Hu W, Qu YZ, Sun XZ. Bootstrapping object coreferencing on the Semantic Web. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 26(4): 663–675 July 2011. DOI 10.1007/s11390-011-1166-z

Bootstrapping Object Coreferencing on the Semantic Web

Wei Hu[1,2,*] (胡 伟), Yu-Zhong Qu[1,2] (瞿裕忠), *Senior Member*, *CCF*, and Xing-Zhi Sun[3] (孙行智)

[1] *State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China*

[2] *Department of Computer Science and Technology, Nanjing University, Nanjing 210093, China*

[3] *IBM Research — China, Beijing 100193, China*

E-mail: {whu, yzqu}@nju.edu.cn; sunxingz@cn.ibm.com

Received February 15, 2011; revised June 7, 2011.

Abstract An object on the Semantic Web is likely to be denoted with several URIs by different parties. Object coreferencing is a process to identify "equivalent" URIs of objects for achieving a better Data Web. In this paper, we propose a bootstrapping approach for object coreferencing on the Semantic Web. For an object URI, we firstly establish a kernel that consists of semantically equivalent URIs from the same-as, (inverse) functional properties and (max-)cardinalities, and then extend the kernel with respect to the textual descriptions (e.g., labels and local names) of URIs. We also propose a trustworthiness-based method to rank the coreferent URIs in the kernel as well as a similarity-based method for ranking the URIs in the extension of the kernel. We implement the proposed approach, called ObjectCoref, on a large-scale dataset that contains 76 million URIs collected by the Falcons search engine until 2008. The evaluation on precision, relative recall and response time demonstrates the feasibility of our approach. Additionally, we apply the proposed approach to investigate the popularity of the URI alias phenomenon on the current Semantic Web.

Keywords object coreference, entity identification, URI alias, data fusion, Semantic Web

1 Introduction

The Semantic Web (SW) is an ongoing effort by the W3C Semantic Web Activity, with the purposes of realizing data integration and sharing among different applications and organizations. To date, a number of prominent ontologies have been developed for publishing data in specific domains, such as the Friend of a Friend (FOAF), which recommend some common identifiers for classes and properties in the form of *URIs* that are widely used across data sources.

At the instance level, however, there is a lack of agreement between sources on the use of common URIs to identify a specific *object*[1]. In fact, due to the decentralized and dynamic nature of the Semantic Web, it frequently happens that different URIs from various sources, more likely originating from different RDF documents, denote the same real-world object, i.e., refer to an identical thing (also known as URI aliases[2]). Such examples exist in the domains of personal profiles, academic publications, media or geographical resources, etc.

Object coreferencing, also called object consolidation or identification[3], is a process for identifying multiple URIs of the same real-world object, that is, determining URI aliases (called *coreferent URIs* in this paper) that denote a unique object[4]. Object coreferencing is important for many data-centric applications, such as heterogeneous data integration or mining systems, Semantic Web search engines and browsers.

At present, object coreferencing has attracted significant attentions from the Semantic Web community, particularly driven by the Linking Open Data (LOD) movement[5]. At an early stage, many researchers focused on exploiting coreferent URIs between local and pairwise data sources[6-8]. With the development of Semantic Web search engines such as Sindice[9] and Falcons[10], a number of online systems have been implemented for performing large-scale, distributed and uncertain object coreferencing[1,4,11-12]. Additionally, identifying duplicate entities, which is also under the names of duplicate detection, record linkage, coreference resolution and many others, has been extensively studied in the database and natural language processing (NLP) areas[3,13-15].

Regular Paper

This work is supported in part by the National Natural Science Foundation of China under Grant Nos. 61003018 and 60973024, in part by the National Research Foundation for the Doctoral Program of Higher Education of China under Grant No. 20100091120041, and also in part by the IBM CRL UR Joint Project.

*Corresponding Author

In this paper, we propose a novel approach for bootstrapping object coreferencing on the Semantic Web. The overview architecture of the proposed approach is illustrated in Fig.1. By accepting an object URI as input, the approach in Phase 1.1 iteratively establishes a *kernel* that is mandated by the standard OWL semantics[16] of `owl:sameAs`, `owl:InverseFunctionalProperty` (`owl:IFP` for abbreviation), `owl:FunctionalProperty` (from now on, `owl:FP`), `owl:maxCardinality` and `owl:cardinality`. These five built-in vocabulary elements in OWL are considerable in number and frequently used to infer the equivalence relation in many systems[17], and combining them together can establish a larger kernel.

Then, in Phase 1.2, the approach *extends* the kernel based upon the textual descriptions of URIs. More specifically, the approach finds the objects that have exactly the same label(s) or local name (a string after the last hash "#" or slash "/" of a URI) as the ones in the kernel, after normalizing their strings in terms of punctuation, cases, stopwords, etc.

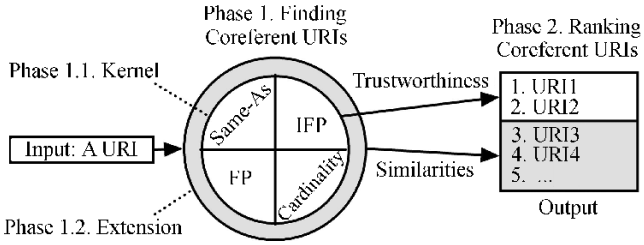


Fig.1. Overview of our approach.

In Phase 2, our approach *ranks* the coreferent URIs based on their coreference confidences. We propose a *trustworthiness*-based method for ranking the coreferent URIs in the kernel. The dereferenced documents of URIs[2] are retrieved to classify each same-as, IFP, FP or cardinality relation into three different levels of trustworthiness. The trustworthiness between the input URI and each URI in the kernel is the least trustworthy edge (i.e., same-as, IFP, FP or cardinality relation) in their most trustworthy path.

We also design a *similarity*-based method to rank the URIs in the extension of the kernel. The similarities are obtained by *matching* the contexts of URIs[18] (via RDF sentences[19]) with a linguistic matcher. Also, the *popularity* for each URI, i.e., the number of Semantic Web documents and RDF triples referring to the URI, is taken into account. The evidences for object coreferencing, including the formed equivalence relations and the matched textual descriptions of URIs, are displayed as snippets for browsing, which can help users understand the relevances between the coreferent URIs.

We implement an online prototype system named ObjectCoref (http://ws.nju.edu.cn/objectcoref/), and run it over a large-scale dataset that is collected by the Falcons search engine[10] up to September 2008. The dataset contains more than 76 million URIs and about 600 million RDF triples. The evaluation on *precision*, *relative recall* and *response time* demonstrates that, comparing with three other coreferencing systems, our proposed approach achieves a higher relative recall, but at the cost of lower precision (so ranking coreferent URIs is needed). Furthermore, we randomly pick up 100 000 URIs from the dataset and give preliminary results which indicate the prevalence of URI aliases on the current Semantic Web.

The rest of this paper is structured as follows. Related work is discussed in Section 2. Section 3 introduces a bootstrapping approach to find coreferent URIs. Section 4 describes a trustworthiness-based as well as a similarity-based methods to rank coreferent URIs. The experimental results on a large-scale dataset are reported in Section 5. Section 6 analyzes the URI alias phenomenon. Finally, Section 7 concludes this paper and gives future work.

2 Related Work

Object coreferencing is a very important mechanism for establishing semantic interoperability and realizing effective data integration. Early studies focused on performing local, pairwise and domain-specific coreferencing. For example, the work in [6] disambiguated geographical resources, the work in [7] interlinked music data and the work in [8, 20] integrated movie descriptions. Compared with them, our approach is domain-agnostic and does not tailor to any specific domain of data.

Accompanied with the development of Semantic Web search engines, such as SWSE[1], Sindice[9] and Falcons[10], and the rapid growth in Linked Data, the study of object coreferencing is drifting to Web-scale. For instance, the work in [1, 9] conducted large-scale object consolidation in terms of the analysis of IFPs, the work in [4] implemented a coreference resolution service (CRS) mainly using `owl:sameAs`, the work in [12] applied statistical methods to identify "quasi"-key properties for object coreferencing in Linked Data, and KnoFuss[17] utilized schema matching to improve the accuracy of coreferencing. The publishing-centric approach called Silk[21] enables the derivation of `owl:sameAs` relations between datasets by using manually specified mapping criteria. Additionally, the studies in [22-23] investigated the use of `owl:sameAs` on the Semantic Web, and the observation in [23] on the popularity of `owl:sameAs` is in accordance with ours.

Technically speaking, the architecture of our approach is similar to [24]. The study in [24] dedicated a large-scale clustering to ontology terms. It looked up synonyms to establish a kernel, and extended the kernel with identical terms in labels or identifiers. There are two differences between our approach and [24]: 1) we adopt OWL built-in vocabulary elements to build the kernel, which have standard semantics and usually are trustworthy, while [24] depended on thesauri, which may be imprecise in some cases; and 2) we propose ranking methods for the coreferent URIs, while [24] used a uniform threshold to filter wrong URIs, which is hard to decide across different domains.

Furthermore, our work is relevant to the problem known as duplicate detection and coreference resolution in the database and NLP fields, which have been extensively studied in the past several decades[3,13,15,25-26]. These methods are treated as similarity-based due to lack of formal semantics to define equivalence. On the Semantic Web, however, OWL provides well-defined semantics for the equivalence relation, which must be carefully considered. There also exist many works that address instance matching in ontology (or schema) matching[27-28]. But they have not been aware of the characteristics of the Web, while our method uses the dereferenceability of URIs to classify the coreference confidence.

3 Finding Coreferent URIs

In the section, we will introduce our algorithm BOCr for bootstrapping object coreferencing.

The overview of BOCr is illustrated in Fig.2, which consists of two major iterations. Taking an object URI u as input, the algorithm initializes an empty queue Q and pushes u into Q. E is defined to record the equivalence relations between URIs. For building the kernel (Lines 4∼19), BOCr iteratively picks up each unchecked URI v in Q, and starts four parallel threads: `CorefBySameAs`(), `CorefByIFP`(), `CorefByFP`() and `CorefByCard`(), in order to perform coreferencing on v (see Lines 6∼13). Different equivalence relations with different marks (e.g., "same-as") are put into E through `AddEquivRel`() for further ranking, while U' keeps the newly found URIs in each iteration to avoid duplicate coreferencing in Q. The kernel iteration converges when there is no URI in Q. Then, the normalized textual descriptions of the URIs in the kernel are extracted for extension. `ExtendByDesc`() searches the objects with the same descriptions as the ones in the kernel (Line 22). BOCr returns a set of URIs U that denote the same object as u, and a set of same-as, IFP, FP and cardinality relations E.

```
Input: A URI u that denotes an object.
Output: A coreferent URI set U, and a set of
same-as, IFP, FP and cardinality relations E.
 1  U ← {u};
 2  Q.Push(u);                    /* Q is a queue */
 3  E ← ∅;
 4  while Q ≠ ∅ do                /* Kernel */
 5      v ← Q.Pop();
 6      U_v^s ← CorefBySameAs(v);
 7      AddEquivRel(E, v, U_v^s, "same-as");
 8      U_v^i ← CorefByIFP(v);
 9      AddEquivRel(E, v, U_v^i, "IFP");
10      U_v^f ← CorefByFP(v);
11      AddEquivRel(E, v, U_v^f, "FP");
12      U_v^c ← CorefByCard(v);
13      AddEquivRel(E, v, U_v^c, "cardinality");
14      U' ← (U_v^s ∪ U_v^i ∪ U_v^f ∪ U_v^c)\U;
15      for each v' ∈ U' do
16          Q.Push(v');
17      end
18      U ← U ∪ U';
19  end
20  for each v ∈ U do              /* Extension */
21      s ← Desc(v);
22      U_v^d ← ExtendByDesc(s);
23      U ← U ∪ U_v^d;
24  end
25  return U, E;
```

Fig.2. Algorithm BOCr.

Example. Fig.3 illustrates a set of coreferent URIs regarding Chris Bizer at the Free University Berlin, which are represented with the solid pattern for the kernel and the dotted one for the extension. Supposing that `sw:chris-bizer` is the URI to start from. By searching for the objects linking with `owl:sameAs`, several coreferent URIs are discovered, such as `ontoworld:Chris_Bizer`. At the time, if we have knowledge of some IFPs, which are shown with asterisks (∗) in the figure, we find the objects having the same values as well, e.g., `bizer:chris`. Notice that the construction of the kernel is an iterative process, and the same-as or IFP relations (see the solid lines in the figure) are recorded. For instance, `dblp:Christian_Bizer` is reached from `bizer:chris`, and they have a same-as relation. Next, the kernel is extended by using the descriptions of the URIs from the kernel (e.g., "chris bizer", "chris" and "christian bizer"). This may cause errors, so ranking strategies to reflect their confidences are required. We will introduce our methods in Section 4.

3.1 Coreferencing by Same-As

Let U be a set of URI references, B be a set of blank

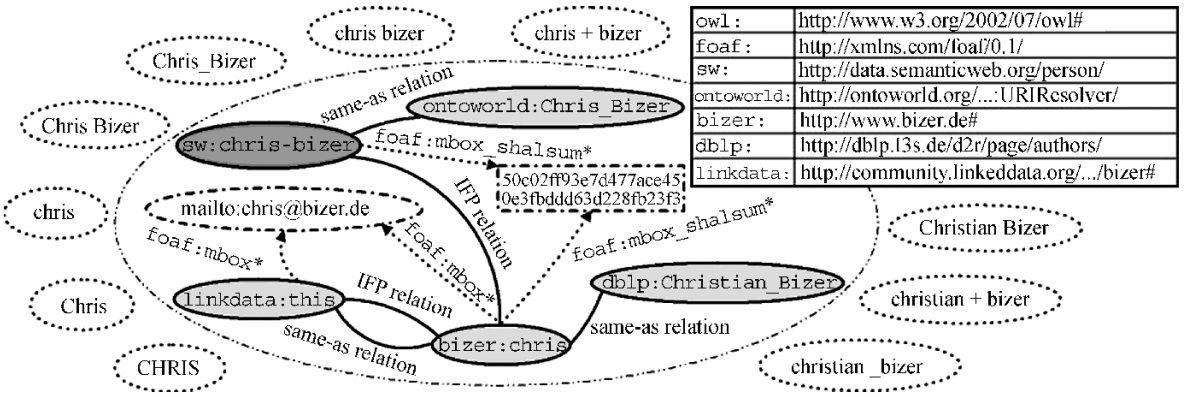


Fig.3. Example.

node IDs and L be a set of literals. A triple $\langle s, p, o\rangle \in (U \cup B) \times U \times (U \cup B \cup L)$ is called an RDF triple[29].

The semantics of `owl:sameAs` specifies that all the URIs linked with this property, in the representation of $\langle s, \texttt{owl:sameAs}, o\rangle$ (called *same-as triple* for simplicity from now on), have the same identity[16], implying that the subject and object should be the same resource.

Definition 1 (Same-As Relation). *Let U be a set of URIs. The same-as relation, denoted by $\mathbb{S}$, is defined as the minimal reflexive and symmetric relation on U, which satisfies the following conditions*: 1) $\forall s \in U$, $\langle s, s\rangle \in \mathbb{S}$; 2) $\forall s, o \in U$, *if there is a same-as triple* $\langle s, \texttt{owl:sameAs}, o\rangle$, *then* $\langle s, o\rangle \in \mathbb{S}$ *and* $\langle o, s\rangle \in \mathbb{S}$.

3.2 Coreferencing by IFPs

The semantics of an IFP guarantees that a value can only be the value of this property for a single object, i.e., two separate objects are indirectly inferred to be identical based on having the same value of that property[16].

To identify IFPs, we parse ontologies to find the properties whose `rdf:type` is explicitly given as `owl:IFP`. This is done at preprocessing time. Note that IFPs can be inferred in multiple ways based on OWL semantics. For example, the work in [30] did reasoning over pD* that includes rules for handling `owl:sameAs`, `owl:IFP` and `owl:FP` axioms. However, anyone can define anything on the Semantic Web, and thus inferring IFPs over different sources may cause errors and inconsistency. For instance, we inferred `dc:title` as an IFP in the Falcons dataset. The work in [31] studied the reasoning problem of new ontologies published on the Web redefining the semantics of existing entities resident in other ontologies (called ontology hijacking), which enlightens us to only consider dereferenceable[2] IFPs in our approach for avoiding ontology hijacking.

Definition 2 (IFP Relation). *Let U be a set of URIs. The IFP relation, denoted by $\mathbb{I}$, is defined as the minimal reflexive and symmetric relation on U, satisfying that*: 1) $\forall s \in U$, $\langle s, s\rangle \in \mathbb{I}$; 2) $\forall s_1, s_2 \in U$, *if there exist an IFP p and two IFP triples* $\langle s_1, p, o\rangle$, $\langle s_2, p, o\rangle$, *then* $\langle s_1, s_2\rangle \in \mathbb{I}$ *and* $\langle s_2, s_1\rangle \in \mathbb{I}$.

To find the IFP relations, we match the values of objects in the RDF triples with the same IFPs as the predicates. If the values are exactly the same (by implementing a trivial string comparison algorithm), then we construct an IFP relation between the subjects of those IFP triples. For example in Fig.3, because both `sw:chris-bizer` and `bizer:chris` have an IFP `foaf:mbox_sha1sum`, and the sha1sum values are the same, we bridge an IFP relation between them. This approach has been shown to be feasible in [1, 9]. Also, the lexical forms of some literals can be empty, where for example the values of `foaf:mbox_sha1sum` are blank in a few triples. We omit these triples to avoid wrong coreferencing.

Due to several heterogenous ways for expressing email addresses, we develop an ad hoc method for identifying identical email addresses for two widely-used IFPs: `foaf:mbox` and `foaf:mbox_sha1sum`. Given an email address, we compute its `sha1sum` value and utilize `foaf:mbox_sha1sum` to find new coreferent URIs. We bridge the IFP relations between the URIs using the two IFPs.

3.3 Coreferencing by FPs and (Max-) Cardinalities

The process of using FPs to find coreferent URIs is similar to that of IFPs. We first recognize the dereferenceable properties whose types are explicitly defined as `owl:FP`, and then use these FPs to construct the FP relations.

Definition 3 (FP Relation). *Let U be a set of URIs. The FP relation, denoted by $\mathbb{F}$, is defined to be the minimal reflexive and symmetric relation on U, which satisfies that*: 1) $\forall o \in U$, $\langle o, o\rangle \in \mathbb{F}$; 2) $\forall o_1, o_2 \in U$, *if there*

exist an FP p and two FP triples $\langle s, p, o_1\rangle$, $\langle s, p, o_2\rangle$, then $\langle o_1, o_2\rangle \in \mathbb{F}$ and $\langle o_2, o_1\rangle \in \mathbb{F}$.

The cardinality constraint `owl:cardinality` (or `owl:maxCardinality`) is a built-in OWL property that links a restriction class with a data value. A restriction having an `owl:cardinality` (or `owl:maxCardinality`) constraint describes a class of all objects that have exactly (at most) N semantically distinct values for the property concerned, where N is the value of the cardinality constraint. If $N = 1$, its semantics is similar to FPs (but with respect to a particular class) and can be applied to produce coreferent URIs.

Definition 4 (Cardinality Relation). *Let U be a set of URIs. The cardinality relation, represented by $\mathbb{C}$, is defined as the minimal reflexive and symmetric relation on U, which satisfies that*: 1) $\forall o \in U$, $\langle o, o\rangle \in \mathbb{C}$; 2) $\forall o_1, o_2 \in U$, *if there exist a (max-)cardinality restriction $\langle c,$ `owl:onProperty`$, p\rangle$, $\langle c,$ `owl:maxCardinality`$,$ "1"$\rangle$ (or $\langle c,$ `owl:cardinality`$,$ "1"$\rangle$), where c, p are the restriction class and property respectively, and two cardinality triples $\langle s, p, o_1\rangle$, $\langle s, p, o_2\rangle$ with $\langle s,$ `rdf:type`$, c\rangle$, then $\langle o_1, o_2\rangle \in \mathbb{C}$ and $\langle o_2, o_1\rangle \in \mathbb{C}$.*

3.4 Kernel

Based on the same-as, IFP, FP and cardinality relations, we define the equivalence relation below.

Definition 5 (Equivalence Relation). *Let $\mathbb{S}, \mathbb{I}, \mathbb{F}, \mathbb{C}$ be the same-as, IFP, FP and cardinality relations on a set U of URIs, respectively. $\mathbb{K}$ is the transitive closure on $\mathbb{S} \cup \mathbb{I} \cup \mathbb{F} \cup \mathbb{C}$.*

It is worth noting that $\mathbb{K}$ is an equivalence relation on $\boldsymbol{U}$, because $\mathbb{S}, \mathbb{I}, \mathbb{F}, \mathbb{C}$ are all reflexive and symmetric.

Theorem 1 (Kernel). *Let U be a set of URIs. For a URI $u \in U$, the algorithm BOCr establishes an equivalence class $[u]_{\mathbb{K}} = \{v \mid v \in U, \langle u, v\rangle \in \mathbb{K}\}$, called a kernel, of u under the equivalence relation $\mathbb{K}$.*

Proof. See the algorithm BOCr. The detailed proof is straightforward. □

It is worth noting that more and more vocabularies are being published on the Semantic Web, we will consider more vocabulary elements for the kernel, e.g., `skos:exactMatch`.

3.5 Extension from the Kernel

On the current Semantic Web, there are a considerable amount of coreferent URIs that have not been interlinked with `owl:sameAs` or others. For instance, we find more than 70 URIs denoting Tim Berners-Lee in our dataset, where only 9 of them are resident in the kernel. This inspires us to extend coreferent URIs from the kernel.

The method `ExtendByDesc`() in the algorithm searches those URIs satisfying the condition that their textual descriptions exactly match the one of any URI in the kernel. The rationale is that URIs having identical textual descriptions are likely to refer to the same object. The exact match implies that the approach would discard the URIs whose textual descriptions are only a part (substring) of the ones in the kernel. In [32], the same method is applied to match two hundred biomedical ontologies effectively. In this paper, we use $\mathbb{E}$ to denote the extension relation.

Specifically, the textual description of a URI is derived from the `rdfs:label`(s) in its dereferenced document or from its local name (the string after the last hash "#" or slash "/" of the URI). The textual descriptions are normalized via uniforming string cases and removing some employed delimiters. For example, "Tim Berners-Lee" equals "Tim_Berners_Lee" and "tim+berners-lee". It is also important in practice to remove some non-content-bearing stopwords like "user", "xml" and "rdf". Currently, we defined 616 stopwords.

Looking up synonyms in thesauri, e.g., WordNet or Geonames, may also bring benefits to the extension by increasing the number of coreferent URIs, especially for the multilingual case. But we do not use any background thesaurus at present, since it is often difficult to find proper synonyms at the instance level. For example, WordNet cannot return any synonym for "Tim Berners-Lee". Also, if the description is polysemic, the extension tends to involve wrong or ambiguous results. Nevertheless, we will consider the use of appropriate thesauri in future work.

We store the normalized textual descriptions of URIs in a database table, and create indexes on them for quick search.

4 Ranking Coreferent URIs

In the previous section, we constructed a number of coreferent URIs. In this section, we propose a trustworthiness-based method to rank the coreferent URIs in the kernel and a similarity-based method for ranking the URIs in the extension, depending on the input URI. Coreferencing evidences like the same-as, IFP, FP and cardinality relations and the matched textual descriptions of URIs are displayed as snippets for user browsing. In general, we employ the following heuristic rules:

• For the kernel, a URI can be reached from the input URI via an equivalence relation. Each equivalence relation is constituted by at least one path of the same-as, IFP, FP and cardinality relations (i.e., edges). The dereferenced documents of URIs[2] are retrieved to classify the trustworthiness of a same-as, IFP, FP or

cardinality relation into different levels. The trustworthiness between the input and any URI in the kernel is the least trustworthy edge in their most trustworthy path. We assume that the URIs in the kernel are always ranked higher than the ones in the extension.

• The context for each URI in the extension (as well as the input URI) is constructed by using the notion of RDF sentences[19]. The similarity between any URI in the extension and the input URI is calculated by matching their contexts with a linguistics-based matching algorithm V-Doc[18]. The greater the similarity, the higher the rank of the URI in the extension.

• Furthermore, when the coreference confidences of the URIs in the kernel (in the extension) are equal, a popularity-based ranking takes effect to impose the total order of the ranking. We count the number of Semantic Web documents (SWDs) and RDF triples referring to each URI. The rationale is that, if some URI appears in more SWDs or RDF triples, people might be more familiar with the URI, and they would like to see the URI being ranked higher. We also prefer the number of SWDs to RDF triples for each URI. Please note that certain linkage-based techniques, e.g., PageRank[33] and HITS[34], can also be adopted to rank coreferent URIs. But the size of a kernel is usually small and the similarities computed by V-Doc are unlikely to be the same, thus the linkage-based analysis cannot bring much benefit to the ranking.

4.1 Ranking for the Kernel

Because anyone can say anything on the Semantic Web, the trustworthiness of the same-as, IFP, FP and cardinality relations needs to be classified. The act of retrieving a representation of a resource identified by a URI is referred to as *dereferencing* that URI[2]. The document retrieved by dereferencing the URI of an object can be considered as the *authoritative* definition of that URI[31]. This inspires us to compare the dereferenced documents of URIs with the documents that declare the same-as, IFP, FP or cardinality triples.

Let $\langle s, o\rangle \in \mathbb{S}$ be a same-as relation. $deref(s)$ and $deref(o)$ be the dereferenced documents of s and o, respectively. *doc* is a document that contains the corresponding same-as triple. We distinguish the trustworthiness of $\langle s, o\rangle$ into three different levels (due to the symmetry of the same-as relation), which are shown in Fig.4.

• Level 1: $deref(s) = doc = deref(o)$. This case happens where s, o are dereferenced to the same document of the same-as triple. It is considered to be more trustworthy than the other two levels.

• Level 2: $deref(s) = doc \neq deref(o)$ or $deref(s) \neq doc = deref(o)$. This is caused by the different dereferenced documents of s and o, and only one of them equals *doc*. For example, one party declares the same-as relation with existing objects. Another such case is either s or o is dereferenceable, while the other is not.

• Level 3: $deref(s) \neq doc \neq deref(o)$, because 1) neither s nor o is dereferenceable; or 2) a third-party document suggests the relation, which does not equal $deref(s)$ or $deref(o)$. This constitutes the least trustworthy level.

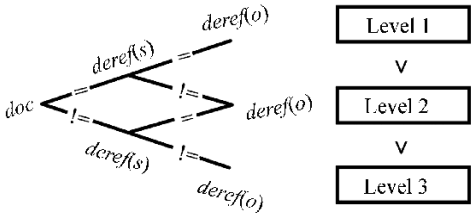


Fig.4. Classification of same-as relation.

Additionally, there are two special cases to be addressed: 1) a same-as relation may be formed from different same-as triples, thus there can be more than one *doc* for a single same-as relation. In our approach, we use the highest level of trustworthiness as its level; and 2) reciprocal same-as triples[23] exist in Level 2. For example, assuming that there exists a same-as relation $\langle s, o\rangle$ that is derived from two documents doc_1 and doc_2. In doc_1, $deref(s) = doc_1 \neq deref(o)$; while in doc_2, $deref(s) \neq doc_2 = deref(o)$. We upgrade the case to the same trustworthiness as Level 1. In our dataset, we found the reciprocal same-as triples among two or three documents, but no more than three documents.

Similar to the same-as relation, given two IFP triples $\langle s_1, \text{IFP}, o\rangle$ and $\langle s_2, \text{IFP}, o\rangle$ declared in doc_1 and doc_2 respectively, we distinguish the trustworthiness of the IFP relation $\langle s_1, s_2\rangle \in \mathbb{I}$ into three levels, based on the dereferenceability of s_1 and s_2 (see Fig.5). We select the highest level of trustworthiness of an IFP relation as its level.

• Level 1: $deref(s_1) = doc_1$ and $deref(s_2) = doc_2$. The case is considered as most trustworthy in the three levels of trustworthiness of the IFP relation.

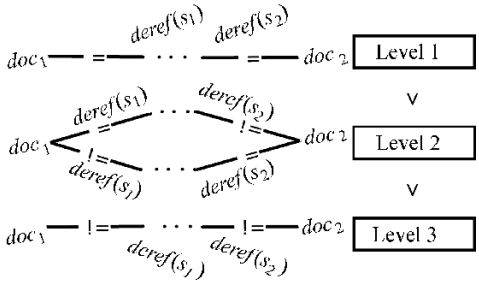


Fig.5. Classification of an IFP relation.

• Level 2: $deref(s_1) = doc_1$ and $deref(s_2) \neq doc_2$, or $deref(s_1) \neq doc_1$ and $deref(s_2) = doc_2$. It is assumed to be more trustworthy than Level 3.

• Level 3: $deref(s_1) \neq doc_1$ and $deref(s_2) \neq doc_2$.

Based on the same manner (the dereferenceability of subjects) as the IFP relation, we divide an FP relation or a cardinality relation into three different levels of trustworthiness as well. We omit the details in this paper.

A URI in the kernel is reachable from the input URI through an equivalence relation, which is composed of at least a *path* of the same-as, IFP, FP and cardinality relations under our definition of the equivalence relation. An *edge* on a path is a same-as, IFP, FP or cardinality relation. The *distance* (length) of an edge is assigned according to its trustworthiness level: 1 for Level 1, 2 for Level 2 and 3 for Level 3. We define the *shortest* path from the input URI to a URI in the kernel as the most trustworthy path between them. The trustworthiness between any URI and the input is the *least* trustworthy edge ($= 1/edge\_distance$) in the shortest path.

Example. In Fig.3, `sw:chris_bizer` can reach `linkeddata:this` through two IFP relations, or through an IFP relation and a same-as relation. Let us assume that the two IFP relations between `sw:chris_bizer` and `bizer:chris` and between `bizer:chris` and `linkeddata:this` are in Level 2, and the same-as relation between `bizer:chris` and `linkeddata:this` is in Level 3. The edge distance from `bizer:chris` to `linkeddata:this` is 2. So, the shortest path from `sw:chris_bizer` to `linkeddata:this` contains two IFP relations, and the shortest path distance is 4 ($= 2 + 2$). Therefore, the trustworthiness between them is 0.5.

4.2 Ranking for the Extension

To rank the URIs in the extension, we propose a matching-based method to compute the similarity between each URI in the extension and the input URI.

A URI is a compact sequence of characters which identifies an abstract or physical resource. Although URIs contain a few textual descriptions, e.g., the local names for the objects they denote, they alone are not adequate to determine whether the denoted objects are similar.

In an RDF graph, a URI is involved in a set of RDF triples that form a context for the URI. We use the notion of RDF sentences[19] to extract contexts, which guarantees the completeness of blank nodes in the contexts. Blank nodes are a class of existentially quantified resources whose meanings exist in the scope of the triples they appear. RDF triples that share a blank node form an integrated structure to indicate a joint context of that blank node. If such triples are separated, the context is broken. But, RDF semantics[29] provides no intrinsic mechanism to preserve this kind of structures.

We define that two RDF triples are *b-connected* if they share some blank nodes. The b-connected relation is defined as transitive, that is, two RDF triples are b-connected if they both b-connect to another triple. An RDF sentence in an RDF graph is the maximum closure of the b-connected RDF triples (called a minimum self-contained graph in [35], which proved that an RDF graph can be decomposed into a unique set of RDF sentences).

Definition 6 (RDF Sentence). *An RDF sentence, st, is a subset of RDF triples T in an RDF graph g, which satisfies the following conditions*:

• $\forall t_i, t_j \in st$, t_i, t_j *are b-connected*;

• $\forall t_i \in st, t_j \in g \setminus st$, t_i, t_j *are not b-connected.*

Furthermore, we define:

$$subj(st) = \{s \mid \exists \langle s, p, o\rangle \in st\}, \quad (1)$$

$$pred(st) = \{p \mid \exists \langle s, p, o\rangle \in st\}, \quad (2)$$

$$obj(st) = \{o \mid \exists \langle s, p, o\rangle \in st\}. \quad (3)$$

Next, we introduce our definition of the *context* for a URI using RDF sentences as basic units instead of RDF triples, where the URI appears as the subject of some triples in each RDF sentence. This is similar to the Concise Bounded Description[36].

Definition 7 (Context). *Let u be a URI. The context of u, denoted by $ctx(u)$, is a set of RDF sentences $ctx(u) = \{st_1, st_2, \ldots, st_n\}$, where each st_i $(i = 1, 2, \ldots, n)$ satisfies that $u \in subj(st_i)$.*

With regard to the given URI for constructing a context, only forward links ($u \in subj(st_i)$) are considered in the context. However, the direction of a property in RDF is somehow arbitrary so that backward links may also be considered. Although it is not difficult to include backward links, only forward links are included here because we think that the predicates and objects in RDF triples are mainly used for describing the subjects, and the number of triples in each context can be reduced. The impact of including backward links in contexts will be considered in our future work.

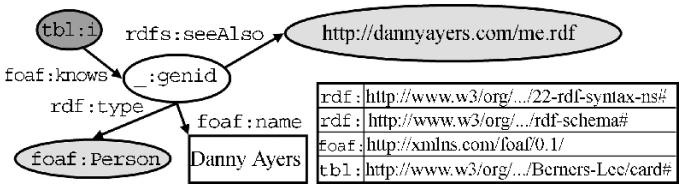


Fig.6. Context for `tbl:i`.

Example. Fig.6 shows an RDF sentence that consists of four RDF triples. The RDF sentence forms a context stating that `tbl:i` knows a person whose name is "Danny Ayers", etc. If we only extract the triple $\langle$`tbl:i`, `foaf:knows`, `_:genid`$\rangle$ for `tbl:i`, it can hardly give any useful information due to the blank node `_:genid`.

After forming the contexts for all the URIs in the extension and the input URI, we now adopt a linguistics-based ontology matching algorithm V-Doc[18] to compare the context of each URI in the extension with the input (rather than all the URIs in the kernel due to computation cost).

Terms in the context of a URI are extracted to construct the virtual document for that URI. To formalize, we firstly define the sets of neighboring URIs and literals within a context. Let U, L be the sets of all URIs and all literals, respectively. Given the context $ctx(u)$ of a URI u, the neighboring URIs and literals in the context are:

$$neighURI(u) = \{v \in U \mid \exists st(st \in ctx(u) \wedge v \in (pred(st) \cup obj(st)))\}, \quad (4)$$

$$neighLit(u) = \{l \in L \mid \exists st(st \in ctx(u) \wedge l \in obj(st))\}. \quad (5)$$

We use the vector space model to represent the virtual document of a URI. Given $\forall u \in \boldsymbol{U}$, let $\boldsymbol{ln}(u)$ be the term vector representing its local name, and $\boldsymbol{lbl}(u)$ be the term vector representing its label(s) encoded in its dereferenced document. Given $\forall l \in \boldsymbol{L}$, let $\boldsymbol{lexForm}(l)$ be the term vector that denotes its lexical form. The virtual document of a URI u, denoted by $VD(u)$, is computed by involving not only the local textual description of u, but also the descriptions of neighbors to reflect its intended meaning:

$$\boldsymbol{VD}(u) = \alpha \cdot \boldsymbol{ln}(u) + \beta \cdot \boldsymbol{lbl}(u) + \gamma \cdot \sum_{v \in neighURI(u)} (\boldsymbol{ln}(v) + \boldsymbol{lbl}(v)) + \theta \cdot \sum_{l \in neighLit(u)} \boldsymbol{lexForm}(l), \quad (6)$$

where $\alpha, \beta, \gamma, \theta$ are weighting coefficients, and we set them to 2, 2, 1 and 1 respectively, according to our past experience of using virtual documents in ontology matching[18].

The problem of computing similarities between the input URI and a URI in the extension is transformed into the problem of computing similarities between two virtual documents, which has been extensively studied in information retrieval. For a URI u, let $\boldsymbol{VD}(u)$ be the term vector representing its virtual document calculated according to (6). $\boldsymbol{VD}(u)$ is refined by inverse document frequency (IDF) factors, that is, a higher weight is assigned to a term in a virtual document if the term occurs in fewer documents. The similarity between the input URI u and any URI v in the extension is computed based on the cosine similarity measure:

$$sim(u, v) = \frac{\boldsymbol{VD}(u) \bullet \boldsymbol{VD}(v)}{|\boldsymbol{VD}(u)| \cdot |\boldsymbol{VD}(v)|}, \quad (7)$$

where the numerator is the dot product of the two term vectors, and the denominator is the product of their magnitudes.

We utilize these cosine similarities to rank the coreferent URIs in the extension. The greater the similarity, the higher the rank of the URI in the extension.

5 Evaluation

We implemented an online system, called ObjectCoref, for our proposed method. In this section, we report the experimental results on a large-scale dataset collected by the Falcons[10] search engine till September 8th, 2008. All the tests were carried out on an Intel Core2 Duo 2.40 GHz CPU, 4 GB memory with Windows 7 Professional and Java 6. The dataset was indexed on four Xeon Quad 2.40 GHz CPUs, 16 GB memory with Red Hat Linux Enterprize Server 5.3 and MySQL 5.0.

5.1 Dataset Characteristics

The characteristics of the dataset are listed in Table 1. This dataset contains 596 418 935 RDF triples (tr.) from 11 719 608 SWDs which refer to 76 389 570 URIs, equal to an average of 7.81 RDF triples per URI.

Table 1. Characteristics of the Dataset

URIs	Same-As tr.	IFP tr.	FP tr.	Card. tr.
76 389 570	7 880 906	27 686	35 652	34 635

By investigating the dataset, 7 880 906 unique same-as triples were found, where 474 869 triples are in Level 1, 5 228 423 in Level 2 and 2 177 614 in Level 3. Furthermore, there are 828 086 reciprocal same-as triples in Level 2 between two documents and 1 020 reciprocal ones among three documents, which are mainly from http://bio2rdf.org and http://dbpedia.org. Considering the purpose of ObjectCoref, blank nodes provide no meanings outside their original scopes. So, we omitted the same-as (and IFP, FP, cardinality) triples with blank nodes in our experiment.

We discovered 1 791 unique IFPs from the dataset, where 413 ones (23%) are dereferenceable. Based on

these dereferenceable IFPs, 27 686 IFP triples without any blank nodes were retrieved, and most of them are about `foaf:mbox_sha1sum` (11 903) and `foaf:mbox` (2 981). We also found that 11 760 IFP triples have the IFP relations with others.

For FPs, we found 11 765 dereferenceable FPs from 21 067 ones in total, and derived 35 652 FP triples without blank nodes, in which 440 triples have the FP relations with others.

In addition, we discovered 113 dereferenceable properties that are involved in 34 635 cardinality triples without blank nodes, where 6 123 have the cardinality relations with others.

Based on the statistical data, we can see that the bulk (99.8%) of equivalence relations are given by explicit same-as triples.

5.2 Precision and Relative Recall

In this experiment, we tested the top-k precision and relative recall of ObjectCoref on performing object coreferencing on the Semantic Web. We analyzed 364 408 query logs in the Falcons search engine, and chose 10 popular query URIs in object search, which are depicted in Table 2. These URIs cover a wide range of real-world domains (such as people, geography), and some of their local names have several meanings. For instance, "Jaguar" can be either a mammal or a car brand. We selected each URI having a clear meaning among its polysemic local names, to observe whether or not ObjectCoref can identify the correct coreferent URIs and how well.

Table 2. Summary of Sample URIs

	HTTP URI
1	www.w3.org/People/Berners-Lee/card#i
2	data.semanticweb.org/person/chris-bizer
3	www.cs.vu.nl/...#Frank+van+Harmelen
4	www4.wiwiss.fu-berlin.de/factbook/.../China
5	dbpedia.org/resource/United_States
6	dbpedia.org/resource/Berlin
7	dbpedia.org/resource/Semantic_Web
8	dbpedia.org/resource/Apple_Inc.
9	dbpedia.org/resource/Jaguar [mammal]
10	semanticweb.org/wiki/Java [programming]

We employed 5 students to perform peer reviews for the top-10 and top-20 coreferent URIs found by ObjectCoref. A student judges if each returned URI is coreferent with the input, in terms of the provided evidences like equivalence relations, textual descriptions, snippets summarized by Falcons or dereferenced documents. But, there still exist a few URIs that are hard to identify if they are coreferent. We treated them as correct in our experiment.

The top-k precision is computed by $prec(k) = |correct(k)|/k$, and the relative recall is the number of coreferent URIs retrieved by one system divided by the total number of unique URIs from all systems. The relative recall provides a practical, if imperfect, solution to the problem that the total number of results is unknown, and it has been widely used to assess the quality of large ontology matching[27]. One may think to use the correct number of coreferent URIs of one system over the number of correct ones from all systems as the relative recall, but this might be impossible if the number is very large. For example, to evaluate the relative recall of a search engine, it is infeasible to determine the correct answers from ten thousands of returned results for a query.

The coreferent URIs without ranking (denoted by W/O-rank) was adopted to compare the top-k precision. Furthermore, three coreferencing websites called sameas.org (http://sameas.org/), rkbexplorer (http://www.rkbexplorer.com/sameAs/, by CRS[4]) and sig.ma (http://sig.ma/, by Sindice[9]) provided services for object coreferencing in terms of the same-as or IFP relations. We input the 10 sample URIs in Table 2 into them, and collected the coreferent URIs that they returned. It is worth noting that all the services above are built on different datasets, therefore the returned results only have partial overlap.

The comparisons on the top-10 and top-20 precision of W/O-rank and ObjectCoref are depicted in Figs. 7(a) and 7(b), respectively. These figures indicate that both W/O-rank and ObjectCoref are effective to achieve a good precision for Semantic Web object coreferencing. In addition, by ranking the coreferent URIs, the precision can be further improved. In particular, by using the matching-based ranking, the top-20 precision averagely increases, especially in No.2 ("chris-bizer") and No.8 ("Apple_Inc."). The reason is that, with the context matching, the URIs that are more similar to the input are ranked higher than the less similar ones. For instance, in No.2, with the help of contexts, another "Chris" (Chris Lilley at the W3C) in the extension was moved to the end of the query result. In addition, our experiment also suggested that the coreferent URIs in the kernel are usually correct (see the top-10 precision).

The comparison on the top-20 precision of ObjectCoref, sameas.org, rkbexplorer and sig.ma is shown in Fig.8(a), while the relative recall is in Fig.8(b). From the experiment, we saw that sameas.org and rkbexplorer used a closely similar dataset but with different ranking schemes, while sig.ma focused on data mashup and limited the size of returned URIs per query no more than 20.

From the figures, we observed that the relative recall

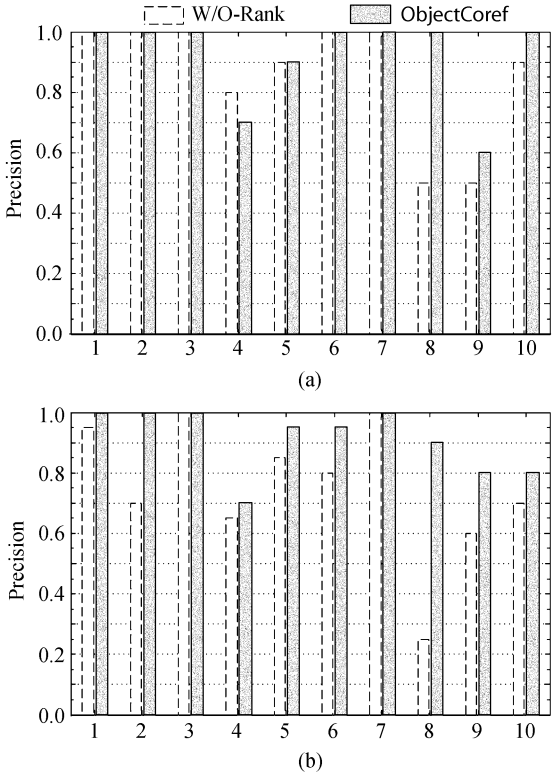


Fig.7. W/O-rank vs. ObjectCoref. (a) Top-10 precision. (b) Top-20 precision.

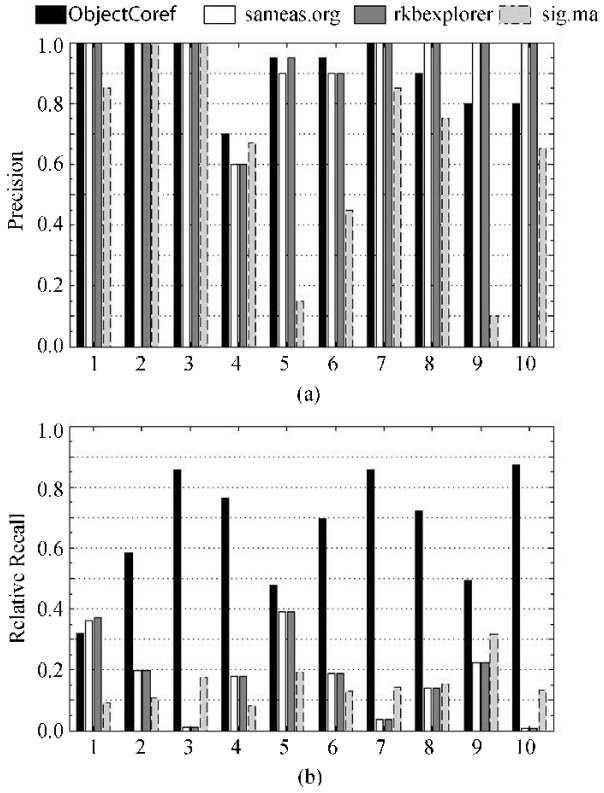


Fig.8. ObjectCoref vs. others. (a) Top-20 precision. (b) Relative recall.

of sameas.org, rkbexplorer and sig.ma are significantly smaller (usually less than 0.2) than the one of ObjectCoref, even excluding the wrong coreferent URIs. This proves that ObjectCoref performed better than the rest three systems in performing object coreferencing, since iteratively combining the use of same-as, IFPs, FPs and cardinalities forms a more complete kernel, and merely using a part of them would miss some correct coreferent URIs. Additionally, the extension and ranking are feasible, ObjectCoref achieved a comparable precision (with much more coreferent URIs) as compared with the three other systems.

5.3 Response Time

We analyzed the response time of ObjectCoref with regard to the size of coreferent URIs per input. The background intuition is that the more returned URIs associated to an input, the longer ObjectCoref spends to deal with it. Our objective is to figure out whether ObjectCoref can always complete object coreferencing in a controllable time. The response time includes the time for coreferencing, ranking and generating user-friendly snippets for display.

We randomly chose 5 000 sample URIs in our dataset, and repeated the experiment 10 times to measure the average response time. Fig.9 shows the response time per size of coreferent URIs. Each point in the figure corresponds to a sample URI. From this figure, we observed that 99% of cases correspond to the samples with $\leqslant 100$ coreferent URIs, while the average response time is 162.5 ms. The maximum size of the coreferent URIs is 924, and ObjectCoref took 10.5 s to complete the case. This experiment shows that ObjectCoref is capable of being used for online purposes. We tested various curves to fit the response time, and found that the best is a linear regression with equation $y = 12.82x + 58$, where the correlation coefficient is

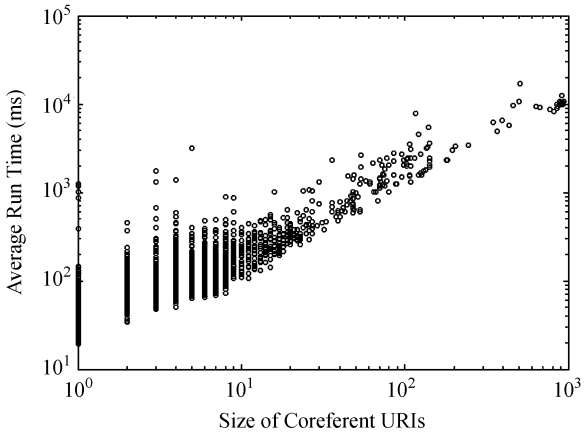


Fig.9. Response time (in log-log scale).

0.934. This proves that the response time of ObjectCoref can be approximately predicted using a linear function over the number of coreferent URIs returned.

6 Analysis of URI Alias Phenomenon

Just as one may expect to use different names to refer to the same person, Web architecture allows the association of more than one URIs to a resource. As defined in [2], the URIs denoting the same object are called *URI aliases*. Although there are benefits from URI aliases such as naming flexibility, overusing URI aliases are not encouraged because they weaken the "network effect". In this section, we investigate the popularity of URI alias phenomenon on the current Semantic Web, based upon our equivalence relation.

It is very hard, if not impossible, to identify all URI aliases on the Semantic Web. In this paper, we try to apply the equivalence relation $\mathbb{K}$ and the extension relation $\mathbb{E}$ (see Section 3) to observe the popularity of the URI alias phenomenon, and the popularity $P_{\mathbb{K}}, P_{\mathbb{E}}$ are defined as the average sizes of all unique equivalence classes under $\mathbb{K}, \mathbb{E}$, respectively.

Table 3. Popularity of URI Aliases

Indicator	Popularity
$P_{\mathbb{K}}$	1.42
$P_{\mathbb{E}}$	15.33
$P_{\text{sameas.org}}$	1.46
$P_{\text{rkbexplorer}}$	1.20
$P_{\mathbb{M}}^{300}$	4.23

We randomly chose 100 000 object URIs in our dataset and fed them into ObjectCoref. As listed in Table 3, the popularity under $\mathbb{K}$ is 1.42, while under $\mathbb{E}$ is 15.33, which indicate that there is a large gap between $P_{\mathbb{K}}$ and $P_{\mathbb{E}}$.

The top-5 largest equivalence classes under $\mathbb{K}$ and $\mathbb{E}$ are listed in Tables 4 and 5 respectively, where $[n]$ denotes the number of similar URIs having the same size. Most of the largest kernels derive from the same-as relation and fall into the biomedical domain. For the URIs in the extensions, because they are not always accurate, we resorted to manpower for obtaining the "real" popularity and the average precision of the extensions.

We randomly chose 300 URIs from the 100 K samples and employed 7 students to manually identify correct URIs in the extensions of ObjectCoref. It turns out that the popularity from the manual observation, denoted by $P_{\mathbb{M}}^{300}$, is 4.23, as compared with $P_{\mathbb{E}}^{300} = 14.99$ under the same 300 URIs. So the average precision of the extensions is about $4.23/14.99 = 0.28$. If our sampling is typical and the manual observation is relatively accurate, we can conclude that the kernel would omit some coreferent URIs, while the extension can find considerable candidates but lacks accuracy.

Table 4. Top-5 Largest Kernels

HTTP URI	Size
www.biopax.org/...#biochemicalReaction5	204
bio2rdf.org/accession:af...[3]	133
bio2rdf.org/accession:af107466	79
rdfweb.org/.../cwm-crawler-output.rdf#_g...[2]	63
bio2rdf.org/accession:af048837	47

Table 5. Top-5 Largest Extensions

HTTP URI	Size
dbpedia.org/data/...[4]/birth_date_and_age	18 939
dbpedia.org/resource/...[2]/flagicon	4 032
dbpedia.org/resource/...[8]/coor_dms	3 766
www.advogato.org/person/...[11]/diary.html	3 606
www.informatik.uni-trier.de/...[6]:Michael.html	3 116

We also checked some inaccurate extensions and observed that the improper extraction of the textual descriptions of URIs mainly causes the inaccuracy. For instance, the size of the extension for http://dbpedia.org/data/Leon_Bott/dateofbirth/birth_date_and_age is 18 939. This is because we used "birth_date_and_age" for extension, which is prevalent in our dataset as there are many data in DBpedia that describe the birth date and age of different people. However, if we could identify "Leon_Bott" is the key description, the extension would be more accurate. On the other hand, if the local name of a URI is specific, e.g., "Tim Berners-Lee" or "Semantic Web", the extension is more accurate. As a future work, we will consider possible ways to identify the discriminative parts of URIs for extension.

7 Conclusion

The main contributions of this paper are summarized as follows.

• We introduced a bootstrapping approach to perform object coreferencing on the Semantic Web. We firstly established a set of semantically equivalent URIs (called a kernel) for an input object URI based on the same-as, IFP, FP and cardinality relations, and then extended the kernel in terms of the textual descriptions of URIs.

• We defined a trustworthiness-based measure for ranking the coreferent URIs in the kernel. The dereferenced documents of URIs were retrieved to classify each same-as, IFP, FP or cardinality relation into different trustworthiness levels. We also proposed a similarity-based method to rank the URIs in the extension. The context of each URI was extracted in terms of

its RDF sentences, and the similarities between different contexts were calculated by the V-Doc algorithm.

• We implemented an online prototype system called ObjectCoref, and evaluated its performance on a large-scale dataset collected by the Falcons search engine in 2008. The experimental results demonstrated that ObjectCoref achieves both good precision and relative recall on object coreferencing with acceptable response time. We also analyzed the popularity of the URI alias phenomenon based on the average sizes of the kernels and extensions of 100 000 sample URIs.

In future work, we will incorporate different measures to rank the coreferent URIs discovered by our approach. In addition, we would like to exploit more possible semantics, rules and thesauri to find coreferent URIs. We also hope to test our approach on various datasets from other Semantic Web search engines.

Acknowledgment The authors would like to thank their colleagues for participating in the experiments. They also appreciate the anonymous reviewers for their helpful comments.

References

[1] Hogan A, Harth A, Decker S. Performing object consolidation on the semantic web data graph. In *Proc. WWW Workshop on I3: Identity, Identifiers, Identification*, Banff, Canada, May 8, 2007.

[2] Jacobs I, Walsh N. Architecture of the World Wide Web, volume one. http://www.w3.org/TR/webarch/, Dec. 15, 2004.

[3] Bleiholder J, Naumann F. Data fusion. *ACM Computing Surveys*, 2008, 41(1): 1-41.

[4] Glaser H, Jaffri A, Millard I C. Managing co-reference on the Semantic Web. In *WWW Workshop on LDOW*, Madrid, Spain, Apr. 20, 2009.

[5] Bizer C, Heath T, Berners-Lee T. Linked data — The story so far. *International Journal on Semantic Web and Information Systems*, 2009, 5(3): 1-22.

[6] Volz R, Kleb J, Mueller W. Towards ontology-based disambiguation of geographical identifiers. In *Proc. WWW Workshop on I3: Identity, Identifiers, Identification*, Banff, Canada, May 8, 2007.

[7] Raimond Y, Sutton C, Sandler M. Automatic interlinking of music datasets on the Semantic Web. In *WWW Workshop on LDOW*, Beijing, China, Apr. 22, 2008.

[8] Hassanzadeh O, Consens M. Linked movie data base. In *WWW Workshop on LDOW*, Madrid, Spain, Apr. 20, 2009.

[9] Tummarello G, Delbru R, Oren E. Sindice.com: Weaving the open linked data. In *Proc. ISWC/ASWC*, Busan, Korea, Nov. 11-15, 2007, pp.552-565.

[10] Cheng G, Qu Y Z. Searching linked objects with Falcons: Approach, implementation and evaluation. *International Journal on Semantic Web and Information Systems*, 2009, 5(3): 49-70.

[11] Bouquet P, Stoermer H, Niederee C, Maña A. Entity name system: The back-bone of an open and scalable web of data. In *Proc. IEEE ICSC*, Washington DC, USA, Aug. 4-7, 2008, pp.554-561.

[12] Hogan A, Polleres A, Umbrich J, Zimmermann A. Some entities are more equal than others: Statistical methods to consolidate linked data. In *ESWC Workshop on NeFoRS*, Heraklion, Greece, May 31, 2010.

[13] Elmagarmid A K, Ipeirotis P G, Verykios V S. Duplicate record detection: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 2007, 19(1): 1-16.

[14] Wang S, Du X Y, Meng X F, Chen H. Database research: Achievements and challenges. *Journal of Computer Science and Technology*, 2006, 21(5): 823-837.

[15] Li Y, Musílek P, Reformat M, Wyard-Scott L. Identification of pleonastic it using the web. *Journal of Artificial Intelligence Research*, 2009, 34(1): 339-389.

[16] Dean M, Schreiber G. OWL web ontology language reference. http://www.w3.org/TR/owl-ref/, Feb. 10, 2004.

[17] Nikolov A, Uren V, Motta E, de Roeck A. Overcoming schema heterogeneity between linked semantic repositories to improve coreference resolution. In *Proc. ASWC*, Shanghai, China, Dec. 6-9, 2009, pp.332-346.

[18] Qu Y Z, Hu W, Cheng G. Constructing virtual documents for ontology matching. In *Proc. WWW*, Edinburgh, UK, May 23-26, 2006, pp.23-31.

[19] Hu W, Qu Y Z, Cheng G. Matching large ontologies: A divide-and-conquer approach. *Data and Knowledge Engineering*, 2008, 67(1): 140-160.

[20] Ferrara A, Lorusso D, Montanelli S. Automatic identity recognition in the Semantic Web. In *Proc. ESWC Workshop on IRSW*, Tenerife, Spain, Jun. 2, 2008.

[21] Volz J, Bizer C, Gaedke M, Kobilarov G. Discovering and maintaining links on the web of data. In *Proc. ISWC*, Chantilly, USA, Oct. 25-29, 2009, pp.650-665.

[22] Halpin P, Hayes P J, McCusker J P, McGuinness D L, Thompson H S. When owl:sameAs isn't the same: An analysis of identity in linked data. In *Proc. ISWC*, Shanghai, China, Nov. 7-11, 2010, pp.305-320.

[23] Ding L, Shinavier J, Shangguan Z N, McGuinness D L. SameAs networks and beyond: Analyzing deployment status and implications of owl:sameAs in linked data. In *Proc. ISWC*, Shanghai, China, Nov. 7-11, 2010, pp.145-160.

[24] Gracia J, d'Aquin M, Mena E. Large scale integration of senses for the Semantic Web. In *Proc. WWW*, Madrid, Spain, Apr. 20-24, 2009, pp.611-620.

[25] Fellegi I P, Sunter A B. A theory for record linkage. *Journal of the American Statistical Society*, 1969, 64(328): 1183-1210.

[26] Cheng T Y, Wang S. A novel approach to clustering merchandise records. *Journal of Computer Science and Technology*, 2007, 22(2): 228-231.

[27] Euzenat J, Shvaiko P. Ontology Matching. Heidelberg: Springer, 2007.

[28] Wang S, Englebienne G, Schlobach S. Learning concept mappings from instance similarity. In *Proc. ISWC*, Karlsruhe, Germany, Oct. 26-30, 2008, pp.339-355.

[29] Klyne G, Carroll J J. Resource description framework (RDF): Concepts and abstract syntax. http://www.w3.org/TR/rdf-concepts/, Feb. 10, 2004.

[30] Urbani J, Kotoulas S, Maassen J, van Harmelen F, Bal H. OWL reasoning with WebPIE: Calculating the closure of 100 billion triples. In *Proc. ESWC*, Heraklion, Greece, May 30-Jun. 3, 2010, pp.213-227.

[31] Hogan A, Pan J Z, Polleres A, Decker S. SAOR: Template rule optimisations for distributed reasoning over 1 billion linked data triples. In: *Proc. ISWC*, Shanghai, China, Nov. 7-11, 2010, pp.337-353.

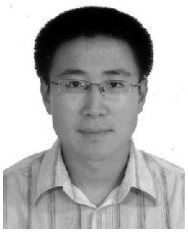
[32] Ghazvinian A, Noy N F, Jonquet C, Shah N, Musen M A. What four million mappings can tell you about two hundred ontologies. In *Proc. ISWC*, Chantilly, USA, Oct. 25-29, 2009, pp.229-242.

[33] Page L, Brin S, Motwani R, Winograd T. The PageRank citation ranking: Bringing order to the web. Technical Report, Stanford University, 1998.

[34] Kleinberg J. Authoritative sources in a hyperlinked environment. In *Proc. SODA*, San Francisco, USA, Jan. 25-27, 1998, pp.668-677.

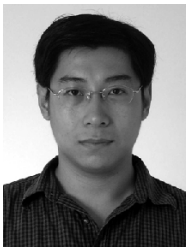
[35] Tummarello G, Morbidoni C, Bachmann-Gmür R, Erling O. RDFSync: Efficient remote synchronization of RDF models. In *Proc. ISWC/ASWC*, Busan, Korea, Nov. 11-15, 2007, pp.537-551.

[36] Stickler P. CBD — Concise bounded description. http://www.w3.org/Submission/CBD/, Jun. 3, 2005.



Wei Hu is a lecturer at the Department of Computer Science and Technology, Nanjing University. He received his B.Sc. degree in computer science and technology in 2005, and his Ph.D. degree in computer software and theory in 2009 both from the Southeast University. His research interests include Semantic Web, ontology engineering and data fusion.

Yu-Zhong Qu is a professor and Ph.D. supervisor at the Department of Computer Science and Technology, Nanjing University. He received his B.Sc. and M.Sc. degrees in mathematics from Fudan University in 1985 and 1988 respectively, and his Ph.D. degree in computer software from Nanjing University in 1995. His research interests include software methodology, Semantic Web and Web science.



Xing-Zhi Sun is a researcher in the Semantic Technology group, IBM China Research Laboratory. He received his B.Sc. degree in electrical engineering from the Shanghai Jiao Tong University in 2000, and his Ph.D. degree in computer science from the University of Queensland, Australia, in 2005. His research interests include ontology data management and data analysis in healthcare.