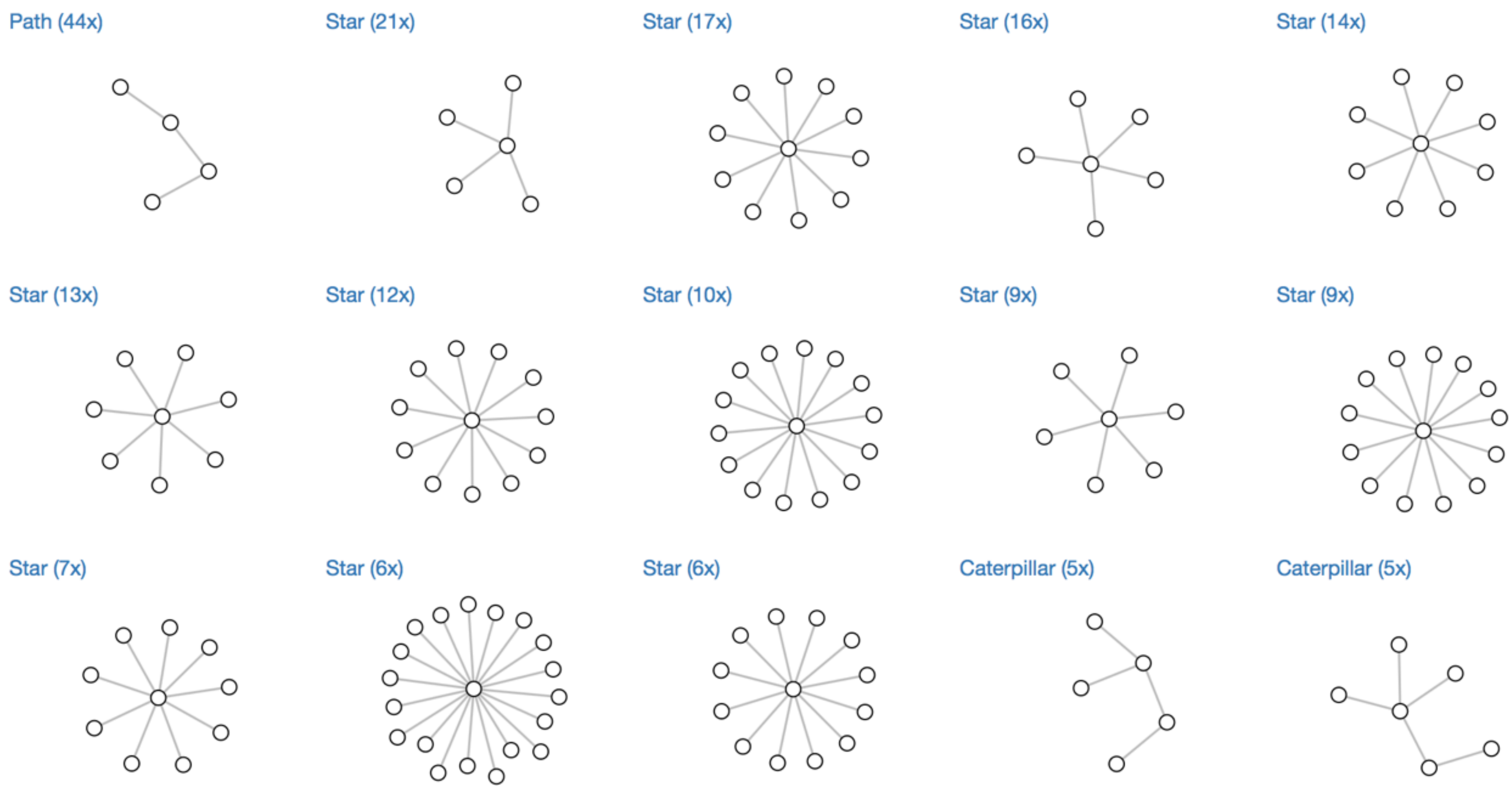
- Explorative research on Linked Dataset graphs to find
  - frequent graph patterns
  - common graph patterns for classes
  - general graph model for Linked Datasets

https://prolod.org

Jentzsch, A., Dullweber, C., Troiano, P., Naumann, F. Exploring Linked Data Graph Structures. ISWC 2015.

ProLOD++
Graph pattern mining

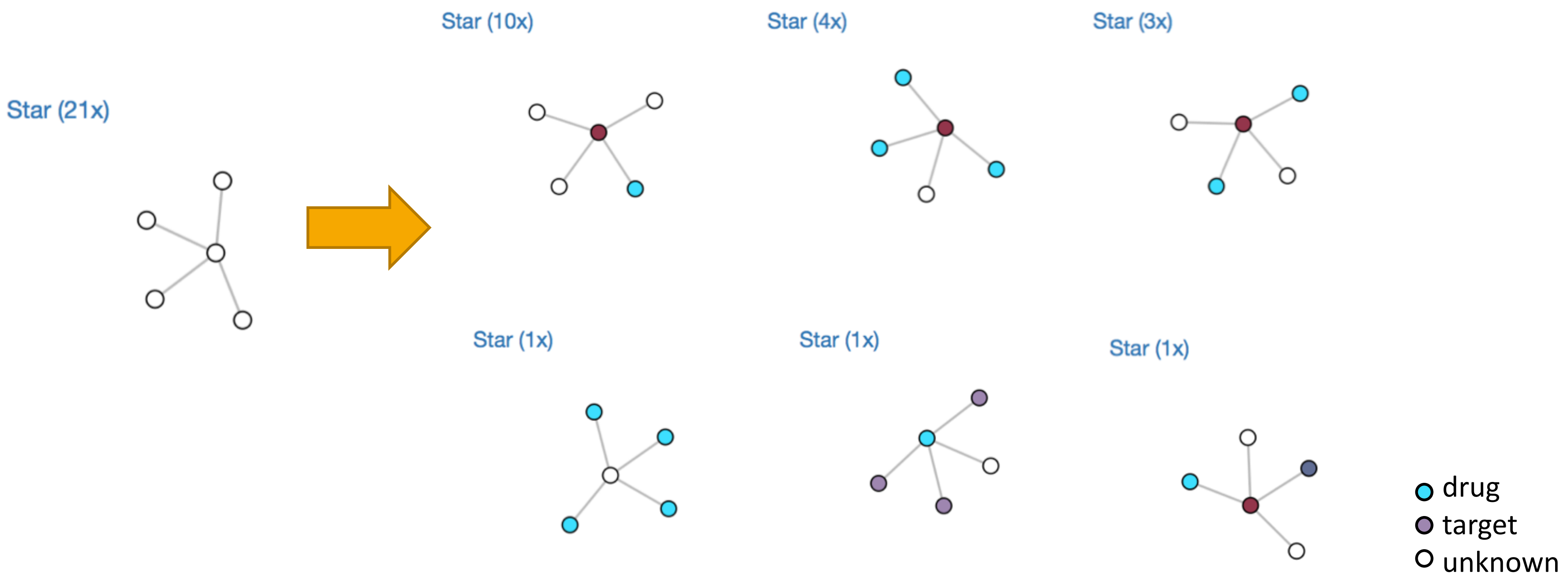
Definition of core set of frequent graph patterns in Linked Datasets based on satellite component analysis



Jentzsch, A., Dullweber, C., Troiano, P., Naumann, F. Exploring Linked Data Graph Structures. ISWC 2015.

ProLOD++
Graph patterns

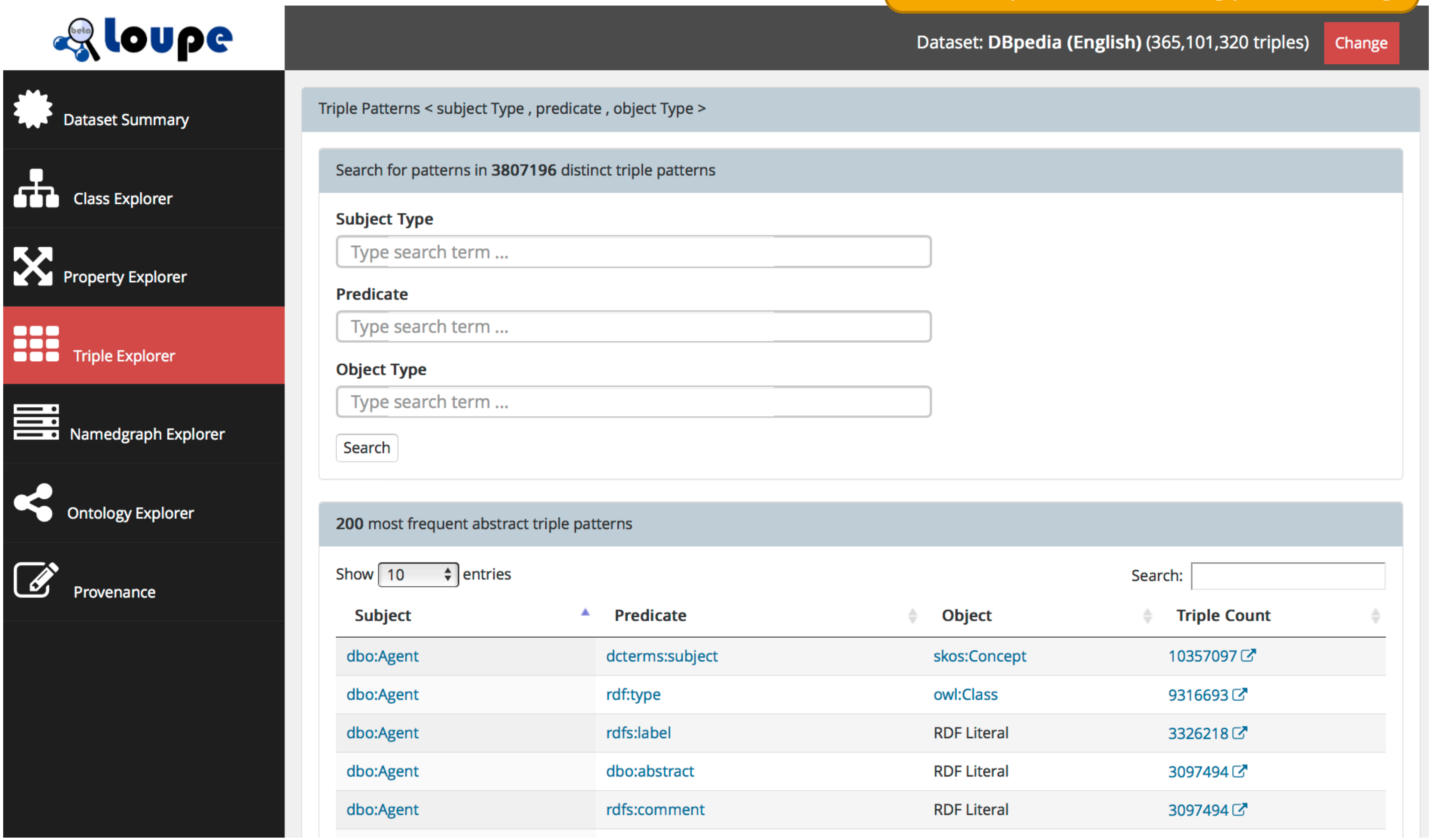
- Group class-coloured graphs by their permutation groups [Luks82]
  - Permutation group: the set of all automorphisms of a graph



Jentzsch, A., Dullweber, C., Troiano, P., Naumann, F. Exploring Linked Data Graph Structures. ISWC 2015.

Loupe

- Frequent triple patterns
- Graphical ontology browsing



Mihindukulasooriya, N., Poveda-Villalón,M., García-Castro, R. and Gómez-Pérez, A. Loupe - An Online Tool for Inspecting Datasets in the Linked Data Cloud. ISWC 2015.

Requirements for Linked Data exploratory search systems

- The system provides efficient overviews
- The system helps the user to understand the information space and to shape his mental model
- The user can explore multiple, heterogeneous results and browsing paths
- The system eases the memorization of relevant results
- The system inspires the user and shapes his information need
- The system provokes discoveries

Challenges

- Displaying the graph for exploration
  - E.g. by clustering of topical domains
  - Allowing the user to drill down
- Live graph exploration
  - E.g. via federated SPARQL queries
    - Requires knowledge on endpoint URIs
    - Slow in real-time
- Guiding the user to interesting parts of the graph
  - Usually done by entity inlinks
    - Limited insights

Tutorial outline

Background (5 min)

Graph models, subgraph isomorphism, subgraph mining, graph clustering

Exploratory Graph Analysis (35 min)

Focused Graph Mining (35 min)

Refinement of Query Results (35 min)

Real World-Use Case (15min)

Linked Data graphs

Challenges and discussion

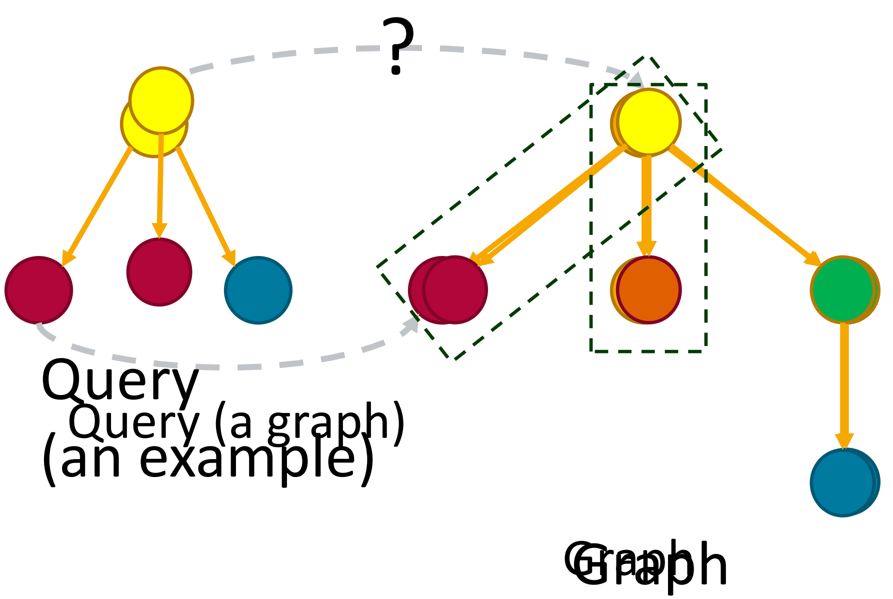
Summary of Exploratory Graph Analysis

Approximate Queries

- User query is imprecise

By-Example methods

- User query is an example result



- Only need a partial knowledge on the data
- No need for complicate query languages (use examples, partial descriptions)
- The query adapts to user need
- Enable exploratory search by using small queries on the data

Challenges for Exploratory Graph Analysis

Database

- Unsupported in most of the current graph databases
- No ”universal” index to answer multiple type of queries
- Partitioning only for exact query answering

Data mining

- User interactivity in the exploration process
- No solutions for probabilistic graphs
- Respond to queries while the graph changes
- Find examples in streaming settings

Information retrieval

- Exploiting query logs for personalized query answering
- Retrieve results in form of documents converting the query structures

Summary of Focused Graph Mining

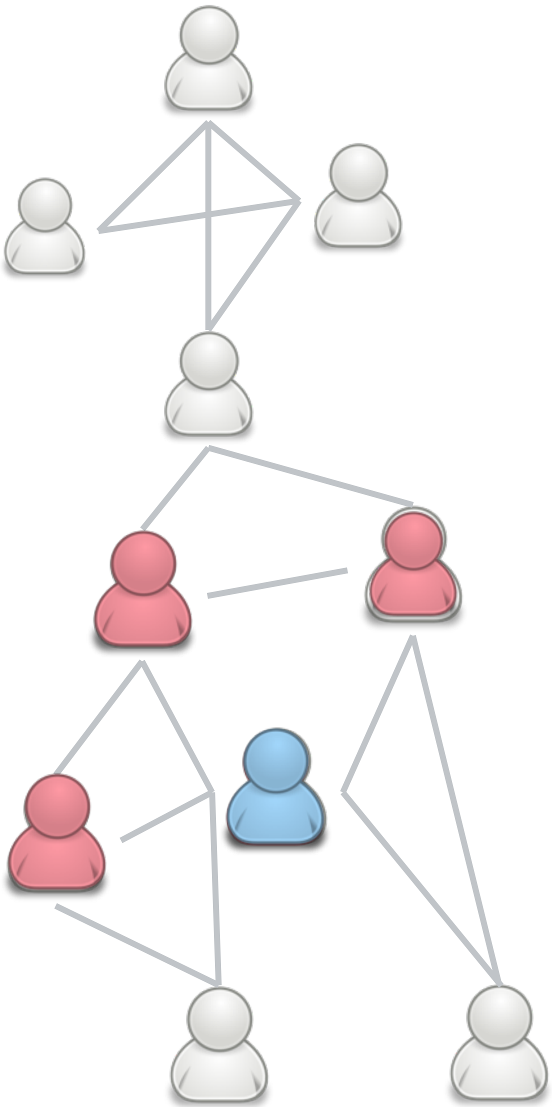
The focus on individual user interest

... as **Query** to the Graph Mining System

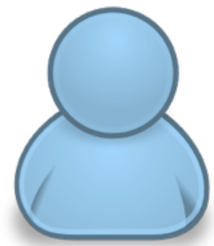
... as **Seed Node(s)** for Local Search

... as **Attributes** and **Weights**

- **get or infer user interest**
  → unexpected results
- **interactive exploration**
  → intuitive parametrization
- **adaptive graph mining**
  → individual local search



Challenges for Focused Graph Mining



User interactivity in the graph mining process
- unsupported in most of the current graph mining algorithms
- huge variety of user interactions possible
- feedback loop needs to be unified and become exchangeable

Data mining

Revolution of formal models and search algorithms
- insufficient extensions of existing models and algorithms
- adaptive steering of algorithms vs. fixed parametrization
- evaluation of algorithms with user studies

Scalability of algorithms for real-time interaction
- NP-hard problems, heuristic algorithms, …, still not scalable
- exploit the user interest for pruning the search space

Summary of Refinement of Query Results

Refinement

- The user query is too restrictive or too generic

Top-k Results

- Queries typically have inexact matches

Skyline Queries

- Find small set of interesting items with many dimensions and incremental updates

- The user might have a very generic idea of how to describe the structure of interest
- The system guides the user towards the answer with simple steps
- The results are explained with reformulations
- The query matches are inexact and interesting

Challenges for Refinement of Query Results

Database

- Real time performance
- Profiling of queries for optimized performance

Data mining

- Personalized reformulations and interactivity
- Facet search discovery in graphs

Information retrieval

- Uncertain graph data

The missing tiles in graph exploration

Interactivity

Adaptivity

Personalization

Scalability

Slides: https://hpi.de//mueller/tutorials/graph-exploration.html