Data cleansing base on subgraph comparison

Huang Li [1,2*]

[1] *College of Computer Science and Technology, Wuhan University of Science and Technology*

[2] *Hubei Province Key Laboratory of Intelligent Information Processing and Real-time Industrial System*

Received 1 February 2014, www.tsi.lv

Abstract

With the quick development of the semantic web technology, RDF data explosion has become a challenging problem. Since RDF data are always from different resources, which may have overlap with each other, they could have duplicates. These duplicates may cause ambiguity and even error in reasoning. However, attentions are seldom paid to this problem. In this paper, we study the problem and give a solution, named K-radius sub graph comparison (KSC). The proposed method is based on RDF-Hierarchical Graph Model. KSC combines similar and comparison of 'context' to detect duplicate in RDF data. Experiments on publication datasets show that the proposed method is efficient in duplicate detection of RDF data. And KSC is simpler and less time-costs than other methods of graph comparison.

Keywords: RDF data cleansing, K-radius sub graph comparison

1 Introduction

The Web enables persons to link related documents. Similarly, it enables persons to link related data. Linking open data project, one of endeavors to link data, aims to use the Web to connect related data. Nowadays, according to its statistics the data sets consist of over 13.1 billion RDF triples, which are interlinked by around 142 million RDF links in 2010. Comparing with its statistics of April 2008 (2 billion RDF triples, and around 3 million RDF links) [1], the linked data have more than six times. The trends show the linked data will be even larger in the future.

However, due to diverse sources of data, it is quite possible that there exist duplicates in linked data. It may lead to many duplicate search results or wrong statistics on entities of Web etc. Let us take DBLP (http://dblp.uni-trier.de/) which data are generally integrated from IEEE, ACM Portal et al, as an example. When we search 'Jim Smith' in DBLP, DBLP returns 34 papers which DBLP thinks the single author named as 'Jim Smith' wrote. But in fact those papers were written by 5 different 'Jim Smith's. Furthermore, 'Jim Smith' is generally written as 'Smith, J.' in ACM database, but as 'J. Smith' in IEEE database. Those data make persons have to spend lots of time to explore the truth. Thus, it is highly necessary to cleanse linked data.

Linked Data uses URIs and RDF to connect pieces of data, information, and knowledge on the Web. The RDF is based upon the idea of making statements about resources (in particular Web resources) in the form of subject-predicate-object expressions. A collection of RDF statements intrinsically represents a labelled, directed multi-graph. Although some solutions were proposed to cleanse relational database, few research are found on RDF data cleansing. Graph data might be transferred to relational data, but not all graph data contain uniform and enough relationship among entities. This make the performance of traditional approaches is not very well when using those approaches to cleanse graph data.

Thus in this paper, we study the cleansing problem of linked data, and to solve the duplicate problem in linked data. The primary contributions of this article are as follows:

- Propose a simple and intuitive graph model for RDF data, named RDF-Hierarchical Graph Model, which improves Bipartite Statement-Value Graph model [5].
- Propose a simple and efficient method for graph comparison. To avoid the time-consuming comparison of graph, we introduce a new concept of K-radius sub graph. The K-radius sub graph is the 'context' of a node. We propose a K-radius Sub graph Comparison method, denoted KSC, to compare the 'context' between two nodes in RDF-Hierarchical Graph. KSC extends sub graphs of two nodes partiwisely. The sub graphs contain the relationship-information of two nodes, partiwisely. KSC compares the two sub graphs, and calculate the similarity of them. Compared to other methods, KSC is more simple and efficient for RDF graph comparison.
- Propose a solution for RDF data cleansing, based on above two. To the best of our knowledge, we are the first ones to study RDF data cleansing. In this solution, KSC, which is based on RDF-Hierarchical Graph Model, utilizes the attribute of RDF data and the links

[*] Corresponding author: Tel: +86-130-988-82821; fax: +86-027-6889-3420;
E-mail: huangli82@wust.edu.cn

among them, namely ‘context’ of a node to detect duplicate.

The rest of the paper is organized as follows. The RDF-Hierarchical Graph model is defined in section 2. Section 3 describes the proposed method K-radius Subgraph comparison method. In section 4, we report the performance of our approach. Related work is discussed in section 5. Finally, we conclude this paper in section 6.

2 Hierarchical Graph Model for RDF Data

RDF (Resource Description Framework) which is proposed by WWW Consortium[*] is used to describe metadata about information resource. RDF statement is a triple which is consisted of a subject, a predicate and an object. A set of RDF triples are RDF graph. RDF graph is a sort of hypergraph. In this hypergraph, the hyperedges represent the statements, and the hypernodes denote subjects, predicates and objects.

Since hypergraph is hardly analyzed, Hayes et al. proposed bipartite graphs model to represent RDF data [5]. In a bipartite graphs model, both hyperedge and hypernode are represented by a node in the graph, and they are called hyperedge-node and hypernode-node respectively. Edges in bipartite graph model are used to connect hyperedges and their hypernodes. Then the hypergraph is transformed into a bipartite graph. According to RDF, subject and object are different from predicate in types. For duplicates must be of the same type, we distinguish the two parts in a triple by layers. The nodes of same types are in the same layers. Subject and object are set on the same layer, and predicate is on a different layer. We define the layer of predicate is higher. Then we get a new model of RDF graph, called RDF-Hierarchical graph. We cannot find that there are subject of one type and object of another type in one triple. That is because subject and object of different types could not construct a statement of RDF data. Therefore, the model graph is Hierarchical.

We can take a triple as a unit in RDF-Hierarchical graph, for RDF-Hierarchical graph is consisted of triples. Figure 1(a) shows a unit in a sub graph of RDF-Hierarchical graph. Figures 1(b) (c) show different kinds of sub graphs in RDF-Hierarchical graph.

A unit is also a sub graph of RDF-Hierarchical graph. Figure. 1(b) is consisted of two units, and (c) has three units. Shadowed triples in Figure. 1(d) cannot exist.

The operations on RDF-Hierarchical graph are most unit-oriented. We will give some definitions of concepts on RDF-Hierarchical graph. Assume the target node is p, the target unit u, and G is a RDF-Hierarchical graph.

[*] http://www.w3.org/

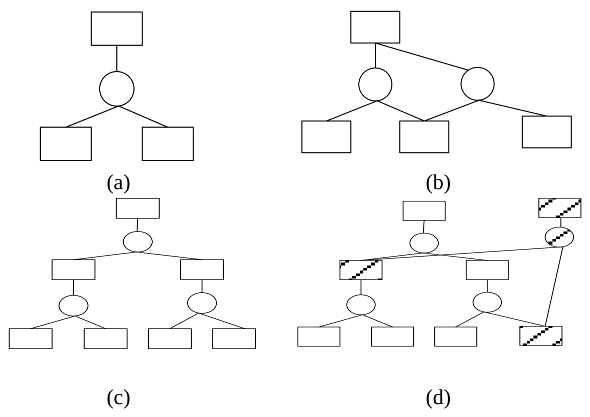

FIGURE 1 Sub graphs of RDF-Hierarchical graph model.

Definition 1. n-distance. The n-distance is used to measure the distance from a unit to a node, denoted as D: $D(u, p)$

$$
= \begin{cases} min\,(\{j|(u_1, u_2, ..., u_{j-1})\}) & \\ \quad p \notin u,\ \exists\ sequence\ (u_1, u_2, ..., u_{j-1}) & \\ where\ u_i \in G,\ u_i \cap u_{i-1} \neq \phi,\ i=1,...,j-1, p \in u_1,\ u \cap u_{j-1} \neq \phi & \\ 1 & p \in u \\ 0 & else \end{cases}
$$

Definition 2. K-radius sub graph. The n-distance between any units in a sub graph to node P is not bigger than k. this sub graph is the k-radius sub graph of P, denoted $SG_k(P)$. If $u_i \subset SG_k(P)$, then $\max(D(u_i, P)) = k$.

In Figure 2, we give a sub graph of RDF-Hierarchical graph. From Figure. 2, we can get $D(u_1, A) = D(u_2, A) = D(u_3, A) = D(u_4, A) = 1$, and $D(u_5, A) = D(u_6, A) = 2$. We also can find all distance from units in Figure 2 to node A is not more than 2. So the sub graph shown in Figure. 2 is a 2-radius sub graph of A.

We also introduce some operations on graph. Assume $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$ are two graph.

Operation 1. Subtraction. This operation, denoted as ‘-’, means to subtract the nodes and edges in another graph. The result is a set calculated as $G_1 - G_2 = \{V_1 - V_2, E_1 - E_2\}$. The operation of subtraction is not suitable for any pairs of graphs. If $V_1 \subseteq V_2$, $V_1 - V_2$ is error. Therefore, V_1 should be larger than V_2.

Operation 2. Intersection. This operation, denoted as ‘$\bigcap$’, means to intersect the nodes and edges in another graph. The result is $G_1 \bigcap G_2 = \{V_1 \bigcap V_2, E_1 \bigcap E_2\}$.

Operation 3. Union. This operation, denoted as '$\bigcup$', means to unite the nodes and edges in another graph. The result is $G_1 \bigcup G_2 = \{V_1 \bigcup V_2, E_1 \bigcup E_2\}$.

3 K-Radius Sub graph Comparison

Generally, in a graph, the nodes, which have smaller distance to a node P, have stronger relationship with P. These nodes restrict P, and help to disambiguate P. We call these constraints one node's 'context'. If two nodes are duplicates, they would have the same or similar 'context's. Thus, the principle of KSC is that a node is disambiguated by the 'context's, which is presented by other near nodes and edges in a graph.

The biggest difference between RDF-Hierarchical graph and a general graph is the operations are unit-oriented in RDF-Hierarchical instead of node-oriented. In RDF-Hierarchical graph, a node is disambiguated by the units around it. The reason for using units instead of nodes is that every node has inherent neighbourhoods in RDF-Hierarchical graph. The nearest nodes of them are statement nodes, which cannot help for disambiguating. According to last section, K-radius sub graph contains most units, which the n-distance from the target node is not more than k. Thus, we use K-radius sub graph to reflect the 'context' around the node.

In the following sections, we will state the method how to get the K-radius sub graph of one node, and the calculation method of similarity between K-radius sub graphs.

3.1 K-RADIUS SUBGRAPH

Here, we will describe the process of finding the K-radius sub graph of a node.

From Figure 2, we can find units are not shown clearly in RDF-Hierarchical graph. For presenting the process of create K-radius sub graph intuitively, we simplify the graph in this subsection. In the simple RDF-Hierarchical graph, each unit in RDF-Hierarchical graph is considered as a node. If two units share the same nodes, there is an edge between the two units in RDF-Hierarchical graph, and the number of nodes two units sharing is the weight of the edge. As the RDF-Hierarchical graph is hierarchy, the simple RDF-Hierarchical graph also has hierarchy. The level of the target node is level 0, and all the levels are positive. The level of a unit equals the highest level of the note in it. Figure 3 is the result simply from Figure. 2.

According to the Definition 1 and 2, the n-distance from units, which contain the target-node is 1. Therefore, these units consist of 1-radius sub graph. If k is bigger than 1, the process of finding K-radius sub graph is an iteration of extending.

However, if we try to extend all the nodes, which are within an n-distance of k to the target node, the result k-radius subgraph will be huge and complex. The goal of finding K-radius sub graph is to give some constraint to the target-node. Thus the units, which have stronger relationship with the target-node, have more worth.

In a simple RDF-Hierarchical graph, the edge expresses the two units have overlap. The weight of the edge represents the degree of the overlap. The larger the weight is the more overlap two units have. More overlap means stronger relationship. The hierarchy of RDF-Hierarchical graph also implies the degree of the relationship. The nodes in the same layer are in the same type in RDF-Hierarchical graph. The units in different layers have weaker relationship. Therefore, each time, we choose the lowest level of units. Then we conclude two properties of KSC as follows:

Assume the target node is P, $k > 1$, the RDF-Hierarchical graph is G.

Property 1. The set S of candidate units is $S = \{v | \max(|v \bigcap u_i|), \text{ and } u_i \in SG_{k-1}(P), v \in G\}$.

Property 2. The final extending set $S' = \{t | \min(level(t)), t \in S\}$, where S is the candidate set getting from rule 1.

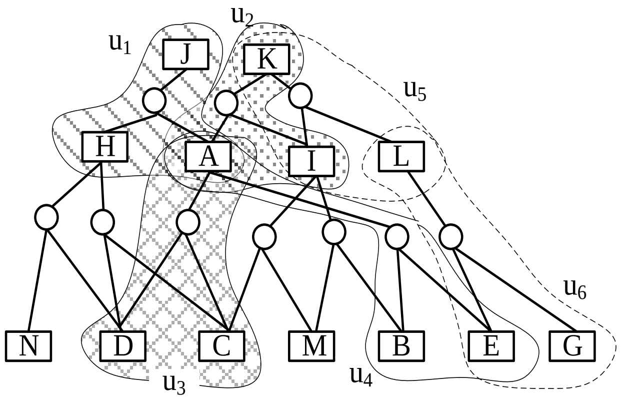

FIGURE 2 Sub graph of RDF-Hierarchical graph.

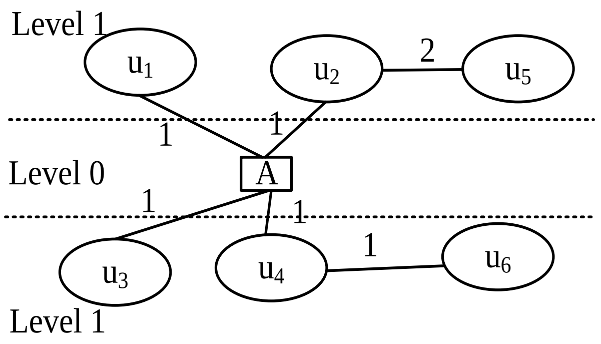

FIGURE 3 Simple RDF-Hierarchical graph.

In each extending, KSC chooses the units having strongest relationship with the units, which are already in the subgraph. Therefore, in Figure 3, we will take u_6 out from the 2-radius sub graph of A.

3.2 SUB GRAPH COMPARISON

In this section, we will find the K-radius subgraph of a node, which not only shows information of a node but also helps disambiguating the node. Thus, duplicates should have similar K-radius sub graphs. According to the above analysis, if the K-radius sub graphs of two similar nodes are similar, the two nodes have a higher probability to be duplicates. Here we will give the method of comparing two K-radius sub graphs.

There are many connections among nodes in different units. These connections contain much information. Therefore, the comparison of sub graphs is node-oriented. In generally, most methods of comparing graphs are usually used in biology, and each edge refers to a regular bond. In that situation, they need only consider the construction of the graph without considering the values of nodes. That is a graph isomorphism problem. Many of them are complex and time-consuming. However, in RDF-Hierarchical graph, the values of nodes are different and very important for duplicate detection. Both construction and value should be taken into comparing of two K-radius sub graphs. More information is supplied from K-radius subgraph. It includes the nodes, the edges, and level of the nodes, the n-distance from nodes to the target node. All this information help subgraph comparison. Thus, a special method for calculating similarity between two K-radius sub graphs is needed.

Let us analyse the contributions of different information implied in K-radius sub graphs. Nodes and edges are more important for comparing. However, not all the nodes in K-radius sub graphs have the same contributions. The nearer two nodes are the stronger relationship they have. So the nodes have smaller n-distances from the target node are more important for identifying the target node. Therefore, the n-distance should be taken into calculation of similarity. The definition of n-distance orients units. The three value nodes in a unit are equal, so the nodes in a unit have the same n-distance as the unit has.

According to the above analysis, we give the calculation of similarity between two K-radius subgraph, which is shown in formula (1).

Assume two nodes are P and Q. The K-radius sub graphs are $SG_k(P)$, $SG_k(Q)$, and $SG_0(P)=SG_0(Q)=0$.

$$SimEnv(SG_k(P),SG_k(Q)) = \frac{1}{k}\sum_{i=1}^{k}\left(SimG((SG_i(P)-SG_{i-1}(P)),(SG_i(Q)-SG_{i-1}(Q)))\right)^{i} . \quad (1)$$

In formula (1), $(SG_i(P)-SG_{i-1}(P))$ is a set, which contains the nodes with n-distances is i and the edges among them. Because SG_{k+1} contains SG_k, the subtract operation is valid on them. The intersection is to find the common nodes and edges. Along with i increasing, the nodes farther from the target-node have smaller contributions, so the Similarity increase slowly.

Since there are some ambiguity entities in the graph, it is difficult to judge that whether two other nodes in two sub graphs are the same one or not. How do we judge the two sub graphs share the node P? In [12], they proposed a method, called 'Greedy Matching' (GM), to match the nodes in two graphs. This method can avoid ambiguity problem effectively. However, that method only compares two nodes, and our comparison must consider edges as well. Therefore, we improve the GM method.

Assume two graphs $G_1(V_1,E_1)$, $G_2(V_2,E_2)$, and a threshold is ρ.

For any $v_{1i}\in V_1$, if there is a node v_{2j} in V_2, and $SimString(v_{1i},v_{2j})>\rho$, the pair (v_{1i},v_{2j}) is the candidate pair from V_1. All these pairs are cluster in set S_1. As the same, we can get the candidate set S_2 of pairs from V_2. Then according to [12], we can get the bound of $Sim(V_1,V_2)$ in formula (2):

$$USimV(V_1,V_2)=\frac{\sum_{S_1\cup S_2} SimString(v_{1i},v_{2j})}{|V_1|+|V_2|-|S_1\cup S_2|},$$
$$LSimV(V_1,V_2)=\frac{\sum_{S_1\cap S_2} SimString(v_{1i},v_{2j})}{|V_1|+|V_2|-|S_1\cap S_2|}. \quad (2)$$

If two edges are similar, they could have similar nodes. Therefore, if one edge in E_1 has the similar nodes with an edge in E_2, the two edges are similar. The similarity between two edges depends on the similarities of the nodes. For any edge $(v_{1i},v_{1j})\in E_1$, we find all candidates sets of v_{1i} and v_{1j} respectively, denoted as S_i and S_j. Then put all the edges from S_i r S_j, which belongs to E_2 into a set E_{ij}. Similar to the notes, we can get a bound of $Sim(E_1,E_2)$ shown in formula (3):

$$USimE(E_1,E_2)=\frac{\sum_{(v_i,v_j)\in E_1}\frac{\sum_{(v_i',v_j')\in UE_{ij}}(SimString(v_i,v_i')+SimString(v_j,v_j'))}{|US_i \cap US_j|*2}}{|E_1|},$$

$$LSimE(E_1,E_2)=\frac{\sum_{(v_i,v_j)\in E_1}\frac{\sum_{(v_i',v_j')\in LE_{ij}}(SimString(v_i,v_i')+SimString(v_j,v_j'))}{|LS_i \cap LS_j|*2}}{|E_1|}, \quad (3)$$

where $US_i=\{v_i'|(v_i,v_i')\in S_1\cup S_2\}$, $LS_i=\{v_i'|(v_i,v_i')\in S_1\cap S_2\}$, $UE_{ij}=US_i \cap US_i$, and $LE_{ij}=LS_i \cap LS_i$.

Therefore, the bound of similarity between two sub graphs SG_1 and G_2 can be calculated in formula (4):

$$USimG(G_1,G_2)=USimV(V_1,V_2)+USimE(E_1,E_2),$$
$$LSimG(G_1,G_2)=LSimV(V_1,V_2)+LSimE(E_1,E_2). \quad (4)$$

We set the average value to the similarity between two sub graphs. The calculation is show in formula (5):

$$SimG(G_1,G_2)$$
$$=(USimG(G_1,G_2)+LSimG(G_1,G_2))/2. \quad (5)$$

Combining the similarity of the nodes, the final similarity of two nodes is calculated in formula (6):

$$S_{total}(P,Q)$$
$$=\alpha SimString(P,Q)+\beta SimEnv(SG_k(P),SG_k(Q)), \quad (6)$$

where α and β are factors. $SimString(P,Q)$ is the similarity of strings between P and Q. If the similarity is bigger than a threshold, P and Q are duplicates.

4 Experiment

We do some experiments and show results to examine the efficiency and accuracy of the proposed approach, in this section. All experiments are performed on an IBM eServer with a 1.25GHz Power4 processor and 4GB of memory, running Suse Linux Enterprise 10.0. All approaches are implemented and tested in Java.

4.1 DATASET

We experimentally study the proposed approach on DBLP, which is a real dataset on publication. We store all data in DBLP in RDF triples. DBLP now lists more than 1.2 million publications. Since there are huge amount of triples, it is hard to measure the results by hands. Therefore, we picked up 3 groups of triples from DBLP. Each contains 5,000 triples. We use a semiautomatic method to calculate the numbers of duplications. Table 1 shows more details of the 3 groups.

TABLE 1 Detail of groups

	Number of triples	Number of entities	Number of duplicates
Group 1	5000	13057	73
Group 2	5000	2978	29
Group 3	5000	7843	57

In a RDF triple, each item is considered as an entity. In table 1, three groups contain different numbers of entities, although they contain the same number of RDF triples. This is induced by relationships among RDF triples. If all triples have fewer relationships with each other, there are fewer triples sharing entities. As we can see, the maximum number of entities in each group is 15,000 (3*5000=15000). In this situation, all the triples have no relationships with each other.

The three groups we pick up for testing represent 3 typical situations. In group 1, the number of entities is closed to the maximum number. The triples have fewer relationships with each other in this situation. In group 2, the number of entities is smaller than the number of triples. This indicates most triples share entities in this situation. The number of entities in group 3, is between the above two groups. The ratios of the number of triples to entities are shown in Figure 4.

4.2 EXPERIMENTAL ANALYSIS

The process of the proposed approach can be divided into three steps. The first step is to transfer the RDF triples into RDF-Hierarchical graph. Second, calculate the similarities of the data pairwisely. The similarity calculation has been studied in [6]. All the similarity calculation pairs are in the same layer. Here, we use the LFDW [6] to calculate similarity. If the similarity of the pair is higher than the threshold, the pair is a candidate. The candidate pairs need to be verified. The last step is to compare the ‘context’s between the data. The ‘context’ of a node is measured by the K-radius subgraph of the node. Then we combine the two similarities together as a total similarity between the two nodes. If the total similarity is bigger than the threshold, the two data are duplicates.

We first pick up two pairs of nodes in RDF-Hierarchical graph to test the similarity of K-radius sub graphs. One pair has smaller similarity, and the other pair has bigger similarity. The details of the test data are as follows.

The test pairs: pairs 1 ("*Thomas Cormen*", "*Charles H. Leiserson*"), pairs 2 ("*John R. Smith*", "*Smith R. John*").

$$SimString(pair1) = 0.098; SimString(pair2) = 0.975$$

The result shows both similarities increase along with k in Figure. 5. This result can be easily proved by formula (1). The similarity is a summation. We also find some interesting phenomenon in the result. Similarity of K-radius sub graphs in pair 1 increases slowly at the beginning, but quickly when k is bigger than 4. And the similarity of pair 2 almost stop increasing when k is bigger than 5. After comparing strings of nodes, pair1 has less possibility to be duplicates than pair 2. When k is small, the two nodes in pair 2 share little nodes. Most units in sub graphs need to be extended. Along with increasing of K, more and more units are extended in sub graphs. The probability of sub graphs sharing nodes is higher. Meanwhile, most units of K-radius sub graphs in pair 2 stop extending. Small nodes are extended in the sub graphs. Thus, the above phenomenon happened.

From the above test, the value of k should be smaller than 4 for distinguishing two kind pairs, for the different is smaller when k is bigger than 4. Then we will test the proposed approach in different Ks, and K is smaller than 4. The results are shown in Figure 6 and Figure 7. The threshold is the same in different k.

Figure 6 shows the trend of f-measure (denoted F_1) when k increases in different groups. The trends of the f-measure are the same in three groups. The f-measures increase quickly when k is small, and slow down when k is bigger. The critical values are reached at different k in three groups. In group 1, when k is 1, the change of f-measure is small. The triples in group 1 have fewer relationships with each other. There is few information of 'context' around each entity. In this group, most duplicates are detected through similarity comparison between entity pairs. In group 3, triples share more entities with each other. Then more information of 'context' joins to help detecting. In group 3, when k is 2, f-measure reaches critical values.

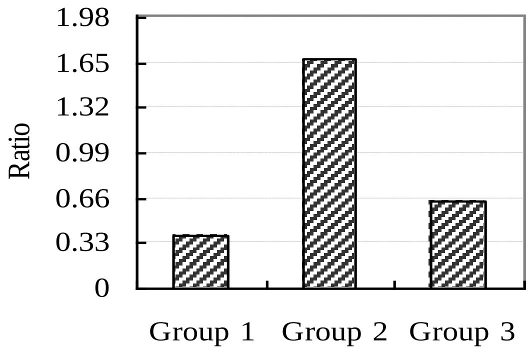

FIGURE 4 Ratios of number of 3-tuples to entities.

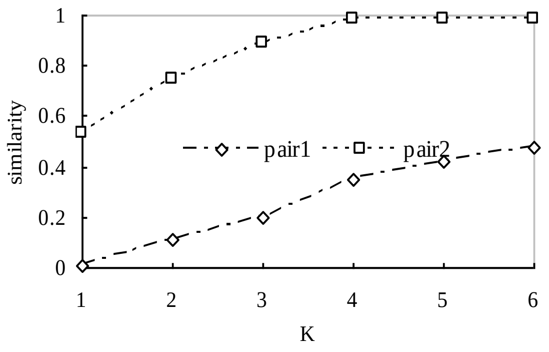

FIGURE 5 Similarities of K-radius sub graphs in two pairs.

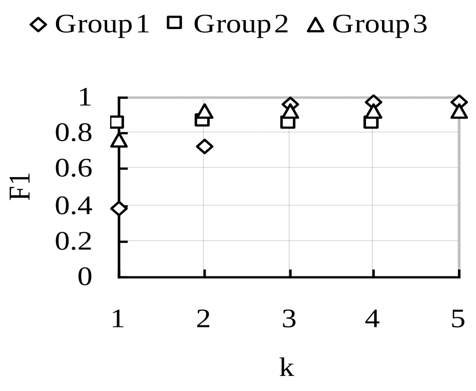

FIGURE 6 Results of detection duplicate with different k

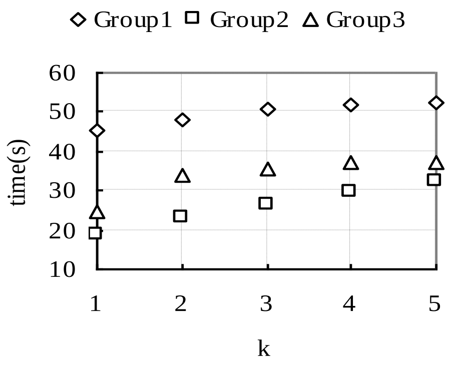

FIGURE 7 Time needed in different k

We compare the f-measures in the three groups. Group 2 is the highest, and group 1 is the lowest. This shows that 'context' is an important factor in duplicate detection.

Figure 7 shows the running times in different k. In all groups, the running time increases when k increases. The trends are similar in the three groups.

We take group 2 as an example to describe the trend in detail. The time increases slowly when k is small, quickly when k is bigger than 3, and slowly again when k is bigger than 4. When k is small, the scales of the k-radius sub graphs are small. The time cost in comparison is small. When k increases, more and more units extended to the K-radius subgraph. More time are needed for comparing. When k is still bigger, most units in K-radius sub graphs stop extending. The scales of the K-radius sub graphs increase slowly, and then the time needed to compare is also increasing slowly. According to Figure. 6and 7, although when *k* is 4, the f-measure is highest, more time is needed. The f-measure is litter higher when *k* is 4 than *k* is 3, but much more time is needed when k is 4. Thus, the optional value of *K* for group 2 is 3.

Through similar analysis on group 1 and group 3, we can get the optional values of *k* are 1 and 2, respectively. And when the ratio of triples to entities is 1/3, there are no triples sharing entities with each other. In this situation, *k* is equal to 0. Then Figure. 8 gives the relationship between the optional value of *k* and the ratio of triples to entities.

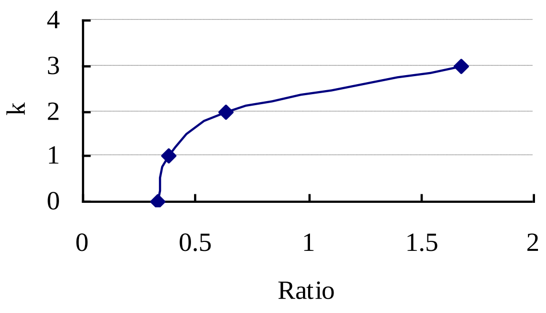

FIGURE 8 Relationship between k and ratio of triples to entities.

From Figure 8, we can induce a formula to calculate *k* in a dataset. We can calculate the optional value of k using the formula (7):

$$k = \begin{cases} 0 & ratio = 1/3 \\ \lceil (\ln(ratio) + 1.1) * 1.8 \rceil & others \end{cases}, \qquad (7)$$

where $ratio = \dfrac{Number_{triple}}{Number_{entity}}$.

Since RDF graph could be transfer to both Entity-relationship model and simple graph model, many methods for Entity-relationship model and graph model are also available for RDF graph. On Entity-relationship model, many methods are proposed for duplicate detecting [2, 3, 4, 9, 10] on publication data. A typical one is proposed by Han [3], which also consider the relationship among data. For graph model, Kalashnikov proposed a domain impendent method in [7], which combines the similarity and connection for identify entities. For few methods have been proposed for RDF data cleansing, we compare the proposed method KSC with methods which are used for Entity-relationship model and graph model instead. The result is shown in Figure 9. In Figure 9, for simplicity, we use '*R*' to refer to the method for entity-relationship model, and use 'C' to refer to the method proposed by Kalashnikov in [7]. '*KSC*' is the proposed approach. Here the connect-path is the shortest and least resistance path. If two units share more nodes, the connection between them is stronger. The longer the path is, the bigger resistance two nodes have, the less connection they have.

For KSC considers 'context's of data, the more 'context's the data have, the higher f-measure KSC gets. Therefore, in Figure 9(a-c), KSC on Group2 gets the highest recall, precision and f-measure. Furthermore, KSC has more advantage than other methods in Group2. The more relationships the entities have, the more effective KSC is. For there may be missing some links when transfer RDF graph to Entity-relationship model, the Entity-relationship model have lowest f-measure. The 'context' among data can better improve the precision than 'connections' do, as shown in the result.

Figure 9(d) also shows the running time of the three methods. The proposed method 'KSC' cost least time. For the number of entities is smallest in Group2, the least time is cost in Group2, although many 'context's need more time to deal with. For many extra works have to do when consider the relationship between data in Entity-relationship model, 'R' needs most time.

To present the results intuitively, we introduce another measure, called efficiency, denoted *EFF*. The calculation of EFF is shown in formula (8):

$$EFF = F_1 * Num(duplicate)/time. \qquad (8)$$

EFF refers to the number of duplicates, which are correctly detected in a unit time. In Figure 9(e), we also show the *EFF*s of the three methods. By comparing EFFs, the proposed method 'KSC' has highest efficiency of duplicate detection in any situation. When there are more relationships among entities, the advantage of KSC is more outstanding. From Figure 9, we can find out that, the f-measure of KSC is highest, and it costs least running time in all groups. KSC is effective for any kind of dataset whether the data in it have more or less relationships.

5 Related Works

Our research solves the duplicate detection of RDF data. This research is related to two main studies: RDF data modelling and duplicate detection.

RDF is the W3C standard model for describing metadata. The RDF data also has the problem of duplicates. RDF data do not only represent the value of the data, but also represent the relationships among the data. In [8], Klyne et al proposed a directed labelled graphs to represent the RDF data. A triple of RDF statement is a label edges. This model is easy to implement and represents the relationships among data clearly. However, if the relationships are complex, much information would be lost. The RDF graph is different from a common graph. It is a hypergraph, because there may have more than one edge between two nodes. Therefore, in [11], Morales proposed a direct hypergraph model. In this model, each RDF statement is a hyperedge in the hypergraph. This model can represent the complex relationships among data. However, it cannot deal with the scale of RDF data, and hard to more process on it. And in [5], Hayes proposed a bipartite graph model. This model transfers the hypergraph to a common bipartite graph. It is easy to manage and operate.

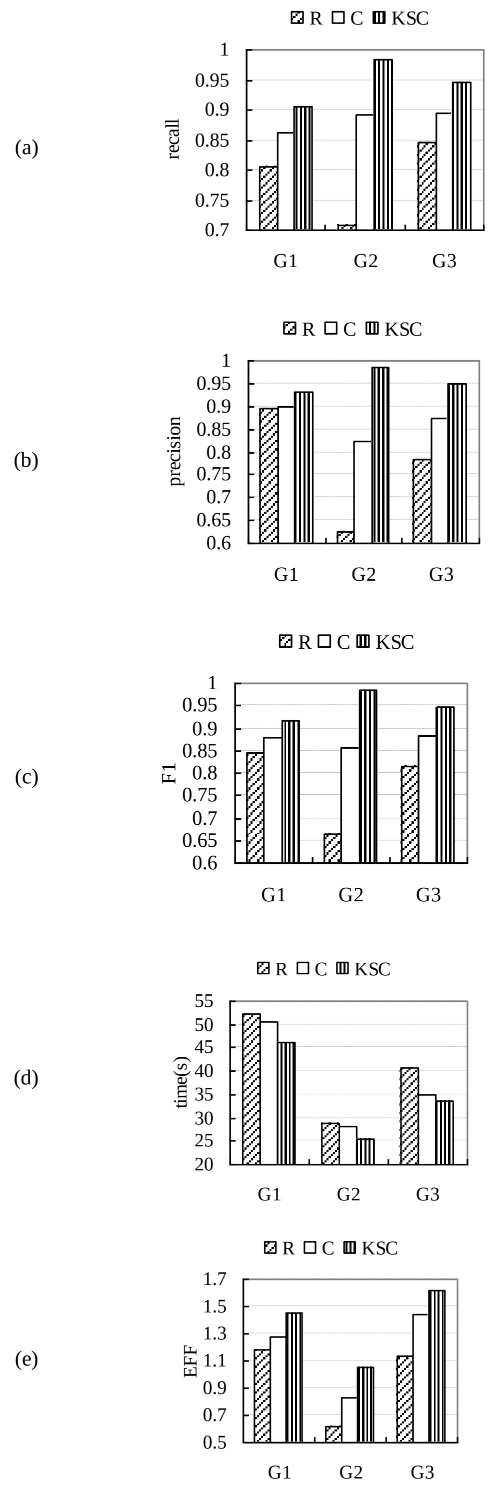

FIGURE 9 Result of comparison between R, C and KSC. Here, 'G1','G2','G3' refer to Group1, Group2, and Group3, respectively.

Our model is improved from this model. All the models introduced are used for semantic retrieval, like similar query and related query. They do not focus on the duplicate detection.

Duplicate is the main inducement of data dirty. Duplicate detection belongs to data cleansing. It is an important study in data mining. The basic method to detect duplicates is to compare entities. The main issue of compare entities is field matching [9], which could be achieved by recursive field matching algorithm [10], Smith-Waterman algorithm and R-S-W algorithm [2] etc. The above methods all consider the records themselves and omit the relationships among them. Recently, researchers shifted their attention to the associations among entities. Han et al. proposed an unsupervised learning approach using K-way spectral clustering that disambiguates authors in citations. It utilizes three kinds of citation attributes: co-author names, paper titles, and publication venue titles [3]. A general object distinction methodology is introduced in [13]. The approach combined two complementary measures for relational similarity: set resemblance of neighbour tuples and random walk probability, and then analysed subtle linkages effectively. Han et al. investigated two supervised learning approaches to disambiguate authors in the citations [4]. In [7], Kalashnikov proposed a domain-independent method. This method analysed not only object features but also inter-object relationships to improve the disambiguation quality. They used the shortest path algorithm to find the connect path connects two entities. And the path is measured by the association strength between the two entities. However, it is difficult to set the association strength between two entities correctly.

6 Conclusion and Future Works

Nowadays, links among data are increasing explosively. Due to variety sources of data, RDF data have duplicates. Duplicate may cause the inference and inquire error. However, studies are seldom made on this problem. Thus in this paper, we proposed an approach to detect the duplicates in RDF data.

The proposed approach combines both similarity and 'context' among RDF data to detect the duplicates. And considering the complexity of the associations among RDF data, we proposed a model for RDF, called RDF-Hierarchical graph, which is improved from Bipartite Statement-Value Graphs [4]. On the model, we give a K-radius subgraph comparison method to detect the 'context' of the two similar nodes, to avoid the complex and high cost of graph comparison. This method explores the K-radius sub graphs of the two nodes, which reflect the 'context's. By comparing two K-radius sub graphs (KSC), we get the similarity of 'context's between the nodes. Finally, we combine the similarity and the 'context' between the two nodes to decide whether they are duplicates or not.

We implement the proposed method on publication datasets, and compare the method with the methods on entity-relationship model and graph model, for seldom methods are proposed on RDF data. The results show that the proposed method improves accuracy and efficiency in detecting duplicates obviously. And the KSC is convinced to be a more simple and quick method.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grant Nos. 61100055), The major projects of the National Social Science Foundation of China(Grant Nos.11&ZD189), Natural Science Foundation of Hubei Province (Grant No.500104), the Hubei Province Key Laboratory of Intelligent information processing and real-time industrial systems (Wuhan University of Science and Technology, Grant Nos. znss2013B013).

References

[1] Zander S, Schandl B 2012 *Journal Semantic Web* **3**(2) 131-155
[2] Franek L, Jiang X, He C 2014 Weighted mean of a pair of clusterings *Pattern Analysis and Applications* **17** 153-166
[3] Williams K, Giles C 2013 Near duplicate detection in an academic digital library *Proceedings of the 2013 ACM symposium on Document engineering-2013* (*Florence, Italy, September 10-13 2013*) 334-43 http://clgiles.ist.psu.edu/pubs/DOCENG2013-near-duplicate-detection.pdf
[4] Yuan D, Mitra P, Lee Giles C 2013 Mining and indexing graphs for supergraph search *Proceedings of the VLDB Endowment*, 2013 **6**(10) 829-840 http://www.vldb.org/pvldb/vol6.html
[5] Gutierrez C, Hurtado C, Mendelzon A 2011 *Journal of Computer and System Sciences* **77**(3) 520-41
[6] Huang L, Jin H, Yuan P, Chu F 2008 Duplicate Records Cleansing with Length Filtering and Dynamic Weighting *The 4th International Conference on Semantics, Knowledge and Grid* 2008, Dec. 3-5 2008 95-102 http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4725901&isnumber=4725879
[7] Ioannou E, Nejdl W, Niederée C 2010 On-the-fly entity-aware query processing in the presence of linkage *Proceedings of the VLDB* **3**(1) 429-38
[8] Klyne G, Carroll J 2004 Resource description framework (RDF): Concepts and Abstract Syntax *W3C Recommendation. World Wide Web* http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/ February 10 2004
[9] Minton S N, Nanjo C, Knoblock C A, Michalowski M, Michelson M 2005 A Heterogeneous Field Matching Method for Record Linkage *Proceedings of the Fifth IEEE International Conference on Data Mining (Houston Texas USA 27-30 November 2005)* 314-21
[10] Shvaiko P 2013 IEEE *Transactions on Knowledge and Data Engineering* **25**(1) 158-76
[11] Morales A, Serodio M 2007 A Directed Hypergraph Model for RDF *Proceedings of Knowledge Web PhD Symposium* 2007
[12] Korn F, Saha B, Srivastava D, Ying S 2013 On Repairing Structural Problems In Semi-structured Data *Proceedings VLDB* **6**(9) 601-12
[13] Fan X, Wang J, Pu X 2011 *Journal of Data and Information Quality* **2**(2) 1242-6

Author

Huang Li, born 1982, Wuhan, China

Current position, grades: lecturer of computer science
University studies: PhD in computer science
Scientific interests: Data management, Semantic web and knowledge.
Publications : 10
Experience: PhD in computer science. Currently, she is a lecturer of computer science of the School of Computer Science and Technology, WUST, Wuhan, China. Her research interests include data management, semantic web and knowledge