Crowdsourcing Tasks in Open Query Answering

Elena Simperl  
Karlsruhe Institute of Technology, Germany  
elena.simperl@kit.edu

Barry Norton  
Ontotext AD, Bulgaria  
barry.norton@ontotext.com

Denny Vrandečić  
Karlsruhe Institute of Technology, Germany  
denny.vrandecic@kit.edu

Abstract
Open query answering is the idea of answering queries that are not given using the vocabulary of the queried knowledge base but instead the vocabulary of the inquirer. Many aspects of open query answering can be tackled through the combination of human effort with algorithmic techniques. In this paper we explore its applicability to crowdsourcing, using a framework in which human and computational intelligence can co-exist by augmenting existing Linked Data and Linked Service technology with crowdsourcing functionality. We analyze how the task can be decomposed and translated into Mechanical Turk projects in order to achieve this vision.

Introduction
Answering open queries, that are, queries that are not using a predefined vocabulary, required by the knowledge base, but are open to the inquirer using their own vocabulary, requires a number of tasks to be completed, namely identity resolution, vocabulary mapping, and metadata completion. These tasks have been subject to a wide array of techniques that attempt to perform them automatically, but yet require human input to produce training data and validate their results or data of sufficient quality.

We have extensively discussed in previous work the importance of combining human and computational intelligence to handle such inherently human-driven tasks. Abstracting from their technical flavor in the context of Linked Data, such tasks tend to be highly contextual and often knowledge-intensive, thus challenging to fully automate through algorithmic approaches (Cuel et al. 2011; Siorpaes and Simperl 2010). Instead of aiming at such fully automated solutions, which often do not reach a level of quality required to create useful results and applications, we propose a framework in which such human computation becomes an integral part of existing Linked Data and Linked Service technology, through crowdsourcing functionality exposed via platforms such as Amazon’s Mechanical Turk. We will demonstrate how the types of tasks that are required to answer open queries can largely be uniformly decomposed, and a formal, declarative description of the domain, scope and purpose of the application can form the basis for the automatic design and seamless operation of crowdsourcing features to overcome the limitations and complement computational methods and techniques. This paper instantiates our general description of a crowdsourcing-enabled framework for linked data management tasks (Simperl, Norton, and Vrandečić 2011) for the use case of open query answering.

The following section describes open query answering. Section then identifies the tasks required in order to answer open queries, before Section describes how they can be formalized using our framework. Related work is given in Section.

Open query answering
We define the task of open query answering as answering a query that is given using the vocabulary of the inquirer instead of the vocabulary of the knowledge base. Whereas query answering systems usually require the usage of their vocabulary and conceptualization, in open query answering this is not the case. The client application can simply use their own ontology and not worry about mapping or translating neither the results nor the query to the ontology of the knowledge base.

To give an example: assume a SPARQL endpoint that contains knowledge about a number of friends using the FOAF vocabulary (Brickley and Miller 2005), e.g. like the following set of triples:

```
hp:Harry foaf:givenName "Harry" ;
    foaf:familyName  "Potter" ;
    foaf:mbox <mailto:scarface@hogwarts.ac.uk> .
foaf:knows hp:Hermione , hp:Ron .
hp:Hermione foaf:givenName  "Hermione" ;
    foaf:familyName  "Granger" .
hp:Ron foaf:givenName  "Ron" ;
    foaf:familyName  "Weasley" ;
    foaf:mbox <mailto:weasley17@hogwarts.ac.uk> .
```

An address book application might want to query that SPARQL endpoint in order to learn about all the acquaintances of its user and to allow to select them. Assume that the application is using the vCard vocabulary (Dawson and Howes 1998), which is widely used for address books applications. If Harry uses the application, it might send the following query to the endpoint:

```
SELECT ?name ?email
WHERE {
  hp:Harry foaf:givenName "Harry" ;
    foaf:familyName  "Potter" ;
    foaf:mbox <mailto:scarface@hogwarts.ac.uk> .
foaf:knows hp:Hermione , hp:Ron .
hp:Hermione foaf:givenName  "Hermione" ;
    foaf:familyName  "Granger" .
hp:Ron foaf:givenName  "Ron" ;
    foaf:familyName  "Weasley" ;
    foaf:mbox <mailto:weasley17@hogwarts.ac.uk> .
FILTER (?name = "Hermione")
FILTER (?email = "mailto:scarface@hogwarts.ac.uk")
FILTER (?email = "mailto:weasley17@hogwarts.ac.uk")
}
```

Copyright © 2012, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

1http://www.mturk.com/
In order to achieve this goal, three steps need to happen:

1. The SPARQL processor needs to resolve the identity of the two Harrys.
2. The SPARQL processor needs to map the vCard properties used in the query to the appropriate properties in FOAF.
3. The email of Hermione needs to be added to the knowledge base.

Note that for the sake of simplicity we have chosen a fairly trivial example, but both the mappings and the identity resolution can become arbitrarily complex.

### Human intelligence tasks in open query answering

In the previous section we identified three steps required for answering the query in our example. These three steps correspond to the following three tasks in linked open data management (Heath and Bizer 2011):

1. **Identity resolution**: Although identifiers for individual resources can be directly reused in assertions in other data sets (usually as the object in triples), it is often the case that an identifier scheme is created for an overlapping set of resources. One of the Linked Data best practices stands that the predicate owl:sameAs should be used to equate the two identifiers. Identity resolution, then, involves the creation of owl:sameAs links, either by comparison of metadata or by investigation of links on the human Web.

2. **Vocabulary mapping**: It is similarly encouraged to reuse vocabulary terms over the Web, but often certain vocabularies are used in a specific context, or cover a given aspect of a domain better than another. In our example the vCard vocabulary is widely used for contact data, which is why the address book application assumes it, whereas social network data is more aligned with the FOAF vocabulary. But both ontologies cover a fairly widely overlapping domain, thus allowing to map the vocabulary and the conceptualization. Note that in the example above we only have simple and complete one-to-one mappings. In many cases the mapping may be incomplete (i.e. use the rdfs:subPropertyOf or rdfs:subClassOf predicates instead of owl:equivalentProperty respectively owl:equivalentClass) or no appropriate concepts may exist and thus would need more complex descriptions (e.g. the value for the property vcard:fn (full name) would be the concatenation of the values for foaf:givenName and foaf:lastName).

3. **Metadata completion**: It is clear, given the state of the current Web of Data (and may always be true), that certain properties, necessary for a given query, may not be uniformly populated. Manually conducted research might be necessary to transfer this information from the human readable Web, especially where scraping or database conversion have failed. In the same way, manual inspection (e.g., during browsing) or automated quality checks may indicate that the data required from a query is of low quality. Fixing this is typically a human-driven tasks, even if problem areas can be identified automatically.

### Technical realization

Our approach to use Mechanical Turk for open query answering envisions the enhancement of SPARQL query processors with integrated crowdsourcing features that deal with the packaging of specific tasks as Mechanical Turk HITs, and with the integration of the crowdsourcing results with existing, semi-automatic functionality. The former has to include user interface management capabilities, in order to produce optimal human-readable descriptions of specific tasks operating on specific data, to increase workers' productivity, and to reduce unintended behavior such as spam.

We propose to use SPARQL patterns to drive the generation of HITs interfaces. The corresponding components of the SPARQL endpoint need to interact with Mechanical Turk to post specific HITs, assess the quality of their results, and exploit these results in a particular query. In order for this interaction to occur online we may require specific optimizations to predict the time-to-completion of the crowdsourced tasks.

Technically the tasks introduced in Section can be characterized according to graph patterns from the SPARQL query language representing input to the human task and its desired output. In particular, the former expresses a query over the existing collection of Linked Data, necessary to provide the input to the task. The latter represents a query that should succeed when the additional assertions that result from the human task have been added to the knowledge. This is similar to the characterization of Linked Open Services (Norton and Krummenacher 2010) and Linked Data Services (Speiser and Harth 2011), and also the characterization of discovery goals in (Norton and Stadtmüller 2011). The approach is described in further detail in (Simperl, Norton, and Vrandečić 2011).

The inclusion of commonly-used predicates that have pre-configured screen representations in Linked Data Browsers means that tasks can be given HTML-based representations simply based on existing technology. This is a significant advantage of the Linked Data nature of the data sets being subjected to crowdsourcing in this approach. Below we sketch the form of the input and output, across the previously mentioned tasks.

---

1. E.g. the retrieval of an object connect by foaf:depiction and display it as an image.
2. See also http://km.aifb.kit.edu/sites/spark/.
Identity resolution  The identity resolution between the two Harrys can be represented by a single input graph:

```
{ OPTIONAL { ?vperson owl:sameAs ?fperson } }
```

and an output graph:

```
{ OPTIONAL ( ?vperson owl:sameAs ?fperson ) }
```

Existing Linked Data browsing technology can be used to transform the graphs satisfying these queries into an HTML page, showing the labels and the email address, with a button that will assert the `owl:sameAs` link, and another which will not (hence the `OPTIONAL` keyword in the output graph).

Vocabulary mapping  The vocabulary mapping is structurally similar to the identity resolution task in the given example, we merely need to compare the properties instead of the instances and use the `owl:equivalentProperty` predicate.

Metadata completion  In order to encode this task we check if a the required triple is part of the graph (note that the `NOT EXISTS` filter was added in SPARQL 1.1):

```
{ ?person vcard:given-name ?given ;
  vcard:family-name ?vfamily ;
  vcard:email ?vemail .
  ?person foaf:givenName ?fgiven ;
  foaf:familyName ?ffamily ;
  foaf:mbox ?fmail .
  FILTER NOT EXISTS { ?person vcard:email ?m } }
```

with the output graph:

```
{ ?person vcard:email ?email }
```

Related work

Combining traditional data management technology and crowdsourcing has recently received some attention in the area of data bases (Doan, Ramakrishnan, and Halevy 2011), with approaches such as CrowdDB (Franklin et al. 2011) and TurkDB (Parameswaran and Polyzotis 2011) proposing extensions of established query languages and processing techniques to deal with the challenging inherently arising when dealing with less deterministic computational resources such as humans in environments with high performance and scalability constraints.

There is an increasing body of research available that looks into methods and techniques to optimize worker productivity and HITs design, with the most promising findings being published at the annual HCOMP workshop.

```
^See http://www.humancomputation.com/
```

Conclusions

In this paper we have outlined a framework for crowdsourcing the steps required for the task of open query answering. We have shown for an exemplary query how it breaks down into tasks, and then how the well-defined structure and semantics of standard-compliant LOD datasets can support the automatic breakdown of tasks for further crowdsourcing. Especially the possibility to gather further information by using resolvable identifiers enables us to set up human tasks that contain all the necessary information.

We believe that the proposed solution could enhance today’s possibilities for answering open queries, providing viable and effective solutions for its challenges that are known to be reliant on human input in one way or another.

Acknowledgements

The research leading to this paper was partially supported by the Network of Excellence PlanetData, funded by the European Community’s Seventh Framework Programme FP7/2007-2013 under the contract 257641.

References


