Introducting HypoGator: generating coherent hypotheses from biased geo-political data

?

***Abstract*—[ABSTRACT]**

***Index Terms*—[Items]**

I. Introduction

Modern Knowledge Base Management Systems allow the ingestion of various data representations, such as relational, graph and full-text data [**?**]. Current KBMS usually ingest reliable data sources, such as encyclopaedic data [**?**], business data [**?**] and medical journals and clinical data [**?**]. Despite such KBMS are able to reconcile different representation towards an uniform one [**?**], no technique is currently exploited for detecting *contradicting* facts for hypothesis generation: in particular both data collected from on-line social network [**?**] or even medical diagnoses [**?**] may contain contradictions.

Example 1: One of the simplest ways to find contradictions is to check whether there are facts that are both affirmed and denied at the same time. E.g., fact "Casu Marzu is not a good cheese" is a rebuttal of the fact "Casu Marzu is a exquisite cheese". Alternatively, we can focus on factual representations that do not allow the contemporaneity of two alternative hypotheses, thus violating a functional dependency. E.g., fact "Yesterday Alice flew to Berlin" contradicts "Yesterday Alice took a trip to Kearney" because Alice cannot be found in two different places at the same time.

As previously stated, the inherent inconsistency of spurious data sources leads to the generation of conflicting hypotheses in response to a query. The inability of detecting such inconsistencies prohibits to weight how many KBMS facts either support or discard a given hypothesis, thus preventing from correctly ranking the generated hypotheses. This whole scenario demands for a query language which interpretation is able to detect such inconsistencies from the data, and an hypothesis generation task that is able to generate hypotheses which do not contain contradictions.

In order to solve this practical problem, we focus on querying domain-specific knowledge bases containing the aforementioned form of inconsistencies. Such setting allows to use ontologies to represent knowledge, thus allowing to represent both structured (tables), semistructured (wikis, web pages) and unstructured (full texts, videos, images) documents with a uniform representation [**?**], [**?**]. MeSH[1] is an example of an ontology for medical settings. Ontologies prompt the intended semantics and properties of such final representations: they differentiate *entities* from *fillers* (e.g., properties), where the formers are individual units that can be described by different qualities represented as the latter. They also distinguish the *(binary) relationships* involving such entities, and the *facts* providing a description of the observed world, which may involve multiple entities and fillers. Events are a specific case of facts. As previously mentioned, this ideal data description does not prevent the data to be biased: as an example, a different point of view may provide biased factual descriptions, that need to be detected. In order to do so, taxonomies and semantic networks focus more on describing entities and fillers through *relationships*, thus allowing to detect data inconsistencies [**?**]. ICD-11[2] is a well-known medical taxonomy used to identify and categorize specific diseases and medical procedures uniquely. Ontologies and taxonomies provide the pillars sustaining the HypoGator framework, which consists of a pipeline composed of the following blocks, which are described in more detail in the homonymous paper sections (§):

- **Query Interpretation** (§**??**): any generic query (SPARQL, CYPHER, SQL) is represented as a graph q [**?**] and then used to perform an approximated graph matching with the knowledge base [**?**].
- **Hypothesis Generation** (§**??**): the previous approximated graph matching generates the preliminar hypotheses h: similar or coherent hypotheses are merged ("synthesis"), while conflicting hypotheses are kept separated.
- **Hypothesis Evidence and Scoring** (§**??**): before returning such hypotheses, we must know which knowledge base element support the provided hypotheses and which does not. Such evidence extraction is used to provide a first scoring function counting how many supporting facts validate the hypothesis against the remaining ones (either supporting or discarding the hypothesis).
- **Hypothesis Query Answering** (§**??**): after the previous filter phase, each hypothesis h is possibly rewritten into an hypothesis h' maximizing the similarity with q. Finally, we rank the hypotheses by combining the previous section's ranking and the edit distance between h and h'.

Before providing details of such phases, we analyse the state of the art of query interpretation (§**??**), information extraction (§**??**-**??**) and knowledge base construction (§**??**).

II. Related Work

A. Natural language query answering

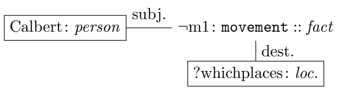
- Natural language querying in relational databases: [**?**], [**?**]

[1]https://www.nlm.nih.gov/mesh/

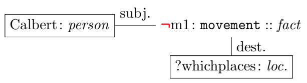
[2]https://icd.who.int/

TABLE I
ASSOCIATION BETWEEN THE VARIABLES' INSTANTIATION (**Candidate**) TO THE DATA ASSOCIATING IT (**Knowledge Base Facts**).

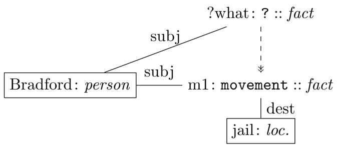
Candidate	**Knowledge Base Facts**
Rome	Abigail came back to Rome.
Rome	On September 1988, Abigail moved from Bologna's to La Sapienza University, Rome.
Latium	Yesterday (2018/06/22) Abigail was seen in Latium.
Turin	Abigail was rushed to the "Umberto I" hospital in Turin.
Minneapolis	Abigail did a trip to Minneapolis on 1987.
Duluth	Then, Abigail travelled from Minneapolis to Dulduth.



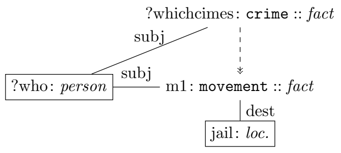
(a) "*Where did Abigail go?*"



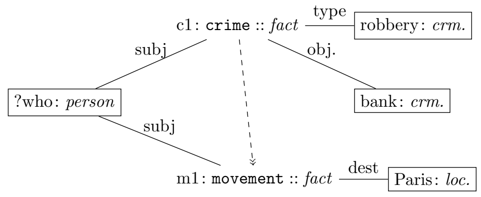
(b) "*In which places did Calbert not go?*"



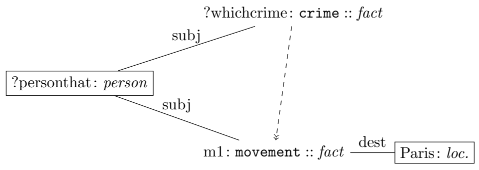
(c) "*Which crimes did Bradford committed before going to jail?*"



(d) "*Who robbed the bank before travelling to Paris?*"



(e) "*Which crimes did Bradford committed before going to jail?*"



(f) "*Which crime was committed by the person that first travelled to Paris?*"

Fig. 1. Each entity/filler is represented by a rectangle, facts have no shape and fact's fields are represented by straight edges. Variables are represented by names preceded by a question mark, ?. Temporal associations between facts are represented by dashed edges.

- Natural language querying in graph databases: [**?**]

B. Rule Mining

Miguel, here we should quote the paper on temporal rule mining.

C. Approximated graph pattern matching

- Graph matching over dimensions: [**?**]
- Approximated graph matching: [**?**], [**?**]

D. Knowledge Bases

- KB with precise information: [**?**]
- KB with uncertainty: [**?**], [**?**]

III. QUERY INTERPRETATION

Given that (natural language) queries can be expressed either as a declarative language [**?**], [**?**] or using the same format as the data [**?**], [**?**] (e.g., graphs), this paper uses the latter approach because its representation is language independent. Moreover, the latter approach is also able to express most of the natural language queries of interest within our scenario `[TODO]`.

With reference to Figure **??**, each entity and filler can be expressed as a vertex, while each binary relationship can be expressed as a edge, and facts can be expressed as hyperedges connecting different vertices [**?**]. Negations can be expressed by juxtaposing the ¬ symbol. Natural language uses adverbs or pronouns to identify which are the elements to be returned by the query, namely *variables*, which are represented as nodes (or edges) which name starts with a question mark. Such variable instantiation can be performed by approximate graph matching the query graph to the data represented in the knowledge base [**?**], thus creating a set of *morphisms* linking each variable to the corresponding matched KB elements (*candidate*). The KB's subgraph containing all the candidates for a given morphism is an *hypothesis* h. E.g., Table **??** provides in its left column a set of possible hypotheses' candidates generated from the knowledge base data and answering the query "*Where did Abigail go?*'.

We now analyse four possible queries of interest within our scenario `[TODO]`:

1) **Return an entity or filler associated to a given fact or relationship.** E.g., for "*Where did Abigail go?*" (Figure **??**), *go* is the keyword identifying the fact for the specific `movement` type, while *where* identifies the element to be returned and its type, that is a *location*. Therefore, the query asks for all the facts expressing a movement (*go*) of Abigail towards a specific location (*where*).
2) **Return a fact associated to a given fact or relationship.** E.g., in "*What did Bradford do before going to jail?*" (Figure **??**), the keywords *What* and *do* specify that we need to return a fact which is liked via a temporal association (*before*) to another fact or relationship (*Bradford ... going to jail*). In this case, the to-be-returned fact may have any possible type. The same question can be also refined to return a specific fact type as in the former example: e.g., "*Which crimes did Bradford committed before going to jail?*" (Figure **??**) may specify to return all the facts pertaining to Bradford's criminal record.
3) **Variables appearing in multiple facts/relationships.** E.g., "*Who robbed the bank before travelling to Paris?*" (Figure **??**) asks to return an entity (*Who*) which first appears in a `movement` fact (*Who ... travelling to Paris*) and, later on, performed a `crime` (*Who robbed the bank*). We want to return an entity appearing in

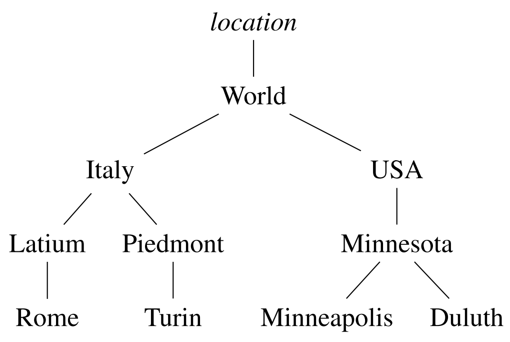


Fig. 2. An example of a geographical taxonomy for geographical dimensions (*location*). Taxonomies may be used to aggregate compatible hypotheses.

two distinct events having a temporal association. This corresponds to a join between two facts where the criminal corresponds to a moving person.
We can rephrase the same query to return a fact having a variable binding as follows: *Which crime was committed by the person that first travelled to Paris?* (Figure **??**). In this case the variable binding is made explicit by the keywords *person* and *that*, while *Which* remarks that we now want to return only the criminal facts.

4) **Negated queries.** E.g., for "*In which places did Calbert not go?*" (Figure **??**), we can provide either all the *places* appearing in negated place relationships or `movement` facts, or return a set of all the possible *places* not appearing in non-negated `place` relationships and `movement` facts. Please note that such queries can be directly expressed in graph query languages which, on the other hand, do not usually query data that may contain negations. On the other hand, current graph pattern matching frameworks may only express positive facts.

IV. Hypothesis Generation

The hypotheses generator acts as the first part for the query evaluation, which identifies and aggregates the data in the Knowledge Base (approximately) matching the query.

Let us now focus on hypotheses containing only one candidate as in Table **??**; by using some geographical taxonomy as the one in Figure **??**, we may also coarsen the former candidates in order to get a better hypothesis' support: we have that the hypothesis with candidate (h.w.c.) *Italy* is supported by three entities over five, while h.w.c. *USA* or *Minnesota* is supported by two entities over five. Please note that trivial hypotheses should be discarded: an hypothesis is trivial when it contains all the other generated hypotheses (e.g., "*World*"). Please note that single candidates cannot be inconsistent with themselves, because conflicting candidates are separated in different hypotheses by definition (e.g., all the cities are different to each other, except from Rome that appears twice). Nevertheless, not all the hypotheses are inconsistent to each other (e.g., "*Rome*" is compatible with "*Latium*" and "*Italy*", but not with "*Turin*" and "*Piedmont*").

Last, this approach can be generalized over multiple candidates by combining multiple hierarchies together as described in [**?**], where candidates may be extracted at different possible representation levels `[Is this enough or do I have to provide more details on how I would do that?]`.

V. Hypothesis Evidence and Scoring

In contrast to the evidence in the former section, having coherent candidates does not prevent us from having hypotheses that either contain contradictions or are contradicted by other facts or relationships. It is then required to associate each generated hypothesis with the set of facts and relationships providing either evidence or counterevidence.

Let us analyse some possible inconsistencies: if we suppose that the relationship "*Abigail has never been in the USA*" is stored within our knowledge base, then we now that this evidence contradicts the facts in Table **??** generating the candidates "*Minneapolis*" and "*Duluth*". On the other hand, the relationship "*Abigail was born in Detroit on 1989*" contradicts with the second "*Rome*" candidate, because the entity *Abigail* did not existed prior to 1989. Last, the fact "*On the 22nd of June 2018, Abigail died in Turin*" contradicts with the hypothesis "*Latium*" because the same person cannot be located in two different places at the same time.

The former inconsistencies may be observed by comparing different KB facts and relationships with the hypothesis, which combination describes a subset of the whole knowledge graph, κ. We can now represent facts and tuples from κ as tuples, thus allowing to use logical inconsistent detection frameworks which are independent from the graph representation of the data [**?**]. Rules may be directly provided by the ontologies' TBox-es `[cite:TODO]`, which describe the correlations between different facts and relationships and provide an homogeneous representation of different facts and relationships. E.g., the TBox may state that each `movement` fact towards a given destination at a given time implies a `location` relationship of the same subject at the same time. After converting each possible fact and relationships into "finer-grained" facts and relationships, it is now possible to recognize all the spatio-temporal inconsistencies as in [**?**]. This approach can be therefore extended to any other type of inconsistencies foreseen by the TBox rules in ontologies.

Last, we can now measure its discrepancies via support scores, thus providing a first ranking of the generated hypotheses.

VI. Hypothesis Query Answering

As previously stated, each "minimal" connected (hyper)graph containing the candidates describes an hypothesis h. Now, we want to align h towards the query representation: such alignment can be represented by instantiating the query variables in q with the candidates, thus obtaining the to-be-returned hypothesis h'. Given that h was obtained through approximate graph matching of q, we know that the greater the edit distance between the two h and h', the less the reliability on h' as a possible candidate. By doing so we assign higher scores to the hypotheses that are directly represented within the data, and inferior ones to all the remaining hypotheses that were generated over imprecise(?) inference rules. The combination of the support measure with the edit distance provides the score assigned to h'.

VII. Conclusion

Acknowledgment

This work is supported by DARPA under FA8750-12-2-0348-2 (DEFT / CUBISM).