Interactive Knowledge Graph Querying through Examples and Facets

Yael Amsterdamer, Laura Gáspár

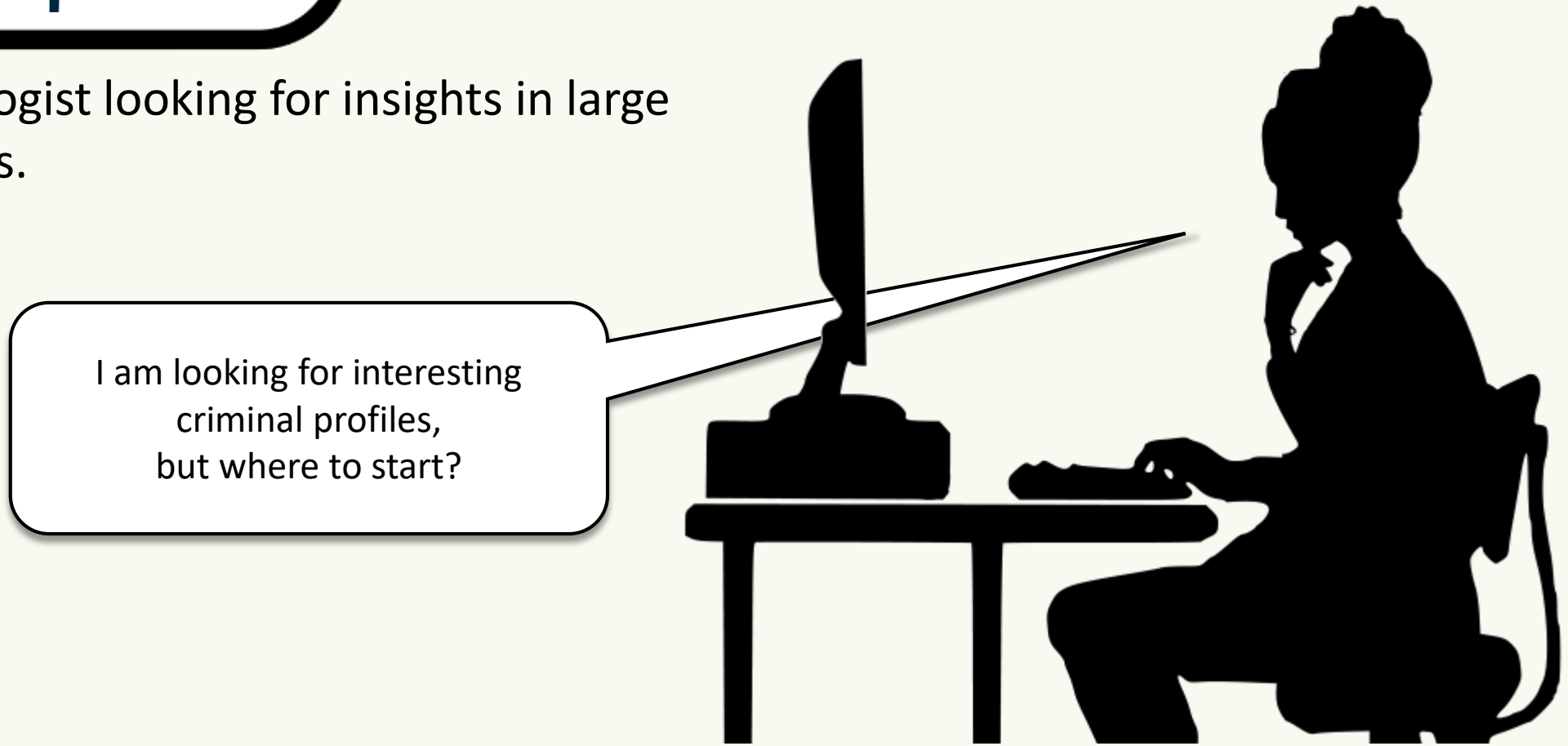
Pronounced "ya-ELLE"

Like the city

Department of Computer Science
Faculty of Exact Sciences
Bar-Ilan University

Motivating Example

Lucy is a criminologist looking for insights in large knowledge graphs.



Motivating Example

How to find sufficient examples?

DBpedia

Properties not in broader context, process may get "stuck"

Query by Example

- [ ] Angela Merkel
- [x] Silvio Berlusconi
- [ ] Saddam Hussein
- [ ] Other:

Faceted Search

Type: criminal

Birth Place:

- [ ] China
- [ ] India
- [ ] U.S.A

Office holder:

- [ ] Any

“Marrying” Examples and Facets

Lord George Gordon

PROPERTIES ABSTRACT EXTERNAL LINKS

Statement		Must	Must not	Don’t care
DEATH PLACE:	ANY	☐	☑	☐
TYPE:	Leader	☑	☐	☐
CONVICTED OF:	ANY	☑	☐	☐
CHILD OF:	ANY	☐	☐	☑
MEMBER OF:	UK Parliament	☐	☐	☑

More...

REFINE!

Our contributions

1. **Model**
2. **Encoding in SPARQL**
3. **Choosing the questions**

Model

Facts as RDF triples:

```
Saddam_Hussein type  Leader. Saddam_Hussein birthYear 1937.
∈Ent           ∈Pred  ∈Ent/Lit
```

Basic graph patterns (BGPs):

```
 ?x          type   ?y.          ?x       birthYear 1937.
```

sSPARQL (SPARQL fragment, simplified):

```
SELECT DISTINCT ?e
WHERE {?e type Person. ?e convictedOf []
MINUS {?e deathPlace []}
MINUS {?e convictedOf WarCrimes}.}
```

stands for an undistinguished variable, allowing any value to be assigned to it

User Interaction

sSPARQL (SPARQL fragment, simplified):

```
SELECT DISTINCT ?e
WHERE {?e type Person. ?e convictedOf []
MINUS {?e deathPlace []}
MINUS {?e convictedOf WarCrimes}.}
```

Multiple Semantics

Saddam Hussein is not an answer

"Negative" patterns must never hold.

- **Closed world:**
  "positive" patterns must hold

No matching answers

- **Step-wise semantics:**
  maximal subset of "positive" patterns must hold

Silvio Berlusconi and Angela Merkel

- **Weighted semantics:**
  maximal subset of "positive" patterns with maximal weight must hold

Only Silvio Berlusconi

- **Open world:**
  "positive" patterns may or may not hold

Silvio Berlusconi and Angela Merkel

Example query

```
SELECT DISTINCT ?e
WHERE {?e type Person. ?e convictedOf []
MINUS {?e deathPlace []}
MINUS {?e convictedOf WarCrimes}.}
```

Example Knowledge Graph:

```
Saddam_Hussein deathPlace Baghdad.
Saddam_Hussein type Person.
Silvio_Berlusconi convictedOf Fraud.
Angela_Merkel type Person.
```

Multiple Semantics - Results

- **Proposition** [containment between semantics]: for any Q and G, it holds that
$$Q_{Closed-world}(G) \subseteq Q_{weighted}(G) \subseteq Q_{step-wise}(G) \subseteq Q_{open-world}(G)$$

- **Proposition** [monotonicity]**:** for any sequence of questions and answers, it holds for the queries that encode them that
$Q^{i+1}_{Closed-world}(G) \subseteq Q^{i}_{Closed-world}(G)$ and $Q^{i+1}_{open-world}(G) \subseteq Q^{i}_{open-world}(G)$,
i.e., answer candidate set monotonically decreases

- **Proposition** [encoding in closed-world semantics]**:** for every sSPARQL query Q and a KG G, there exist sSPARQL queries Q^1, Q^2, Q^3 such that
  - $Q_{step-wise}(G) = Q^1_{Closed-world}(G)$
  - $Q_{weighted}(G) = Q^2_{Closed-world}(G)$
  - $Q_{open-world}(G) = Q^3_{Closed-world}(G)$

Problem definition

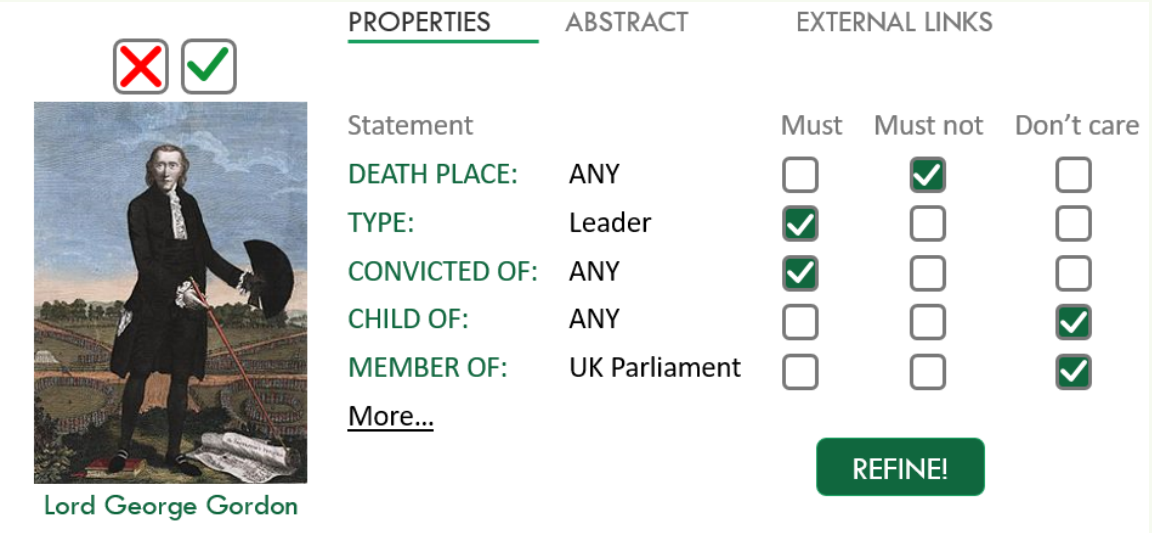
For a single question: choose a question $q_{p,o}$ that maximizes the number of eliminated candidate answers

$$E\left[\left|E_{\varphi'} - E_{\varphi}\right|\right]$$
$$= \Pr\left["\text{must}"\middle|q_{p,o}\right]\left|E_{\varphi} - E_{p,o}\right| + \Pr\left["\text{must not}"\middle|q_{p,o}\right]\left|E_{\varphi} \cap E_{p,o}\right|$$

- Depends on the semantics
- In the paper: deriving answer probabilities
- In the paper: selecting an example entity with multiple properties

Summary

- An interactive framework for knowledge graph exploration
- Selecting example entities with example properties to obtain feedback
  - Adding constraints to an sSPARQL query
- Multiple semantics for dealing with incompleteness and avoiding “dead-ends”
- See the paper for more details



Thank you!