Relationship Queries on Extended Knowledge Graphs

Mohamed Yahya, Denilson Barbosa, 
Klaus Berberich, Qiuyue Wang, 
Gerhard Weikum
Knowledge Graphs = SPO Triples

- **SPO triples** about named entities
  - AlbertEinstein wasBornIn Ulm
  - AlbertEinstein diedIn Princeton
  - AlbertEinstein type SwissPhysicists
  - AlbertEinstein influences KarlPopper
  - Princeton isLocatedIn NewJersey
  - NewJersey isLocatedIn UnitedStates
Triple Pattern Queries

- How to get **answers** from knowledge graphs?
  - Physicists with their place of birth and place of death

- **SPO triples** with **variables**
  - `?p type Physicist .`
  - `?p wasBornIn ?b .`
  - `?p diedIn ?d`

- **Answers** are **variable bindings**
Schema Knowledge

- Users need **schema knowledge** to formulate **queries**
Schema Knowledge

- **Users** need **schema knowledge** to formulate **queries**

  Scientists born in the United States
Schema Knowledge

- **Users** need **schema knowledge** to formulate **queries**

Scientists born in the United States

```query
?p type Scientist.
?p wasBornIn UnitedStates
```
Users need schema knowledge to formulate queries

Scientists born in the United States

?p type Scientist .
?p wasBornIn UnitedStates

People are born in cities, not countries!
Users need schema knowledge to formulate queries

Scientists born in the United States

?p type Scientist .
?p wasBornIn ?l .
?l locatedIn UnitedStates
Users need schema knowledge to formulate queries

Scientists born in the United States

?p type Scientist .
?p wasBornIn ?l .
?l locatedIn UnitedStates

CEOs influenced by Albert Einstein
Users need schema knowledge to formulate queries

Scientists born in the United States

?p type Scientist .
?p wasBornIn ?l .
?l locatedIn UnitedStates

CEOs influenced by Albert Einstein

?p type CEO .
?p influencedBy AlbertEinstein
Users need **schema knowledge** to formulate queries.

**Scientists born in the United States**

```
?p type Scientist .
?p wasBornIn ?l .
?l locatedIn UnitedStates
```

**CEOs influenced by Albert Einstein**

```
?p type CEO .
?p influencedBy AlbertEinstein
```

It's called influences.
Users need schema knowledge to formulate queries

Scientists born in the United States

```sparql
?p type Scientist .
?p wasBornIn ?l .
?l locatedIn UnitedStates
```

CEOs influenced by Albert Einstein

```sparql
?p type CEO .
AlbertEinstein influences ?p
```
Incompleteness

- Knowledge graphs are incomplete, with lots of missing triples, so that there may simply be no answer

  \[
  \text{AlbertEinstein actedIn } ?m .
  \]

  \[
  ?m \text{ type Documentary}
  \]

- Some interesting information is hard to capture as triples in a knowledge graph

  “Albert Einstein began an affair with Betty Neumann”

  “Dr. Thomas Harvey conducted the autopsy on Albert Einstein”
Entity Search

- **Entity search** [e.g., Balog et al., TOIS 2012] eliminates need for schema knowledge, but is **limited** to questions with **individual entities** as answers.

  Operating systems to which Steve Jobs related

- Star-shaped triple pattern queries with a **single variable**

  \[ ?a \text{ type OperatingSystem} . \]
  \[ ?a \text{ createdBy SteveJobs} \]
Relationship Search

- **Relationship search** goes after **tuples of entities** as answers, i.e., queries with **multiple variables**

  - **Bond movies** with **performers** of their **title tracks**

  ![Bond Movies](image1)
  ![Performer](image2)
  ![Title Track](image3)

  - **CEOs of tech companies** with their **alma mater**

  ![CEO](image4)
  ![Tech Company](image5)
  ![Alma Mater](image6)
TriniT [this work] enables relationship search and mitigates need for schema knowledge as well as incompleteness.

- **Extended Knowledge Graph** with additional triples harvested from entity-annotated Web contents
- **Query Relaxation** for triple pattern queries with automatic rewritings, e.g., to replace structural predicate by textual phrase
### SELECT

```sparql
SELECT ?x
WHERE {
  <Albert_Einstein> <affiliation> ?x .
  ?x <memberOf> <Ivy_League> .
  subject predicate

  LIMIT 10
}
```

### Entities
- Ivy League
- Ivy Queen
- Poison Ivy (comics)
- Operation Ivy

### Relaxations

```sparql
?x <affiliation> ?y .
?x <affiliation> ?z ;
?z 'housed in' ?y (0.8)

?x <affiliation> ?x :- ?x 'lectured at' ?y (0.7)
```
Outline

- Motivation
- Extended Knowledge Graph
- Query Relaxation
- Experiments
- Summary
Extended Knowledge Graph

- Knowledge graph extended by triples harvested using **Named Entity Disambiguation** and **Open Information Extraction** methods

Albert Einstein lectured at Princeton. He won a Nobel for the discovery of the PE effect. The IAS is housed at Princeton.

<table>
<thead>
<tr>
<th>AlbertEinstein “lectured at” PrincetonUniversity</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlbertEinstein “won Nobel for” “discovery of PE effect”</td>
</tr>
<tr>
<td>IAS “is housed at” PrincetonUniversity</td>
</tr>
</tbody>
</table>

- Text can appear anywhere in triples
Outline

๏ Motivation
๏ Extended Knowledge Graph
๏ Query Relaxation
๏ Experiments
๏ Summary
Triple Pattern Queries

- **Triple pattern queries** for extended knowledge graphs allow **text** in any component of a **triple pattern**

  ?c type TechCompany .

  ?p "ceo of" ?c .

  ?p graduatedFrom ?a

- **Extension** of knowledge graph and **triple pattern queries** addresses **incompleteness** but **schema knowledge** is still **required** to formulate queries
Query Relaxation

- **Queries rewritten using relaxation rules**, e.g.:
  
  - **Structural relaxation**
    
    ```
    ?p wasBornIn ?l
    -------------------------
    ?p wasBornIn ?c .
    ?c locatedIn ?l
    ```
  
  - **Predicate paraphrasing**
    
    ```
    ?p wasBornIn ?l
    -------------------------
    ?p “birthplace” ?l
    ```
  
  - **Predicate inversion**
    
    ```
    ?p graduatedFrom ?u
    --------------------
    ?u “alumnus” ?p
    ?m “assassinated” ?v
    --------------------
    ?v “killed by” ?m
    ```
Query Relaxation

- **Relaxation rules** can be **defined manually** or be **mined automatically** from the data.

- **Structural relaxations** through **rule mining** on knowledge graph [Galárraga et al., WWW 2013]

- **Predicate paraphrases** through **sequence mining** on annotated Web contents [Nakashole et al., EMNLP 2012]

- **Relaxation rules** come with a **weight** that indicates how much the relaxed query deviates from original
Answer Scoring

- Large number of relaxed queries to be considered
- Same answer from multiple relaxed queries

Score of answer $a$ (i.e., variable binding) for query $Q$

$$\text{score}(a, Q) = \prod_{q_i} P[a(q_i) \mid q_i]$$

with $P[a(q_i) \mid q_i]$ as probability of drawing triple $a(q_i)$ among all triples matching the triple pattern $q_i$
Answer Scoring

- Score of answer $a$, query $Q$, and set of relaxation rules $r$

$$
\text{score}(a, Q, r) = \max_{Q' = r_n(\ldots r_1(Q)) : r_i \in r} \left( \prod_{i=1}^{n} w(r_i) \right) \text{score}(a, Q')
$$

is the maximum score under any relaxation of the query, taking into account weights of relaxations.

- Scoring model is amenable to incremental top-$k$ query processing [Ilyas, VLDB 2003; Theobald, SIGIR 2005], allowing judicious invocation of relaxation rules.
Outline

๏ Motivation

๏ Extended Knowledge Graph

๏ Query Relaxation

๏ Experiments

๏ Summary
Methods

- **ES** [Balog et al., TOIS 2012]
  - type restriction and ranking based on text

- **ERQ** [Li et al. TIST 2012]
  - triple patterns consisting of text and variables
  - underlying data is entity-annotated text (Wikipedia)
  - scoring based on proximity in text
Methods

- **SPOX** [Elbassuoni et al., CIKM 2009]
  - *triples extended* by *text* giving extraction context
  - *structural matching* with *ranking* based on *text*

- **TriniT** [this work]
  - *extended knowledge graph* but *no relaxation*

- **TriniT+Relax** [this work]
  - *extended knowledge graph* and *relaxation*
Extended Knowledge Graph

- **YAGO2** as knowledge graph
  - **48 million triples** (44 M type triples, 4.4 M other triples)
  - **ClueWeb’09** corpus with **Google FACC1** annotations

= **Extended knowledge graph** includes all triples \((x, t, y)\) with \(x\) and \(y\) as **named entities** within **same sentence**, separated by **text** \(t\) of a most 50 characters

- **65 million unique triples** (392 M triples in total)
Benchmarks

- **Entity Search Queries** [Balog & Neumayer, SIGIR 2013]
  - 218 INEX-XER, TREC Entity, SemSearch LS, QALD-2
  - “Operating systems to which Steve Jobs related”

- **Entity Relationship Queries** [Li et al., TIST 2012]
  - “Films starring Robert de Niro, and their directors”
  - “Novels and their Academy Award winning film adaptations”
  - **22/28** are about entities and their relationships
Benchmarks

- **Complex Queries** [this work]
  - sample chain of entities from extended knowledge graph (e.g., ALGOL – JohnBackus – TuringAward)
  - ask human for question with chain as answer (e.g., “programming languages by Turing Award winners”)
  - ask another human for corresponding triple pattern query
    - `?x` type `ProgrammingLanguage`.
    - `?y` type `Person`.
    - `?y “invented” ?x`.
    - `?y won TuringAward`
Results

Entity Search

Entity Relationship Search

Complex

nDCG@5  MAP

ES  ERS  SPOX  TriniT  TriniT+Relax
The figure shows the results of various metrics for different search types on knowledge graphs. The metrics include nDCG@5 and MAP. The search types include Entity Search, Relationship Search, and Complex. The methods compared are ES, ERS, SPOX, TriniT, and TriniT+Relax. The values for each metric are displayed in the bars, with the methods distinguished by different colors.
Results

![Bar chart showing results for Entity Search, Entity Relationship Search, and Complex tasks.](image-url)
Outline

- Motivation
- Extended Knowledge Graph
- Query Relaxation
- Experiments
- Summary
Summary

- **Knowledge graphs** are **incomplete** and users have to have **schema knowledge** to get answers from them.

- Our approach **TriniT** addresses these problems by:
  - **extending knowledge graphs** with triples **extracted** from **entity-annotated Web contents**.
  - **automatically rewriting queries** based on **relaxation rules** defined manually or mined from the data.