# Robust Discovery of Positive and Negative Rules in Knowledge-Bases

**Paolo Papotti**

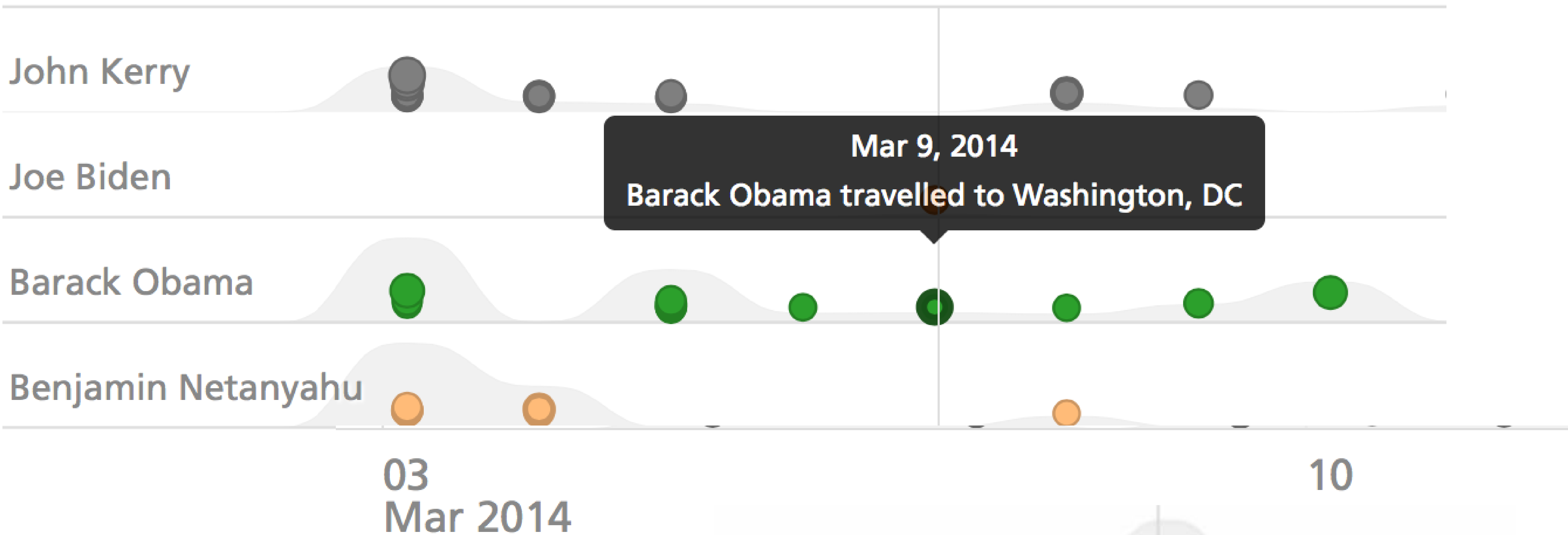
EURECOM
Sophia Antipolis

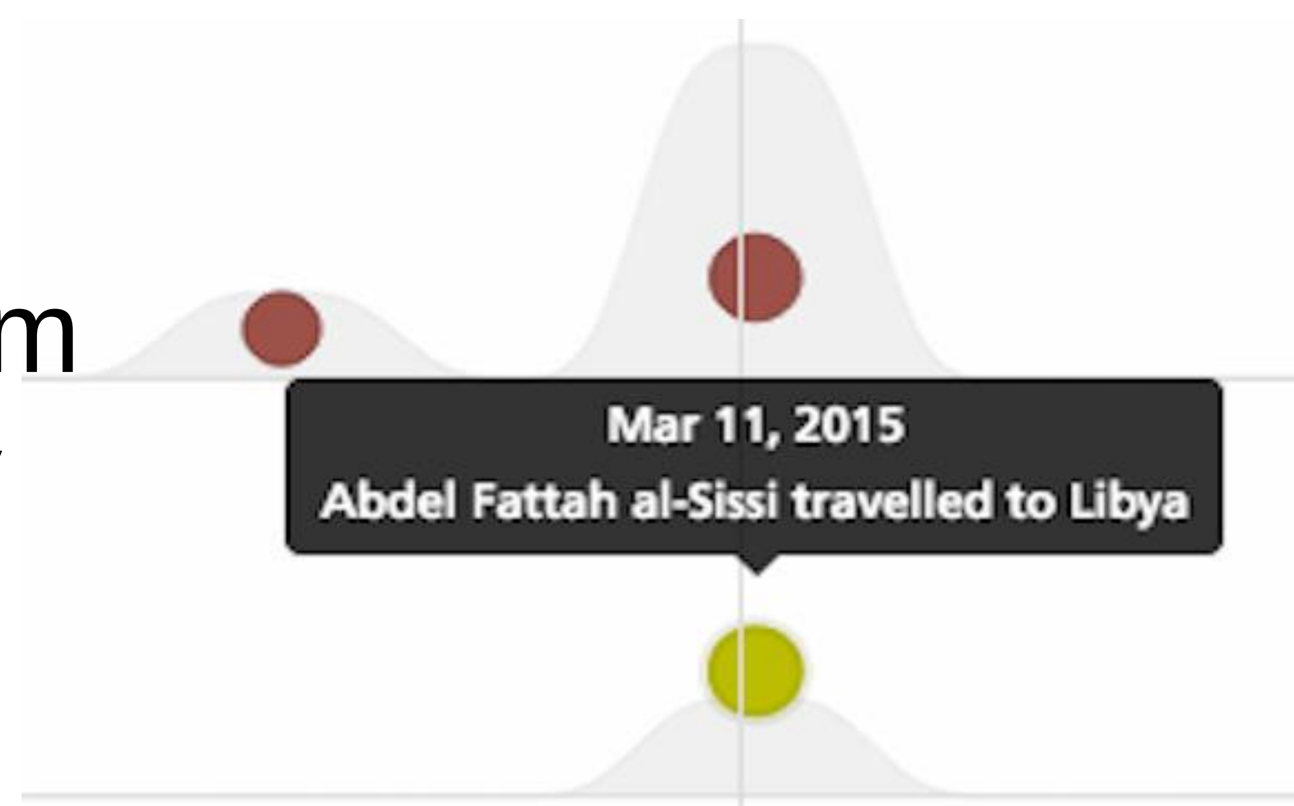
joint work with S. Ortona (Meltwater) and V. Meduri (ASU)

http://www.eurecom.fr/en/publication/5321/detail/robust-discovery-of-positive-and-negative-rules-in-knowledge-bases

Lyon – 12 Dec 2017

Recorded Future
CREATING AN INSIGHTFUL WORLD




- RF integrates facts from 1M web sources every day to run analytics

Goal: obtain cleaning programs that

- Effectively detect and fix problems
- Efficiently process large datasets
- Easy to interpret for validation

- Data cleaning rules

# Cleaning RF data with Temp FDs

- Nine relations, 4000 manually annotated tuples

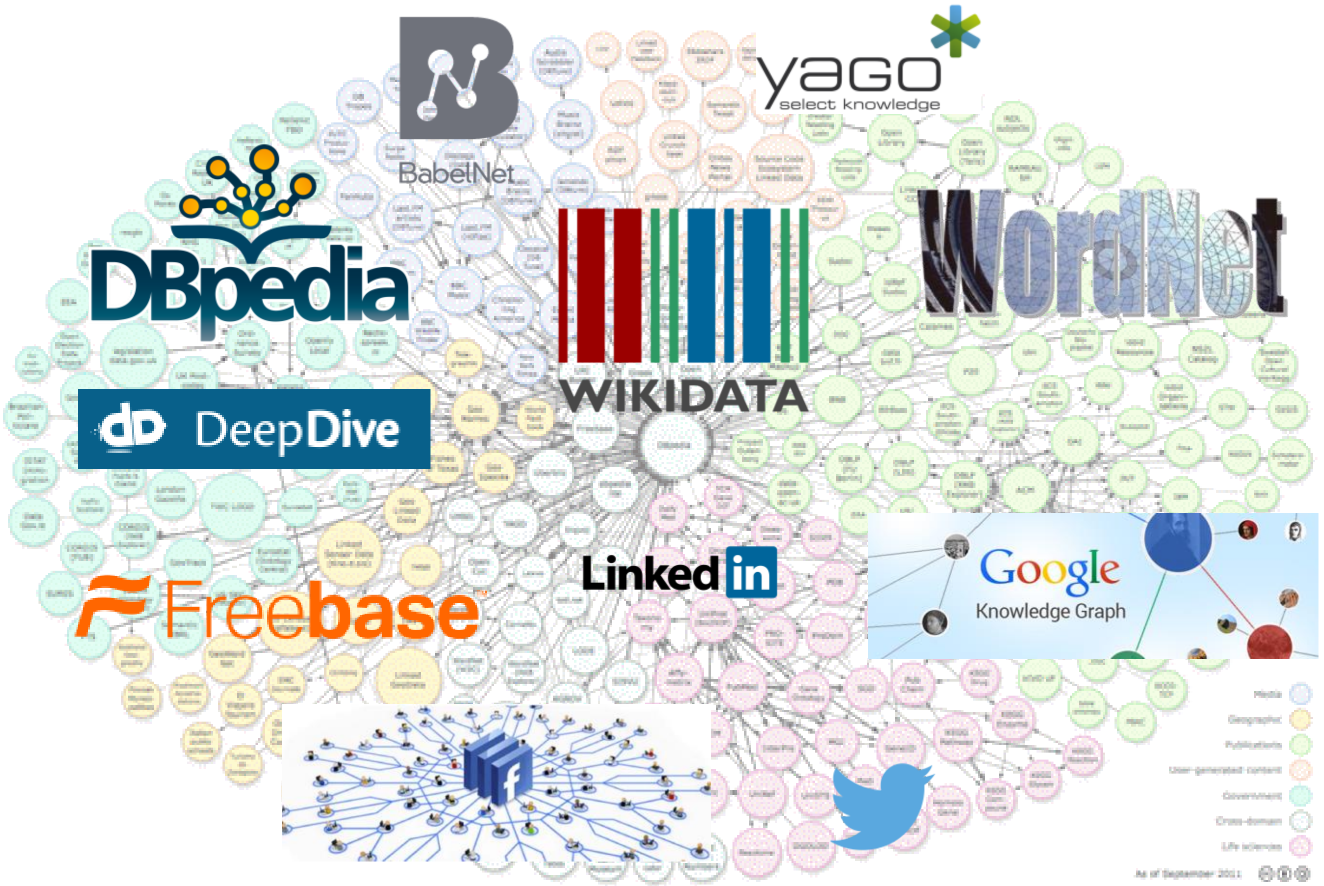
| | | $P$ | $R$ | $F$ |
|---|---|---|---|---|
| Company Employees # | 24 | 0.74 | 0.17 | **0.27** |
| Company Meet. | 336 | 0.94 | 0.5 | **0.65** |
| Credit Rating | 48 | 0.6 | 0.75 | **0.67** |
| Employment Change | 24 | 1.0 | 0.88 | **0.94** |
| Natural Disaster | 24 | 0.8 | 0.5 | 0.62 |
| Person Travel | 48 | 0.61 | 0.82 | 0.7 |
| Political Endorsement | 48 | 1.0 | 0.59 | **0.74** |
| Product Recall | 177 | 0.9 | 0.9 | **0.9** |
| Voting Result | 24 | 1.0 | 0.6 | **0.75** |
| | | 0.84 | 0.54 | |

[Abedjan et al, 2015]

# Relational Data

| Name | Location | Timestamp |
|---|---|---|
| B. Obama | Rome | 8.00 Feb 1 |
| B. Obama | Rome | 8.15 Feb 1 |
| B. Obama | NYC | 11.00 Feb 1 |
| B. Obama | Paris | 18.00 Feb 1 |
| B. Obama | Paris | 18.49 Feb 1 |

# Knowledge Bases




WalMart, KPMG, Amadeus, …

# RDF KBs

<Barack Obama> <spouse> <Michelle Obama> .

<Barack Obama> <birthDate> “1961-08-04” .

<Michelle Obama> <birthPlace> <Illinois> .

SUBJECT PREDICATE OBJECT

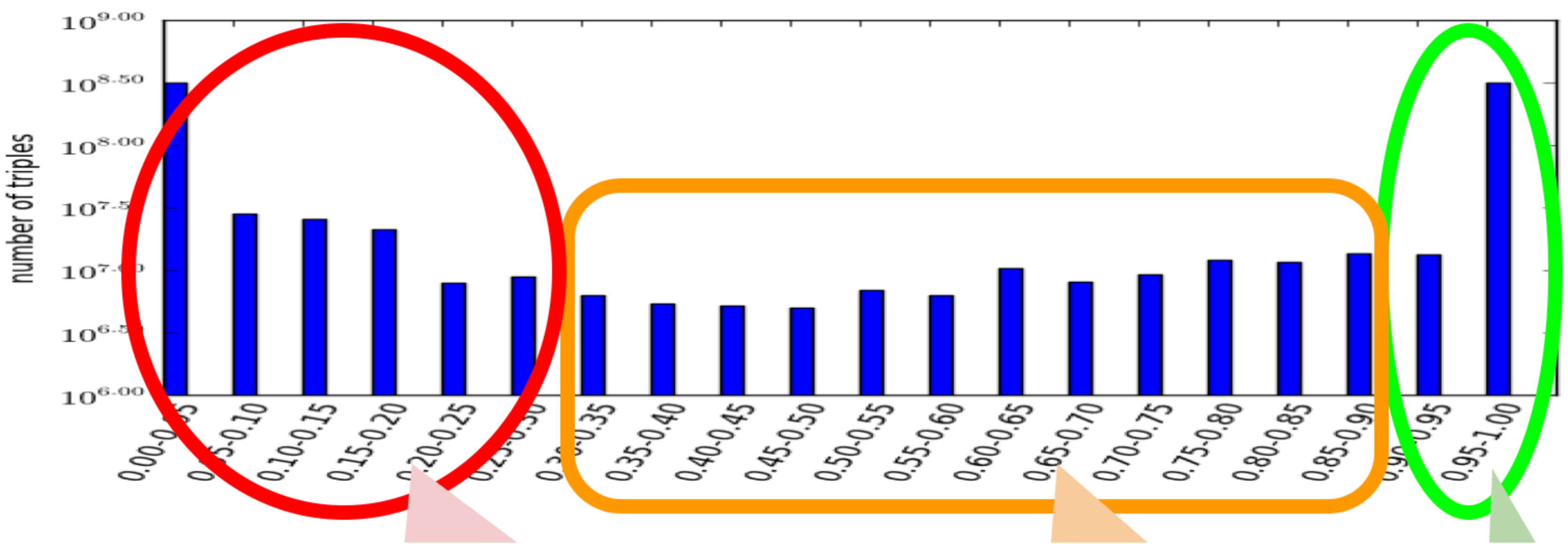
| Name | # Entity types | # Entity instances | # Relation types | # Confident facts (relation instances) |
|---|---|---|---|---|
| Knowledge Vault (KV) | 1100 | 45M | 4469 | 271M |
| DeepDive [21] | 4 | 2.7M | 34 | 7M[a] |
| NELL [6] | 271 | 5.19M | 306 | 0.435M[b] |
| PROSPERA [20] | 11 | N/A | 14 | 0.1M |
| YAGO2 [16] | 350,000 | 9.8M | 100 | 4M[c] |
| Freebase [4] | 1,500 | 40M | 35,000 | 637M[d] |
| Knowledge Graph (KG) | 1,500 | 570M | 35,000 | 18,000M[e] |

**Table 1: Comparison of knowledge bases [9]. KV, DeepDive, NELL, and PROSPERA rely solely on extraction, Freebase and KG rely on human curation and structured sources, and YAGO2 uses both strategies. Confident facts means with a probability of being true at or above 0.9.**

[Dong and Srivastava, 2015]

# Need for rules?

## Usage of Probabilistic Knowledge




[Dong and Srivastava, 2015]

"ML did not reach the required 92% precision threshold, [with **20,459** rules] precision consistently in the range 92-93%"

[Suganthan et al, 2015]

"[to build Kosmix KB —> WalmartLabs KB] analysts have written **several thousands** of rules " [Deshpande et al, 2013]

# Data Quality issues in KBs

**Incomplete data**
DBPedia: 1.7M Person, birth dates reported only for 1M

**Errors**
Yago: 9K cases where child is born before parent

# Horn Rules Discovery

Body (conjunction of atoms) $\Rightarrow$ Head (atom)

$\text{child}(x,z) \land \text{child}(y,z) \Rightarrow \text{spouse}(x,y)$

Atom = predicate from KB

AMIE:

- **memory based**
- language bias
  - max body size **2**, **equality comparison only, no literals**
- **"positive" rules →** Incomplete data **only**

[Galarraga et al, 2013]

# Negative Rules

**Positive Rule:**

child(x,z) ∧ child(y,z) ⇒ spouse(x,y)

**Negative Rule:**

birthDate(x,w) ∧ birthDate(y,z) ∧ w ≤ z ⇒ **¬child**(y,x)

Atom = - positive/**negative** predicate from KB
- value comparison (**<, ≤, =, >, ≥**)

**Constants** allowed for conditional rules

(e.g., rule applies only in US)

# Negative Rules

**Positive Rule:**

$\text{child}(x,z) \wedge \text{child}(y,z) \Rightarrow \text{spouse}(x,y)$

**Negative Rule:**

$\text{birthDate}(x,w) \wedge \text{birthDate}(y,z) \wedge w \leq z \wedge \textbf{child}(y,x) \Rightarrow \text{error}$

Denial constraints [Chu et al, 2013]

# Problem Definition

Given a pair of entities (a,b) in the KB, rule **r**: body → p(x,y) *covers* (a,b) if there exists an instantiation in KB such that the body of **r** holds

Input:

- target predicate **p** (*spouse*)
- generation set **G** (examples of *married couples*)
- validation set **V** (examples of *unmarried couples*)

Output:

a set of positive (*negative*) rules covering all elements in G, and none of V (*none of G, all of V*)

# Exact Solution May Fail

$$\text{child}(x,z) \wedge \text{child}(y,z) \Rightarrow \textbf{spouse}(x,y)$$

# Exact Solution May Fail

? ⇒ **spouse**(x,y)

**+** <Barack Obama> <spouse> <Michelle Obama> .
**+** <Beyonce'> <spouse> <Jay-Z> .
**-** <Tom Cruise> <spouse> <Serena Williams> .
**-** <Leonardo Da Vinci> <spouse> <Hillary Clinton> .

Miss valid rules because do not hold on all examples in G
- rule does not apply for all (couples w/out children)
- for missing values in the data - OWA (missing children)
- for mistakes in the data (wrong parent)

—> failure or overfitting

# Exact Solution May Fail

? ⇒ ¬**spouse**(y,x)

**-** <Barack Obama> <spouse> <Michelle Obama> .
**-** <Beyonce'> <spouse> <Jay-Z> .
**+** <Tom Cruise> <spouse> <Serena Williams> .
**+** <Leonardo Da Vinci> <spouse> <Hillary Clinton> .

Miss valid rules because do not hold on all examples in G

- for missing values in the data - OWA (missing DOB)
- for mistakes in the data (wrong DOB)

—> failure or overfitting

# Problem Revised

<u>Input</u>:

- target predicate **p** (*spouse*)
- generation set **G** (examples of *married couples*)
- validation set **V** (examples of *not married couples*)

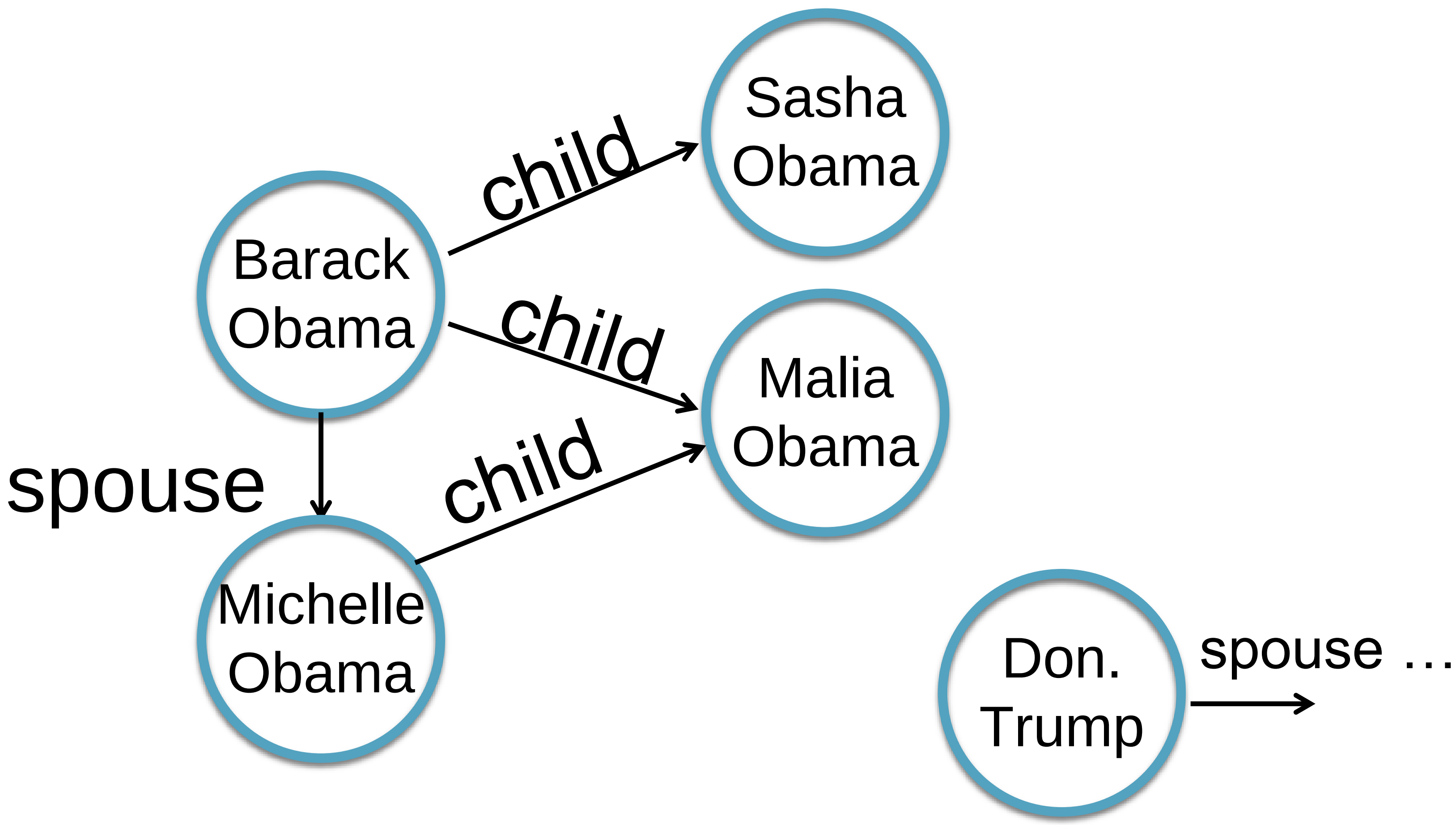
<u>Output</u> (positive rules):

weighted set cover: G universe of elements, each rule is an element of S, and V used for computing weights

$$w(r) = \alpha \cdot \Big(1 - \underbrace{\frac{| C_r(G) |}{| G |}}_{\text{max coverage G}}\Big) + \beta \cdot \Big(\underbrace{\frac{| C_r(V) |}{| U_r(V) |}}_{\text{min coverage V}}\Big)$$

# Generation and Validation Sets

- Generation set **G**: straightforward from KB (all people connected by a spouse predicate)




- Validation Set **V**: all negative example of spouse relationship

# Negative Examples

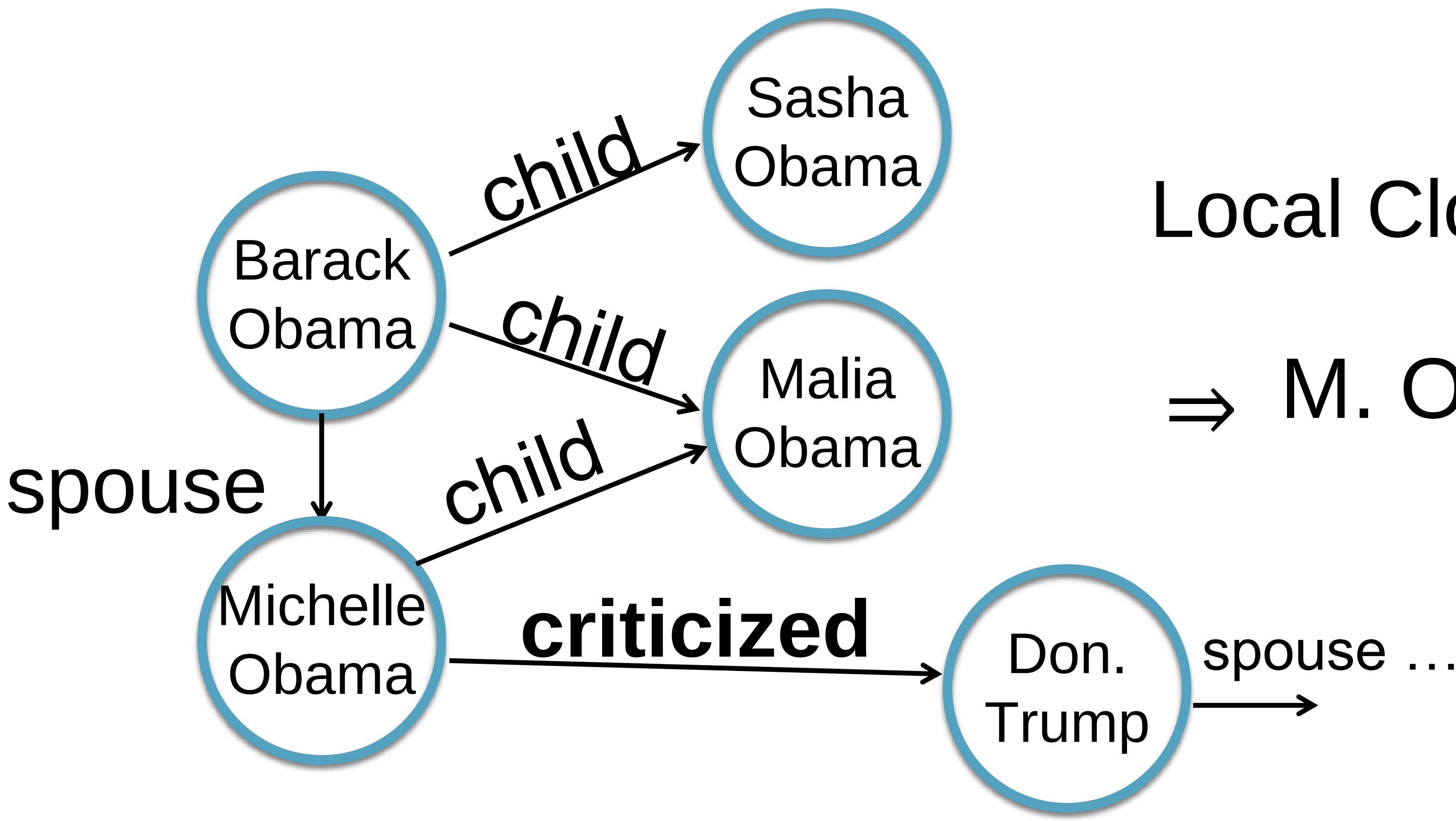
- A true negative, only if the entities have no missing relationships
  - Cannot assume that *what is not in KB is false* (OWA)
- Naïve creation method: Cartesian product
  very small fraction of pairs are semantically related
  → miss meaningful paths!
  - Always true for positive examples: they have target predicate in common
- New method using Local-Closed World Assumption

# Negative Examples

- For predicate *child*, negative example is a pair x,y s.t.
  - x has some children in the KB who are not y, or y is the child of someone who is not x (LCWA on subject and predicate)
  - x,y are connected via a predicate that is different from the target predicate (semantically related)

# Generation and Validation Sets

- Generation set **G**: straightforward from KB (all people connected by a spouse predicate)



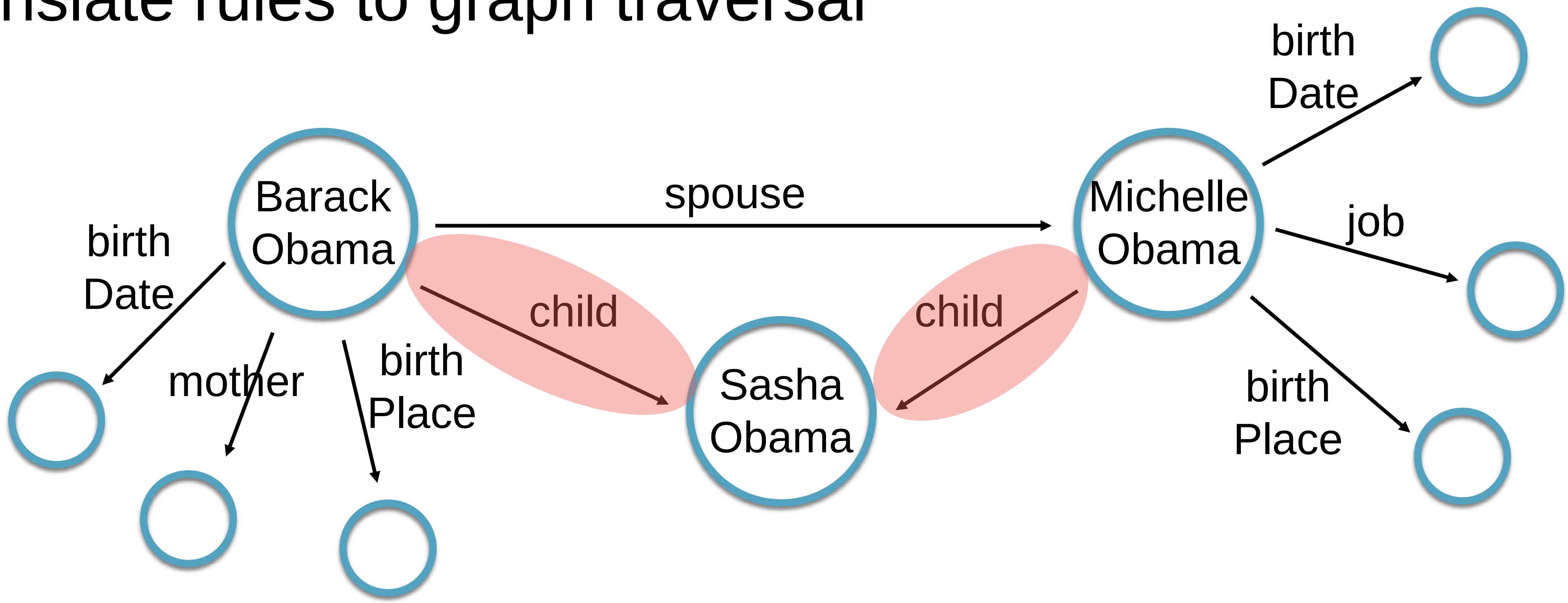

Local Closed World Assumption

⇒ M. Obama does not have another spouse

- Validation Set **V**: all pairs of people (x,y) where either x or y are in a spouse relationship with someone else
- At least another predicate between (x,y): crucial when V is the generation set, size comparable to G

# Naive Rule Generation

- Translate rules to graph traversal




$$\text{child}(x,z) \quad \wedge \quad \text{child}(y,z) \quad \Rightarrow \text{spouse}(x,y)$$

- Generate all possible rules (body size 3) from G, compute weights from G&V, compute weighted set cover

# Greedy Algorithm

- Greedy traversal: at each iteration follow the **most promising path** according to marginal weight
- Build graph **incrementally**: query the KB only when needed to follow a given path
- **Prune** paths that do not lead to good solutions

Advantages:

- A* guarantees optimal if estimation is admissible
- No need to generate all possible rules
- Load in memory only the needed portion of the graph
  - Lexical values: more expressive rules
  - Running time in seconds/minutes

# Experiments

Java with any SPARQL endpoint (Virtuoso)
i5 CPU at 2.80GHz and 16GB RAM

TABLE I. DATASET CHARACTERISTICS.

| *KB* | *Version* | *Size* | *#Triples* | *#Predicates* |
|---|---|---|---|---|
| DBPEDIA | 3.7 | 10.06GB | 68,364,605 | 1,424 |
| YAGO 3 | 3.0.2 | 7.82GB | 88,360,244 | 74 |
| WIKIDATA | 20160229 | 12.32GB | 272,129,814 | 4,108 |

5 most popular predicates for every KB
Output triples manually checked (30) for every rule

http://www.eurecom.fr/en/publication/5321/detail/robust-discovery-of-positive-and-negative-rules-in-knowledge-bases

# Experiments

**Positive Rules: new triples**

- $notableWork(y,x) \Rightarrow creator(x,y)$ (Wikidata)
- $hasChild(z,y) \wedge isMarriedTo(x,z) \Rightarrow hasChild(x,y)$ (Yago)

**Negative Rules: erroneous triples**

- $foundingYear(x,z) \wedge birthYear(y,w) \wedge (z \leq w) \Rightarrow \neg founder(x,y)$ (34 errors DBPedia)

- $isMarriedTo(x,y) \Rightarrow \neg hasChild(x,y)$ (200 errors Yago)

# Experiments

TABLE II. RuDiK Positive Rules Accuracy.

| *KB* | *Avg. RunTime* | *Avg. Precision over Predicates with Rules (All)* | *# Labeled Triples* |
|---|---|---|---|
| DBPedia | 35min | **97.14**% (63.99%) | 139 |
| Yago 3 | 59min | **84.44**% (62.86%) | 150 |
| Wikidata | 141min | **98.95**% (73.33%) | 180 |

TABLE III. RuDiK Negative Rules Accuracy.

| *KB* | *Avg. Run Time* | *# Pot. Errors* | *Precision* |
|---|---|---|---|
| DBPedia | 19min | 499 (84) | **92.38**% |
| Yago 3 | 10min | 2,237 (90) | **90.61**% |
| Wikidata | 65min | 1,776 (105) | **73.99**% |

# AMIE as baseline

TABLE IV. AMIE DATASET CHARACTERISTICS.

| *KB* | *Size* | *#Triples* | *#Predicates* | *#*`rdf:type` |
|---|---|---|---|---|
| DBPEDIA | 551M | 7M | 10,342 | 22.2M |
| YAGO 2 | 48M | 948.3K | 38 | 77.9M |

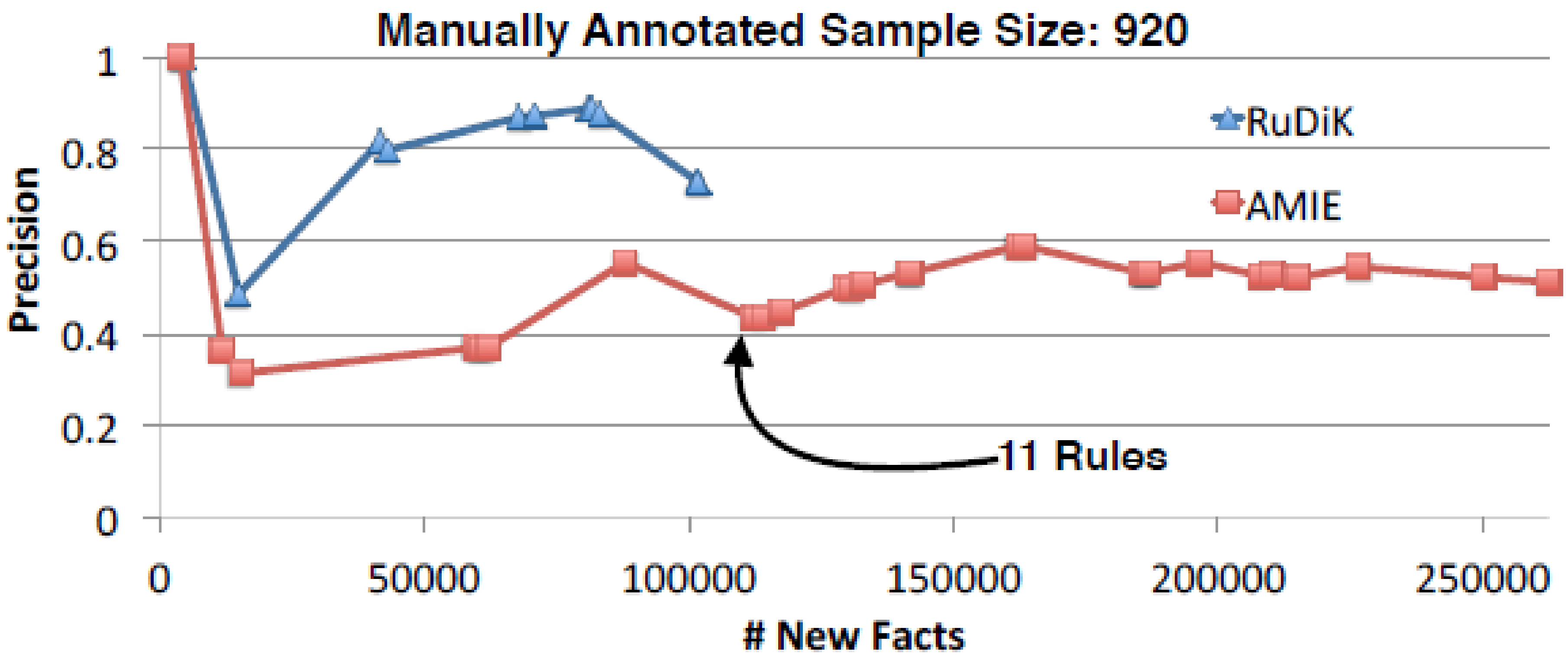



Fig. 3. Accuracy for new facts identified by executing rules in descending AMIE's score on YAGO 2 (no literals).

[Galarraga et al, 2013]

# AMIE as baseline

- Modified KBs to use AMIE for negative rule discovery
- Added notSpouse predicate for each negative example

TABLE V. NEGATIVE RULES VS AMIE.

| | AMIE | | RuDiK (no literals) | |
|---|---|---|---|---|
| *KB* | *# Errors* | *Precision* | *# Errors* | *Precision* |
| DBPEDIA | 457 (157) | 38.85% | 148 (73) | **57.76**% |
| YAGO 2 | 633 (100) | 48.81% | 550 (35) | **68.73**% |

[Galarraga et al, 2013]

# Directions

- Rule discovery combined with other signals
- How to involve users in monitoring/evolution
- Applications beyond error detection

# Robust Discovery of Positive and Negative Rules in Knowledge-Bases

Relational Data

| Name | Location | Timestamp |
|---|---|---|
| B. Obama | Rome | 8.00 Feb 1 |
| B. Obama | Rome | 8.15 Feb 1 |
| B. Obama | NYC | 11.00 Feb 1 |
| B. Obama | Paris | 18.00 Feb 1 |
| B. Obama | Paris | 18.49 Feb 1 |

Knowledge Bases


**Paolo Papotti**
EURECOM