VLDB 2017 tutorial

New Trends on Exploratory Methods for Data Analytics

Davide Mottin, Matteo Lissandrini,
Yannis Velegrakis, Themis Palpanas

HPI

db Trento

UNIVERSITÉ PARIS DESCARTES

Who we are

Davide Mottin

Graph Mining, Novel Query Paradigms, Interactive Methods

https://hpi.de/en/mueller/team/davide-mottin.html

Matteo Lissandrini

Knowledge Graphs , Novel Query Paradigms, Graph Mining

https://disi.unitn.it/~lissandrini

Yannis Velegrakis

Big Data Management & Analytics, Information Integration

https://velgias.github.io

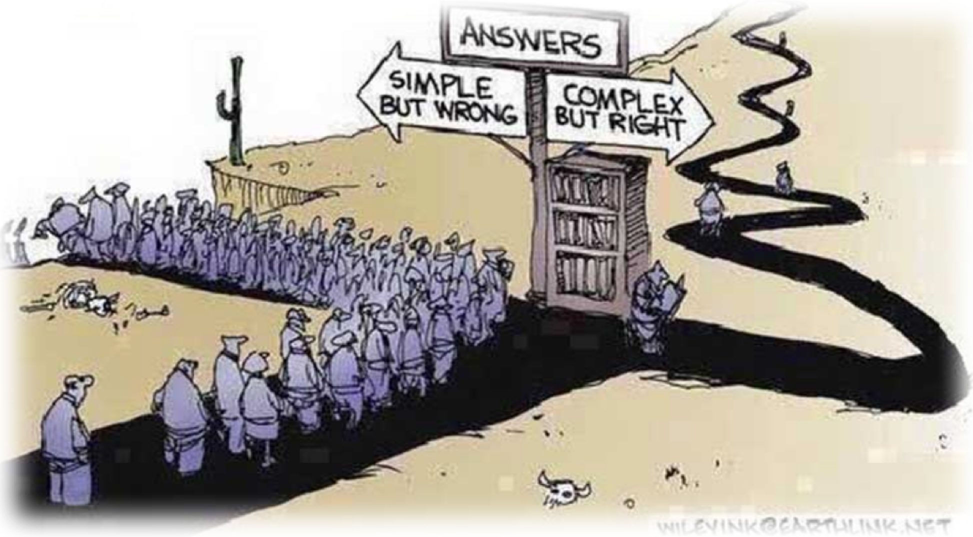
Themis Palpanas

Data Series Indexing & Mining, Data Management, Data Analytics

http://www.mi.parisdescartes.fr/~themisp/

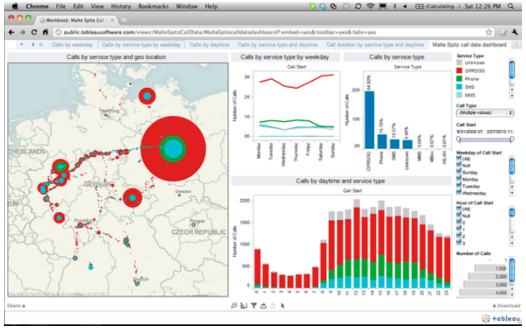
Slides. http://j.mp/DataExplore

Big data – Easy value?



Exploring

Traditional



On data

Data exploration

Cleaning and profiling

Visualization

Analysis

Data exploration software

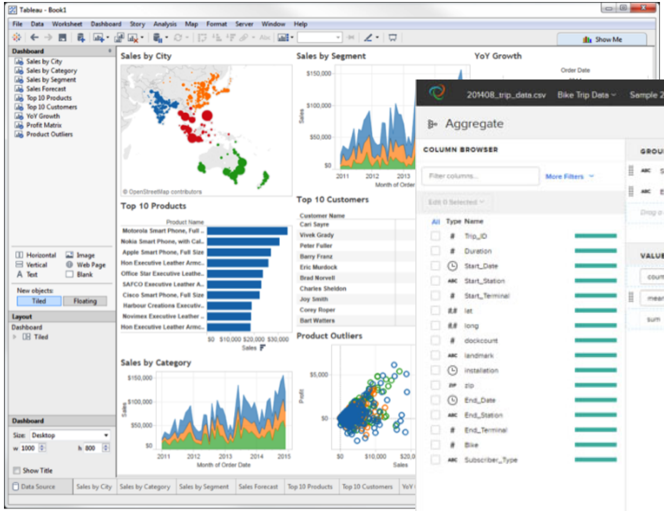
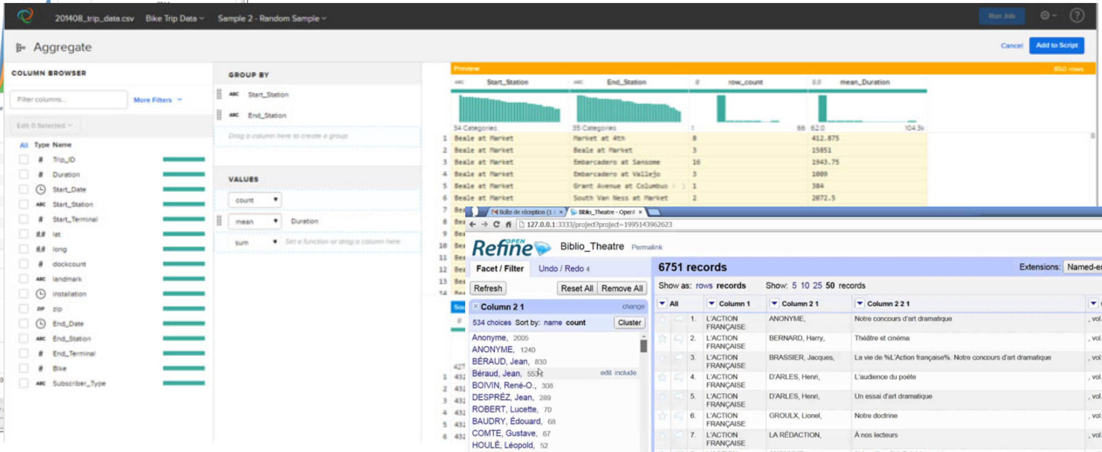
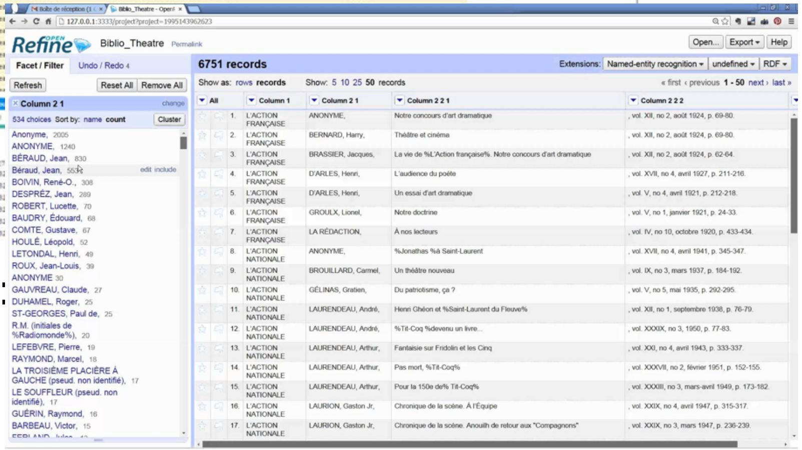


Tableau: analysis and statistics



Trifacta:



OpenRefine: data preparation and cleanup

Traditional data exploration methods

[Idreos et al., 2015]

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

```
SELECT avg(system-stars)
FROM Universe
WHERE system-stars > 10
GROUP BY galaxy
```

Not easy for novices

Declarative Exploratory methods

SELECT galaxy_name
FROM Universe.Galaxy

```
SELECT g.galaxy_name, SUM(s.stars) as st_s
FROM Universe.Galaxy  AS g
JOIN Universe.Systems AS s
ON g.galaxy_name = s.galaxy_name
WHERE
        g.st_s > 100B
        AND diameter > 100k AND diameter > 180k
        AND has_black_hole = TRUE
GROUP BY g.galaxy_name
```

Simple query (exploratory)

Over generic
100 billions results

Complex query
(for data experts)

Specific
Few results

Examples as Exploratory Methods

Answers

Historical perspective: Query-by-example

[Zloof et al. 1975]

Specify a query by example tables, or skeletons.

Name	Stars	Diameter	Black_hol e	Color	Life
P._	> 10B	>100k	TRUE		
		<180k			

- Intuitive GUI for simple queries
- SQL not required

- Restricted to SQL semantics
- Not example-based

Tutorial's goals

- Exploratory methods using examples
- Algorithms for retrieving data without using query languages
- Interactive methods and user-in-the-loop feedback
- Machine learning for adaptive, online methods

But NOT

- Declarative query methods
- User interfaces and visualization
- Optimizations for fast data access
- Dynamic data

Tutorial structure

Relational databases (25 min)

Textual data (10 min)

Graph and networks (25 min)

Machine learning (10 min)

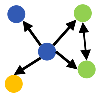
Challenges and Remarks

Example-based methods

- Query suggestion using examples
- Reverse engineering queries

- Entity extraction by example text
- Web table completion using examples
- Search by example

- Community-based Node-retrieval
- Entity Search
- Path and SPARQL queries
- Graph structures as Examples



Where we are

Relational databases

Textual data

Graphs and networks

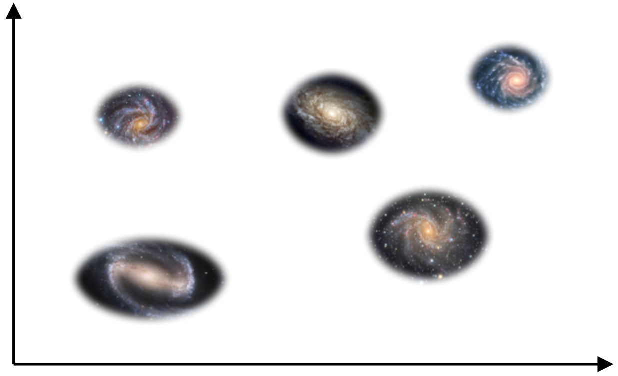
Machine learning

Challenges and Remarks

Reverse engineering queries (REQ)

Given a set of examples, find the query that generated that set of tuples

Example tuples



How do you find such queries?

SELECT g.galaxy_name, SUM(s.stars) **AS** st_s
FROM Universe.Galaxy **AS** g
JOIN Universe.System **AS** s
ON g.galaxy_name = s.galaxy_name
WHERE
    g.st_s > 100B
    AND diameter > 100k AND diameter > 180k
    AND has_black_hole = TRUE
GROUP BY g.galaxy_name

SELECT galaxy_name
FROM Universe.Galaxy

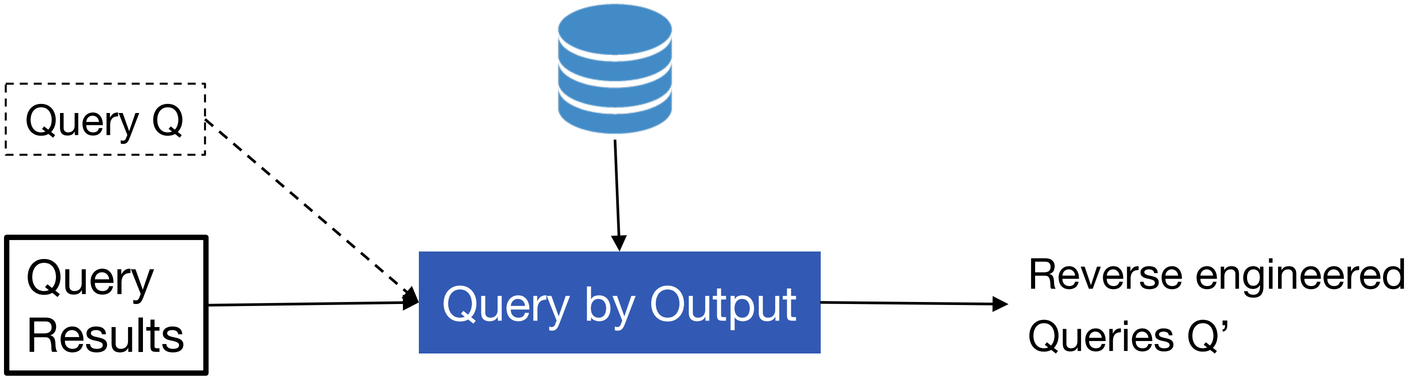
Reverse engineering queries (REQ)

- REQ
  - Exact
    - One-shot
      - Query by output - TALOS
      - REQ SPJ queries from examples
    - Interactive
      - Query From examples (QFE)
      - Interactive inference of join queries
  - Approximate
    - Minimal
      - Discovering Queries based on Examples
    - Top-k
      - S4: Top-k Spreadsheet style

Query by Output - TALOS

[Tran et al. 2013]

Main idea: Find the set of queries that exactly return a set of examples



Two queries Q and Q’ are instance equivalent on a database D, if the results of Q are the same of the results of Q’

TALOS

[Tran et al. 2013]

pID	name	country	weight	bats	throws
P1	A	USA	85	L	R
P2	B	USA	72	R	R
P3	C	USA	80	R	L
P4	D	Germany	72	L	R
P5	E	Japan	72	R	R

(a) Master

pID	year	stint	team	HR
P1	2001	2	PIT	40
P1	2003	2	ML1	50
P2	2001	1	PIT	73
P2	2002	1	PIT	40
P3	2004	2	CHA	35
P4	2001	3	PIT	30
P5	2004	3	CHA	60

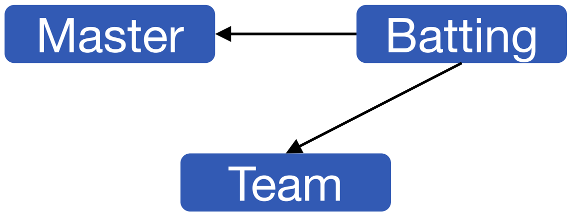
(b) Batting

team	year	rank
PIT	2001	7
PIT	2002	4
CHA	2004	3

(c) Team

Input

B	PIT
E	CHA



Join graph computation

Join table

$J = \texttt{Master} \bowtie \texttt{Batting} \bowtie \textit{Team}$

	name	bat	throw	stint	HR	team
t_1	A	L	R	2	40	PIT
t_2	A	L	R	2	50	MT1
t_3	C	R	L	2	35	CHA
t_4	D	L	R	3	30	PIT
t_5	B	R	R	1	73	PIT
t_6	B	R	R	1	40	PIT
t_7	E	R	R	3	60	CHA

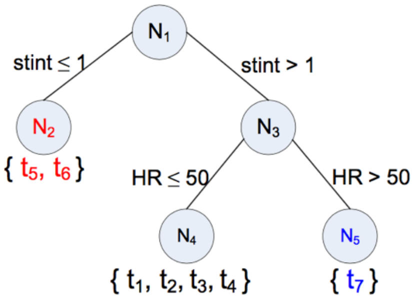
TALOS

[Tran et al. 2013]

	name	bat	throw	stint	HR	team	
t_1	A	L	R	2	40	PIT	✗
t_2	A	L	R	2	50	MT1	✗
t_3	C	R	L	2	35	CHA	✗
t_4	D	L	R	3	30	PIT	✗
t_5	B	R	R	1	73	PIT	✓
t_6	B	R	R	1	40	PIT	✓
t_7	E	R	R	3	60	CHA	✓

Idea: treat the problem as a binary classification

1. **Strict**: all tuples must be captured
2. **At-Least-one**: one tuple for example must be captured



Decision tree

$$Gini(S) = 1 - (f_+^2 + f_-^2)$$

Positive and negative tuples in S

$$Gini(S_1, S_2) = \frac{(|S_1| Gini(S_1) + |S_2| Gini(S_2))}{|S_1| + |S_2|}$$

How complex is exact REQ?

[Weiss et al., 2017]

Database D

Relational Operators:

σ selection $\{=, \neq, \geq, \leq\}$

π projection

$\bowtie$ natural join

E^+ Positive examples

E^- Negative examples

REQ

Q such that results contain

- All positive examples
- No negative example

How difficult is to find:
A bounded size Q? an unbounded Q?

Complexity - No parameters

[Weiss et al., 2017]

Operator	Unbounded Queries	Bounded Queries
π	P	P
$\bowtie$	P	NPC
σ	P	NPC
$\sigma, \bowtie$	P	NPC
π, σ	NPC	NPC
$\sigma, \bowtie$	DP	DP
$\pi, \sigma, \bowtie$	DP	DP

Only projections: Easy

Unbounded selections: Easy

Unbounded selections: HARD

Combination of operators: HARD!!!

Unbounded Select

[Weiss et al., 2017]

	A	B	C	D	E
☑	1	2	3	4	5
☒	1	3	2	3	4
	2	4	4	1	3
	5	3	2	4	2
☒	4	2	3	1	2
	2	2	4	3	2
☑	1	1	2	1	5
☑	1	5	4	2	3

Possible queries?

$A = 1$ AND

$B \geq 1$ AND $B \leq 5$ AND

$C \geq 2$ AND $C \leq 4$ AND

$D \geq 1$ AND $D \leq 4$ AND $D \neq 4$

$E \geq 3$ AND $E \leq 5$ AND $E \neq 4$

Bounded select

Reduction from Set Cover

NP-C

INPUT: Database D, Examples E, Query size k

OUTPUT: Does there exist a query satisfying D and E, of size at most k?

U = {1,2,3,4,5}   S = { {1,2,3}, {2,4}, {3,4}, {4,5} }

	S_1	S_2	S_3	S_4
☒	1	0	0	0
☒	1	1	0	0
☒	1	0	1	0
☒	0	1	1	1
☒	0	0	0	1
☑	0	0	0	0

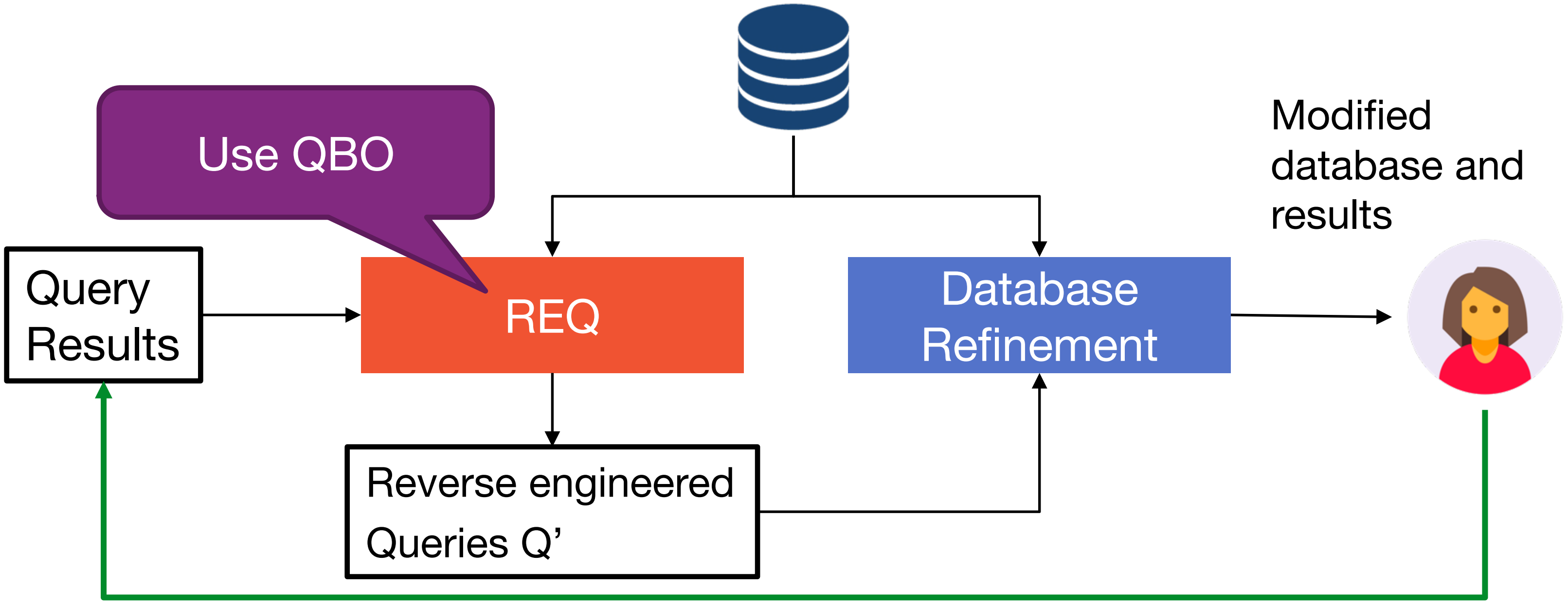
Complexity - Parameters

	No param	Schema	Examples	No param	Query	Schema	Examples
π	P	-	-	P	-	-	-
$\bowtie$	P	-	-	NPC	P $W[2]C$	P FPT	NPC
σ	P	-	-	NPC	P $W[2]C$	$\{=\}: P$, $\{\neq\}: NPC$ $\{=\}: FPT$	P FPT
$\sigma, \bowtie$	P	-	-	NPC	P $W[2]C$	$\{=\}: P$, $\{\neq\}: NPC$ $\{=\}: FPT$	NPC
π, σ	NPC	$\{=\}: P$, $\{\neq\}: NPC$ $\{=\}: FPT$	P $W[3]C$	NPC	P $W[3]C$	$\{=\}: P$, $\{\neq\}: NPC$ $\{=\}: FPT$	NPC
$\pi, \bowtie$	DP	P $W[1]H$, co-$W[1]H$	DP	DP	P $W[2]H$, co-$W[1]H$	P $W[1]H$, co-$W[1]H$	DP
$\pi, \sigma, \bowtie$	DP	$\{=\}: P$, $\{\neq\}: NPC$ $\{=\}: W[1]H$, co-$W[1]H$	DP	DP	P $W[3]H$, co-$W[1]H$	$\{=\}: P$, $\{\neq\}: NPC$ $\{=\}: W[1]H$, co-$W[1]H$	DP

Interactive REQ – Query from Examples

[Li et al., 2015]

Main idea: Interactively remove candidate queries proposing a new set of query results from a modified database



Database Refinement

[Li et al., 2015]

eid	name	gender	dept	salary
1	Alice	F	Sales	3700
2	Bob	M	IT	4200
3	Carol	F	Service	3000
4	Dave	M	IT	5000

Results

name
Bob
Dave

REQs =

- $Q_1 = \sigma_{gender=M}(D)$
- $Q_2 = \sigma_{salary>3700}(D)$
- $Q_3 = \sigma_{dept=IT}(D)$

Database Refinement

eid	name	gender	dept	salary
1	Alice	F	Sales	3700
2	Bob	M	IT	3000 ~~4200~~
3	Carol	F	Service	3000
4	Dave	M	IT	5000

name
Bob
Dave

Result $R_1' = Q_1(D') = Q_3(D')$

name
~~Bob~~
Dave

Result $R_2' = Q_2(D')$

Cost model

[Li et al., 2015]

Number of modified tables

Number of new result sets

$$cost(D') = \boxed{edit(D,D') + \beta \cdot n} + \boxed{\sum_{i=1}^{k} edit(R,R_i)} + \boxed{N \cdot \frac{edit(D,D')}{\mu} + \beta} + \boxed{\frac{2}{k}\sum_{i=1}^{k} edit(R,R_i)}$$

DB cost | Results cost | Effort to examine D’ | Effort to examine new results

Current cost | Residual cost

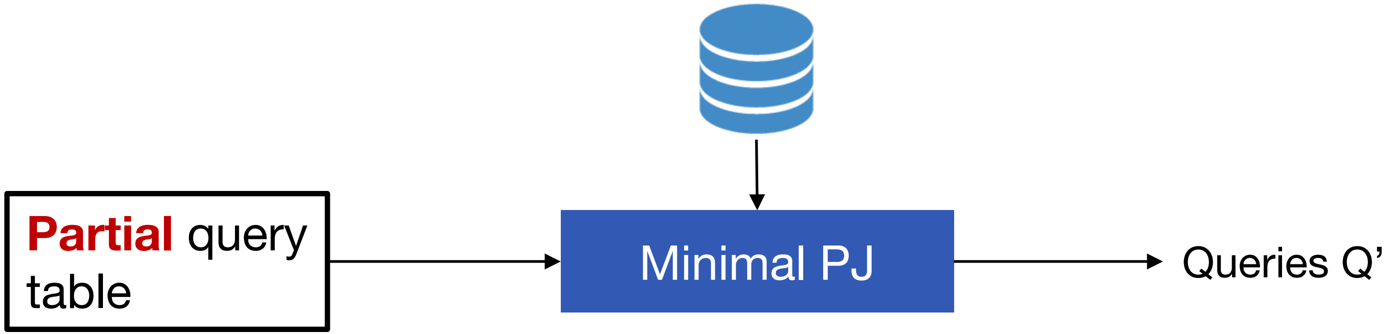
Main idea: Find a refined db D’ and results $R_1, \dots R_k$ with:

1. Minimum number of results k
2. Minimum differences i the database
3. The query are balanced (less interactions)

Minimal Project Join REQ

[Shen et al., 2014]

Main idea: Find the set of queries that approximately return a set of examples



	A	B	C
1	Mike	ThinkPad	Office
2	Mary	iPad	
3	Bob		Dropbox

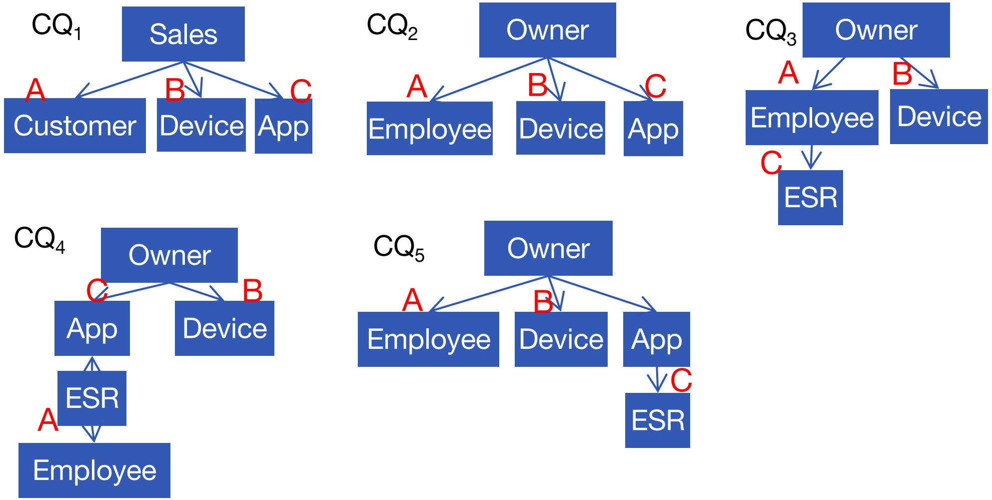
- valid: every tuple is present in query results
- minimal: any removal in query tree gets to an invalid query

Candidate Query Generation

[Shen et al., 2014]

- Use candidate network generation algorithm (Hristidis 2002)

	A	B	C
1	Mike	ThinkPad	Office
2	Mary	iPad	
3	Bob		Dropbox



1. Generate join tree J
2. Generate mapping ϕ
3. Check minimal:
   - Every leaf node contains a column that is mapped by an input column

Validity verification

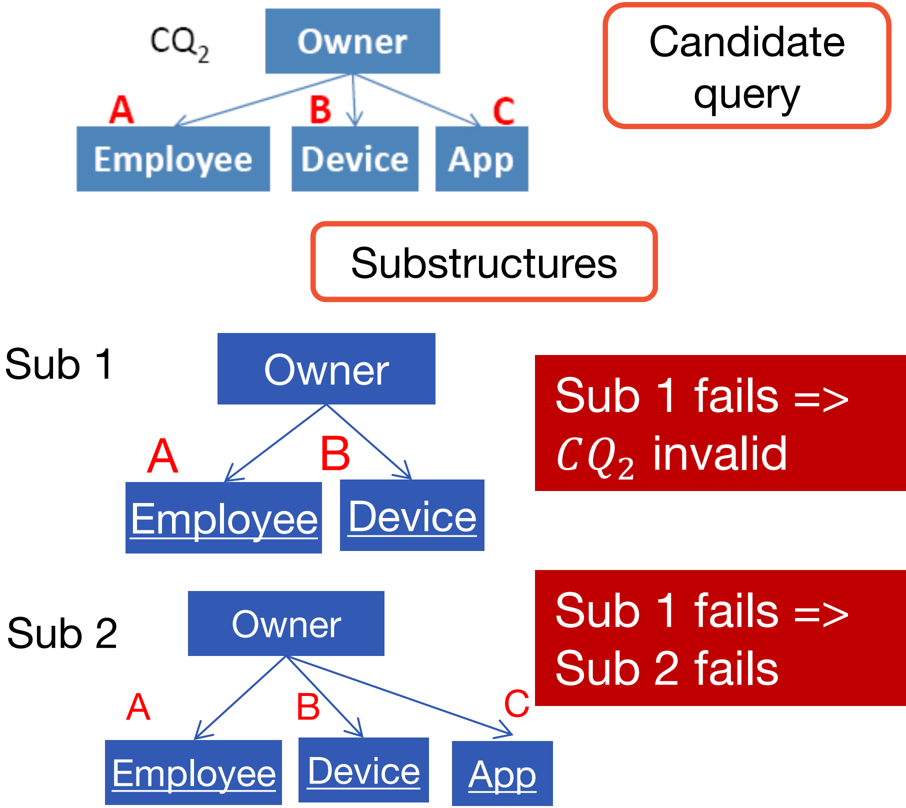
[Shen et al., 2014]

Naïve: check all candidate queries singularly if they return ALL examples

Better: exploit substructures in candidate queries for pruning

Best: adaptively select the substructures to have the min number of evaluations

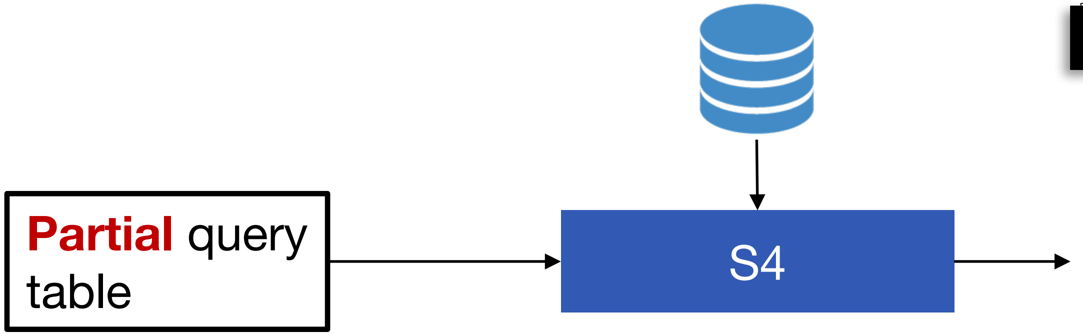
NP-hard

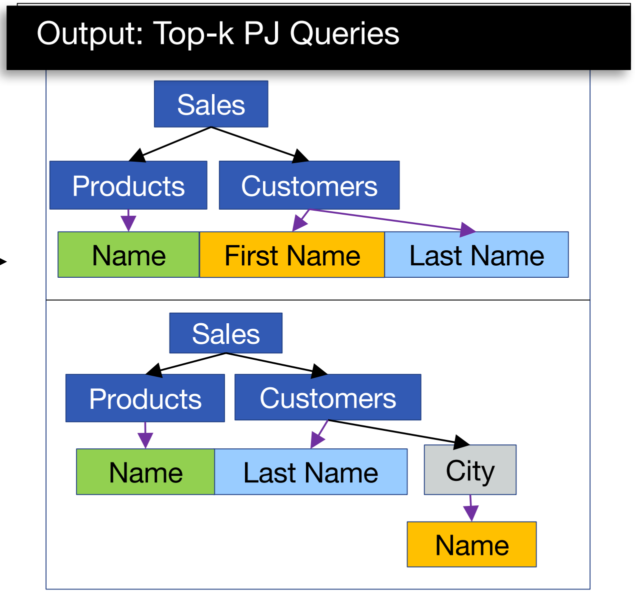


Minimal Project Join REQ

[Psallidas et al., 2015]

Main idea: Allow missing rows/columns and rank the k best queries





	A	B	C
1	John	Smith	Xbox
2	Jill	Hans	Surface

Ranking score

[Psallidas et al., 2015]

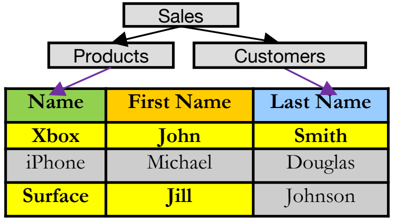
Linear combination of row score and column score

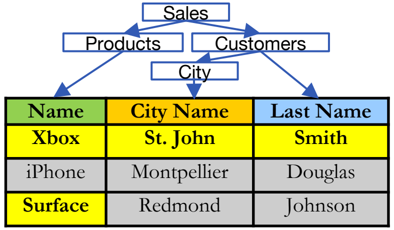
$$\frac{\alpha * score_{row}(Q) + (1-\alpha) * score_{col}(Q)}{|Q|}$$

- $\alpha = 1$ penalizes missing rows
- $\alpha = 0$ penalizes missing columns

Row score

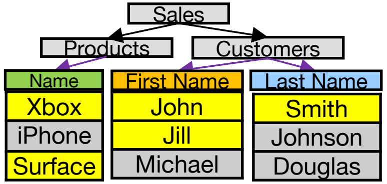
			Row Score	
John	Smith	Xbox	3	3
Jill	Hans	Surface	2	1
			5	4

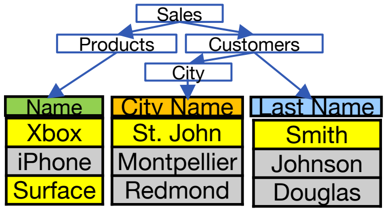




Column score

	John	Smith	Xbox	
	Jill	Hans	Surface	
Column Score	2	1	2	5
	2	1	1	4





S4 Optimizations

[Psallidas et al., 2015]

Upper bound	Row score is always bounded by the column score (row containment is more restrictive) Exploit inverted indexes on columns/rows
Early termination	Stop when current upper bound score is less than the k-th ranked evaluated query Scan queries on decreasing upper bound
Caching	Reuse common subparts in the candidate queries

Reverse engineering queries (REQ)

Lack of user models!

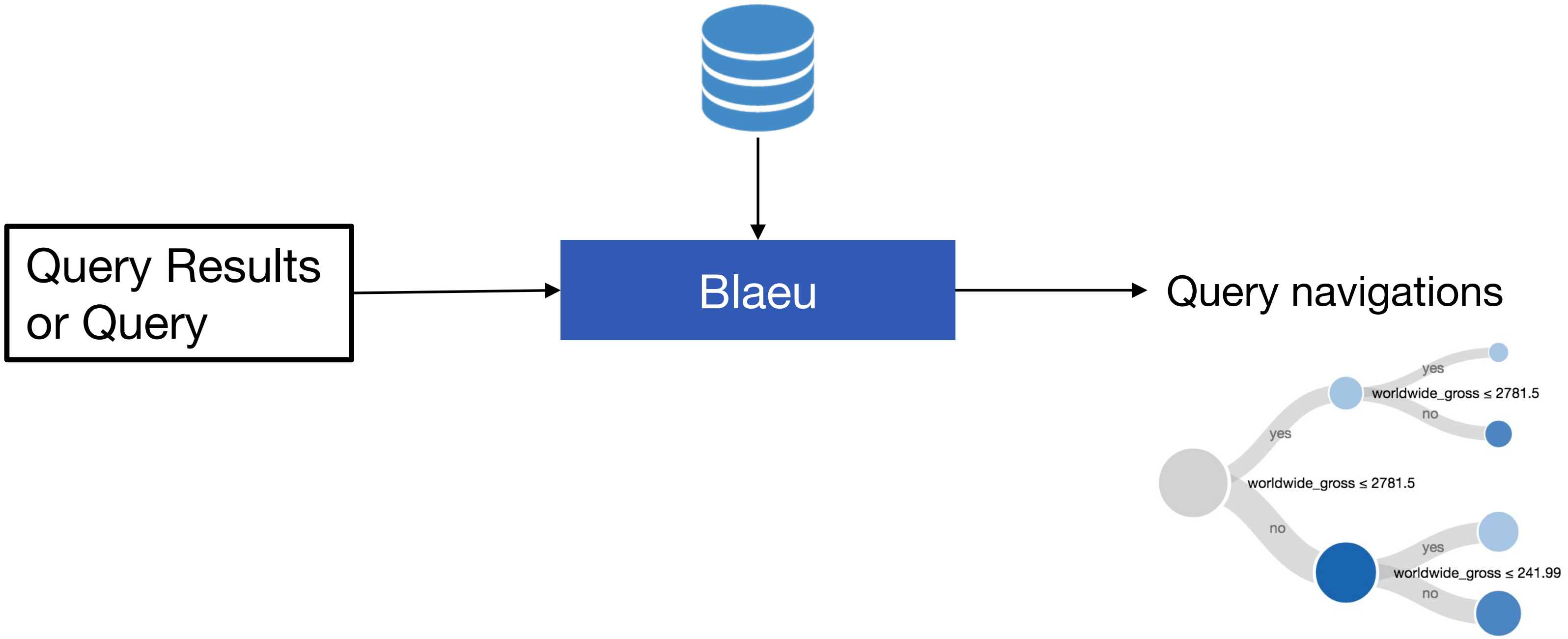
REQ

- Exact
  - One-shot
    - Query by output - TALOS
    - REQ SPJ queries from examples
  - Interactive
    - Query From examples (QFE)
    - Interactive inference of join queries
- Approximate
  - Minimal
    - Discovering Queries based on Examples
  - Top-k
    - S4: Top-k Spreadsheet style

Examples for query suggestion: Blaeu

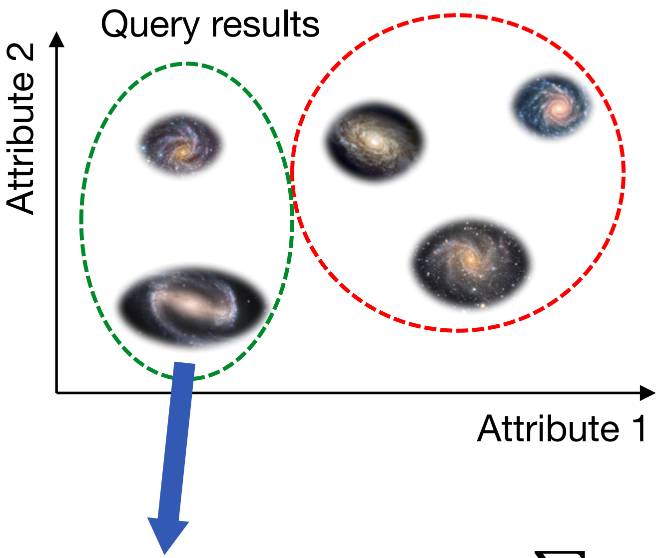
[Sellam et al., 2016]

Main idea: Allow interactive navigation of the query space in a hierarchy



Examples for query suggestion: Blaeu

[Sellam et al., 2016]



$$u: DB \rightarrow \{-1,1\}, U(Q) = \sum_{t \in Q} u(t)$$

User utility

Given a result of an example query Q, explore the data through data maps = partitions

Output: Set of query refinements

Problem: User utility is unknown

- Cluster analysis for result exploration
- Zoom and projection operations
- User model

D. Mottin, M. Lissandrini, T. Palpanas, Y. Velegrakis

Examples for query suggestion: Blaeu

[Sellam et al., 2016]

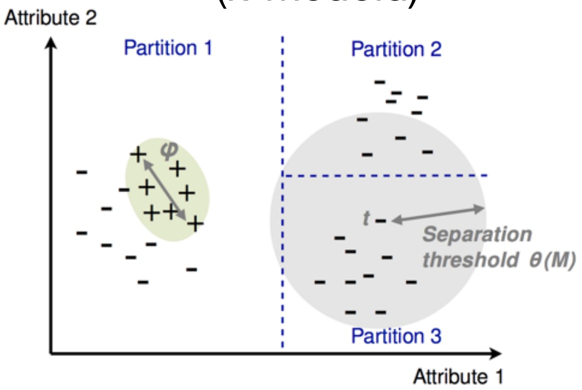
$$u: DB \rightarrow \{-1,1\}, U(C) = \sum_{t \in C} u(t)$$

Unknown User utility

Find the partition $\mathcal{C} = \{C_1, \dots, C_n\}$ of the results of Q such that exists $C_j \in \mathcal{C}: U(C_j) > U(Q)$

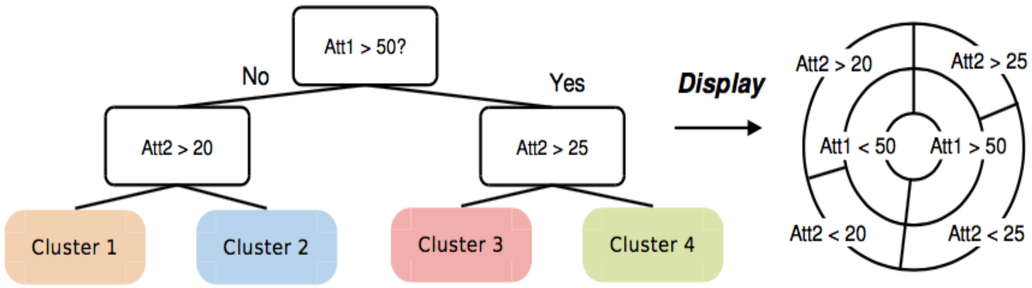
Solution: interesting tuples are close to each other within a maximum separation threshold $\theta(\mathcal{C})$

Detect clusters (k-medoid)



Inference

Organize clusters



Where we are

Relational databases

Textual data

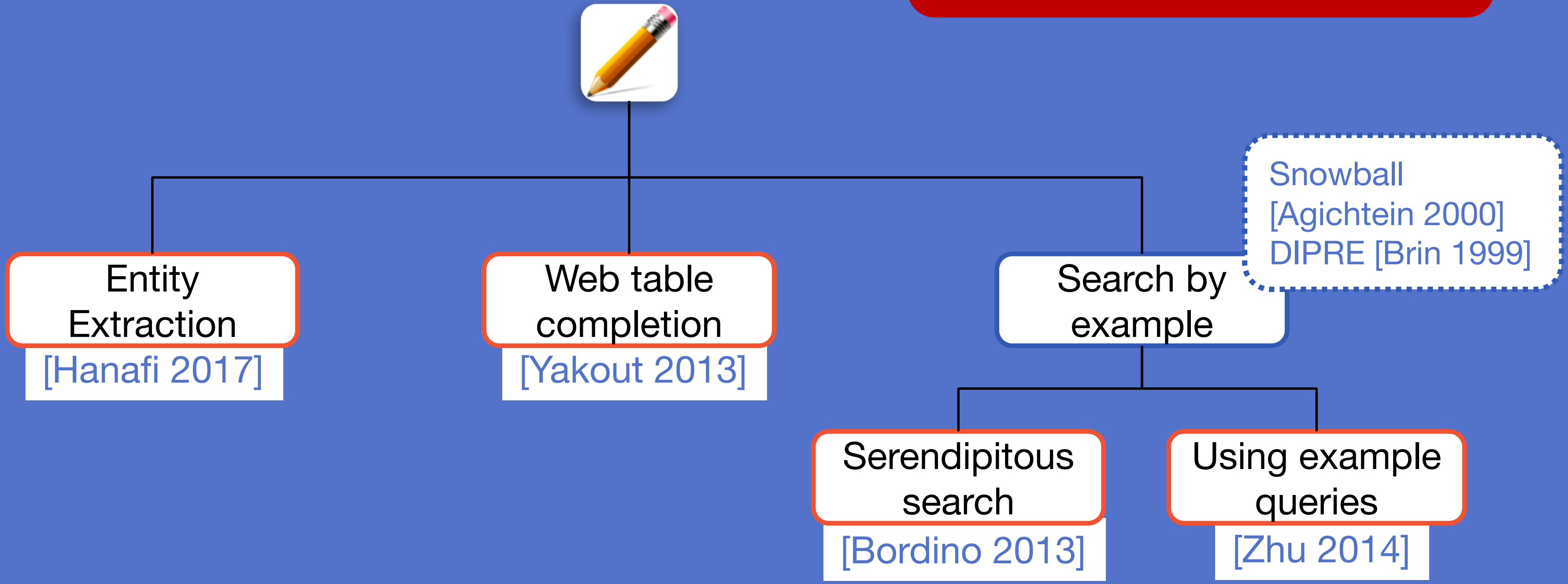
Graphs and networks

Machine learning

Challenges and Remarks

Examples for textual data

Few methods for textual data using examples



Entity extraction by-example (SEER)

[Hanafi et al., 2017]

Main idea: Create rules to extract wanted information from documents using examples

definition) increased 9.6 percent, the number of murders increased 6.2 percent, aggravated assaults increased 2.3 percent, the number of rapes (revised definition) rose 1.1 percent, and robbery violations were up 0.3 percent.
Violent crime increased in all but two city groupings. In cities with populations from 50,000 to 99,999 inhabitants, violent crime was down 0.3 percent, and in cities with 500,000 to 999,999 in population, violent crime decreased 0.1 percent. The largest increase in violent crime, 5.3 percent, was noted in cities with 250,000

SEER

Output: Extraction rules

P: Percentage = 1.0		= 1.0
D: {5, 6} = 0.4	D: {percent, %} = 0.4	= 0.4
R: [0-9]+ = 0.2	D: {percent, %} = 0.4	= 0.3

Learning rules

[Hanafi et al., 2017]

Example: **5 percent up**

1. **Enumerate** possible primitives per example token

5		percent	
	P: Number		L: ‘percent’
	P: Integer		R: [A-Za-z]+
	L: ‘5’		T: 0-1
	R: [0-9]+		

. . .

2. **Assign** scores to primitives

Token gap Regex < Literal Dictionary < Pre-builts

Dubai : T: 0-1 < L: ‘Dubai’ < P: City

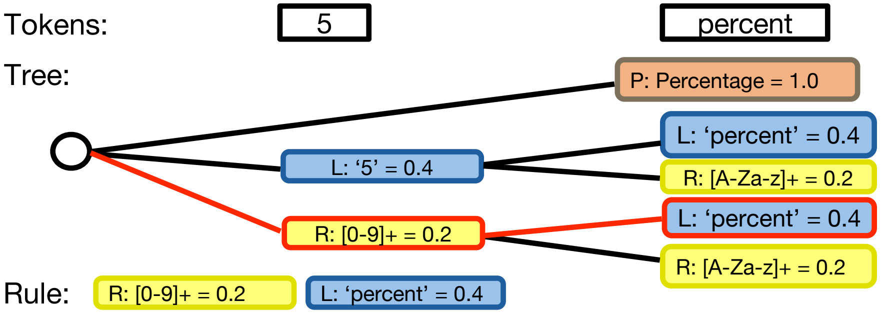
0 — 1

Learning rules (cont'd)

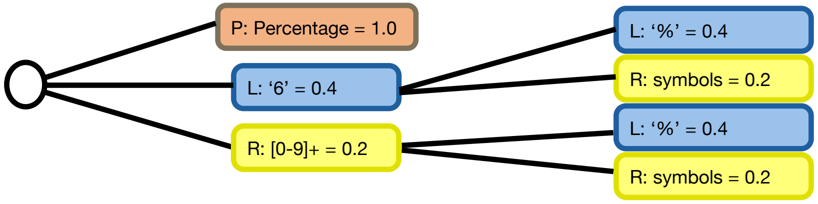
[Hanafi et al., 2017]

3. Generate rules

Example: **5 percent**

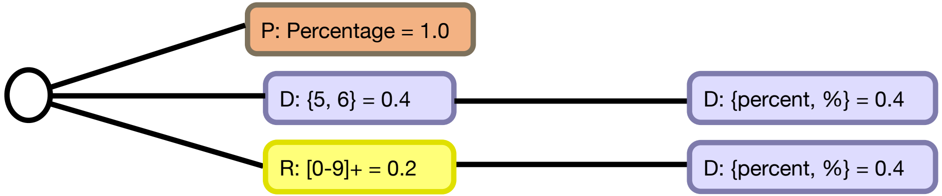


Example: **6%**



4. Merge

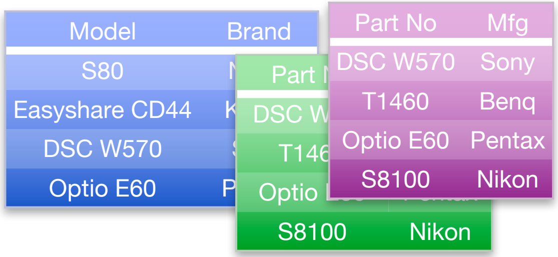
Intersection: [**5 percent, 6%**]



Web tables completion (InfoGather)

[Yakout et al., 2012]

Main idea: Complete tables using partial information about tuples



Web tables

Model	Brand
S80	
A10	
GX-1S	
T1460	

Incomplete table

→ **InfoGather** →

Model	Brand
S80	Benq
A10	Innostream
GX-1S	Samsung
T1460	Benq

Complete table

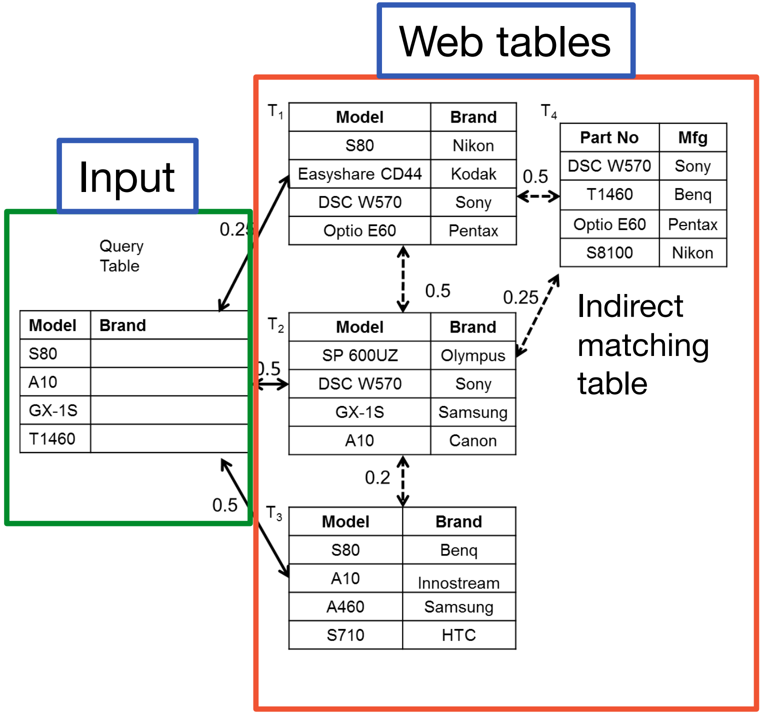
Augmentation framework

[Yakout et al., 2012]

Direct Match Approach (DMA)

- Traditional schema matching techniques using the attribute names and the values in the column

$$S_{DMA}(T) = \begin{cases} \dfrac{|T \cap_K Q|}{\min(|Q|, |T|)} & if\ Q.A \approx T.B \\ 0 & otherwise \end{cases}$$



Ranking tables using PageRank

- **PageRank**

- **Personalized PageRank (PPR)**

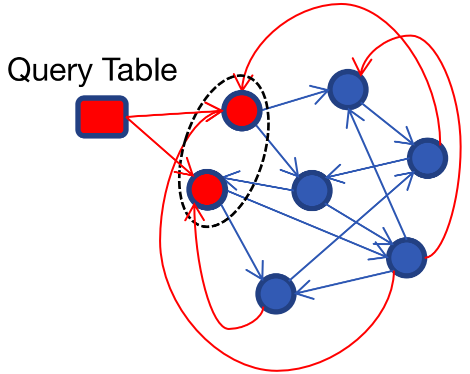
Adjacency matrix

$$\pi_u(v) = \epsilon\,\delta_u(v) + (1-\epsilon) \sum_{\{w|(w,v)\in E\}} \pi_u(w)\alpha_{w,v}$$

- **Topic Sensitive Pagerank (TSP)**

$$\pi_{\vec{\beta}}(v) = \epsilon\,\vec{\beta} + (1-\epsilon) \sum_{\{w|(w,v)\in E\}} \pi_{\vec{\beta}}(w)\alpha_{w,v}$$

Topic vector



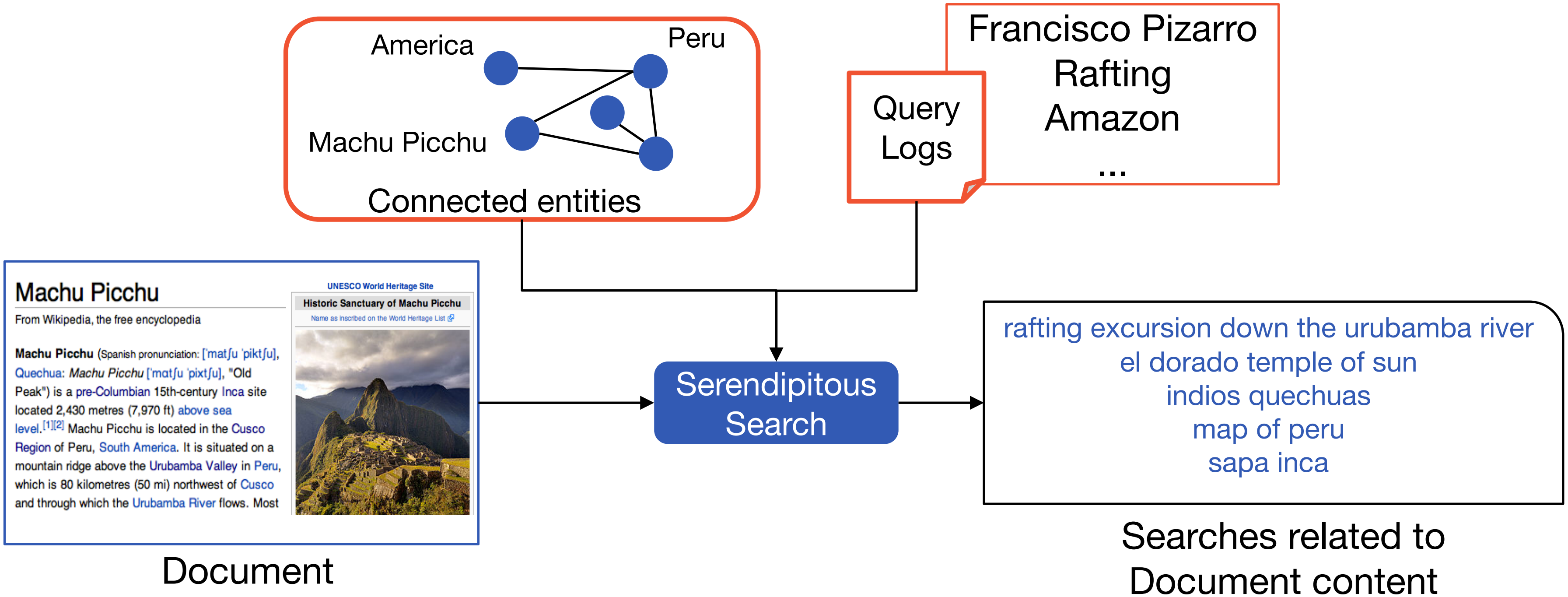
Nodes ➔ Web Tables
Edges ➔ Tables Similarity

Topic weight ➔ DMA score

Serendipitous search

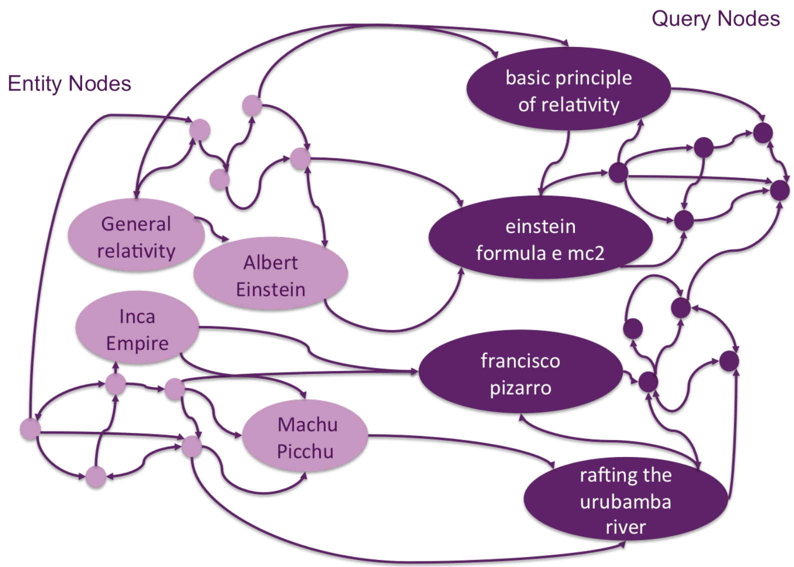
[Bordino et al., 2013]

Main idea: Use related entities and query logs to find serendipitous searches



Find queries using entity-query graph

[Bordino et al., 2013]



Idea: Run Personalized PageRank on entity-query graphs

Query-flow graph with entity nodes

Three types of arcs:

1. **query to query:**

$$w_Q(q_i \to q_j) = w_{QFG}(q_i \to q_j)$$

2. **entity to query**

Frequency-based approach

$$w_{EQ}(e \to q) = \frac{f(q)}{\sum_{q_i | e \in X_E(q_i)} f(q_i)}$$

3. **entity to entity**

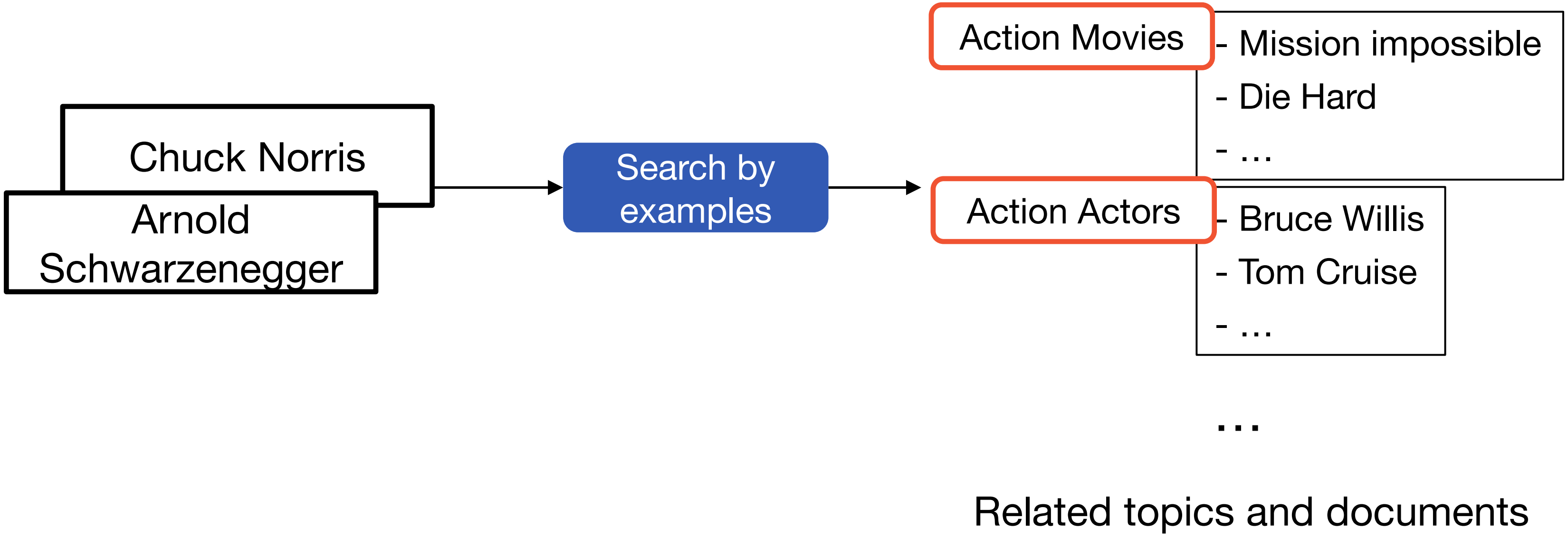
The more queries entities share the higher the probability

$$w_E(e_u \to e_v) = 1 - \prod_{i=1,\dots,r} \left(1 - \boxed{p_{q_{i_s} \to q_{i_t}}(e_u \to e_v)}\right)$$

Search by multiple examples

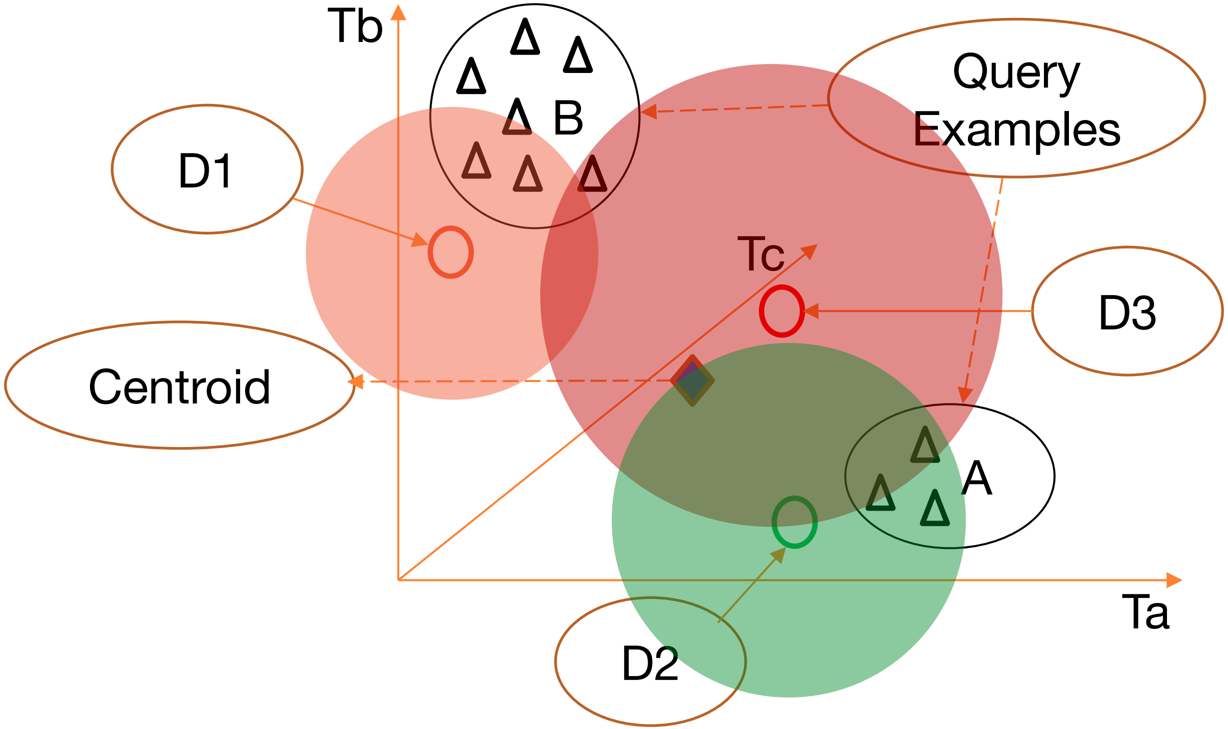
[Zhu et al., 2014]

Main idea: Document examples are used to find topics



Nearest neighbor approach

[Zhu et al., 2014]



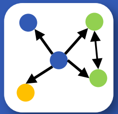
Main Idea:

The similarity is an aggregation over the distances between document D_i and its nearest query example

Where we are

Relational databases

Textual data

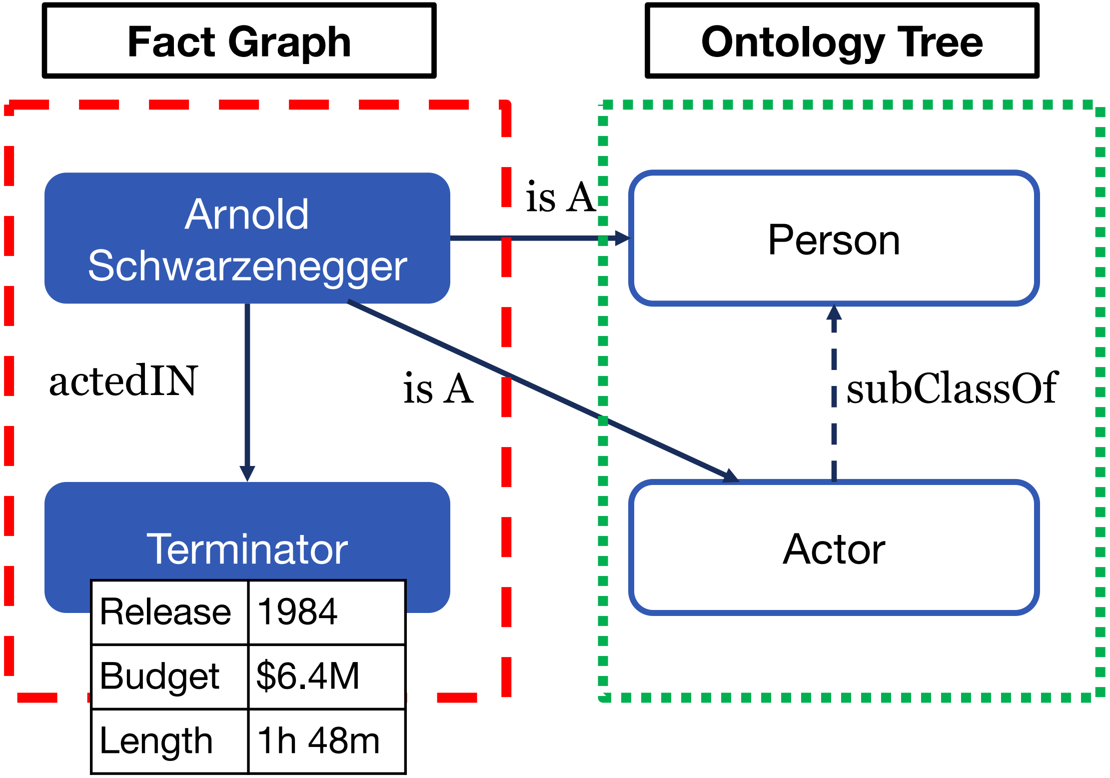


Graphs and networks

Machine learning

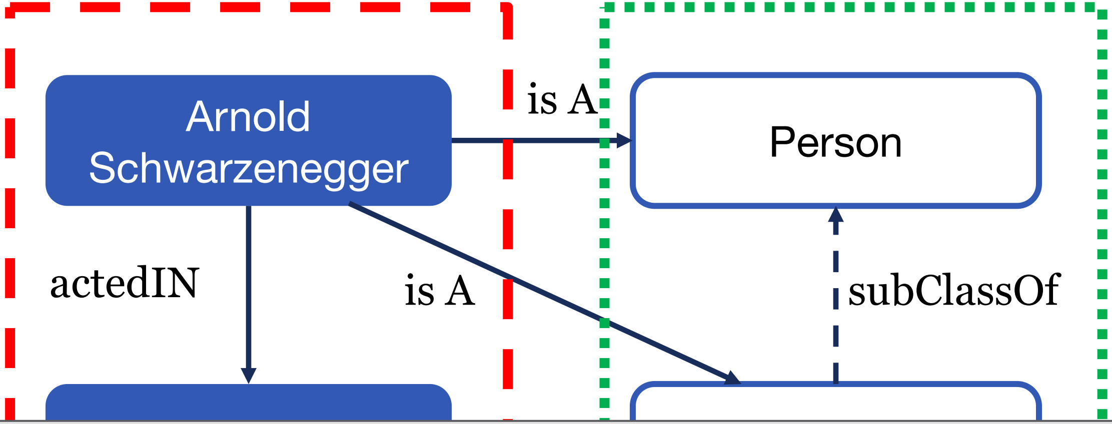
Challenges and Remarks

Graphs



Fact Graph
Ontology Tree
Arnold Schwarzenegger
is A
Person
actedIN
is A
subClassOf
Terminator
Actor
Release 1984
Budget $6.4M
Length 1h 48m

Graphs



RDF

```
(subject,predicate,object)

(Arnold_Schwarzenegger,isA,Person)
(Actor, subClassOf, Person)
(Arnold_Schwarzenegger, actedIn, Terminator)
```

Exemplar Queries

[Mottin et al., 2014]

Input: Q_e, an example ***element*** of interest
Output: set of elements in the desired result set

Nodes/Entities
Edges/Facts
Structures

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*

Exemplar Queries

[Mottin et al., 2014]

Input: Q_e, an example ***element*** of interest

Output: set of elements in the desired result set

Nodes/Entities
Edges/Facts
Structures

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*
- **[OPTIONAL]** return only the subset A^R that are **relevant**

SIMILARITY

Nodes

- **Connectivity**
  - Mediator Nodes [Ruchansky'15]
  - Clusters [Perozzi'14]
- **Properties**
  - Entity Search [Metzger'13, Sobczak'15]

Structures

- **Queries**
  - Path Queries [Bonifati'15]
  - SPARQL [Arenas'16]
- **(Edge-)Labels**
  - Entity Tuples [Jayaram'15]
  - Graph Structures [Mottin'14]

CHALLENGE: DISCOVER USER PREFERENCE

CHALLENGE: EFFICIENT SEARCH

The Minimum Wiener Connector Problem

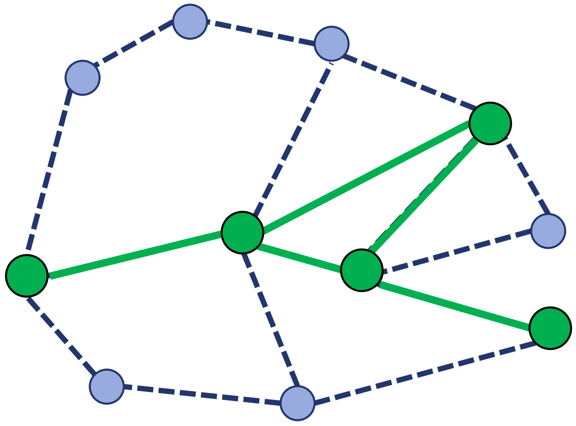
[Ruchansky, et al., 2015]

Model: **Unlabeled Undirected Graph**

Query: A set of **Nodes Q**

Similarity: Shortest-Path **distance**

Output: A Set of **Connector Nodes H**
"*explains*" connections in **Q**



Connectors:
Nodes with **HIGH closeness** to **ALL** the inputs

Similar to a Steiner-Tree but **overall pairwise distances** are optimized

Case: Infected Patients
→ Culprit/Other Infected

Case: Target Audience
→ Influencers

The Minimum Wiener Connector Problem

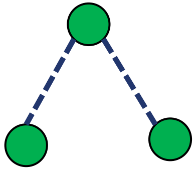
[Ruchansky et al., 2015]

Model: **Unlabeled Undirected Graph**

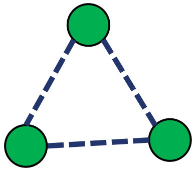
Query: A set of **Nodes Q**

Similarity: Shortest-Path **distance**

Output: A Set of <u>**Connector Nodes**</u> **H**
minimize the sum of **pairwise shortest-path-distances** between nodes in the **connector *H***



W=1+2+1 =4



W=1+1+1 = 3

Sometimes The Best Solution is NOT A Tree

NP-Hard

Called: Wiener Index.

*tradeoff between **size***

*and **average distance***

$$min \sum_{(u,v)\in H} d(u,v)$$

d(u, v) is the shortest-path distance

Approximate minimum Wiener Index Connector

[Ruchansky et al., 2015]

CHOOSE r & $\lambda \in [1, \log_{(1+\beta)} |V|]$

All Pairwise Distances

Distances from a root r

Measure distance in H

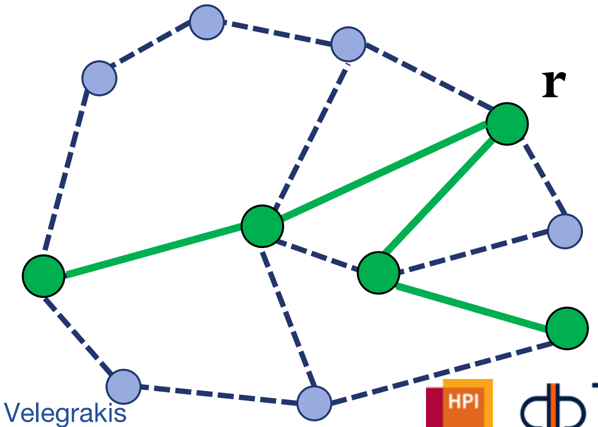
Precomputed distance in G

Edge Weights

$$w(u, v) = \lambda + \frac{max\{d_G(r,u), d_G(r,v)\}}{\lambda}$$

Approximated with
Edge-Weighted SteinerTree

Enumerate Candidate Solutions
for $r \in Q$ & λ
and **keep best**



Focused Clustering and Outlier Detection

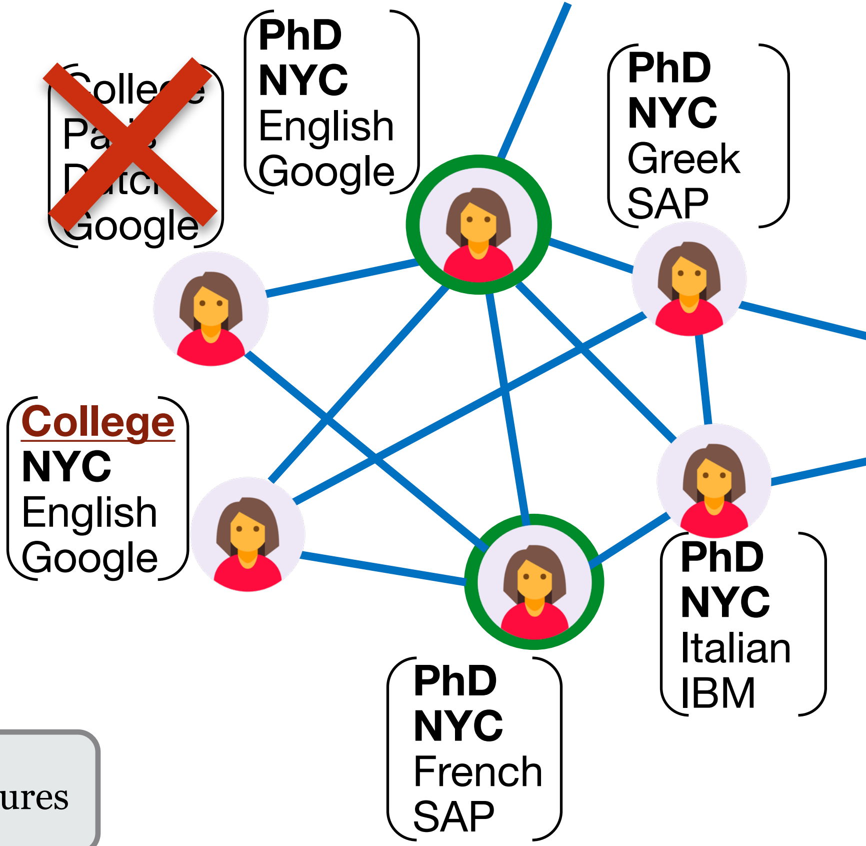
[Perozzi et al., 2014]

Model: **Unlabeled Undirected Graph with Node Attributes**

Query: A set of **Nodes Q**

Similarity: **Attribute Values & Connectivity** (*to be inferred*)

Output: **Clusters** of Nodes: Dense & Coherent **+Cluster Outliers**



Case: Target Users → Community with same interests

Case: Products→ Co-purchased products with similar features

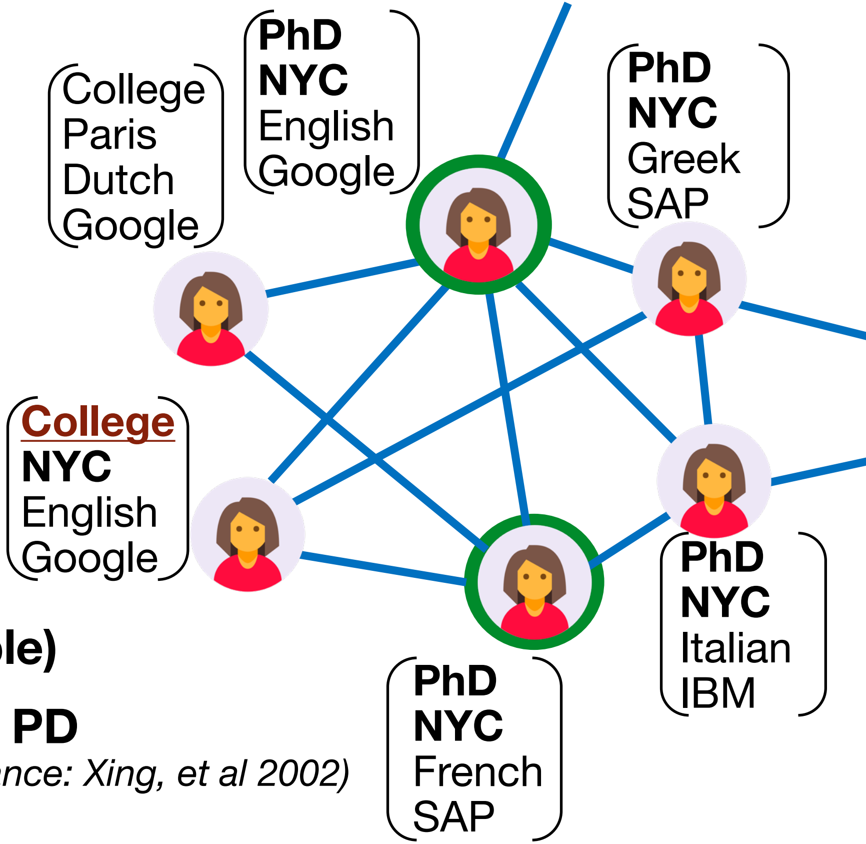
Focused Clustering and Outlier Detection

[Perozzi et al., 2014]

TASK: Infer "FOCUS" , important attributes

attribute weights β

$$\begin{pmatrix} \textbf{PhD} \\ \textbf{NYC} \\ \text{English} \\ \text{Google} \end{pmatrix} \quad \begin{pmatrix} \textbf{PhD} \\ \textbf{NYC} \\ \text{French} \\ \text{SAP} \end{pmatrix} \rightarrow \begin{pmatrix} \mathbf{0.5} \\ \mathbf{0.5} \\ 0 \\ 0 \end{pmatrix}$$



1. Set of similar pairs, PS (from Q)

2. Set of dissimilar pairs, PD (random sample)

3. Learn a distance metric between PS and PD

(Distance Metric Learning, inverse Mahalanobis distance: Xing, et al 2002)

Focused Clustering and Outlier Detection

[Perozzi et al., 2014]

LOCAL
clusters

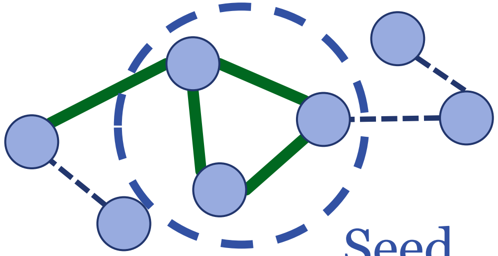
TASK: Extract Clusters on Focused Graph

attribute weights β -> Edge Weight

1. Find Starting Set of Candidates

1.a Drop low-weight edges

1.b Extract **Strongly Connected Component** C_1, C_2, ...



2. Grow Clusters around Candidates

2.a Compute conductance of **C:** **$\phi^{(w)}$ (C, G)**

2.b Select node to add to **C'**: **best improvement to $\Delta\phi^{(w)}$ (C,C')** *(greedy)*

2.c Prune Underperforming nodes

3. Detect Outliers: High *unweighted* conductance

SIMILARITY

Nodes

Connectivity

Mediator Nodes
[Ruchansky'15]

Clusters
[Perozzi'14]

✓

Properties

Entity Search
[Metzger'13,
Sobczak'15]

Structures

Queries

Path Queries
[Bonifati'15]

SPARQL
[Arenas'16]

(Edge-)Labels

Entity Tuples
[Jayaram'15]

Graph Structures
[Mottin'14]

iQBEES: Entity Search by Example

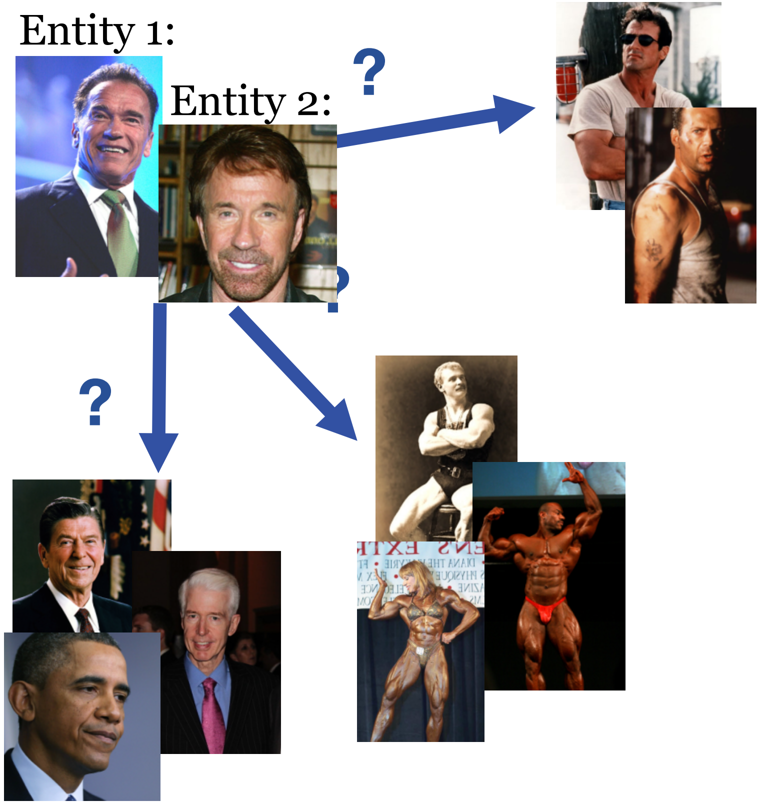
[Metzger et al., 2013, Sobczak et al., 2015]

Model: Knowledge Graph

Query: A set of Entities Q

Similarity: shared semantic properties

Output: A Set of Similar Entities *ranked*

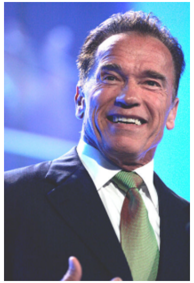


Case: Products→ Find Similar Products

Case: Social Media→ User recommendation

Maximal Aspects

[Metzger et al., 2013, Sobczak et al., 2015]



Prune generic aspects

Rank Set of aspects

?x type BodyBuilder

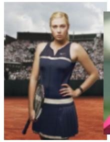
?x type AmericanActor

?x type AmericanActor

?x type GovernorCalifornia

?x hasHeight 1.88m

?x type Entity



?x type AmericanActor

?x actedIn TheExpendables

?x type ActionActor

Adding any aspect → E(A)={Arnold}

Include Typical Types

use most specific type

REPEATABLE *Update Q*

SIMILARITY

Nodes

Connectivity

Mediator Nodes
[Ruchansky'15]

Clusters
[Perozzi'14]

✓

Properties

Entity Search
[Metzger'13,
Sobczak'15]

✓

Structures

Queries

Path Queries
[Bonifati'15]

SPARQL
[Arenas'16]

(Edge-)Labels

Entity Tuples
[Jayaram'15]

Graph Structures
[Mottin'14]

Learning Path Queries on Graphs

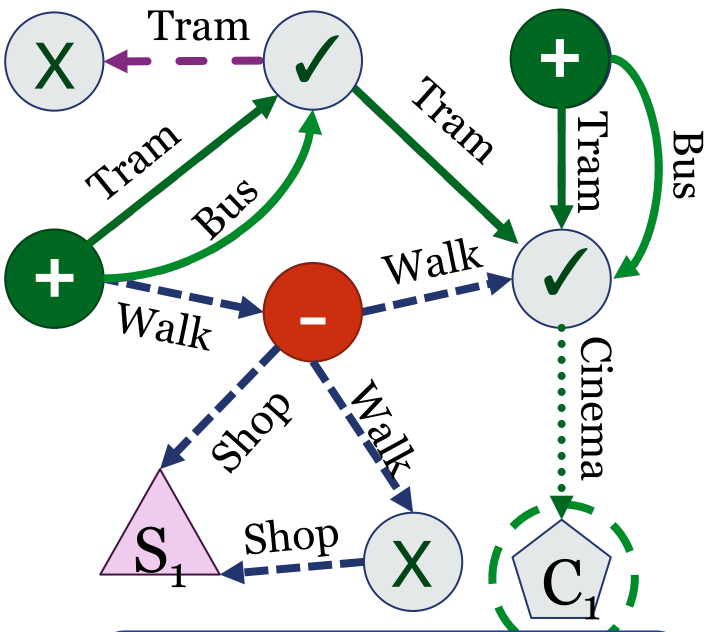
[Bonifati et al., 2015]

Model: Edge Labeled Graph

Query: 2 sets of Entities Q^+ , Q^-
Positive, Negative

Similarity: common path query (RegExp)
(bus|tram)*Cinema

Output: A Set of Nodes Satisfying
some paths(Q^+) but NOT paths(Q^-)



Case: Proteins→ Similar interactions/co-expression

Case: Tasks Initiator→ Similar Processes/Behaviours

MONADIC: only starting nodes
extensible to
BINARY/ N-ARY : path from X to Y

Learnability of Path Queries

[Bonifati et al., 2015]

Query: Q+ & Q- (**Positive** & **Negative** examples)

Consistecy: $\forall v \in Q^{+}.\, paths_G(v) \not\subseteq paths_G(Q^{-})$

1. Selecting the Smallest Consistent Paths

Infinite Paths? **Fix maximal length** K **but…**

When to use **Kleene star** * **?**

$$\mathrm{C} \mid (\mathrm{A} \cdot \mathrm{B} \cdot \mathrm{C}) \rightarrow (\mathrm{A} \cdot \mathrm{B})^{*} \cdot \mathrm{C}$$

Consistency Check:
PSPACE-complete

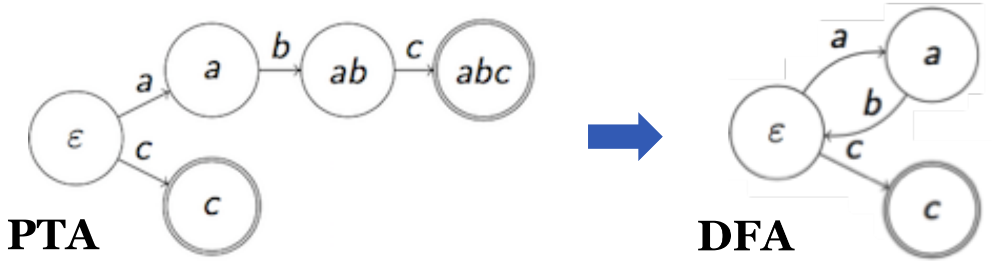
Enumerate Paths
Up to Fixed distance

For paths of Length N
K = 2 X N +1

2. Generalize SCP

a. Construct Prefix-Tree Acceptor

b. Generalize into DFA with Merge



Reverse engineering SPARQL queries

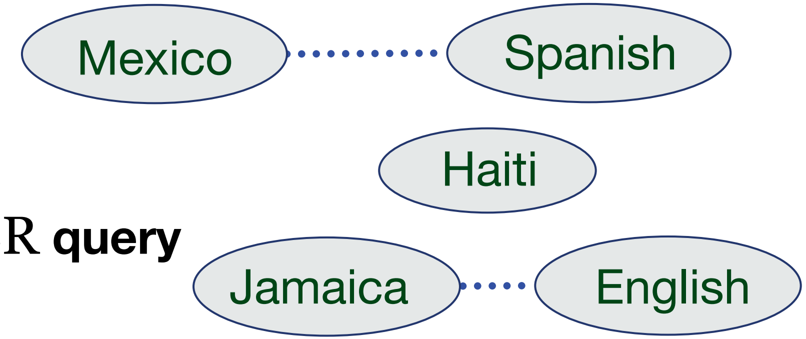
[Arenas et al., 2016]

Model: Knowledge Graph

Query: Set of *ANSWERS**

Similarity: common AND/OPT/FILTER query

Output: A SPARQL QUERY/RESULT



Case: Open Data → Query Unknown Schema

Case: Novice User → Avoid SPARQL

	?e1	?e2
M1	Mexico	Spanish
M2	Haiti	
M3	Jamaica	English

Reverse engineering SPARQL queries

[Arenas et al., 2016]

Query: Set of ***Variable Mappings***

	$?X$	$?Y$	$?Z$
M1	`John`		
M2	`Mary`	`mary@email.eu`	
M3	`Lucy`		`Roses Street`

$\boxed{(?X, \texttt{type}, \texttt{Person})}\ ?X \neq \texttt{me}$

$\swarrow$ OPT $\qquad$ OPT $\searrow$

$\boxed{(?X, \texttt{email}, ?Y)} \qquad \boxed{(?X, \texttt{addr}, ?Z)}$

Enumerate all possible SPARQL queries satisfied by the mappings

INTRACTABLE

Σ_2^p –complete

coNP –complete

Build tree-shaped SPARQL queries IMPLIED by the mappings

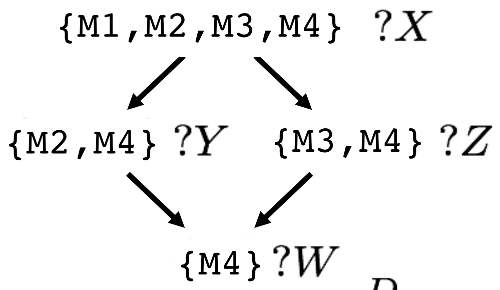
Reverse engineering SPARQL queries

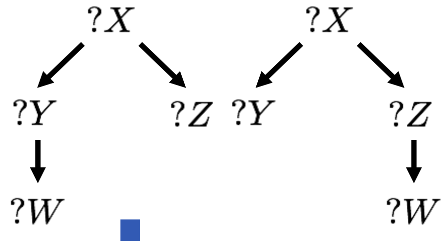
[Arenas et al., 2016]

Query: **Set of *Variable Mappings* Ω**

Ω

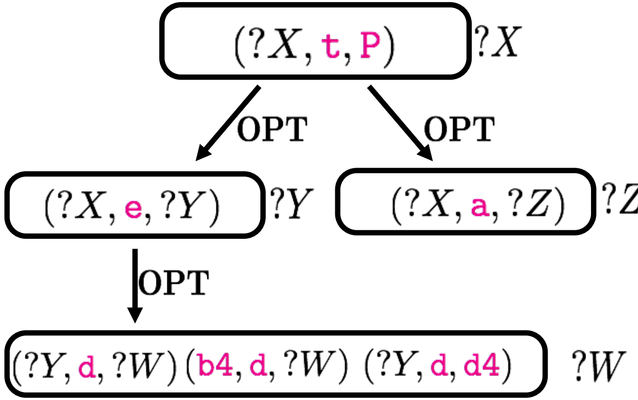
	?X	?Y	?Z	?W
M1	a1			
M2	a2	b2		
M3	a3		c3	
M4	a4	b4	c4	d4





D

M1	(a1, t, P)
M2	(a2, t, P)(a2, e, b2)
M3	(a3, t, P)(a3, a, c3)
M4	(a4, t, P)(a4, e, b4) (a4, a, c4) (b4, d, d4)



Greedy: keep just enough to cover all variables

SIMILARITY

Nodes

Connectivity

Mediator Nodes [Ruchansky'15]

Clusters [Perozzi'14]

✓

Properties

Entity Search [Metzger'13, Sobczak'15]

✓

Structures

Queries

Path Queries [Bonifati'15]

SPARQL [Arenas'16]

✓

(Edge-)Labels

Entity Tuples [Jayaram'15]

Graph Structures [Mottin'14]

Exemplar Queries

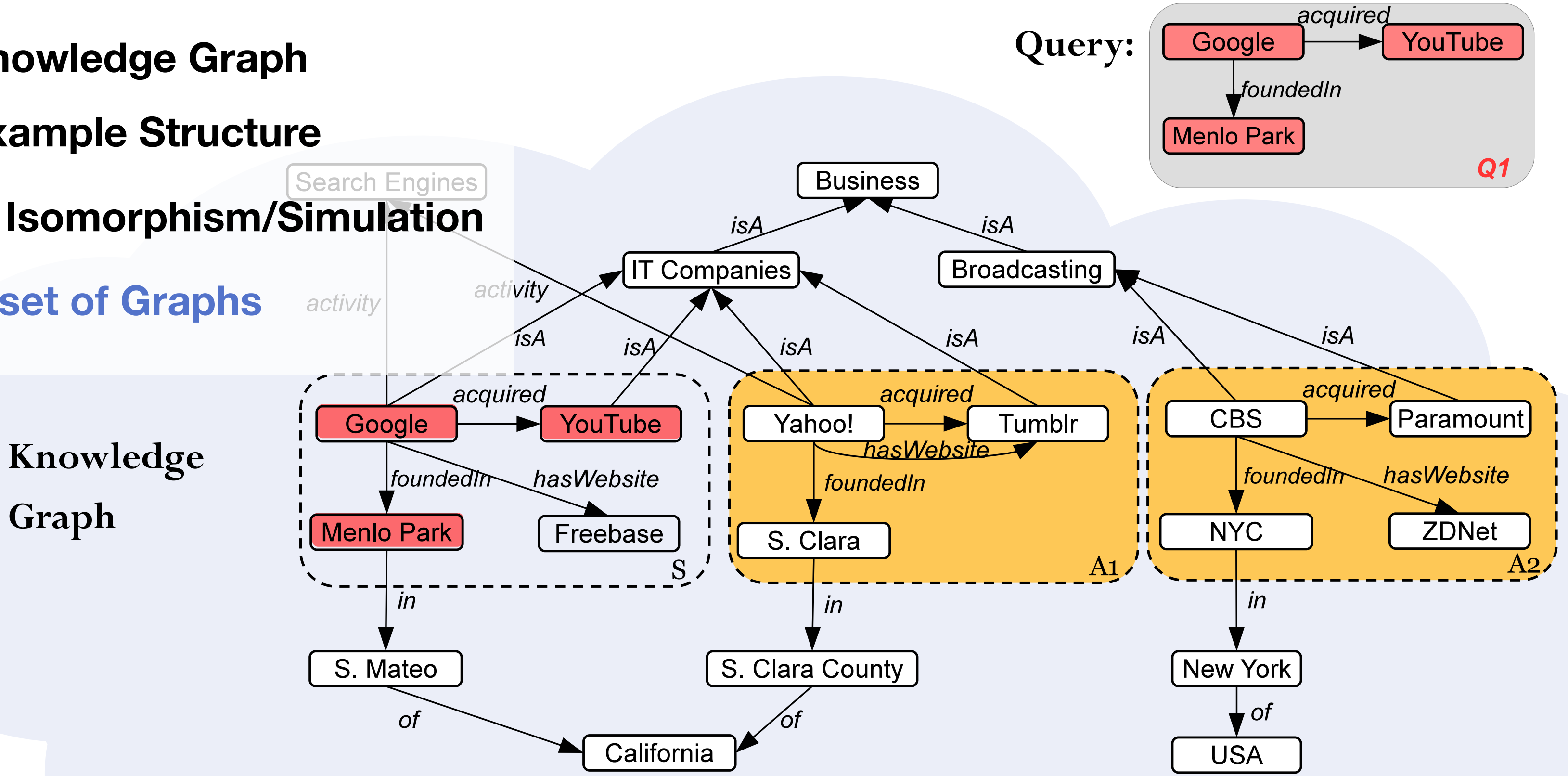
[Mottin et al., 2014]

Model: Knowledge Graph

Input: Example Structure

Similarity: Isomorphism/Simulation

Output: A set of Graphs

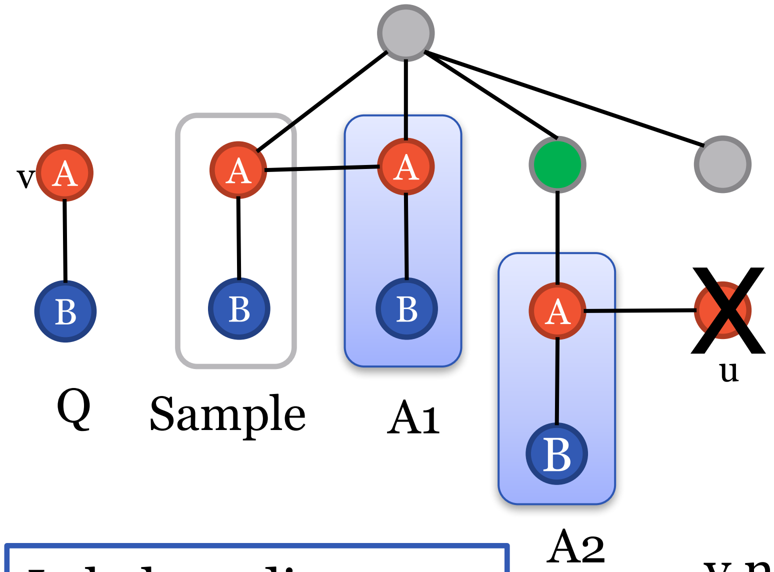


D. Mottin, M. Lissandrini, T. Palpanas, Y. Velegrakis

Computing exemplar queries [Mottin et al., 2014]

NP-complete (subgraph isomorphism)

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



Labels at distance 1

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 iteratively on the query nodes

v neighborhood = {(B,1)}

$\not\subseteq$

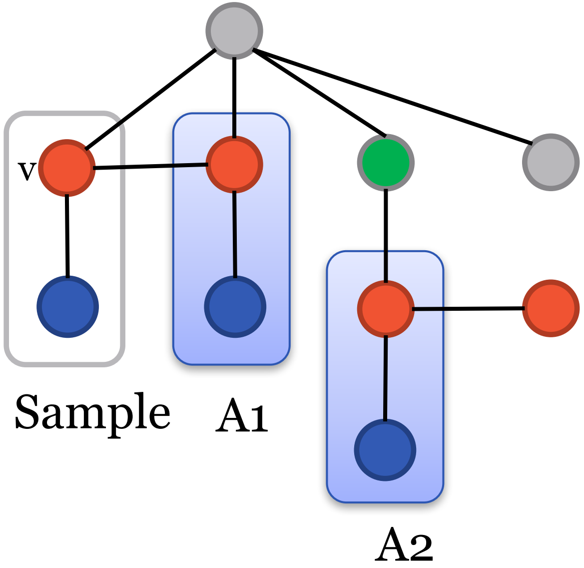
u neighborhood = {(A,1)}

No Match

Computing exemplar queries [Mottin et al., 2014]

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: frequency of the edge-label

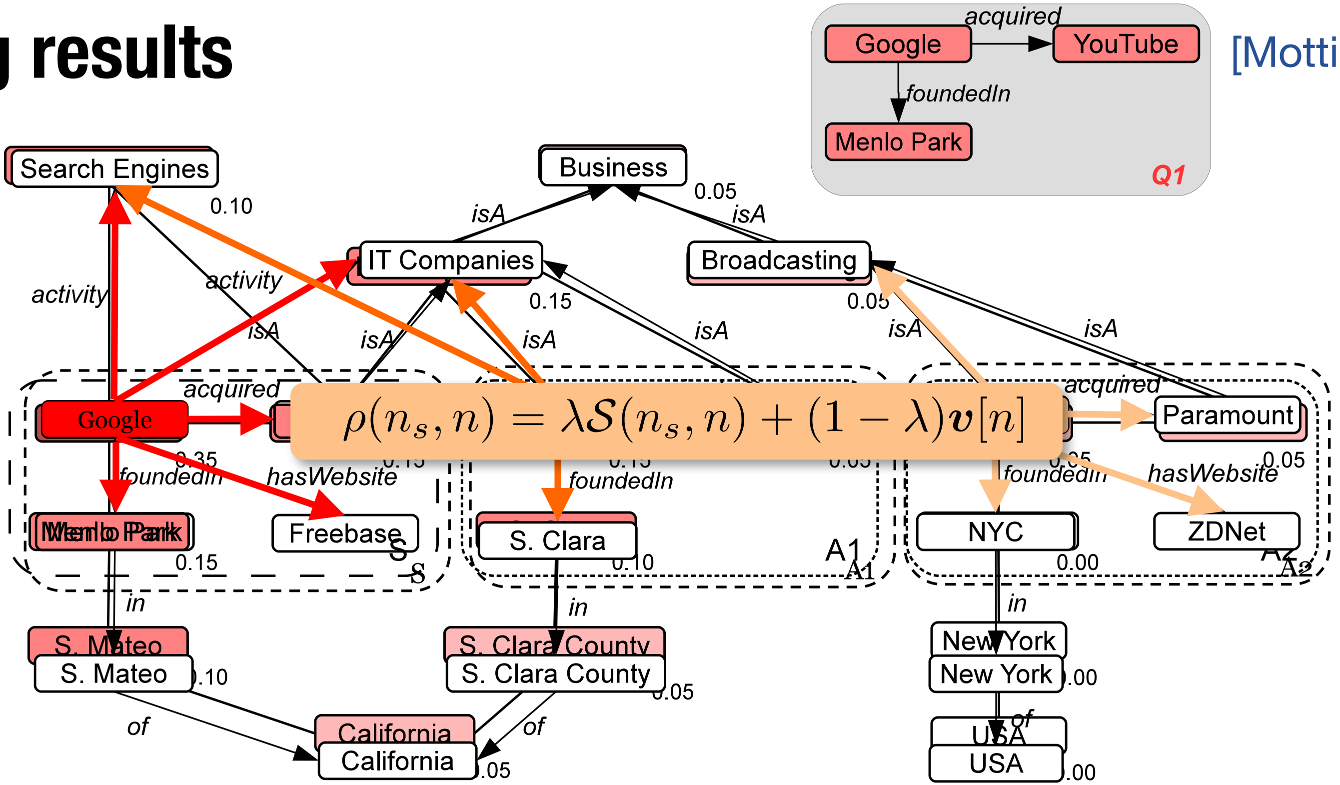
$$I(e_{ij}^{\ell}) = I(\ell) = \log \frac{1}{P(\ell)} = -\log P(\ell)$$

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

D. Mottin, M. Lissandrini, T. Palpanas, Y. Velegrakis

Ranking results

[Mottin et al., 2014]



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

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)

[Jayaram et al., 2015]

Model: Knowledge Graph

Input: Entity Tuples

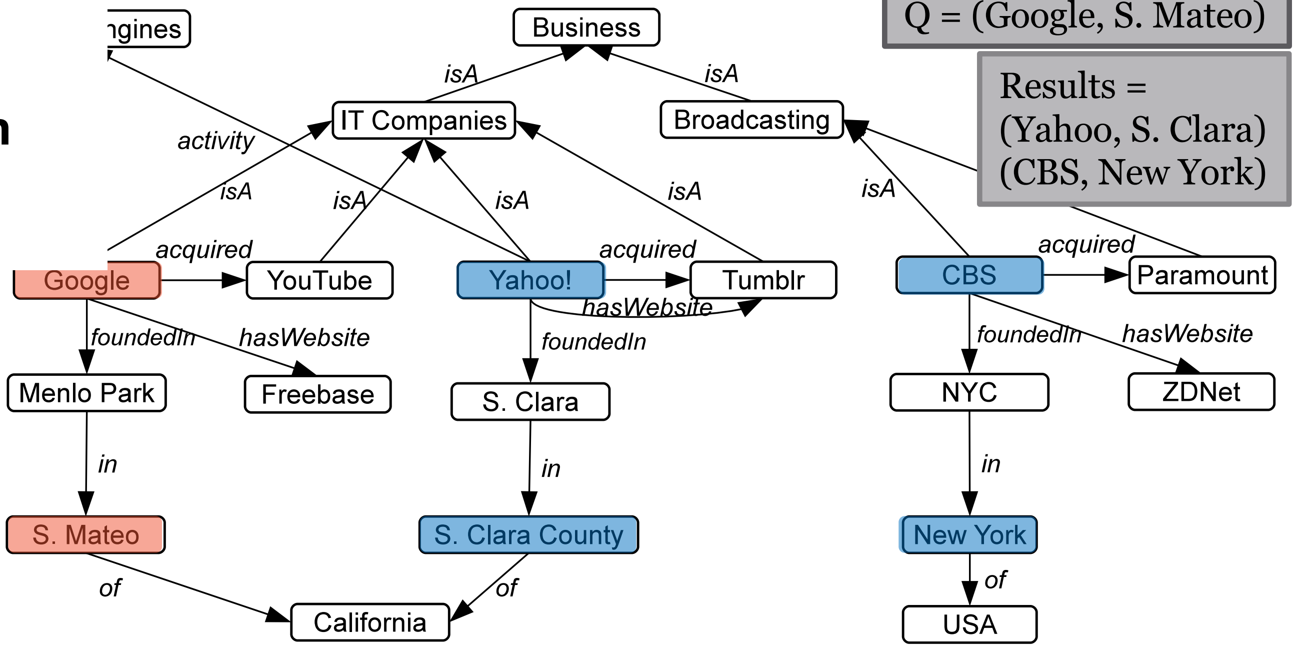
Similarity: Isomorphism

Output: A set of Tuples

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

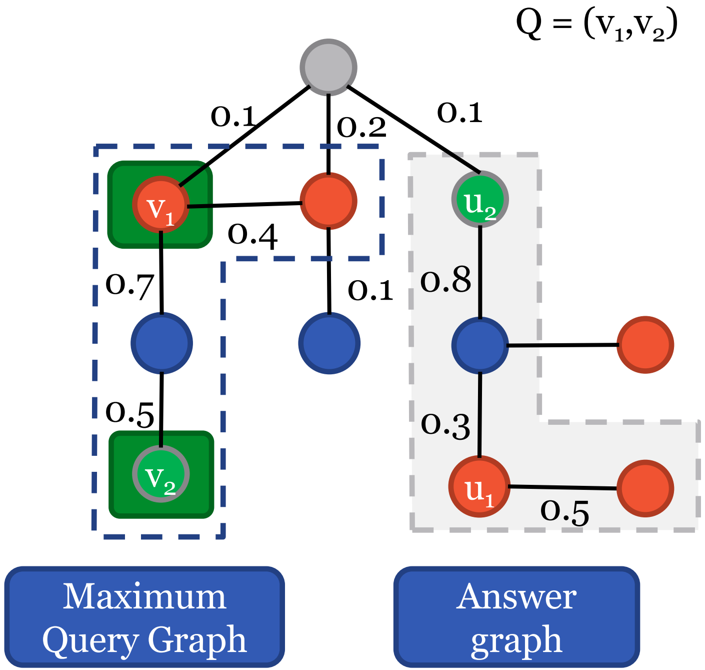
Q = (Google, S. Mateo)

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



GQBE: Maximum Query Graph

[Jayaram et al., 2015]



1. Find the maximum query graph
   - Graph with M edges having the maximum weight
2. Answers subgraph-isomorphic to the query graph

NP-hard

3. Return top-k

Answer score:

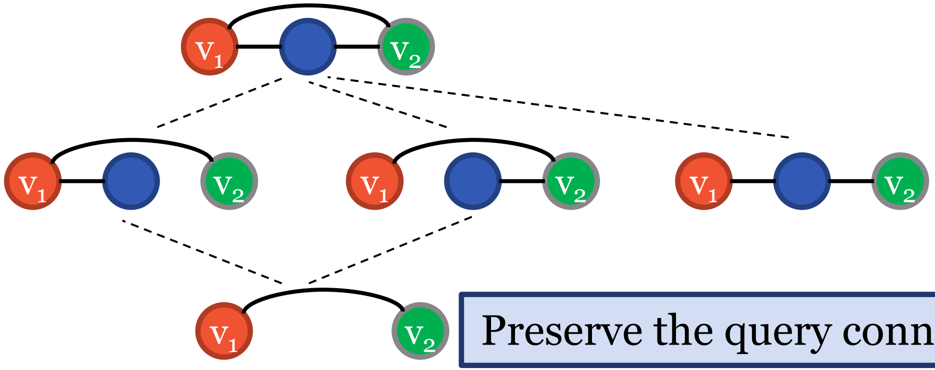
- Sum of query graph weights
- Similarity match between edges in the answer and the query (shared nodes take extra credit)

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

Multiple query tuples

[Jayaram et al., 2015]

Subgraphs of Maximum Query graph



Maximum Query Graph is Very Large

Preserve the query connectivity

Find answers using a lattice obtained removing edges from the union graph

GQBE finds answers for multiple query tuples

1. **Compute a re-weighted union graph of the individual query graphs**

SIMILARITY

Nodes

Connectivity

~~Mediator Nodes [Ruchansky'15]~~

~~Clusters [Perozzi'14]~~

Properties

Entity Search [Metzger'13, Sobczak'15]

Structures

Queries

Path Queries [Bonifati'15]

SPARQL [Arenas'16]

Structures

~~Entity Tuples [Jayaram'15]~~

~~Graph Structures [Mottin'14]~~

Do not Include User Feedback

Where we are

Relational databases

Textual data

Graphs and networks

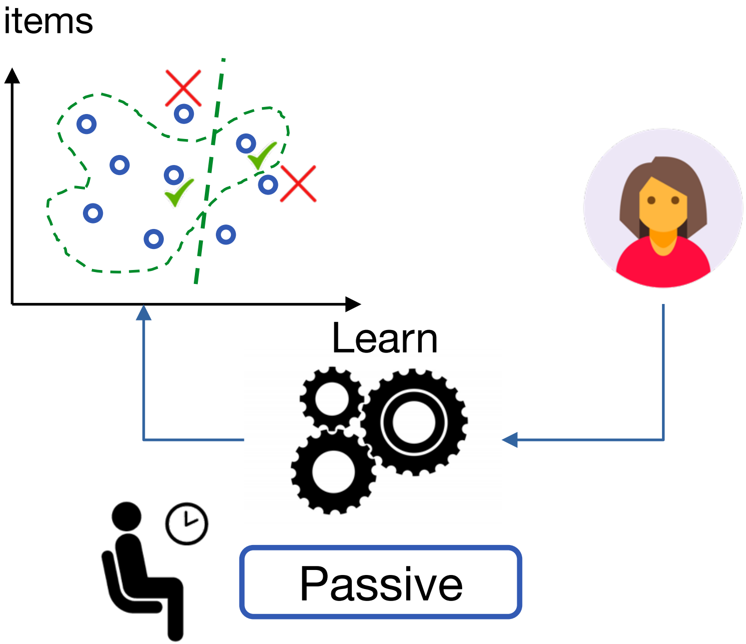
Machine learning

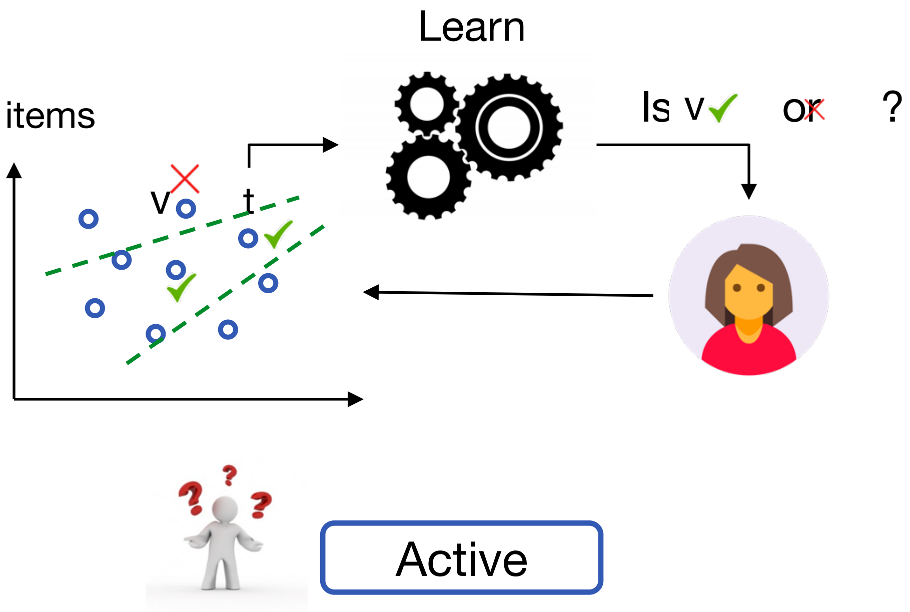
Challenges and Remarks

Online exploration of datasets

Main idea: Learn the items to show online as more points are acquired

Two ways of learning: passive and active





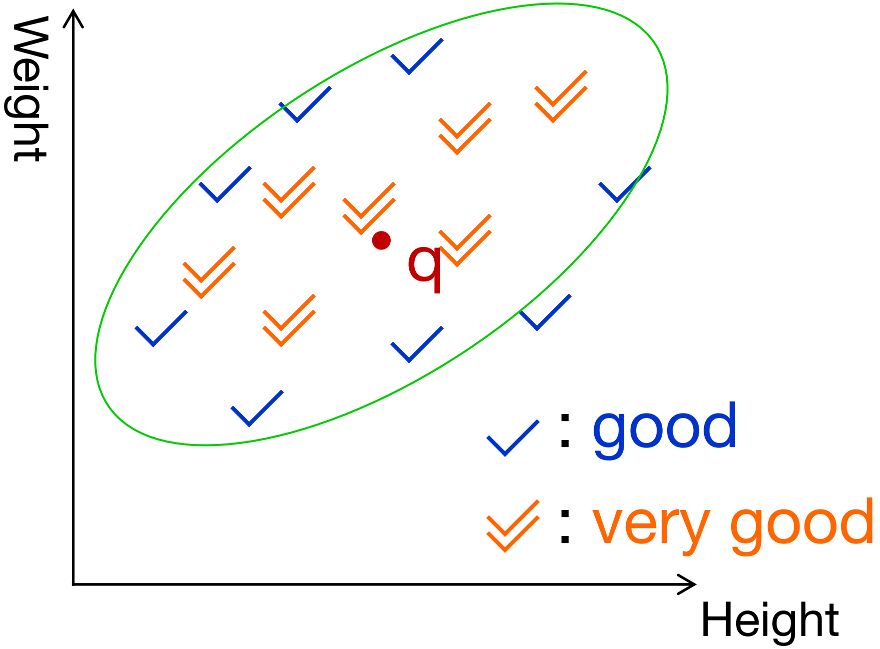
MindReader

[Ishikawa et al., 1999]

Main idea: learn an implicit query from user examples and optional scores

Searching "mildly overweighted" patients

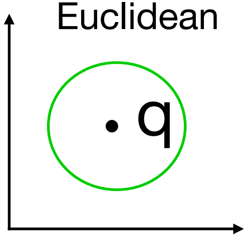
- The doctor selects examples by browsing patient database
- The examples have **"oblique"** correlation
- We can "guess" the implied query



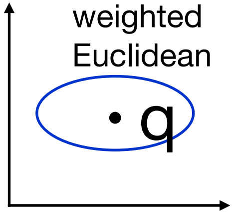
Learning an ellipsoid distance

[Ishikawa et al., 1999]

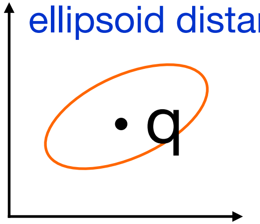
Euclidean



weighted
Euclidean



generalized
ellipsoid distance



Weighted distance matrix

$$D(x,q) = (x-q)^{\top} M (x-q)$$

Implicit query

$$D(x,q) = \sum_{j}^{n} \sum_{k}^{n} m_{jk}(x_j - q_j)(x_k - q_k)$$

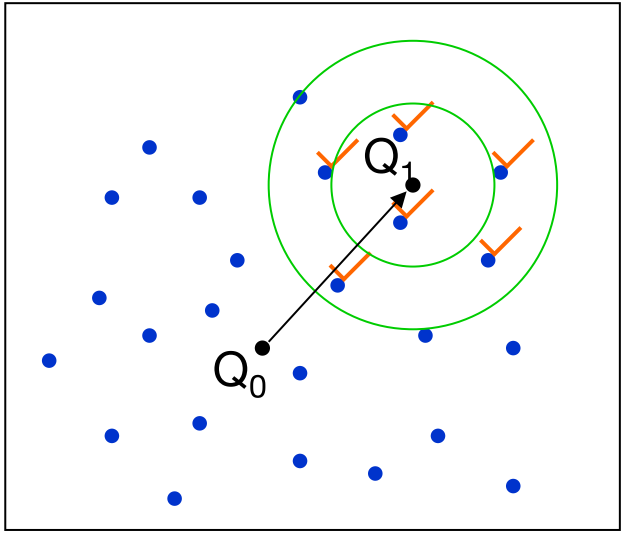
Learn the query minimizing the penalty = weighted sum of distances between query point and sample vectors

$$\begin{aligned} &\textit{minimize} \ \sum_{i} (x_i - q)^{\top} M (x_i - q) \\ &\textit{subject to} \quad \det(M) = 1 \end{aligned}$$

Learning the distance

[Ishikawa et al., 1999]

- Query point is moved towards "good" examples — Rocchio formula in IR



Q_0: query point

● : retrieved data

✓ : relevance judgments

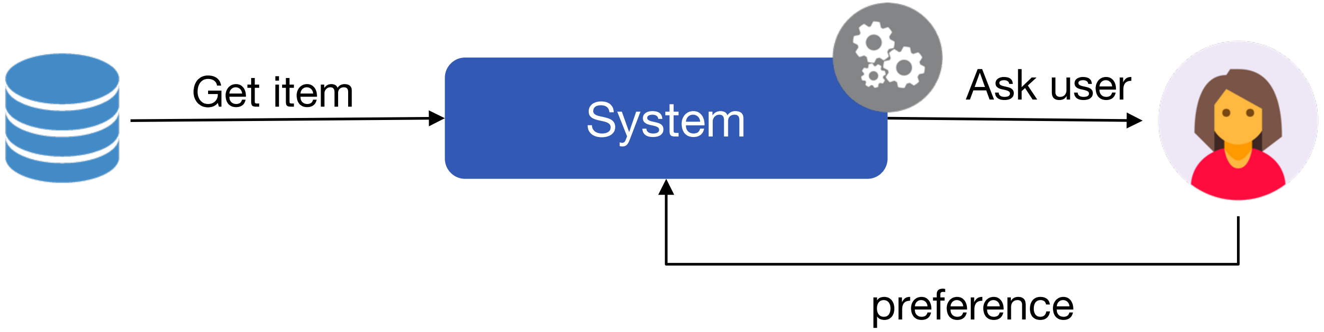
Q_1: new query point

Learning can be done online!!!

Active learning for online query systems

[Vanchinathan et al., 2015]

Main idea: the system “query” the user to understand her preferences

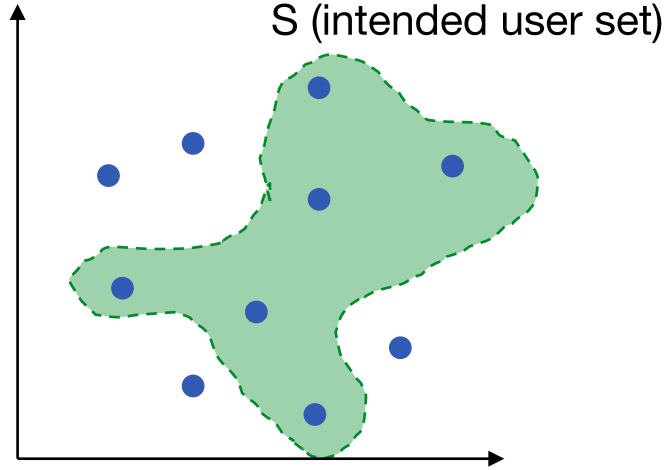


Learn unknown preferences and minimize the number of questions to the user

Learning unknown preferences

[Vanchinathan et al., 2015]

Problem: Find a set S that maximize the user preference within a budget (e.g., number of interactions)



User preferences

$$\arg\max \sum_{v \in S} pref(v)$$

$$\text{subject to } \; Cost(S) \leq budget$$

Cost for the set S

Background: Gaussian processes

[Bishop et al., 2006]

Idea: Model the user preferences as a Gaussian Process

A Gaussian Process (GP) is an infinite set of variables, any subset of this is Gaussian

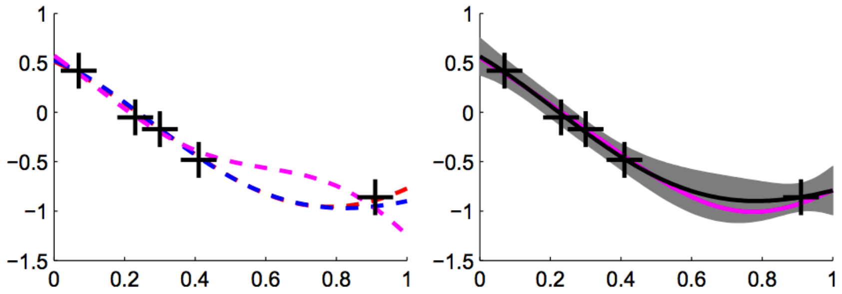
$$P(\mathbf{f}|\Sigma,\mu) = |2\pi\Sigma|^{\frac{1}{2}} \exp(-\frac{1}{2}(\mathbf{f}-\mu)^{\top}\Sigma^{-1}(\mathbf{f}-\mu))$$

Gaussian prior

Specified only by mean and covariance

Given observations $\{x, y\}_{i=1}^{n}$ over an unknown function f drawn from a Gaussian prior, the posterior is Gaussian

$$P(\mathbf{f}|\mathbf{y}) \propto \int dx\, P(\mathbf{f},\mathbf{x},\mathbf{y})$$



GP-Select

[Vanchinathan et al., 2015]

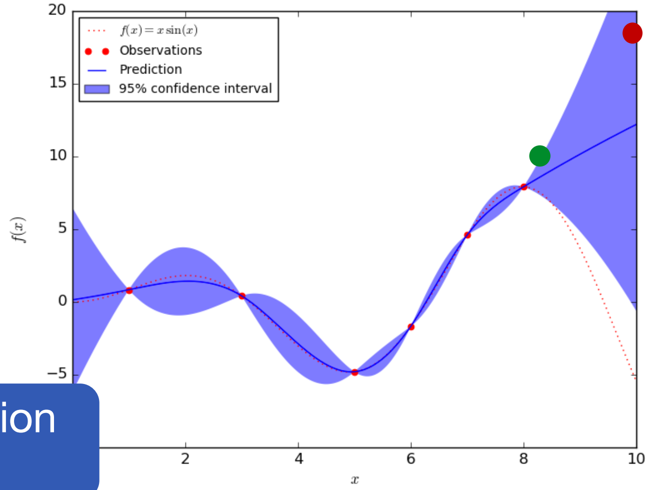
Algorithm 1 GP-SELECT

Input: Ground Set $\mathbf{V}$, kernel κ and budget B
Initialize selection set S
for $t = 1, 2, \ldots, B$ **do**
  Model Update:
    $[\mu_{t-1}(\cdot), \sigma^2_{t-1}(\cdot)] \leftarrow \text{GP-Inference}(\kappa, (S, y_{\{1:t-1\}}))$
  Item Selection:
    Set $v_t \leftarrow \underset{v \in \mathbf{V}/\{v_{1:t-1}\}}{\text{argmax}} \quad \mu_{t-1}(v) + \beta_t^{1/2} \sigma_{t-1}(v)$
  $S \leftarrow S \cup \{v_t\}$
  Receive feedback $y_t = f(v_t) + \epsilon_t$
end for

Learn posterior

Trades off exploration exploitation

Ask user feedback

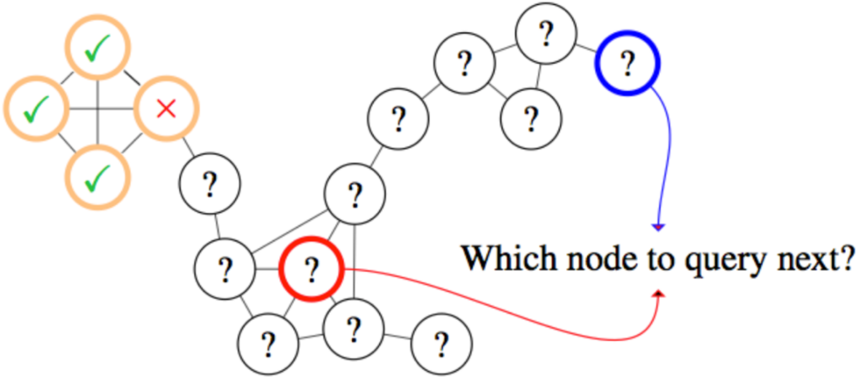


- Exploration: select items with high-variance
- Exploitation: select items with high-value

Active learning on graphs – which prior?

[Ma et al., 2015]

Idea: Use the graph structure to infer the node classes



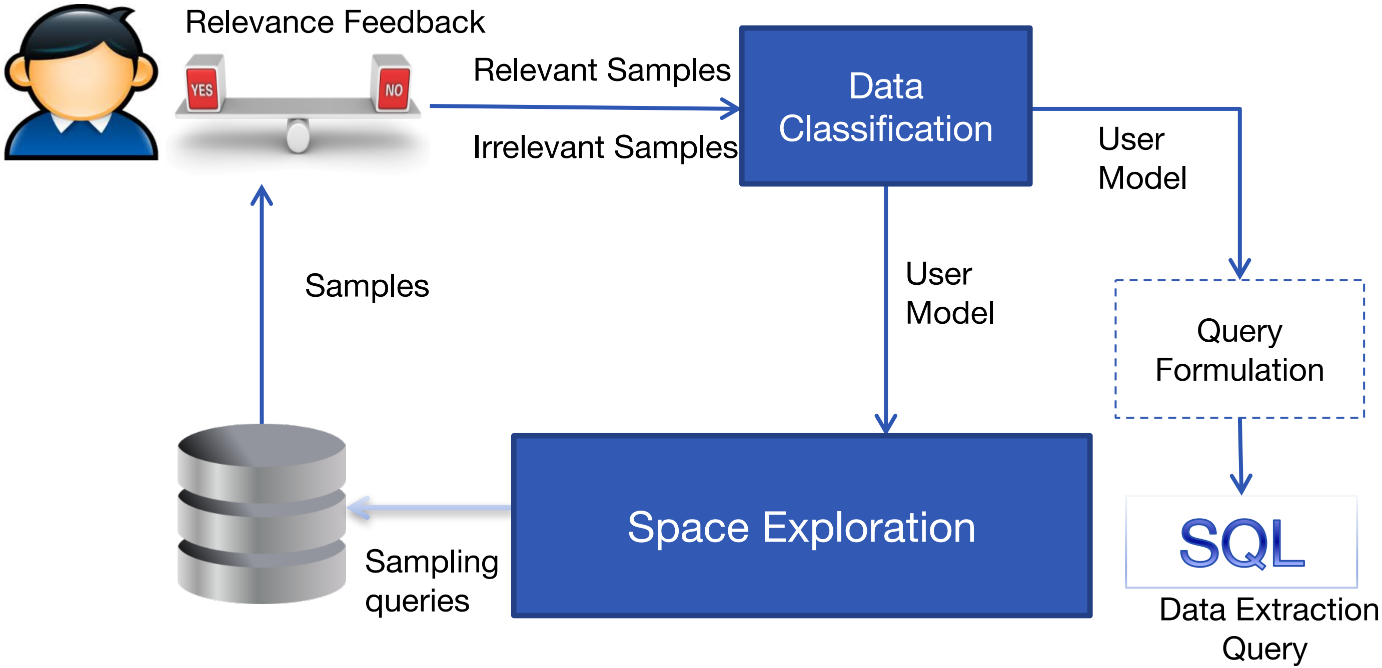
Use graph Laplacian as prior
$L = D - A$, A is the adjacency matrix

$$p(\mathbf{f}) \sim \mathcal{N}(0, L^{-1})$$

Laplacian: higher probability of having the same class if two nodes are connected

Explore-by-Example: AIDE

[Dimitriadou et al., 2015]



The AIDE algorithm

[Dimitriadou et al., 2015]

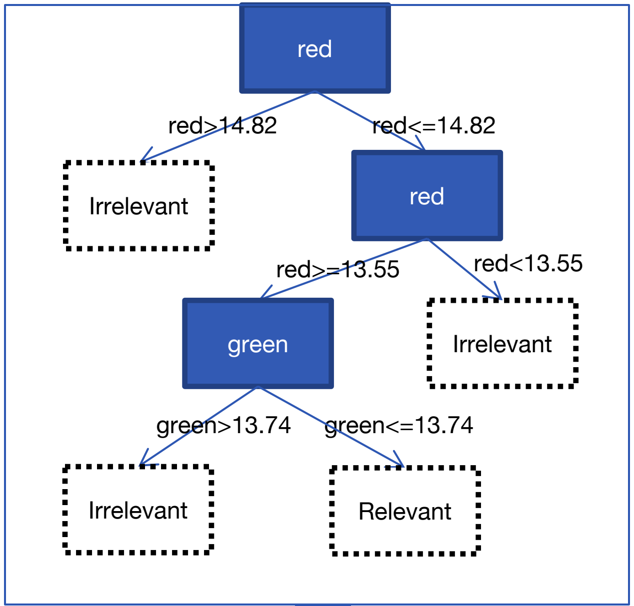
1. Divide the space into d-dimensional cubes
2. Find the sample points in the cubes (medoids)
3. Train the classifier
4. Refine the training sampling from neighbors of misclassified points
5. Boundary refinement

Classification & Query Formulation

[Dimitriadou et al., 2015]

Sample	Red	Green	Relevant
Object A	13.67	12.34	Yes
Object B	15.32	14.50	No
..	..	..	...
Object X	14.21	13.57	Yes

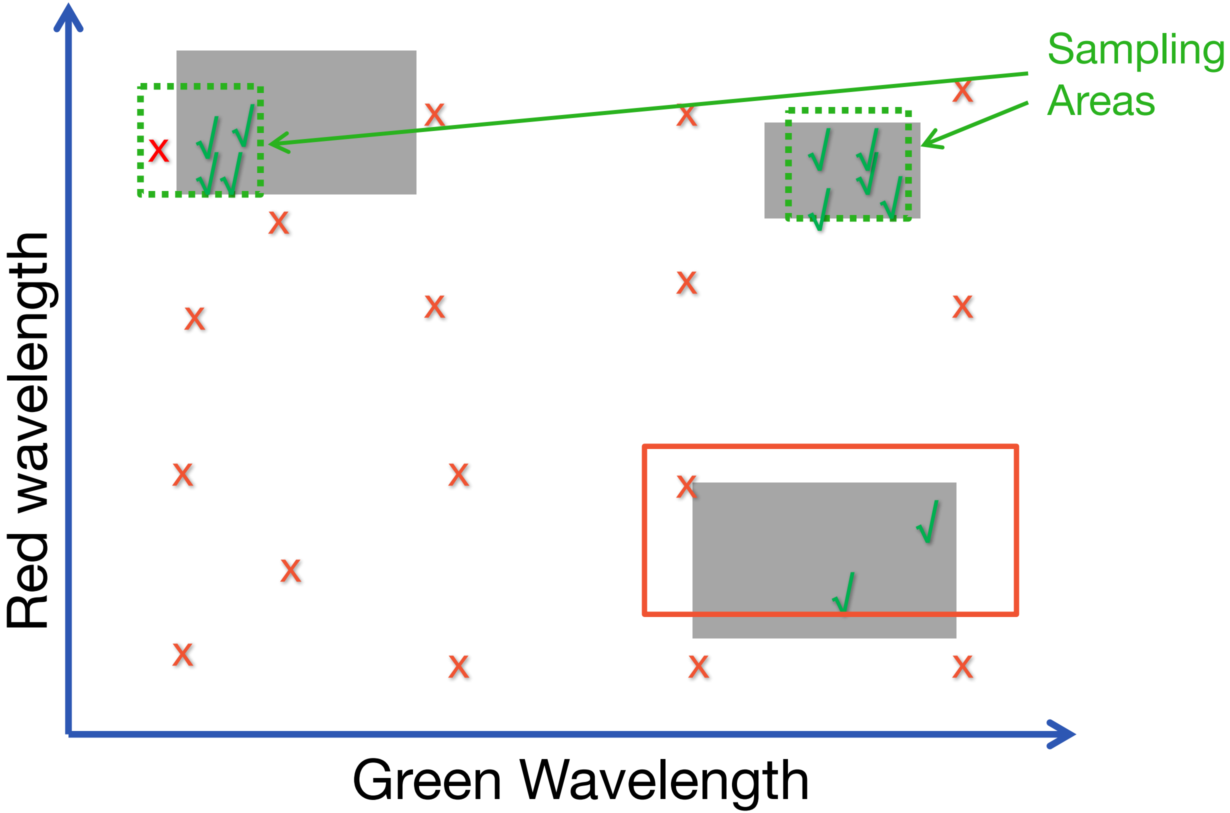
Decision Tree Classifier



SELECT * FROM galaxy WHERE red<= 14.82 AND red>= 13.5 AND green<=13.74

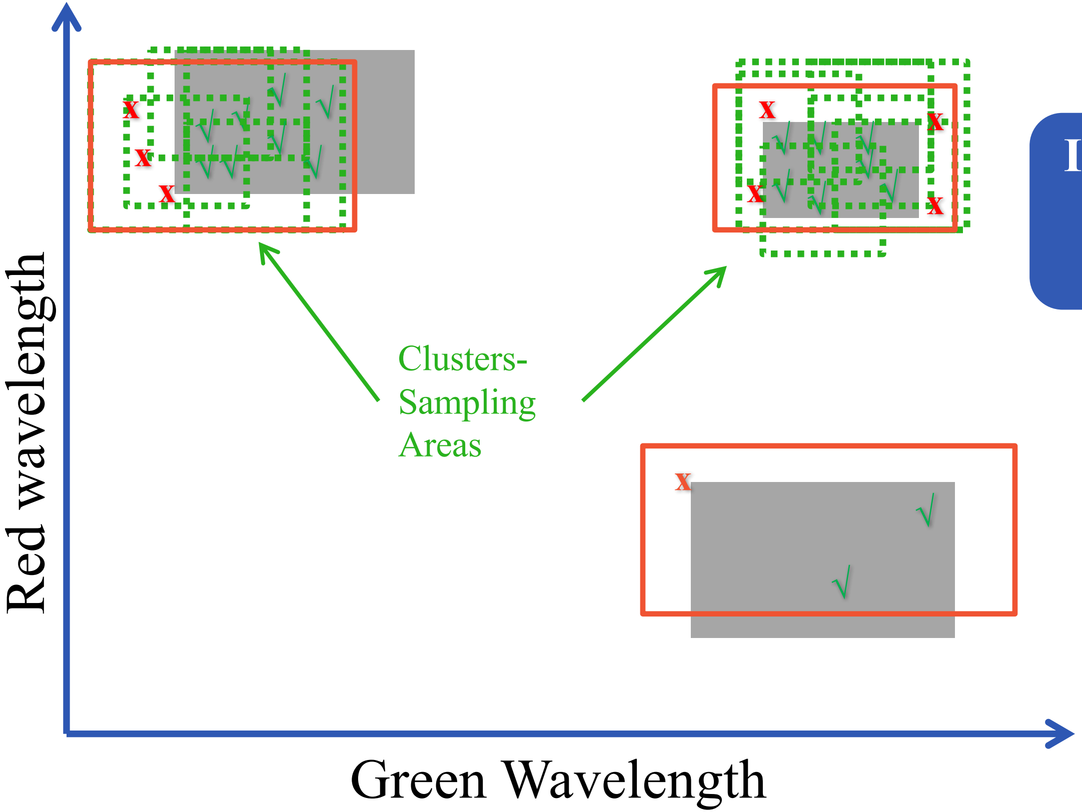
Misclassified Sample Exploitation

[Dimitriadou et al., 2015]



Clustering-based Sampling

[Dimitriadou et al., 2015]



Idea: Use a k-medoid approach to find sampling areas

D. Mottin, M. Lissandrini, T. Palpanas, Y. Velegrakis

Where we are

Relational databases

Textual data

Graphs and networks

Machine learning

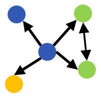
Challenges and Remarks

Example-based methods

- Query suggestion using examples
- Reverse engineering queries

- Entity extraction by example text
- Web table completion using examples
- Search by example

- Community-based Node-retrieval
- Entity Search
- Path and SPARQL queries
- Graph structures as Examples



Example-based methods: takeaways

Relational

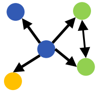
- Complex search space
- Exact and approximate
- Interactivity can improve the quality
- Limited to query inference

Textual

- Allows serendipitous search
- Easier document finding
- Speed up entity matching

Graph

- Exploit locality
- Entity attributes are expressive
- Reverse engineering: good approximations
- Large result-sets require ranking

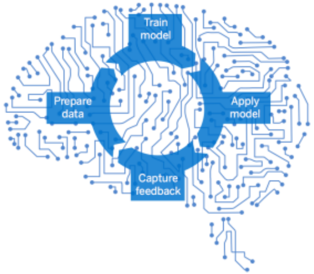


The use of examples

Examples can ease data exploration

- … reduce need for complex queries / simplify user input
- … require no schema knowledge
- … allow uncertainity in search conditions
- … require little data analytics expertise

Where should we invest time



Machine learning

Approximate Methods

User models

Scalability

ADOPT HETEROGENEITY

Need for solutions that
operate across different models

operate on heterogeneous datastores

HPI dbTrento

"The Context of Mobile Interaction"

– Nadav Savio

Nadav Savio | Giant Ant Design | www.giantant.com

PERSONALIZATION

better understand user needs

Meta-data and User Profiles

exploit *query log, prior searches, user context*

DEMOCRATIZATION
easy access to data

tools that work on commodity hardware, mobile devices

data-exploration for everyday use-cases

INTERACTIVITY

gradually understand user need

ADAPTIVITY

build indexes and data structures on-the-go

NATARUAL LANGUAGE INTERFACE

flexible, vague, imprecise input

exploration through conversation

Example is always more efficacious than precept

***Samuel Johnson**, Rasselas (1759), Chapter 29.*

"New Trends on Exploratory Methods for Data Analytics."
Davide Mottin, Matteo Lissandrini, Yannis Velegrakis, Themis Palpanas.
Proceedings of the Conference in Very Large Databases (PVLDB), 10(12), 2017

Slides: http://j.mp/DataExplore

Acknowledgments

We would like to thank the authors of the papers who kindly provided us the slides

Angela Bonifati, Radu Ciucianu, Marcelo Arenas, Gonzalo Diaz, Egor Kostylev, Yaacov Weiss, Sarah Cohen, Fotis Psallidas, Li Hao, Chan Chee Yong, Ilaria Bordino, Mohamed Yakout, Kris Ganjam, Kaushik Chakrabati, Thibault Sellam, Rohit Singh, Maeda Hanafi, Marcin Sydow, Mingzhu Zhu, Yoshiharu Ishikawa, Daniel Deutch, Nandish Jayaram, Bryan Perozzi, Kiriaki Dimitriadou, Yifei Ma, Natali Ruchansky, Quoc Trung Tran, Hastagiri Prakash Vanchinathan

References

M. Arenas, G. I. Diaz, and E. V. Kostylev**. Reverse engineering sparql queries**. WWW, 2016.

Agichtein, E. and Gravano, L. **Snowball: Extracting relations from large plain-text collections.** ICDL, 2000.

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

A. Bonifati, R. Ciucanu, and S. Staworko. **Learning join queries from user examples**. TODS, 2016.

A. Bonifati, U. Comignani, E. Coquery, and R. Thion. **Interactive mapping specification with exemplar tuples.** SIGMOD, 2017.

I. Bordino, G. De Francisci Morales, I. Weber, and F. Bonchi. **From machu picchu to rafting the urubamba river: anticipating information needs via the entity-query graph**. WSDM, 2013.

D. Deutch and A. Gilad. **Qplain: Query by explanation**. ICDE, 2016.

References

G. Diaz, M. Arenas, and M. Benedikt. **Sparqlbye: Querying rdf data by example**. PVLDB, 2016.

K. Dimitriadou, O. Papaemmanouil, and Y. Diao. **Explore-by-example: An automatic query steering framework for interactive data exploration.** In SIGMOD, 2014.

B. Eravci and H. Ferhatosmanoglu. **Diversity based relevance feedback for time series search.** PVLDB, 2013.

A. Gionis, M. Mathioudakis, and A. Ukkonen. **Bump hunting in the dark: Local discrepancy maximization on graphs**. ICDE, 2015.

M. F. Hanafi, A. Abouzied, L. Chiticariu, and Y. Li. **Synthesizing extraction rules from user examples with seer.** SIGMOD, 2017.

Y. Ishikawa, R. Subramanya, and C. Faloutsos. **Mindreader: Querying databases through multiple examples.** VLDB, 1998.

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

References

H. Li, C.-Y. Chan, and D. Maier. **Query from examples: An iterative, data-driven approach to query construction.** PVLDB, 2015.

Y. Ma, T.-K. Huang, and J. G. Schneider. **Active search and bandits on graphs using sigma-optimality.** UAI, 2015.

S. Metzger, R. Schenkel, and M. Sydow**. Qbees: query by entity examples.** CIKM, 2013.

D. Mottin, M. Lissandrini, Y. Velegrakis, and T. Palpanas. **Searching with xq: the exemplar query search engine**. SIGMOD, 2014.

D. Mottin, M. Lissandrini, Y. Velegrakis, and T. Palpanas. **Exemplar queries: a new way of searching.** VLDB J., 2016.

B. Perozzi, L. Akoglu, P. Iglesias Sa´nchez, and E. Müller**. Focused clustering and outlier detection in large attributed graphs.** KDD, 2014.

References

F. Psallidas, B. Ding, K. Chakrabarti, and S. Chaudhuri. **S4: Top-k spreadsheet-style search for query discovery**. SIGMOD, 2015.

R. Rolim, G. Soares, L. D'Antoni, O. Polozov, S. Gulwani, R. Gheyi, R. Suzuki, and B. Hartmann. **Learning syntactic program transformations from examples**. ICSE, 2017.

N. Ruchansky, F. Bonchi, D. García-Soriano, F. Gullo, and N. Kourtellis. **The minimum wiener connector problem**. SIGMOD, 2015.

T. Sellam and M. Kersten. **Cluster-driven navigation of the query space.** TKDE, 2016.

Y. Shen, K. Chakrabarti, S. Chaudhuri, B. Ding, and L. Novik. **Discovering queries based on example tuples.** SIGMOD, 2014.

R. Singh. **Blinkfill: Semi-supervised programming by example for syntactic string transformations.** PVLDB, 2016.

G. Sobczak, M. Chochół, R. Schenkel, and M. Sydow. iqbees: **Towards interactive semantic entity search based on maximal aspects.** Foundations of Intelligent Systems, 2015.

References

Y. Su, S. Yang, H. Sun, M. Srivatsa, S. Kase, M. Vanni, and X. Yan. **Exploiting relevance feedback in knowledge graph search**. KDD, 2015.

Q. T. Tran, C.-Y. Chan, and S. Parthasarathy. **Query reverse engineering**. VLDB J., 2014.

H. P. Vanchinathan, A. Marfurt, C.-A. Robelin, D. Koss- mann, and A. Krause. **Discovering valuable items from massive data**. In KDD, 2015.

C.Wang, A.Cheung, and R.Bodik**. Interactive query synthesis from input-output examples.** In SIGMOD, 2017.

C. Wang, A. Cheung, and R. Bodik. **Synthesizing highly expressive sql queries from input-output examples.** In PLDI, 2017.

Y. Y. Weiss and S. Cohen. **Reverse engineering spj-queries from examples**. SIGMOD, 2017.

M. Yakout, K. Ganjam, K. Chakrabarti, and S. Chaudhuri. **Infogather: Entity augmentation and attribute discovery by holistic matching with web tables.** SIGMOD, 2012.

M. Zhu and Y.-F. B. Wu. **Search by multiple examples**. WSDM, 2014.

M. M. Zloof. **Query by example**. AFIPS NCC, 1975.