Methods of Entity Resolution in Dataspaces

Jia Yuelin[a], Lu Wei*[b], Su Chang[c]
[a] Cangzhou People's Hospital, Cangzhou 061000, China;
[b] Marine Design and Research Institute of China, Shanghai 200000, China;
[c] Harbin Engineering University, Harbin 150001, China;
Corresponding author: Lu Wei (13810156919@139.com)

ABSTRACT

Dataspace is a new way of data integration. Entity resolution identifies two records that point to the same entity in the real world. In this paper, a record graph is constructed by using the records in the data set. The redundant comparisons are removed by pruning the record graph, and the records is divided into blocks according to the pruned graph. The subsequent entity resolution work is only carried out in blocks. When the entity is parsed in the block, the method of attribute mapping and expression representing attribute value is used to further divide the data to ensure the accuracy of parsing. Methods experiments were carried out on real data sets.

Keywords: Entity Resolution, Dataspace, Record Blocking, Property Mapping, Information Merging

1. INTRODUCTION

Entity resolution refers to the process of identifying different descriptions of the same entity, aimed at guaranteeing data quality. It is the key technology of data cleansing, data integration and data mining [1]. Most of the traditional entity resolution work depends on the pattern or semantic mapping between data. Dataspace is a new mode of data integration without strict data pattern and semantic mapping. In accordance with the subject demand to gradually incorporate data and build relationship, it is a heterogeneous data set characterized by its data coming from multiple data sources[2]. However, when the entity is parsed in dataspace, semantic mapping, the powerful tool of entity resolution, will be lost. To solve the problem, in this paper, the data record tagging is divided into blocks regardless of the record semantics. After the block is divided, the records in blocks will be semantically mapped. An expression will be used to represent record attribute value so as to compare and merge the records.

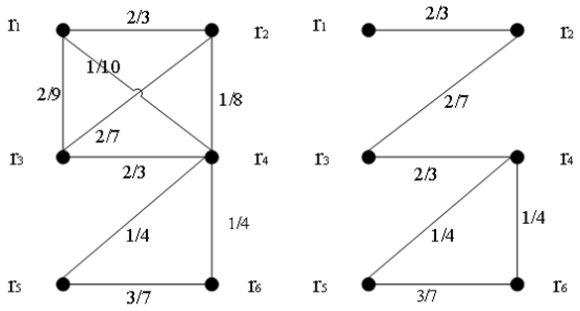
2. BLOCK METHOD BASED ON THE RECORD GRAPH

Entity resolution is to compare records. The matching probability of record pairs in different fields is relatively small, so it is a waste to compare in pairs. To this end, people have proposed the partitioning technology [3], which uses a less expensive calculation method to predict the data, that is, data records that may belong to the same entity are placed in one block, and the records are only compared within the block.

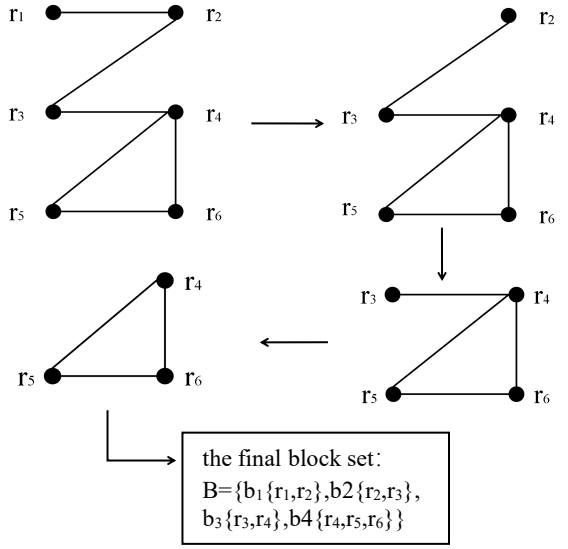
2.1 Construction of the record graph

In this paper, a label method is used to represent data records regarded as an attribute value set. Based upon a common sense hypothesis [4], if two records point to the same entity, they definitely contain some of the same attribute values. Taking into consideration the relationship between data records can improve accuracy [5]. A record graph model is used to represent record nodes in dataspace and their relationships. Calculate the similarity between two records and draw an edge between them and the edge weight is the similarity value. Due to its simple representation, what the tag-style block method only requires is to get the attribute value of the record, without relying on the fixed data pattern and the semantics of the strong mapping. Therefore, in the face of the heterogeneous data set of dataspace, this method is pretty adaptable.

There are recording set R={r1{FullName: Tom Lloyd Malik; Job: producer,Actor; Address: L.A.}, r2{Name: Tom Malik; Producer; birthPlace: L.A.},r3{Label: Mike Styles; Profession: producer; Place_of_birth: L.A.; Place_of_birth: 1964}, r4{Mike Harry Styles; birthPlace: L.A.; Gender: male}, r5{FullName: Harry Green; Address: LOS; Sex: male; Profession: Writher}, r6{Label: Harry Green; Gender: male; birthYear: 1980; married}}.An overview of record graph block method based on the record set R is shown in Figure 1. (To simplify the example figure, the record set doesn't temporarily mark their relationship.)



a. Construct the record graph and prune.



b. Block according to the pruned graph.

Figure 1. Label-based block method process.

2.1.1 To calculate the similarity

Tag conversion function tag() is able to convert one record into a tag set (namely, tag:$r_i \rightarrow T(r_i)$). The label similarity of the two records can be got through calculating the ratio of the intersection to the size of union of the two sets.

Definition 1. Label similarity: convert the record into a tag set through tag conversion function tag() and calculate the label similarity of the two records, which is recorded as $sim_{tag}(r_i, r_j)$:

$$sim_{tag}(r_i, r_j) = \frac{|T(r_i) \cap T(r_j)|}{|T(r_i) \cup T(r_j)|} \quad (1)$$

Where T(r_i) is the normalized tag set converted from the record r_i by the label conversion function.

Definition 2. Relationship similarity: integrate the comprehensive similarity of two records on all the relationships they have, denoted as $sim_{rel}(r_i,r_j)$:

$$sim_{rel}(r_i, r_j) = \frac{\sum_{rel \in REL} \frac{|Nbr(r_i) \cap Nbr(r_j)|}{|Nbr(r_i) \cup Nbr(r_j)|}}{|REL|} \quad (2)$$

Where Nbr(r_i) refers to the record set that has a connection with the record r_i in the *rel* relationship. REL represents all the records relationship sets on the records r_1 and r_2, including partnerships, teacher-student relationships and teaching relationships.

Integrate the above two to get the comprehensive similarity sim(*ri,rj*):

$$sim(r_i, r_j) = \alpha \cdot sim_{tag}(r_i, r_j) + (1-\alpha) \cdot sim_{rel}(r_i, r_j) \quad (3)$$

2.1.2 To construct weighted graph model

Construct the record graph via the records and the record similarity in dataspace.

Definition 3. Record graph: given a dataset R of a dataspace, construct an undirected graph G=(R, E), called a record graph. Where R is a set of nodes representing records in the dataspace; while E is the edge set, and there is an edge between the two records representing the similarity of the record pair.

After constructing the record graph, the matching value of the record pair with a smaller edge weight in the graph is relatively low. By processing the edges of the record graph, unnecessary comparisons between record pairs are reduced.

2.2 Pruning of the graph

The edge with the smaller weight is deleted according to a certain rule, that is, the graph is pruned to reduce redundant matches.

For the convenience of the following description, the definition of the point region is given.

Definition 4. Point region: the appearance form of a record r in the record graph is considered as a node. The area that is formed by the record node itself, the neighbor records with its edges and the edges connecting them is regarded as the point region. The point region is a subgraph of the record graph G, which is denoted as $G_r=\{\{r\} \cup R_r, E_r\}$. Where Rr is a set of neighbor nodes with edges connected to r, and E_r is a set of edges connecting r and neighbor nodes.

The pruning process has two main components: pruning center and pruning rule.

Pruning center can be divided into two kinds. One is the Edge-centralization, that is, select the best pair to be compared by traversing the edge set of the graph to filter out edges that do not satisfy the pruning rule; the other is Node-centralization. Traversing all the nodes in the graph, aiming to find its optimal pair set to be compared for a record node in its point area—that is, the number of records with the largest edge weight associated with this record.

In the light of function, the pruning rule is divided into weight value and cardinality threshold. The former specifies the minimum weight of the reserved edge, deleting all edges below this weight; the latter offers the maximum number of edges of the reserved edges in the graph, leaving the edges with edge weight of top-k. Wherein the cardinality threshold defines the number of pairs to be compared, and is suitable for applications with restrictions on time resources. Based on the matching probability of the record pair itself, the weight value determines whether to prune the edges connecting the record pair. It is applicable to the applications that value effectiveness. According to the scope of action, the pruning rule can be divided into global threshold and local threshold. The global threshold applies to the entire graph, that is, all the edges in the graph; while the local threshold applies to a subset of the graph, that is, the point region of a node.

Combine the above two, four pruning schemes are put forward:

(1) Edge-centralization cardinal pruning: the global cardinality threshold k specifies the total number of edges to be retained in the record graph, namely k edges with the maximum weight. According to the weight, the edge sets can be sorted in descending order to effectively delete the edges with low weights.

(2) Node-centralization cardinal pruning: for each node r_i, retain the edges connecting its edge with weight of top-k as well as the k records that are most similar to it. For the record node r_i, after getting its point region $G_{ri}=\{\{r_i\} \cup R_{ri}, E_{ri}\}$

and the cardinality threshold of the current record node k_{ri}, traverse the edges in the subgraph, reserve the edges with edge weight top-k as well as delete other edges connecting r_i. In general, the cardinality threshold for each node should depend on the edge set size of its point region (eg $k_{ri}=0.1\times|E_{ri}|$).

(3) Edge-centralization threshold pruning: use the weight threshold to prune in the global scope, select the minimum edge weight w_{min}, traverse all the edges in the graph, and delete the edges whose weight are lower than w_{min}. It traverses all the edges in the graph and deletes the edges whose weights are below the preset threshold w_{min}, leaving the remaining edges in the graph and outputting them. Under normal circumstances, the weight between matching records is greater than that between the unmatched records. Therefore, selecting the w_{min} target is to determine the balance between the two.

(4) Node-centralization threshold pruning: for the selection of the pruning range, if a global threshold is adopted, a uniform threshold is used for all nodes of the record graph, and the pruning process is the same as the weight pruning scheme of the Edge-centralization; if a local threshold is adopted, then a specific threshold can be selected for the special node according to the requirements of clients. In essence, it applies the weight pruning of the Edge-centralization to the point region of node r_i. The main difference from the threshold pruning scheme of Edge-centralization is that it can use different thresholds for each node. Firstly, it obtains the point region of r_i $G_{ri}=\{\{r_i\}\cup R_{ri}, E_{ri}\}$, and then specifies the minimum edge weight of the subgraph pruning based on the input local threshold criterion. Then, it iterates through the edges in *Eri* and deletes the edges whose weights are less than the set threshold.

2.3 Blocking of the record graph

The pruned graph is G={R,E}, in which R is the record set and E is the edge set. Take a record r_i randomly, create a block b_i and put r_i into bi. If the node r_j in the r_j point region are connected to all the nodes in b_i with edges, place r_j in b_i and delete all the connected edges with nodes in r_j and b_i. Repeat this operation until all the neighbor nodes of r_j are traversed. At this point, if the node in b_i becomes isolated without edges connecting it in the graph, then this node can be deleted. Repeat until the graph G is empty. The blocking work is completed, and the block set $B=\{b_1, b_2,...,b_{|B|}\}$ is obtained.

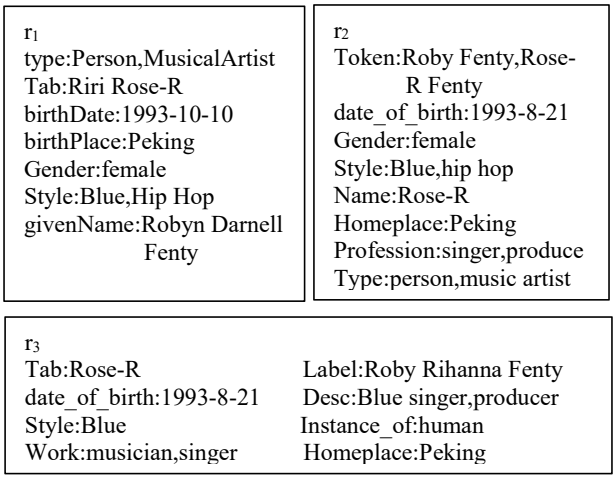
3. RECORD COMPARISON AND CONNECTION BASED ON PROPERTY MAPPING

After blocking, the record information in blocks contains enormous duplicate attribute values, conducive to mapping the attributes. By observing the characteristics of global data and semantic mapping, it could be found that there are some differences in the recorded information of one or more data. Such records should be removed to improve accuracy. By integrating the matching record information, users can take what they need, which reduces the time to process record information.

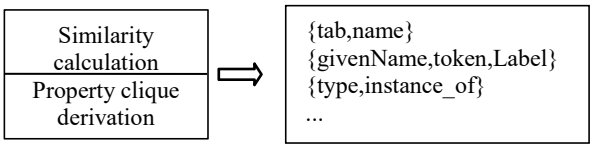
3.1 Mapping of heterogeneous attributes

A commonly way in entity resolution is to use a uniform attribute to calculate the similarity of attribute values. But in the multi-source heterogeneous environment of dataspace, there is no exact attribute mapping, so the attribute value can be used in turn to match the attributes.

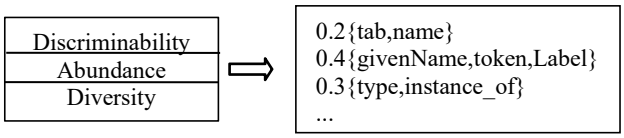
Definition 5. Instantiation: an entity record consists of a set of attributes and values. The value of an attribute may be misspelled or null. Null values cannot be compared. If an entity is described by an attribute whose value is not empty, the attribute is called to instantiate the entity. The entire attribute semantic mapping process is shown in Figure 2.



a. Sample data.



b. Global attribute mapping.



c. Goodness calculation of attribute mapping cluster.

Figure 2. Heterogeneous Attribute Mapping.

3.1.1 Global attribute mapping

For two attributes from different entities, calculate two similarity values:

(1) The attribute name similarity, denoted as S_L, can be obtained by comparing the two normalized property names. It is possible to decide whether or not to include this part in the calculation on the basis of the size of the attribute name similarity in the dataset.

(2) The attribute value similarity, recorded as SV, can be got through comparing all values of the two attributes and retaining the highest similarity score.

In this way, the attribute matching pair can be acquired. Next, from the attribute matching pair set, the attribute matching set is obtained through calculation. The collection of the attribute matching set is called the attribute matching cluster.

The attributes in the attribute matching set match each other exactly. The method rejects an property mapping set containing a number of loosely related attributes and follows the widely-used no-repetition hypothesis[6]. Limit each attribute under a namespace (an entity, or a semantically restricted data source) at most can match an attribute in another namespace. Set a global 1:1 matching constraint [7]. However, the deduction process of property mapping set under the global 1:1 matching constraint is not easy. It's owing to the fact that an attribute often involves the attribute pair with more than one match, so simply selecting the pair with the highest matching probability estimation may result in conflict.

The process of acquiring the whole attribute mapping cluster is called as the global property mapping. It regards a property matching pair set as an input, returning an attribute mapping cluster. Let I be the number of namespaces and J be the number of matching attributes. For two different namespaces n_s and n_t, the number of attributes under them are recorded as M_s and M_t respectively; the a^{th} attribute under n_s and the b^{th} attribute under n_t are denoted as pa^s and pb^t separately. Optimize the overall attribute matching by maximizing the total match probability of all matching attribute pairs to satisfy the global 1:1 constraint:

$$\max \sum_{s=1}^{I}\sum_{t=1}^{I}\sum_{a=1}^{M_s}\sum_{b=1}^{M_t}\sigma\left(p_a^s, p_b^t\right)\cdot\Theta\left(p_a^s \approx p_b^t\right), \tag{4}$$

$$s.t.\sum_{b=1}^{M_t}\Theta\left(p_a^t \approx p_b^t\right)\le 1, \forall s,t\in[1,I], s\ne t, \forall a\in[1,M_s] \tag{5}$$

Where σ (p_a^s, p_b^t) is the matching probability of the attribute pair pa^s and p_b^t. When a property pair p_a^s and p_b^t is selected to form an attribute mapping set, the value of the indicator function Θ (p_a^s, p_b^t) is 1 or 0.

The algorithm 1 for forming an attribute matching cluster by matching attribute pair set is shown below.

Algorithm1 Global attribute matching

Input： Attribute matching pair set J

Output： Attribute mapping cluster N

1 N←{};
2 Sort the attribute pairs in J in descending order according to matching probabilities;
3 **repeat**
4 Pop $p_a \approx p_b \in$ J and $p_a \approx p_b$ has the highest matching probability;
5 Get the attribute mapping set $N_i, N_j \in$ N where $p_a \in N_i$, $p_b \in N_j$;
6 **if** $N_i = \varnothing$ and $N_j = \varnothing$ **then**
7 Construct a new attribute mapping set and join N;
8 **else if *Ni* contains attributes from the same namespace as p_b or *Nj* contains attributes from the same namespace as p_a then;**
9 Delete $p_a \approx p_b$ because of the global 1:1 restriction rule;
10 **else**
11 Merge N_i and N_j to become a larger attribute mapping set N_k;
12 Remove N_i and N_j from N and add N_k to it;
13 **end if**
13 **until** J={};
14 **return** N;

Sort the attribute matching pair set in descending order according to matching probabilities (line 2) and process in order. For the attribute pair p_a and p_b, if N_i the attribute mapping set of p_a and N_j that of p_b are not included in the property mapping cluster N respectively, then add the attribute mapping set {p_a and p_b} to N (line 6-7). If there is an attribute mapping set N_i containing p_a and attributes from the same namespace as p_b, delete the attribute pair $p_a \approx p_b$ (line 8-9). Otherwise, merge N_i and N_j to become a larger attribute mapping set N_k. Then add N_k to N and delete N_i and N_j from N (line 11-12). Repeat until the set J is empty.

3.1.2 The goodness calculation of attribute mapping set

At this time, the property mapping set is obtained. All the attributes within a mapping set have mapped semantically with each other, so the information they correspond to is the same kind. According to Zhen Lingmin[8], the accuracy and efficiency of entity resolution can be improved by assigning higher weights to some important attributes as well as discarding some unimportant attributes. Calculate the goodness of the attribute mapping set to correspond to the above weights. When parsing subsequent entities, perform weight calculation to improve its accuracy.

Definition 6. Goodness (good()) of the property mapping set: within a property mapping set, the attribute names are probably different, but they correspond to the same kind of information. The useful degree of the type of information pointed to by the attribute mapping set, namely its corresponding attribute value information itself to the entity resolution is called the goodness of the property mapping set.

In this paper, the goodness of the property mapping set can be calculated through the following three aspects:

(1) Discriminability: within a property mapping set, if its values are variable and span a lot, it will not be much help with entity resolution. The values the attribute mapping set corresponds to vary within a relatively small range, which will be more helpful to entity resolution. Let R be the record set. For a property mapping set N_i, define a discriminability goodness function in R, denoted as discr(N_i):

$$discr(N_i) = 1 - \frac{\left|\bigcup_{r \in R} norm(val(r, N_i))\right|}{\sum_{r \in R} \left|norm(val(r, N_i))\right|} \tag{6}$$

Therein, val(r, N_i) extracts the attribute values of record r on C_i, and norm() normalizes the attribute values from different sources.

(2) Abundance: The more values the attribute mapping set possesses, the more abundant information it can provide for entity resolution, that is, for an attribute whose records are on this property mapping set, as long as its value is not empty, will be beneficial to the entity resolution. Let R be the record set. For a property mapping set N_i, define an abundance goodness function, denoted as abund(N_i):

$$\Theta\left(val\left(r,N_i\right)\neq\varnothing\right)=\begin{cases}1, \text{any attibute in } N_i \text{has a value on r}\\ 0, otherwise\end{cases} \tag{7}$$

Obviously, there is $\sum_{r\in R}\Theta$ (val(r, N_i)$\neq\varnothing$)$\leq$|R|.

(3) Diversity: in order to increase the diversity and decrease the duplication between different attribute mapping sets, reduce the goodness of the attribute mapping sets with redundant information. After selecting an attribute mapping set each time, compare it with the map set that is previously selected for use. If there are a large number of repetitions between its information and the previous mapping set, its diversity needs to be reduced. Let N_i be the current mapping set, and $N_{selected}$ the selected mapping cluster. For a given $N_{selected}$, the diversity of Ni is recorded as div(N_i|$N_{selected}$):

$$div\left(N_i\middle|N_{selected}\right)=1-\max_{N_j\in N_{selected}} div\left(N_i\middle|N_j\right) \tag{8}$$

$$div\left(N_i\middle|N_j\right)=\frac{\sum_{r\in S_i\cap S_j}\max_{\substack{v_x\in val\left(r,N_i\right)\\ v_y\in val\left(r,N_j\right)}} S_v\left(v_x,v_y\right)}{\left|S_i\cup S_j\right|} \tag{9}$$

Wherein, S_i represents a record set instantiated by N_i, namely, S_i={r$\in$R|val(r, N_i)$\neq\varnothing$ }, and so is the S_j. $S_v(v_x, v_y)$ calculates the similarity between v_x and v_y.

Combine the goodness with two steps. The first step is to combine discriminability and diversity, which reflect the static goodness of a property mapping set. Sort the static goodness of the attribute mapping set in descending order and combine the diversity to update the goodness. The second step is to combine diversity to update the goodness. Let N be the sorted attribute mapping cluster. For a mapping set $N_i\in$N, its overall goodness is good(N_i):

$$good\left(N_i\right)=\gamma\cdot comb\left(N_i\right)+\left(1-\gamma\right)\cdot div\left(N_i\middle|\left\{N_j\right\}_{j=1}^{i-1}\right) \tag{10}$$

$$comb\left(N_i\right)=\alpha\cdot discr\left(N_i\right)+\left(1-\alpha\right)\cdot abund\left(N_i\right) \tag{11}$$

Wherein, $0\leq\alpha,\gamma\leq1$.

After obtaining the goodness of each mapping set in the attribute mapping cluster, further carry out entity resolution work in the block, thereby eliminating the data records that are mistakenly included in the block and point to other entities. The calculation process is as follows: there is a record pair r_i with m attributes and r_j with n attributes. Among them, there are p attributes which are mapped and $p\leq\min(m,n)$. Therefore, the attribute of the mapping is $\{att_1,att_2,...,att_p\}$ and similarity between r_i and r_j is:

$$sim\left(r_i,r_j\right)=\frac{\sum_{l=1}^{p} good\left(N_l\right)\times sim_{content}\left(r_i.att_l,r_j.att_l\right)}{m+n-p} \tag{12}$$

Where $sim_{content}(r_i.att, r_j.att)$ is the attribute value similarity of the two record mapping attributes. att_l is an attribute that a property mapping set corresponds to, which contains a mapping of some certain attribute in r_i or r_j. Compare the similarity between the two records with a preset threshold λ. If the former is bigger than the latter, it is considered a

match. Due to the sequential processing, when the record pair is matched, they will be merged into a new record which covers the information of the original record pairs. The calculation method and integration process of $sim_{content}()$ will be described in detail in Section 3.2.

3.2 Record information integration of Similar to Regular Expression (StRE)

In the process of merging records, a method similar to regular expression can be used to merge information, since there exist some similarities between this information merging and regular expression, both of which are to determine a rule for a type of information. When a record pair is merged, the two values of the mapping properties are combined into one StRE.

3.2.1 Similar to Regular Expression (StRE) concept

Due to the fact that there are similarities among the mapping attribute values, the same part can be unified, and different parts reserved to form a Similar to Regular Expression (StRE), which can effectively merge the matching information.

Definition 7. Similar to Regular Expression (StRE): let $\sum$ be an alphabet and ε be a null character. In a StRE (StRE=S[1]S[2]...S[n]), for arbitrary i($1 \le i \le n$), the element $S[i]=\{c_{i,1},c_{i,2},...,c_{i,ni}\}$; for j($1 \le j \le n_i$), $c_{i,j} \in \sum \cup \{\varepsilon\}$. StRE is hereinafter referred to as expression.

R is an attribute value pair, S is an expression derived from an attribute value pair, and G is a set instantiated by S, at this time $R \in G$. The StRE {t,n}ight can instantiate tight and night.

The expression of an attribute value is its value itself. However, the expression that merges two attribute values or the StRE that combines a property value and an expression requires inference calculation.

3.2.2 Similarity calculation of StRE

The expression of an attribute value is itself. The edit distance can be used to calculate the similarity of two expressions. Denote edit distance function as $D(i, j)$, where i and j represent the lengths of the two strings a and b respectively. Use dynamic programming to calculate the edit distance, and then obtain the edit distance similarity function $sim_{edit}(a,b)$ between the stings:

$$sim_{edit}(a,b) = 1 - \frac{D(|a|,|b|)}{\max(|a|,|b|)} \tag{13}$$

The calculation of the similarity between an expression and an attribute or two expressions, is similar to edit distance. Since there may be multiple elements or null characters in the expression, the edit distance function is slightly modified. The modification principles are as follows: (1) For multiple characters, if two expression elements contain the same character, the expression elements match; (2) for null characters, if they do not match, the expression element containing the null character is taken as a null character. At this time, it does not occupy the length and has no effect on the edit distance.

The minimal edit distance of two expressions can be obtained through the above principles. The edit distance $D(|S_1|,|S_2|)$ of expressions S_1 and S_2 is as follows:

$$D(i,j) = \min \begin{cases} D(i,j-1) + NU(S_2[j]) \\ D(i-1,j) + NU(S_1[i]) \\ D(i-1,j-1) + MU(S_1[i], S_2[j]) \end{cases} \tag{14}$$

$$NU(S[i]) = \begin{cases} 1, \varepsilon \notin S[i] \\ 0, \varepsilon \in S[i] \end{cases} \tag{15}$$

$$MU(S_1[i], S_2[j]) = \begin{cases} 1, S_1[i] \cap S_2[j] = \varnothing \\ 0, S_1[i] \cap S_2[j] \neq \varnothing \end{cases} \tag{16}$$

S[i] represents the i^{th} element of the expression and has an initial condition:

$$D(i,j) = \min \begin{cases} 0, i = 0, j = 0 \\ D(i,j-1) + NU(S_2[j]), i = 0 \\ D(i-1,j) + NU(S_1[i]), j = 0 \end{cases}$$

(17)

Herein, the function MU corresponds to the principle (1): as long as the expression elements contain the same characters, they match. While the function NU corresponds to the principle (2), that is, if there is a null character in the expression element, which can be directly ignored and will not affect the calculation of the edit distance, and the distance can be kept to a minimum. The edit distance of two expressions is $D(|S_1|,|S_2|)$. And the edit distance similarity between expressions is $sim_{edit}(S_1,S_2)$:

$$sim_{edit}(S_1,S_2)=1-\frac{D(|S_1|,|S_2|)}{\max(|S_1|,|S_2|)} \tag{18}$$

$sim_{edit}(S_1, S_2)$ is $sim_{content}()$, and S_1 and S_2 are the two attribute values in the function $sim_{content}(r_i.att, r_j.att)$. In the process of comparing and merging, the records attribute values are regarded as expressions for calculation. After the comparison calculation is completed, the attribute information is generated from the expression.

3.2.3 Generation of StRE

The principle of generating a new expression from two expressions is to introduce the least irrelevant examples. For instance, with the attribute value pair cute kid and cut kind, the expression S_1=cut{e,ε} ki{d,n}{d,ε} with eight examples, introducing six irrelevant instances can be got, and the other expression S_2=cut{e,ε} ki{n,ε}d with four examples, introducing two irrelevant ones also can be obtained. Hence, S_2 is better than S_1.

In this paper, M the edit distance matrix of the expression is used. Start from M[|S_1|,|S_2|] and backtrack to M[0,0] to get the expression of the expression. The generation rules are as follows:

$$S[k]=\begin{cases} S_1[i]\cup\varepsilon, D[i,j]=D[i,j-1]+NU(S_2[j]) \\ S_2[j]\cup\varepsilon, D[i,j]=D[i-1,j]+NU(S_1[i]) \\ S_i[i]\cup S_2[j], D[i,j]=D[i-1,j-1]+MU(S_1[i],S_2[j]) \end{cases} \tag{19}$$

In particular:

$$S[k]=\begin{cases} S_1\cup\varepsilon, j=0 \\ S_2\cup\varepsilon, i=0 \end{cases} \tag{20}$$

Where k is a certain position in the backtracking process, and $S[k]$ is the element of the current position. When i=j=0, the backtracking ends. At this time, the expressions S_1 and S_2 are merged into an expression S=...S[k]...S[0] (wherein k is arranged in reverse order in retrospective sequence). And in the case where the three generated conditions are simultaneously satisfied, the priority of function MU is higher than that of function NU This is conducive to the merging of the same characters, reducing the introduction of irrelevant instances.

Cite an example to illustrate the similarity calculation and generation process of StRE. Let the $attvalue_1$="cute kid" and $attvalue_2$="cut kind". The corresponding expressions are S_1=cute kid and S_2=cut kind respectively. Based on the generating formula of the StRE, the distance matrix can be obtained, as shown in Table 3- 1.

Table 3- 1 Similarity Computation and Generation Matrix of Class Regular Expressions.

		c	u	t	e	k	i	d
	0	1	2	3	4	5	6	7
c	1	**0**	1	2	3	4	5	6
u	2	1	**0**	1	2	3	4	5
t	3	2	1	**0**	**1**	2	3	4
k	4	3	2	1	1	**1**	2	3
i	5	4	3	2	2	2	**1**	2

n	6	5	4	3	3	3	**2**	2
d	7	6	5	4	4	4	3	**2**

Where each value M[i, j] in the matrix represents the edit distance between the first i elements of S_1 and the first j elements of S_2. Since the expression is the attribute value itself at this time, the edit distance of the expressions is equal to the edit distance of the attribute values. The edit distance between S_1 and S_2 is M[7,7]=2, and the similarity of expressions is 1-2/7=5/7. Assuming that the similarity of the two records exceeds the threshold and they are confirmed to match, the two attribute values need to be merged. According to the generation rule of the expression, on the basis of the expression matrix M, backtrack from M[7,7]. M[7,7]=M[6,6]+ MU(S_1[6],S_2[6]), thus backtracking to the position of M[6,6], and so on. The backtracking path is marked in bold in Table 1. Finally, the expression cut{e, ε } ki{ ε ,n}d is obtained.

3.2.4 Generation of entity information

After the block processing, step-by-step record comparison, information combination and generation of the attribute values in the attribute mapping set, finally the ultimate expression of an attribute mapping set is obtained. Record the frequency of occurrence of each element in the process of generating the expression. Use this StRE with frequency to select the character with the highest frequency as the value of the element in each expression element, and at last generate a character string with the highest occurrence frequency as the attribute value of the attribute mapping set. For example, with three attribute values Mike Doe, M. Doe and Mike D, the expression M{i:2,.:1}{k:2,ε:1}{e:2,ε:1} D{o:2,.:1} {e:2,ε:1} can be obtained and in the end based on the frequency, the attribute value Mike Doe is also got.

Typesetting and spelling will lead to wrong characters, which appear to be less frequent than the correctly spelled characters. Thus, selecting the characters with highest frequency is helpful in filtering noise.

4. EXPERIMENT

4.1 Experimental data set

Two data sets are used in the block experiment, abbreviated to D_1 and D_2. With 50796 records, containing 22403 entities, D_1 extracts the movie information dataset shared by DBPedia and IMDB. With 335479 records, containing 89258 entities, D_2 extracts from a dataset consisting of two versions of DBPedia [9].

4.2 Experimental evaluation criteria

The evaluation criteria are used in this paper [10]:

Block criteria: pair completeness(PC), reduction rate (RR), F value (F=2×PC×RR/(PC+RR)).

Parsing criteria: precision (P), recall (R), F value (F=2×P×R/(P+R)).

4.3 Experimental analysis of block method

In the process of blocking, four pruning methods are described in this paper. Set α the parameter of calculating the record similarity to 0.6, then prune: for the Edge-centralization weight threshold pruning scheme (ECWP), the F value is the highest when the threshold w_{min} is 0.6 and 0.5 respectively on the dataset; for the Node-centralization weight threshold pruning scheme (NCWP), the experiment is based on an average distribution hypothesis, using a uniform weight threshold, and the execution process and threshold coincide with the ECWP; for the Edge-centralization cardinal pruning scheme (ECCP), k edges need to be preserved during pruning, and k varies with the total |E|. When k=0.5*|E|, the F values peak on both datasets; for the Node-centralization cardinal pruning scheme (NCCP), during the pruning process, reserve kri for the edges connected to each node r_i. Now, based on the average distribution hypothesis, when kri=0.5*|E_{ri}|, its F value reaches the highest. At this time, take the optimal threshold and compare the four pruning methods, as shown in Table 4- 1.

Table 4- 1 Pruning Method for PC and RR Values on Two Data Sets.

	PC(D_1)	RR(D_1)	PC(D_2)	RR(D_2)
ECWP	99.34%	62.32%	98.62%	65.17%
ECCP	97.82%	70.11%	98.01%	75.19%
NCWP	99.34%	62.32%	98.62%	65.17%

NCCP	98.90%	67.92%	98.32%	68.37%

It can be seen that the Edge-centralization pruning method cuts off more useless record pairs, focuses on efficiency, and is suitable for large-scale entity parsing tasks, especially in the case of less matching entities; while the Node-centralization pruning method retains more matching record pairs. The former algorithm preserves edges with a weight of top-k or a weight greater than a preset threshold, while the latter algorithm ensures that each node is connected to its most similar record, more suitable for applications that attach importance to accuracy. The weight threshold algorithm preserves the record pairs with higher similarity and ensures the accuracy of the algorithm; the cardinality threshold controls the number of record pairs to be compared, and keeps the edges with a weight of top-k, which has an impact on accuracy, but guarantees the efficiency of the method.

The overall accuracy is maintained above 97%, and the reduction rate above 60%, indicating that pruning the block diagrams can discard a certain amount of useless record pairs and ensures the efficiency of the algorithm to some extent.

4.4 Experimental analysis of attribute mapping and expressions

When calculating the mapping goodness of the attribute cluster, the experiment sets the parameter value to α =0.5 and λ =0.4. Select shortest distance method to compare the results. The results are evaluated in two phases: one is the result of entity parsing, the other is the generation of real entity information.

For the results of entity parsing, the accuracy and recall rate of the two methods are shown in Figure 3. It shows that as the threshold increases, the recall rate decreases and the accuracy rate increases significantly. When the threshold is 0.6, the method in this paper has reached a higher F value on both of the two datasets. On the two datasets, the recall rate of the two methods is not much different. However, in this paper, the processing method based on attribute mapping and StRE makes the recall rate slightly better. The accuracy is significantly better than that based on shortest distance method, and higher. It illustrates that the method in this paper is more suitable for the recording characteristics of dataspace. The shortest distance method is more dependent on the semantics of the data, and it can play a greater role in a more semantic environment.

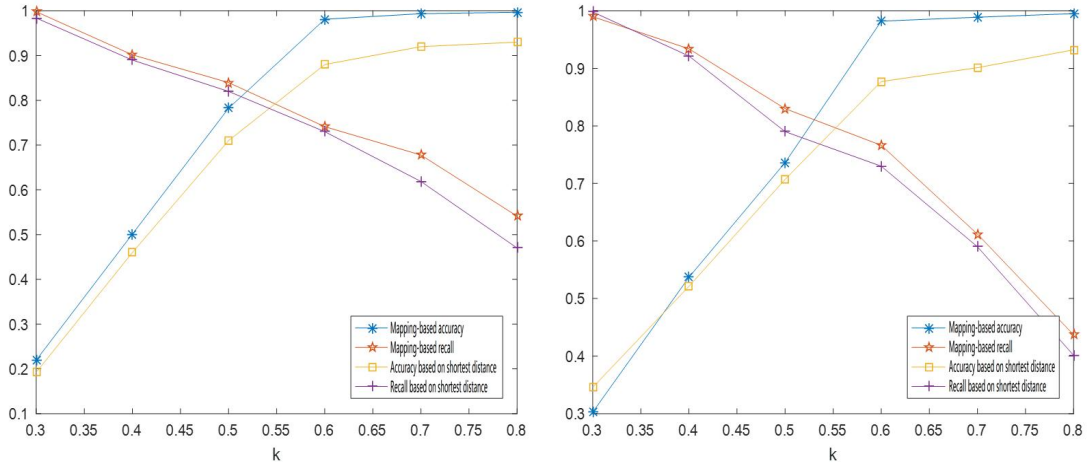


Figure 3. two methods' change with them on two data sets.

In the generation stage of entity information, most entity resolution work focuses on parsing the results, instead of entity information, merging and the generation of entity. Use data set D_1. The number of generated entities and the actual number of entities on it are as shown in Figure 4. Use the optimal threshold generated during the entity resolution process to carry out the entity generation. As can be seen from the figure, the parsing results in this paper are more accurate, and the number of generated entities is close to the number of real entities. However, its accuracy is still obviously higher than the shortest distance method.

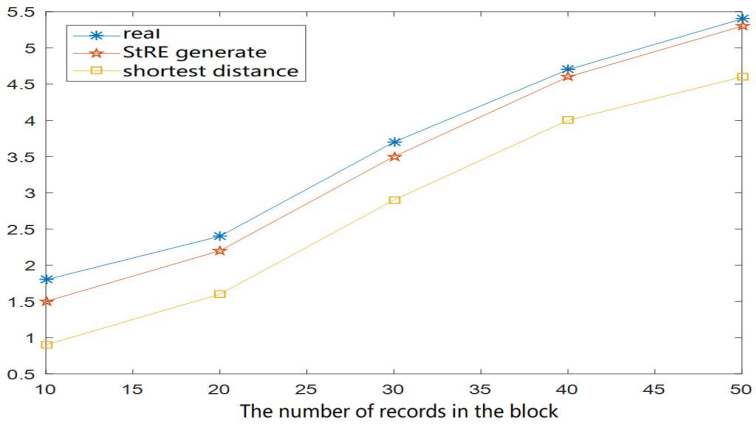


Figure 4 Entity Generation Comparison of Two Algorithms.

5. CONCLUSION

This paper is devoted to the theoretical research on the entity resolution of heterogeneous data sources in dataspaces. Considering that even in the absence of semantic mapping, two records pointing to the same entity have in common with their attribute values, and the relationship between the records is included in the calculation, and the two are combined to construct a record graph. According to the record sets under different conditions, with its applicable pruning methods to simplify the record graph and put forward the algorithm for blocking based on the pruned record graph. Through experimental verification, the proposed method in this paper has a certain positive driving effect on entity resolution.

REFERENCES

[1] Vasilis Efthymiou, Kostas Stefanidis, Vassilis Christophides. Big Data Entity Resolution: From Highly to Somehow Similar Entity Descriptions in the Web[C]// Proceeding Big Data'15 Proceedings of the 2015 IEEE International Conference on Big Data, 2015,11(1):401- 410P.

[2] GE Jingjun, HU Changjun, LIU Xin. Virtual Data Space Sharing Model for Domain Science[J].Minicomputer System,2014,35(13):514-519P.

[3] Batya Kening, Avigdor Gal. MFIBlocks: An effective blocking algorithm for entity resolution[J].Information Systems,2013,38(6):908-926P.

[4] S. Prabhakar Benny, S. Vasavi Dr, P. Anupriya. Hadoop Framework For Entity Resolution Within High Velocity Streams[J].Procedia Computer Science,2016,85: 550-557P.

[5] XIAO Qihua, CHEN Ke, HUANG Dongmei. Data Spatial Feature Extraction Method Considering Spatial Relevance[J].Computer Simulation,2014, 31(12):425-428,433P.

[6] Imen Megdiche, Oliver Teste, Cassia Trojahn. An extensible linear approach for holistic ontology matching[C]. In ISWC, Part I, vol. LNCS 9981. Springer, Kobe, Japan,2016:393-410P.

[7] Chuncheng Xiang, Baobao Chang, Zhifang Sui. An ontology matching approach based on affinity-preserving random walks[C]//Proceeding IJCAI'15 Proceedings of the 24th International Conference on Artificial Intelligence,2015:1471-1477P.

[8] ZHEN Lingmin, YANG Xiaochun, WANG Bin, et al.Entity Analysis Technology Based on Attribute Weight[J]. Computer Research and Development, 2013,50(Suppl.): 281-289P.

[9] George Papadakis, Jonathan Svirsky, Avigdor Gal, Themis Palpanas. Comparative analysis of approximate blocking techniques for entity resolution[J].Proceedings of the VLDB Endowment,2016,9(9):684-695P.

[10] Chirag Nagpal, Kyle Miller, Benedikt Boecking, Artur Dubrawski. An Entity Resolution approach to isolate instances of Human Trafficking online[J].Computer Science,2017,3(18):10-18P.