The novel scalable parallel denoising for Chinese online encyclopedia knowledge base based on the semantic distance of entry tags and Spark cluster

Ting Wang\textsuperscript{a,}\textsuperscript{*}, Jie Li\textsuperscript{b} and Jiale Guo\textsuperscript{a}

\textsuperscript{a} School of Management and Engineering, Capital University of Economics and Business, Beijing 100070, China
E-mails: wangling@cueb.edu.cn, guojiale@cueb.edu.cn
\textsuperscript{b} School of Software, Beijing JiaoTong University, Beijing 100093, China
E-mail: jielile@cueb.edu.cn

Abstract. Because of the open-collaborative of online encyclopedia, a large number of knowledge triples are improperly classified in the online encyclopedia system, so it is inevitable to denoise and refine the open-domain encyclopedia knowledge bases (KBs) to improve its quality and precision. However, the lack and inaccuracy of triple semantic features will lead to poor refining effect. Besides, in the face of large-scale encyclopedia KBs, the processing of massive knowledge will lead to too much computing time and poor scalability of the algorithm. In order to solve the problems of knowledge denoising in the Chinese encyclopedia system, firstly, based on the data field theory, this paper proposes a new Cartesian product mapping-based method for quantifying the quality of entry tags, based on which the semantic quantification of encyclopedia KB is carried out. Secondly, this paper proposes a new method based on multi-feature fusion to calculate the semantic distance between the “out-of-vocabulary” entry tags and embed it into the potential function, so as to further improve the potential function and denoising effect on KBs. Thirdly, in order to make our algorithm have good scalability, the proposed denoising algorithm is implemented and optimized in parallel based on Spark cluster computing framework. Finally, a comprehensive comparative analysis is made on the denoising effect and time efficiency with the state-of-the-art distributed Chinese encyclopedia knowledge denoising algorithm. The experimental results on the real-world datasets show that the parallel denoising algorithm proposed in this paper can improve the efficiency of knowledge denoising and the accuracy of KBs, and outperforms the state-of-the-art methods.

Keywords: Knowledge denoising, Chinese online encyclopedia, Knowledge base, Semantic distance, Potential function, Parallel computing

1. Introduction

At the beginning of the World Wide Web (WWW), the web page on the Internet is only human readable, and machines cannot understand it. Therefore the purpose of Semantic Web is to realize the “Web of Data” and machine-readable. Significantly, the large-scale open-domain and commonsense KBs are the cornerstone to realize the vision of Semantic Web, so it is of great practical significance to research on how to denoise and refine so as to improve its accuracy.

1.1. Motivation

In the era of Semantic Web, the Chinese open-domain KBs based on the online encyclopedia are also booming. However, as the entry editing of online encyclopedia is open-collaborative, and this kind of spontaneous collaboration editing behavior happened on the client does not under standardized control, so there are...
often a lot of noisy knowledge appearing in the constructed KBs. Furthermore, the usage mode of Chinese entry tags will also lead to the improper classification of InfoBox knowledge triples occur in KBs. In order to avoid noise knowledge reducing the accuracy of KBs, it is inevitable to denoise and refine it.

Nevertheless, with the advent of Big Data and Knowledge Graphs (KGs), knowledge in various fields springs up like a tide, and the number of entries in Chinese encyclopedia is growing at a very fast speed. As of April 2020, BaiduBaike, one of the largest Chinese encyclopedias in the world, has published more than 16 million entries. If we denoise the large-scale KBs in the traditional single-machine environment, there will be many problems in time efficiency and fault tolerance; therefore, the time complexity and scalability of the state-of-the-art denoising algorithms are unacceptable at present. From the perspective of High Performance Computing (HPC), since Google proposed the MapReduce framework, the technique of Big Data has made breakthroughs on storage, calculation and query; therefore, due to the lack of optimization on the network communication and I/O by the state-of-the-art methods, the denoising efficiency of algorithm also needs to be improved [1]. Furthermore, in recent years, Spark cluster computing framework based on in-memory mechanism has gradually matured, which provides a reliable platform for large-scale KBs denoising based on its high efficiency and low deployment cost.

1.2. Contributions

Based on the previous works and distributed computing, this paper proposes a novel Spark-based parallel algorithm to implement the denoising of Chinese encyclopedia KBs. The main contributions of this paper can be summarized as follows:

- With the theory of data field, this paper proposes a new Cartesian product mapping-based method for quantifying the quality of entry tags, based on which the semantic quantification of encyclopedia KB is carried out;
- Taking BaiduBaike as the target KB for denoising, in order to make full use of every tag of knowledge triples, the semantic distance of “out-of-vocabulary tag” in the entry’s tags set is matched and calculated by the novel multi-feature fusion method proposed in this paper and embed it into the potential function, so that the Precision and Recall of the knowledge denoising system is further improved;
- The semantic similarity and potential value of triples are calculated based on the Spark in parallel, then the parallel optimization of Cartesian product mapping is implemented by using Spark’s broadcast mechanism, so that the performance efficiency and scalability of the proposed denoising algorithm can be further improved.

Note: the “out-of-vocabulary tags” in this paper refers to the tag not appears in the encyclopedia classification tree.

2. Related work

Since Tim Berners Lee, the father of the WWW, formally proposed the vision of Semantic Web [2], the corresponding technical specifications and standards have been gradually developed and improved. At the same time, the online encyclopedias based on the open-collaborative model have made continuous development, such as Wikipedia 1, Hudong 2 and BaiduBaike 3. They have provided a good data source for the construction and research on the large-scale open-domain and commonsense KBs. Therefore, the related research on KBs based on online encyclopedia has been paid close attention by a great many scholars.

2.1. The construction of KBs

In the aspect of the KBs construction, there have been some typical works, such as Freebase [3], YAGO [4] and DBpedia [5, 6] in the world. Freebase is a semi-automatic KB similar to Wikipedia, but its main feature is that all entries are edited by a structured form. YAGO is a large-scale KB with high coverage and precision, which contains the is-a as well as some other semantic relations between entities. Its datasets are mainly extracted automatically in the classification systems of Wikipedia and WordNet. Furthermore, YAGO allows representing n-ary relations in a natural way while maintaining the compatibility with RDFS. DBpedia is a KB that extracts structured information from Wikipedia to build linked data. As the center of LOD, it can not only semi-automatically build cross-

1 http://www.wikipedia.org
2 https://www.baike.com/
3 http://baike.baidu.com/
domain, multi-language and dynamic updated KB, but also can support knowledge query of SPARQL on the basis of knowledge representation and organization using RDF.

However, the research on the construction of Chinese KBs is still in its infancy. In order to improve the query efficiency of search engines, since Google released the Knowledge Graph in 2012, search engines in China (e.g., Baidu) have also been following the pace to build search optimization services with local characteristics, which has also promoted the research on automatic construction of Chinese KB. Wu et al. [7] use SVMs and Markov Logic Networks (MLNs) to solve the problem of ontology refinement by integrating the Wikipedia with WordNet. Chen et al. [8] extract large-scale knowledge triples from unstructured texts of Chinese encyclopedia based on statistical learning model, and realize the preliminary construction of commonsense Chinese KB. Wang et al. [9] put forward a novel method of automatic building for domain ontology based on TF-IDF algorithm to refine the Chinese online encyclopedia, and by which automatically constructed a Chinese e-Gov KB.

2.2. Linking Open Data

Linking Open Data (LOD) is based on two simple ideas: i) using RDF data model to publish structured data on the web, and ii) building explicit RDF links between entities in different data sources [10]. Bizer et al. [11] firstly propose the concept and technical principles of LOD. Niu et al. [12] identify and extract important structural features on entry’s web page based on Chinese encyclopedia, and finally link Chinese KB with existing LOD according to the multilingual features of Wikipedia. Wang et al. [13] extract the hierarchical relationship between concepts and attributes so as to interlink with DBpedia while building a large-scale Chinese KB. Chen et al. [14] use RDF to describe the concepts in online encyclopedia for resource persistence, and interlink the self-built KB to entities in DBpedia.

With the release of large amounts of RDF triples, Volz et al. [15] introduce a toolkit named Silk framework for discovering and maintaining data links between web datasets. Scharffe et al. [16] present RDF-AI, a framework and tool for managing RDF datasets, which has provided the function of datasets integration in the context of LOD. Hassanzadeh et al. [17] propose a framework to find semantic links from relational data, which allows data publishers to easily find and publish high-quality links to other datasets. In order to solve the shortcomings of the link discovery framework in the face of large datasets, a new link discovery method in metric space is proposed in Ref. [18].

In order to solve the heterogeneity problem caused by different construction patterns in LOD, Niu et al. [19] propose a general method to automatically discover the matching of specific datasets to iteratively improve the matching rules. In terms of entity alignment of KBs in LOD, Pershina et al. [20] present a large-scale KB entity matching algorithm based on Personalized Page Rank. Wang et al. [21] design the entity alignment algorithm based on the semantic tags in multi-source KBs. In the aspect of cross-linguistic LOD construction, Wang et al. [22] study the cross-lingual interlinkings, define features through observation and propose a connection factor graph model. At the same time, Wang et al. [23] also proposes an approach that boosts cross-lingual interlinks by concept annotation, and predicts new cross-lingual links with a regression-based learning model. Wu et al. [24] design a cross-language article-linking method based on Bilingual topic model and SVM model and link articles between Wikipedia and BaiduBaike.

2.3. Knowledge Graph

Nowadays, KB in the form of KGs has been widely used in many applications, and the research of semantic web has entered the era of KG. Furthermore, KG embedding models have been widely applied to address KG completion tasks that aim to predict missing entities or relations based on existing triples in a KG [1]. There are some typical translation-based methods have been proposed to learn entity embeddings. Borders et al. [25] study the embedding problem of entities and relationships of multi-relational data in low-dimensional vector space, and then propose a method to model relationships by interpreting relationships as translation that operates on the low-dimensional embedding of entities. Wang et al. [26] investigate the problem of embedding large-scale KG composed of entities and relations in continuous vector space, then suggest the TransH model to tackle the shortcomings of TransE in embedding mapping. Lin et al. [27] suggests using TransR model to build entity and relation embedding in separate entity space and relation spaces. In order to consider the diversity of relationships and entities, a more fine-grained model named TransD, is designed based on the TransR/CTransR mode by Ref. [28]. Considering that the descriptions of entities in
most KBs are usually concise, Xie et al. [29] propose a method to encode the semantics of entity description by using entity description. Trisedya et al. [30] design a model that can exploit large numbers of attribute triples existing in the KGs and generate attribute character embeddings for capturing the similarity between entities in different KGs. In the process of automatic KB completion for commonsense knowledge graphs, to challenge significantly sparse graph structures, Chaitanya et al. [31] present novel KB completion models by exploiting the structural and semantic context of nodes. Shikhar et al. [32] analyze how the number of interactions that can be captured by ConvE, a reasoning method using low dimensional embedding of entities and relationships, affects the link prediction performance, and conclude that increasing the characteristic interaction is beneficial to the link prediction performance.

Besides, some non-translation based approaches have also been proposed to learn entity embeddings. In order to embed structured knowledge into a more flexible continuous vector space, borders et al. [33] design a learning process based on neural network architecture. And Socher et al. [34] introduce an expressive neural tensor network suitable for reasoning over relationships between two entities, which can reason the discrete entities and relationships. Wang et al. [35] propose a novel KG representation learning method by taking advantage of the rich context information in the text corpus. Nickel et al. [36] use associative memory and cyclic correlation to propose holographic embedding to learn the component vector space representation of the whole KG.

2.4. When the Semantic Web meets Big Data

In the current era of Big Data and KGs, Hadoop and Spark framework have been widely used in the storage, acquisition, query, reasoning and detection of large-scale RDF datasets. Khadilkar et al. [37] have implemented a distributed storage system, which provides users with an extensible RDF storage solution. Jingwei+ [38] is a Bigtable-based distributed RDF repository with high-scalability. Wang et al. [11] propose a parallel algorithm based on MapReduce to refine RDF triples for improving the accuracy of Chinese open-domain KBs.

In terms of query performance optimization, Husain et al. [39] build a framework that using Hadoop to store and retrieve large numbers of RDF triples by exploiting the cloud computing paradigm, settling the query problem of large RDF graph. In order to improve the query performance of SPARQL, Zeng et al. [40] elaborate Trinity. RDF, a distributed memory-based graph engine for web-scale RDF datasets. Gurajada et al. [41] investigate a new design method of distributed shared free RDF engine, which can form a grid-like distributed index through horizontal partitioning based on local RDF. Xu et al. [42] propose an extensible RDF framework based on pipeline, and use key-value to store offline RDF to improve the online query performance. Based on "partial evaluation and assembly" framework, Peng et al. [43] propose a large RDF graph in a distributed environment for SPARQL queries. Wang et al. [44] decomposes the semantic and structural query graph embedded in RDF datasets into star sets based on MapReduce, and then greatly improves the query efficiency of the algorithm by delaying Cartesian product operation. With the development of Spark framework, in order to get over the challenges on RDF queries containing multiple join operations in distributed environment, Jiuyun et al. [45] propose a Spark-based RDF query architecture, and realize more effective data search. Considering that Hadoop-based methods only support specific query patterns, Schütze et al. [46] build a prototype system based on Spark and implement most queries on RDF graphs in sub-second time. Xiong et al. [47] design a distributed parallel query engine based on Spark and HBase for ensuring the storage and query performance of KGs.

Mika et al. [48] put forward the idea of parallelizing semantic reasoning based on MapReduce as early as the emerging of Hadoop. Zhang et al. [49] propose an idea of virtual document based on ontology matching language technology, and seek an economic and effective language matching method based on MapReduce. In order to calculate the maximum number of possible relationships between entities in different datasets, based on MapReduce framework, Torre-Bastida et al. [50] propose a method to extract the relationships between entities in linked data sources. Gu et al. [51] design a large-scale distributed RDFS/OWL backward chain for semantic reasoning based on the Spark, and make optimizations in the query stage to improve the reasoning efficiency.

In order to solve the scalability problem of RDF change detection, Ahn et al. [52] propose an efficient RDF change detection algorithm based on MapReduce. Considering the limitations of existing change detection technologies in scalability or the use of RDF features, Lee et al. [53] describe the implementation
details of similarity-based RDF change detection techniques on the MapReduce.

Under the impact of artificial intelligence and Big Data technology, it can be seen that the research related to KB, RDF semantic reasoning and query, and the construction of LOD have greatly improved in accuracy. Unfortunately, there are still numerous problems in scalability and time efficiency of knowledge refinement and denoising based on the Chinese online encyclopedia system. At present, there are few typical researches on related aspects in the distributed environment.

3. Problem description

According to the characteristics of BaiduBaike classification tree, the sets of entries can be divided into 11 top-categories, namely: Geography, Economy, People, Society, Life, Culture, Art, Nature, Sports, History and Science. Each top-category contains a number of sub-categories, and each category contains an amount of corresponding entities with or not with the is-a relation (as there may be some improper classification entries in categories). Any entry can belong to one or more categories at the same time.

3.1. Knowledge triples

The structured information appeared on the web page of an entry is called InfoBox, and each InfoBox can be parsed into a knowledge triples set according to the logic information, that is, any knowledge triple can be expressed as $K = \langle S, P, O \rangle$, where $S$ is the subject, $P$ is the predicate, and $O$ is the object.

3.2. The tag set of knowledge triples

When BaiduBaike initially established the KBs, it will add the corresponding classification tags set at the same time as creating entries. Because the online encyclopedia entries are edited in an open-collaborative way, the type of tags of each triple is not distinguishable. In this paper, the tags involved in the entry of encyclopedia are divided into the following three types.

(1) Classification tag: It mainly describes the is-a relation between an entry and categories to which it belongs under the classification tree;
(2) Attribute tag: It represents the attribute information related to the entries;
(3) Ambiguous tag: Ambiguities in the entities’ names and arbitrary editing by open-collaborative editors lead to such tags [1].

On the web page of each encyclopedia entries, these three kinds of tags are displayed indiscriminately. But the classification basis of knowledge triples is totally determined by the above three kinds of tags.

3.3. Improper classification of knowledge caused by entry tags

In BaiduBaike, entries are classified according to their tags. According to Section 2.2, it can be known that attribute and ambiguous tags may cause the knowledge triples contained in entries to be improperly classified. For example, as searching the famous ancient Chinese calligrapher "吴让之" (Wu Rangzhi) in BaiduBaike, the returned entry information contains the tag set: "[书法家，书画，人物，名人，扬州]" ([Calligrapher, Painting, People, Celebrity, Yangzhou]), in which the tag "Painting" is the attribute tag, so all knowledge triples under the People entry "Wu Rangzhi" are improperly classified under the sub-category "Painting" (the top-category of it is "Culture"). Because the attributes used to describe "People", such as the "Nationality", "Occupation" and "Graduate School" etc., cannot be used to describe the category "Painting". Finally, it will lead to the imprecision of KBs. The details are shown in Figure 1.

4. System design

In order to solve the problem of imprecision of KBs mentioned above caused by entry tags, taking BaiduBaike as the target KB and Hudong as the referenced KB, a novel method based on data filed is proposed to denoise the BaiduBaike KBs. In this paper, the potential value of tags will be mapped as the degree of intimacy of its corresponding triple in the target triples set. Triples with lower potential value will be eliminated to achieve the purpose of KBs denoising and refining.

4.1. Data field and its potential function

The concept of field, first proposed by physicist Faraday, refers to the non-contact interaction between particles in a certain space. Similar to physical field, data field refers to abstract the mutual relation between data in the data domain space as the issue of effect
between material particles. The spatial distribution of
data field is usually described by scalar potential function.

Assume $P$ is the data field produced by data in $n$-dimensional space $\Omega$, $\forall (X, Y) \in \Omega$ and the function $P(x)$ as its potential function, which represents the potential value produced by the data object $X$ at $Y$. The $P(x)$ must satisfy the following conditions:

1. It is a continuous, smooth and bounded function;
2. It is isotropic;
3. It is a monotone decreasing function about distance $\|X - Y\|$, that is:
   - when the distance is 0, $P(x)$ gets the maximum value;
   - when the distance tends to infinity, $P(x) \to 0$.

Theoretically, all the functions that meet the above criteria can be used to define the potential function of data field. So as to extract the distribution characteristics of the data and cluster the data sets, this paper uses the nuclear field-like potential function to define the scalar potential of the data field.

Suppose that the $n$-dimensional data object $X = \{x_1, x_2, ..., x_n\} \in \Omega$ where $\Omega$ represents the data field space. The calculation of the potential value of $X$ in $\Omega$ is shown in Eq.(1).

$$P(x_i) = \sum_{j=1}^{n} \beta_i \times e^{-\frac{\|x_i - x_j\|}{\sigma}}$$  (1)

Where, $\|x_i - x_j\|$ represents the distance between two objects, $\beta_i \geq 0$ represents the attribute value of object $x_i$, generally defined as the mass of object, which can be understood as semantic similarity in KB [1]. $\sigma > 0$ is called the influence factor, which determines the influence scope of the data object.

4.2. Semantic quantization of KB based on tags set potential function

When the classical data field is used to calculate the potential value between data objects, it usually makes $\beta_i = 1$, that is, only considering the influence of distance on the potential value. However, it often ignores the widespread semantic correlation factors between data objects when this method is applied to KBs. In order to better extract the more accurate classification features of knowledge triples, based on the theory of data field, the tags set of a triple in the KBs is regarded as the data object in the data field in this paper, and the mass of the tag is related to the semantic similarity of its corresponding knowledge triple.

Specifically, the 11 top-categories in BaiduBaike classification tree are set as $E = \{E_1, E_2, ..., E_i, ..., E_{11}\}$, $K_d \subseteq E_i$ as the triples set of a sub-category. For example, the "Painting" is sub-category of the top-category "Culture". $K_d$ is the triples set of "Painting". $a_i \in K_d$ is one of knowledge triples in set $K_d$, and the tags set of $a_i$ is $T = \{t_1, t_2, ..., t_i, ..., t_n\}$ ($n > 0$). Because the tag discussed in this paper only reflects its specific
meaning in the context of its corresponding knowledge triple, and it can be seen from Section 2.2 that each tag in the tags set determines the classification information of the knowledge triple, so we assume that the mass of the tag is the semantic similarity of its corresponding knowledge triple and then should be divided by the number of tags in set. Therefore, in this paper, a new method for measuring the mass of the tag $t_i$ is proposed in Eq.(2).

$$\beta_i = \frac{ES_{a_i}}{N(T)}$$

(2)

Where, $ES_{a_i}$ represents the semantic similarity of triple $a_i$ in its context set $K_a$, $N(T)$ represents the number of tags in set $T$. In the next section 4.2.1, a method for measuring the value of $ES_{a_i}$ will be elaborated, and in section 4.2.2, a new method for computing the semantic distance between tags will be proposed.

4.2.1. Measurement of semantic similarity of triples based on Cartesian product mapping

According to Eq.(2), we can see that the mass of tag $t_i \in T$ is based on the semantic similarity: $ES_{a_i}$ of its corresponding knowledge triple $a_i$, which means we need to calculate the semantic similarity of knowledge triple in the triples set: $E_i$. Therefore, we use a semantic similarity measure method based on Cartesian product mapping, and then calculate the $ES_{a_i}$ of the target triples by multi-strategy comprehensive similarity method. The calculation for the value of $ES_{a_i}$ is mainly based on our previous work: Ref. [1]. But the improvement in this paper is that we use two separate triples sets as the input of algorithm to do Cartesian product mapping, by which can improve the input mode in the form of six-tuple file in the previous work [1]. This improvement can avoid data redundancy, save storage space and further improve the efficiency of Cartesian product mapping.

Assume that the target triple set $K_a = \langle a_1, a_p, a_o \rangle$, the referenced triple set $K_b = \langle b_1, b_p, b_o \rangle$, $a_i = \langle a_{i_1}, a_{i_p}, a_{i_o} \rangle (i \leq m)$ is one of triples in $K_a$, $b_j = \langle b_{j_1}, b_{j_p}, b_{j_o} \rangle (j \leq n)$ is one of triples in $K_b$, $m$ and $n$ are the number of triples in the $K_a$ and $K_b$, respectively. Then the Cartesian product of $K_a$ and $K_b$ is given by Eq.(3), and the Cartesian mapping relationship is shown in Figure 2.

$$K_a \times K_b = \{ (a_i, b_j) | a_i \in K_a \land b_j \in K_b \}$$

According to the mapping relationship, the calculation of $ES_{a_i}$ of triple $a_i$ is defined as shown in Eq.(4).

$$ES_{a_i} = \sum_{j=1}^{m} (a_i \bowtie b_j)$$

(4)

$$= a_i \bowtie b_1 + a_i \bowtie b_2 + \ldots + a_i \bowtie b_j + \ldots + a_i \bowtie b_n$$

Where $\bowtie$ stands for calculating the similarity value between $a_i$ and $b_j$. Specifically, we calculate the two similarity values based on the Edit-distance algorithm and TongYiCiCiLin algorithm respectively. Then, the semantic similarity of triple is obtained by the complementary fusion of the two similarity values according to the preset way.

(1) Similarity computing based on Edit-distance

The Edit-distance is an efficient algorithm with low resource demand in similarity calculation, through which we can get the literal similarity between triples. Suppose triple $a_i = \langle a_{i_1}, a_{i_p}, a_{i_o} \rangle$, $b_j = \langle b_{j_1}, b_{j_p}, b_{j_o} \rangle$, and the similarity of Editing-distance between subjects in $a_i$ and $b_j$ is shown in Eq.(5). The similarity between predicates and objects can be obtained in the same way.

$$ES_{Edit}(a_{i_1}, b_{j_1}) = 1 - \frac{SOP(a_{i_1}, b_{j_1})}{\max(len(a_{i_1}), len(b_{j_1}))}$$

(5)

Where, $SOP(a_{i_1}, b_{j_1})$ represents the minimum steps of adding, deleting and modifying operation times required to make the $a_{i_1}$ and $b_{j_1}$ equal to each other. $len(a_{i_1})$ and $len(b_{j_1})$ respectively represent the characters number of $a_{i_1}$ and $b_{j_1}$.

(2) Similarity calculation based on TongYiCiCiLin

As the Editing-distance can only get the literal similarity of words, in order to consider the semantic similarity of words, we calculate the similarity between triples based on "TongYiCiCiLin" (Extended Version) by Harbin Institute of Technology\(^4\). Considering that TongYiCiCiLin is the semantic dictionary with the most synonym entries (77,343 entries in total) and the most descriptive classification in the field of Chinese natural language processing, which is the reason why TongYiCiCiLin is the best option in this paper.

In the Chinese synonym dictionary "TongYiCiCiLin", each word is encoded and organized in a hierarchical relationship of a tree structure, with five layers from top to bottom. Each level has corresponding...
codes, and the five levels of codes are arranged from left to right to form the CiLin code of lexical unit. For example, take the word “诸位” (Ladies and gentlemen) in the CiLin as an example (TongYiCiCiLin code: "Aa03B02="), its synonym words with the same CiLin code include “各方”, “诸君” (Everybody, Gentlemen). Its code format is shown in Table 1.

<table>
<thead>
<tr>
<th>Sub-code</th>
<th>A</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Meaning</td>
<td>Broad heading</td>
<td>Middle heading</td>
<td>Small heading</td>
<td>Word group</td>
<td>Atomic word group</td>
<td>Equal/ synonymous/ isolated</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Layer</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Specifically, we firstly traverse the TongYiCiCiLin to obtain the corresponding codes and extract the first to the fifth layer of sub-codes, and then give the corresponding similarity weight according to the different levels. The deeper layer of the sub-codes, the higher the similarity weight, otherwise the lower. The semantic similarity between subjects of triples can be obtained by Eq.(6), and the similarity between predicates and objects can be calculated in the same way.

Fig. 2. Semantic measurement of knowledge triples based on Cartesian product mapping

\[ ES_{Word}(a_i, b_j) = \alpha \times \frac{L_n}{5} \times \cos(A_n \times \frac{\pi}{180}) \times \left[ \frac{A_n - D + 1}{A_n} \right] \]

Where \( \alpha \in (0, 1) \) represents the correlation factor for adjusting semantics, setting \( \alpha = 0.9 \) in this paper can get the best overall performance. \( L \) represents the \( n \)-th layer’s number where different sub-codes appearing between \( a_i \) and \( b_j \). \( A_n \) is the total number of nodes (words) under the branch represented by first \( n-1 \) same sub-codes between the two words from left to right, and the \( n \)-th Layer branch is determined by the first appearing different sub-codes between the two words. \( D \) represents code distance determined by the first appearing different sub-codes between the two words from left to right. For example, there are two CiLin code: "Ab03A01=" represents "青年" (Young people) and "Af05A01=" represents "皇帝" (Emperor), then the code distance \( D = 1\)’’-’’d’’=4. The "Young people" is under the "Ab" branch and the "Emperor" is under the "Af" branch. While any one of the two words is an out-of-vocabulary, the result is 0.

(3) Semantic similarity calculation of triples
Considering the semantic complementarity between \( ES_{Edit} \) based on Editing-distance and \( ES_{TWord} \) based on TongYiCiCiLin, we integrate the similarity between the two algorithms. We select the larger value as the semantic similarity of the triple. Specifically, the semantic similarity of any triple in the target triple set is given by Eq.\((7)\).

\[
ES_{ai} = \sum_{j=1}^{n} \left\{ w_1 \times \max(ES_{Edit}(a_i, b_j), ES_{TWord}(a_i, b_j)) \\
+ w_2 \times \max(ES_{Edit}(a_p, b_j), ES_{TWord}(a_p, b_j)) \\
+ w_3 \times \max(ES_{Edit}(a_o, b_p), ES_{TWord}(a_o, b_p)) \right\}
\]  

(7)

Where, we believe that the predicate is the most representative of its core semantics in a knowledge triple, followed by the subject and object, so we set the weight coefficient of \( S \), \( P \) and \( O \) in the calculation of semantic similarity of the triple \( a_i \) as \( w_1 = 0.3 \), \( w_2 = 0.5 \), \( w_3 = 0.2 \).

The specific calculation of \( ES_{ai} \) of triples is shown in algorithm TripleES.

### Algorithm 1: TripleES( \( K_a, K_b \) )

**Input:** The target triple set \( K_a \) and referenced triple set \( K_b \).

**Output:** The set of target triple and its semantic similarity

1. \( (\text{list}_a, \text{list}_b) \leftarrow \text{read}(K_a, K_b) \);
2. \( \text{foreach } a_i \in \text{list}_a \; \text{do} \)
3. \( \text{foreach } b_i \in \text{list}_b \; \text{do} \)
4. \( ES_{ai} \leftarrow 0 \);
5. \( \text{foreach } b_i \in \text{list}_b \; \text{do} \)
6. \( \text{Map}_{K_a, \text{put}}(a_i, ES_{ai}) \);
7. \( \text{return } \text{Map}_{K_a} \);

The dual For loop is used to achieve Cartesian product mapping of the triple set.

#### 4.2.2. Semantic distance calculation of tags based on multi-feature fusion strategy

According to the definition of data field potential function, we calculate the semantic distance of tags.

Suppose a triple \( a_i \in K_a \), \( K_a \) is the triples set of category \( C_a \), and the tags set of \( a_i \) is \( T = (t_1, t_2, \ldots, t_n) \). Firstly, we assume that if a tag \( t_i = C_a \), then \( t_i \) is the central tag of \( a_i \). And the semantic distance between \( t_i \) and \( \forall t_j \in T \) is defined as \( TS_d = |t_i - t_j| \). For example, there is a triple: “<吴让之，中文名, 吴熙载>” (<Wu Rangzhi, Chinese name, Wu Xizai>) belonging to the triples set of sub-category “Painting” of BaiduBaike, and its tags set is [Calligrapher, Painting, People, Celebrity, Yangzhou]. So, in the triples set of sub-category “Painting”, the central tag \( t_i \) in its tags set of this triple is defined as "Painting".

The similarity computing between tags describe in Chinese should not only consider the literal similarity, but also consider the semantic similarity between them. In order to calculate the semantic distance between the central tag and other tags, and to achieve the accurate matching between tags so as to solve the problem of "out-of-vocabulary tag" in the tags set in previous work [1], this paper propose a new multi-feature fusion method to calculate the value of \( TS_d \) in tags set.

Specifically:

- **DisBK:** A semantic distance computing method based on BaiduBaike classification tree;
- **DisLCCS:** A semantic distance computing method based on the Longest Common Continuous Substring;
- **DisT:** A semantic distance computing method based on TongYiCiCiLin.

Where, the DisLCCS algorithm calculates the distance on the literal level between two tags. The DisBK and DisT algorithms consider the distance on semantic relation level between two tags. Finally, the three algorithms will be combined together to get the semantic distance between tags in the tags set.

(1) Encyclopedia classification tree-based semantic distance computing method

The encyclopedia classification tree used in this paper refers to the hierarchical distance tree based on 11 top-categories and their related sub-categories of BaiduBaike. We assume that the semantic distance \( TS_d \) between tags in the encyclopedia classification tree is equal to the path distance. Moreover, when the central tag \( t_i \) and \( t_j \) belong to the same top-category concepts set, if the \( t_j \) is the superclass or subclass of the central tag \( t_i \), then we set the path distance to 1, otherwise to 2. In addition, as the maximum depth of each sub-category in BaiduBaike classification tree is 2, so the maximum path distance between tags is 6. At the same time, set the semantic distance of the "out-of-vocabulary" tag in the classification tree to 6 for punishing this kind of tags. An overview of the semantic distance between tags is shown in Figure 3.

For example, there is a triple "<天河，地理位置，广州>" (<Tianhe, geographic location, Guangzhou>) in the sub-category: "River", its tags set is ["地理，河流，七夕节，广州，歌曲"] ([Geography, River,
Qixi Festival, Guangzhou, Song], in which "River" is the central tag. Although the entry "Tianhe" is a river, it has tags such as "Geography" and "Song". The central tag "River" is a subclass of "Geography", so the path distance between tags: "Geography" and "River" is 1. The tag "Song" is a subclass of "Leisure", "Leisure" is the subclass of "Life", and both "Geography" and "Leisure" are the top-category, so the path between "River" and "Song" can be expressed as: 

River → Geography → ROOT → Life → Leisure → Song.

So the semantic distance between these two tags is 5.

The DisBK algorithm for path distance calculation is given by Eq.(8).

\[
\text{DisBK}(t_i, t_j) = \begin{cases} 
1, & \text{if } t_j \text{ is the parent or child of the } t_i \\
2, & \text{if } t_i \text{ and } t_j \text{ belong to the same category} \\
(4, 5, 6), & \text{if } t_i \text{ and } t_j \text{ do not belong to the same category} \\
6, & \text{"out-of-vocabulary" tag } t_j 
\end{cases}
\] (8)

(2) The Longest Common Continuous Substring-based semantic distance computing method

Because of the existence of "out-of-vocabulary" tags in the encyclopedia classification tree, thus lead to a large number of tags cannot be retrieved in the classification tree, which means that if only using the DisBK algorithm, the correct semantic distance between them cannot be obtained. For example, there is a triple of "<China Construction Bank, headquarters location, Beijing> (<China Construction Bank, headquarters location, Beijing>) belongs to sub-category: "Bank", its tags set is "[Economy, Finance, Bank, Commercial Bank]", in which the tag "Commercial Bank" is an "out-of-vocabulary" tag in the classification tree. Therefore, the semantic distance between two tags: "Bank" and "Commercial Bank" cannot be obtained by the DisBK algorithm.

In order to tackle the semantic limitation of the encyclopedia classification tree, we compute the maximum number of continuous identical characters between two tags based on the Longest Common Continuous Substring (LCCS) algorithm, so as to measure the semantic distance from \( t_j \in T \) to the central tag \( t_i \).

Set \( 0 \leq m \leq \text{len}(t_i) \), \( 0 \leq n \leq \text{len}(t_j) \), where \( \text{len}(t_i) \) and \( \text{len}(t_j) \) represent the character length of tags \( t_i \) and \( t_j \) respectively. The Dynamic Programming (DP) of the length of LCCS between tags \( t_i \) and \( t_j \) is given by algorithm LCCS. For example, the value of LCCS between tags: "Commercial Bank" (Commercial Bank) and "Bank" (Bank) is 2.

Eq.(9) is presented to calculate the semantic distance between tags no matter any one of them is the "out-of-vocabulary" tag or not. In addition, in order to
be consistent with the DisBK algorithm, the semantic distance is set to 6 in DisLCCS algorithm when there is no same character between tags.

\[
\text{DisLCCS}(t_i, t_j) = \begin{cases} 
\min\{\text{len}(t_i), \text{len}(t_j)\} & \text{LCCS}(t_i, t_j) > 0 \\
\text{LCCS}(t_i, t_j) = 0 & \text{LCCS}(t_i, t_j) = 0
\end{cases}
\]

(3) TongYiCiLin-based semantic distance computing method

Although the DisBK algorithm can compute the semantic distance in the classification tree, due to the diversity of Chinese words, the encyclopedia classification tree cannot completely and correctly collect them. Furthermore, the DisLCCS algorithm can only solve the literal similarity and requires that tags have common characters. For example, there is a triple "<苍狼乐队，中文名，苍狼乐队>" (<Canglang band, Chinese name, Canglang band>) in the sub-category: "Band" with tags set "[音乐，歌曲，乐队，专辑，简介]" ([Music, Song, Band, Album, Introduction]). In the classification tree, the tag "Music" belongs to the top-category: "Art", which is the same top-category with the central tag "Band". While the tag "Song" belongs to the top-category: "Life"(cross-category with the central tag "Band"). The tag "Music" and "Song" have the equivalent class relationship between them. However, neither the DisBK nor the DisLCCS algorithm can make full use of the tag "Song" to get the correct semantic distance. Therefore, we propose a new DisT algorithm to calculate semantic distance between tags based on TongYiCiLin to make up this defect.

Specifically, according to the five layers structure of TongYiCiLin, we design the Eq. (10) to calculate the similarity between the tags \(t_i\) and \(t_j\),

\[
\text{TSim}(t_i, t_j) = \lambda_n \times \left[ \frac{A_n - D + 1}{A_n} \right]
\]

Where, \(\lambda_n (n \leq 4)\) represents the similarity control parameter. \(A_n\) is the total number of nodes under the branch represented by first \(n\)-1 same sub-codes between the two words from left to right, and the \(n\)-th Layer branch is determined by the first appearing different sub-codes between the two words. \(D\) represents the code distance between two nodes. We also define \(\lambda_0 = 0.1\) if the first level coding between lexical units is different. If the five levels of coding are the same and the end coding is "#", \(\lambda_1 = 1\), \(\text{TSim} = 0.9\) when the end coding is "#". By parameter optimization, when the initial value of \(\lambda_n\) are set as: \(\lambda_1 = 0.25\), \(\lambda_2 = 0.6\), \(\lambda_3 = 0.7\), \(\lambda_4 = 0.9\), the target triple set can get the maximum denoising effect.

In this paper, we believe that the greater the similarity between tags is, the smaller the semantic distance will be. Therefore, we propose the Eq. (11) to calculate the semantic distance between the tag \(t_i\) and \(t_j\). At the same time, in order to keep consistent with the DisBK and DisLCCS algorithm, we define that if any one of them is the "out-of-vocabulary" in TongYiCiLin (i.e., TSim = 0), then the semantic distance between tags is 6.

\[
\text{DisT}(t_i, t_j) = \begin{cases} 
\frac{1}{\text{TSim}(t_i, t_j)} & \text{TSim} > 0 \\
6, & \text{TSim}(t_i, t_j) = 0
\end{cases}
\]

After obtaining the distance of the tags based on the above three algorithms, considering the complementarity of them, the distance values obtained by the DisBK, DisLCCS and DisT algorithms are integrated in this paper: the minimum value of the three values is selected as the semantic distance between the \(t_j \in T\) and the central tag \(t_i\) as shown in Eq. (12).

\[
TS_d(t_i, t_j) = \min\{\text{DisBK}(t_i, t_j), \text{DisLCCS}(t_i, t_j), \text{DisT}(t_i, t_j)\}
\]
4.3. Knowledge base denoising based on tag set potential function

Based on the data field theory and the tags set of triples, this paper sets the influence factor $\sigma = 10$, $k = 2$, and proposes the potential function of the interaction between the central tag $t_i$ of triple $a_i$ and other tags $t_j$; the tags set potential function is given by Eq.(13).

$$ TF_i(t_i) = \frac{ES_{a_i}}{N(T)} \times e^{-\left(\frac{TS_d(t_i)}{10}\right)^2} \quad (13) $$

Where, $\frac{ES_{a_i}}{N(T)}$ is obtained by Eq.(2), representing the mass of tag $t_i$, $TS_d(t_i, t_j)$ is obtained by Eq.(12), representing the semantic distance between tags $t_i$ and $t_j$.

On the basis of the potential function, in order to extract the correct classification information of triples more accurately, we punish the tags with relatively farther semantic distance. Specifically, the penalty optimization function is given by Eq.(14)[1].

$$ OF_i(t_i) = \begin{cases} +TF_i(t_i), & TS_d \in [0, 3] \\ -TF_i(t_i), & TS_d \in (3, 6] \end{cases} \quad (14) $$

Among them, the sign in front of $TF_i(t_i)$ represents the direction of classified acting force of the current tag $t_j$ to the triple corresponding to the central tag $t_i$. The sign “+” means that the tag has a positive classified acting force on triple $a_i$, otherwise, it means that the tag has a negative classified acting force on triple $a_i$ when it is “-”.

To sum up, the potential value of the central tag $t_i$ corresponding to its triple $a_i$ in the target triples set is given by Eq.(15).

$$ P_T(t_i) = \sum_{j=1}^{n} OF_i(t_i) \quad (15) $$

The specific design is shown in algorithm TriplePV.

After the optimization of experimental parameters, the best precision can be obtained by deleting the latter 38% triples from the target triples set sorted in descending order. Furthermore, In Ref.[11], the best precision can be achieved by deleting the latter 41% in the target triples set. Hence, we believe that near the Golden Section Point (around $\sqrt{5} - 1) / 2 \approx 0.618$, the best precision can be obtained.

Algorithm 3: TriplePV($MapK_a, T$)

Input: The set of target triple and its semantic similarity

$$ MapK_a < a_i, ES_{a_i} > $$

The set of target triple with its corresponding tags:

$$ K_T < a_i, T > $$

Output: Set of target triple and potential value

$$ PK_a < a_i, P_T(t_i) > $$

// Find central tag.

1. $t_i \leftarrow Cycle(T)$;
2. foreach ($a_i, ES_{a_i}$ in $MapK_a$ do
3.  $P_T(a_i) \leftarrow 0$ foreach $t_j$ in $T$ do
4.    $TS_d \leftarrow \min\{\text{DisBK}(t_i, t_j), \text{DisLCCS}(t_i, t_j), \text{DisT}(t_i, t_j)\}$
5.    if 0 $\leq$ $TS_d \leq$ 3 then
6.      $P_T(t_i) \leftarrow P_T(t_i) + \frac{ES_{a_i}}{N(T)} \times e^{-\left(\frac{TS_d}{10}\right)^2}$
7.    else
8.      $P_T(t_i) \leftarrow P_T(t_i) - \frac{ES_{a_i}}{N(T)} \times e^{-\left(\frac{TS_d}{10}\right)^2}$
9. end
10. end
11. $PK_a.put(a_i, P_T(t_i))$
12. return $PK_a$

4.4. An example for denoising of triples set of Painting

As mentioned in section 2.3, the tags set of the entry “吴让之” (Wu Rangzhi) contains the attribute tag "Painting", so all triples of the entry "Wu Rangzhi" are inaccurately classified under the sub-category "Painting" (the top-category of it is "Culture"). On the basis of datasets [54], this paper gives an example of denoising on the triples set of "Painting" based on the value of $P_T(t_i)$. There are 1522 triples under the sub-category "Painting" including triple “<吴让之,中文名,吴熙载>” (<Wu Rangzhi, Chinese name, Wu Xizai>), and 277476 triples under the top-category "Culture". The process of denoising the triples set "Painting" is shown in Figure 4.

Where, the central tag of the triple <Wu Rangzhi, Chinese name, Wu Xizai> is "Painting". According to Eq.(12), $TS_{d, People} = 4$, $TS_{d, Celebrity} = 6$, $TS_{d, Yangzhou} = 6$. It can be seen that these three tags exert a negative acting force on this triple being classified under "Painting". According to the result of $P_T(t_i)$ value in the triples set of "Painting" in descending order, it can be observed that triple <Wu Rangzhi, Chinese name, Wu Xizai> is on the 1507th place, which has been dropped down to the latter 38%. This example reveals that the TriplePV algorithm has the denoising effect on the improper classified triple.
4.5. Parallel optimization for KB denoising based on Spark

4.5.1. Parallel computing for semantic similarity of triples

Because the number of triples in the KB is quite numerous, it will take a lot of time to calculate the value of $ES_{a_i}$ in the Cartesian product mapping procedure directly by serial program. Therefore, we propose a novel parallel method for algorithm TripleES based on Spark.

(1) Data dependency analysis

According to Eq.(7), we firstly study and analyze the data dependency relationship of TripleES algorithm in the execution process, as shown in Figure 5.

It can be seen that, there is a dependency between the calculation of $ES_{a_i}$ and $ES(a_i, b_j)(j \in [1,n])$, because the value of $ES(a_i, b_j)(j \in [1,n])$ must be calculated firstly and then the value of $ES_{a_i}$ can be done. However, the calculation of $ES(a_i, b_1)$, ..., $ES(a_i, b_j)$, ..., $ES(a_i, b_n)$ does not depend on each other. So parallel computing can be realized between $ES(a_i, b_1)$, ..., $ES(a_i, b_j)$, ..., and $ES(a_i, b_n)$.

(2) Parallel implementation for TripleES algorithm based on Spark - ES_Spark

Based on the RDD and the characteristics of Spark framework, compared with the MapReduce-based and six-tuple file as the input in Ref.[1], in order to save the computing and storage resources, this paper separately uploads the two datasets of knowledge triples to be processed onto the HDFS. Then we use Spark’s operation to read two corresponding triples sets and transform them into RDD, the target triples set is $rdd_a$, and the referenced triples set is $rdd_b$. According to TripleES algorithm, firstly, Cartesian product are performed on $rdd_a$ and $rdd_b$ by the cartesian operation of RDD to obtain RDD mapping set. Then, we take the corresponding value to calculate the similarity and use the reduceByKey to accumulate the similarity to get the $ES_{a_i}$ of $a_i$. Finally, the $a_i$ and its corresponding value of $ES_{a_i}$ are extracted and saved back to HDFS. The parallel implementation of TripleES algorithm based on Spark is shown in ES_Spark algorithm.

The execution process of ES_Spark algorithm is summarized in Figure 6.

As shown in the Figure 6, because the cartesian operation is a narrow dependency, which means the partition of each parent-RDD is used by one partition of the child-RDD at most, and the transformation processing of partition data completes in the Stage, so the Shuffle is not caused. But the characteristic of cartesian operation is to match every element in one dataset with all elements in another dataset to generate a large dataset, in this way, the generated large dataset will take up too much memory, resulting in excessive memory consumption. As a result, it may cause the Executor pro-

Fig. 4. Denoising process of triples set “Painting”
cess to pause for the garbage collection at runtime, and lead to the parallel efficiency low. Therefore, when the dataset is enormous, cartesian operation is not ideal in the computational efficiency for Spark based on memory operation. Therefore, we present the parallel optimization for ES_Spark algorithm.

4.5.2. Parallel optimization of triple semantic similarity calculation

As mentioned above, cartesian operation will increase memory consumption, so we optimize the parallel implementation of ES_Spark algorithm based on the characteristics of Spark and datasets as follows.

(1) Broadcast variable with map instead of cartesian operation

When multiple tasks use the same variable, Spark will send this variable separately for each task, so that there will be many multiple copies in the Executor process, resulting in the worse performance with more tasks. The broadcast mechanism in Spark refers to sharing the same immutable and read-only logical vari-
Algorithm 4: ES_Spark(K_a, K_b)

Input: The target triple set K_a and the referenced triple set K_b
Output: The set of target triple and its semantic similarity

RMap < a_i, ES_a_i >
1 (rdd_a, rdd_b) ← RDD(K_a, K_b);
2 rddMap ← rdd_a.cartesian(rdd_b);
3 rddAddMap ← < >;
4 foreach v in rddMap do
5 (a_i, a_j, a_w, b_j, b_p, b_m) ← v.get();
6 ES_a_i ←
7 { 0.3 × max(ES_Edd(a_i, b_j), ES_FWord(a_i, b_j))
8 + 0.5 × max(ES_Edd(a_i, b_p), ES_FWord(a_i, b_p)) } + 0.2 × max(ES_Edd(a_i, b_m), ES_FWord(a_i, b_m))
9 rddAddMap.put(a_i, ES_a_i);
10 RMap ← rddAddMap.reduceByKey(_+_,);
11 return RMap;

Algorithm 5: PV_Spark(RMap, K_f)

Input: The set of target triple with its corresponding ES value: RMap < a_i, ES_a_i >
The set of target triple with its corresponding tags: K_f < a_i, T >
Output: Target triples set and potential value PK_a < a_i, Pr(t_i) >
1 rddKT ← RDD(K_f);
2 rddCom ← RMap.join(rddKT);
3 foreach line in rddCom.mapPartitions do
4 (a_i, ES_a_i, T) ← line.get();
5 t_j ← Cycle(T);
6 Pr(t_i) = 0;
7 foreach t_i in T do
8 TS_d ←
9 \( \min \{ \text{DisBK}(t_i, t_j), \text{DisLCS}(t_i, t_j), \text{DisT}(t_i, t_j) \} \);  
10 if \( 0 < TS_d \leq 3 \) then
11 \( Pr(t_i) \leftarrow Pr(t_i) + \frac{ES}{N(T)} × e^{-\frac{(TS_d)^2}{2}} \);
12 else
13 \( Pr(t_i) \leftarrow Pr(t_i) + \frac{ES}{N(T)} × e^{-\frac{(TS_d)^2}{2}} \);
14 PK_a.put(a_i, Pr(t_i));
15 return PK_a;
So far, we have fully realized the End-to-End parallelism of the serial algorithm of encyclopedia KBs denoising. That is to say, with the explosive growth on the datasets of online encyclopedia KBs, the parallel algorithm proposed in this paper will have high scalability.

5. Experimental environment

5.1. Distributed cluster information

The Spark cluster in this paper consists of 9 Alibaba Elastic Compute Services (ESC), including 1 master-node and 8 slave-nodes. In the 9-ESC cluster, each node is equipped with Intel(R)Xeon(R) Platinum 8163 CPU, 2.50GHz, 2 vCores, 4G DDR4 memory and 40G hard disk. The internal network bandwidth is 0.50Gbps Ethernet switch. Also, 9-ESC nodes are all configured with Ubuntu 16.04 64-bit, JDK 1.8, Hadoop 2.9.1, Scala 2.13 and Spark 2.3.4.

5.2. Spark cluster configuration

According to the above ECS cluster configuration, in order to make full use of the server’s CPU, memory and other performance to achieve the most favorable environment for Spark cluster, this paper configures the Spark parameters as shown in Table 2.

```
<table>
<thead>
<tr>
<th>No.</th>
<th>Parameter name</th>
<th>Value</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SPARK_WORKER_MEMORY</td>
<td>4G</td>
<td>Total memory allocated by the Work to</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Executor at runtime</td>
</tr>
<tr>
<td>2</td>
<td>SPARK_WORKER_CORES</td>
<td>2</td>
<td>Total cores allocated by the Work to</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Executor at runtime</td>
</tr>
<tr>
<td>3</td>
<td>num-executor</td>
<td>8</td>
<td>Total numbers of Executer processes</td>
</tr>
<tr>
<td>4</td>
<td>executor-cores</td>
<td>2</td>
<td>Running cores per Executer processes</td>
</tr>
</tbody>
</table>
```

Moreover, we set the number of partitions of textfile of RDD operation to adjust the parallelism of Spark.
6. Experiment and Analysis

6.1. Datasets and Design

In order to compare with the state-of-the-art methods: Ref. [9] and Ref. [1], the same datasets is used in this paper. The datasets for testing [55] come from the real-world datasets of BaiduBaike KB, and the construction strategy is based on the method in Ref. [13]. The details are as follows:

(1) The 17 sub-categories datasets in target KB BaiduBaike and their corresponding tag set;
(2) The 9 top-categories datasets in the referenced KB Hudong and their corresponding tag set

The mapping relationship between BaiduBaike and Hudong is shown in Table 4. Among them, the number of mapping times are obtained by the Cartesian product operation of triples in the BaiduBaike(m) and Hudong(n) KBs, i.e. \( m \times n \); the number of similarity calculation is \( m \times n \times 3 \), i.e. \(<S, P, O>\) respectively.

In order to test the effectiveness and scalability of the TriplePV algorithm proposed in this paper, two experiments are carried out on the above datasets, respectively:

Experiment 1: Verifying the denoising effect of the TriplePV algorithm on the BaiduBaike KB;
Experiment 2: Verifying the scalability of the PV_Spark algorithm - the parallel implementation of TriplePV algorithm.

6.2. Experiment 1: Denoising for the BaiduBaike KB

The triples set of each sub-category for testing is annotated by the Chinese senior grade students. This is the manually annotated principle with the principle of whether the features of triple can match the related sub-categories or not. If it matches, the Y is marked, otherwise N is marked [1].

In this paper, the following two steps are designed to test the denoising effect of TriplePV algorithm.

Step 1: Ranking changes happened on the improperly classified triples
Step 2: The evaluation on the KB denoising algorithm: TriplePV

6.2.1. Ranking changes happened on the improperly classified triples

The knowledge triples are sorted in descending order based on the \( ES_{a_i} \) and \( PT(t_i) \) value respectively. Then, according to the idea of Golden Section Point (around \((\sqrt{5} - 1) / 2 \approx 0.618\) , taking the number of triples with improper classification (marked with N) in the latter 38%, and comparing the result with Ref. [1] (in latter 41%). The results are shown in Table 5, the difference value represents the number changes of the improperly classified triples based on the \( PT(t_i) \) value compared with the number of triples based on the \( ES_{a_i} \) value and Ref. [1]. The positive sign (+) represents an increase, the negative sign (-) represents a decrease and the 0 represents no change.

The results reveal that the ability of recognizing and denoising the improper classified triples based on TriplePV algorithm is improved on the whole, as compared with Ref. [1], the number of improperly classified triples in the latter 38% increased by 449 in total, while the elimination area of Ref. [1] was latter 41%.
Table 4
Mapping of test datasets

<table>
<thead>
<tr>
<th>Sub-categories of BaiduBaike</th>
<th>The number of triples in BaiduBaike (Precision)</th>
<th>The number of triples in Hudong (Recall)</th>
<th>The number of mapping (billion)</th>
<th>The number of similarity computing (billion times)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Competition</td>
<td>93 (87.10%)</td>
<td>44371</td>
<td>0.0413</td>
<td>0.1238</td>
</tr>
<tr>
<td>Landform</td>
<td>143 (83.92%)</td>
<td>810 (95.07%)</td>
<td>3.687</td>
<td>11.0609</td>
</tr>
<tr>
<td>Collection</td>
<td>180 (85.39%)</td>
<td>455182</td>
<td>5.9102</td>
<td>17.7207</td>
</tr>
<tr>
<td>Bank</td>
<td>322 (81.43%)</td>
<td>164656</td>
<td>1.0966</td>
<td>3.2897</td>
</tr>
<tr>
<td>Lake</td>
<td>426 (15.87%)</td>
<td>2.2304</td>
<td>0.7014</td>
<td>2.1043</td>
</tr>
<tr>
<td>River</td>
<td>1022 (65.80%)</td>
<td>129312</td>
<td>1.0917</td>
<td>3.275</td>
</tr>
<tr>
<td>Landform</td>
<td>1522 (83.92%)</td>
<td>0.6509</td>
<td>5.3052</td>
<td>15.9156</td>
</tr>
<tr>
<td>Architecture</td>
<td>857 (74.02%)</td>
<td>2.378</td>
<td>7.1339</td>
<td></td>
</tr>
<tr>
<td>Prose</td>
<td>2130 (73.62%)</td>
<td>5.9102</td>
<td>17.7207</td>
<td></td>
</tr>
<tr>
<td>Earthquake</td>
<td>128 (89.68%)</td>
<td>3.687</td>
<td>11.0609</td>
<td></td>
</tr>
<tr>
<td>Constellation</td>
<td>322 (81.43%)</td>
<td>5.9102</td>
<td>17.7207</td>
<td></td>
</tr>
<tr>
<td>Scenic spot</td>
<td>3591 (86.66%)</td>
<td>3.687</td>
<td>11.0609</td>
<td></td>
</tr>
<tr>
<td>Teacher</td>
<td>2130 (73.62%)</td>
<td>5.9102</td>
<td>17.7207</td>
<td></td>
</tr>
<tr>
<td>Politics</td>
<td>3591 (86.66%)</td>
<td>3.687</td>
<td>11.0609</td>
<td></td>
</tr>
</tbody>
</table>

Table 5
The number of changes of improper classified triples in the elimination area

<table>
<thead>
<tr>
<th>Sub-categories of BaiduBaike</th>
<th>Latter 41% (number of triples)</th>
<th>Latter 38% (number of triples)</th>
<th>The difference value (number of triples)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Competition</td>
<td>12</td>
<td>2</td>
<td>11</td>
</tr>
<tr>
<td>Landform</td>
<td>17</td>
<td>13</td>
<td>23</td>
</tr>
<tr>
<td>Collection</td>
<td>128</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>Bank</td>
<td>43</td>
<td>2</td>
<td>45</td>
</tr>
<tr>
<td>Lake</td>
<td>301</td>
<td>43</td>
<td>58</td>
</tr>
<tr>
<td>Language</td>
<td>9</td>
<td>283</td>
<td>322</td>
</tr>
<tr>
<td>River</td>
<td>115</td>
<td>104</td>
<td>145</td>
</tr>
<tr>
<td>Architecture</td>
<td>338</td>
<td>280</td>
<td>326</td>
</tr>
<tr>
<td>Prose</td>
<td>312</td>
<td>253</td>
<td>373</td>
</tr>
<tr>
<td>Earthquake</td>
<td>98</td>
<td>89</td>
<td>118</td>
</tr>
<tr>
<td>Constellation</td>
<td>84</td>
<td>68</td>
<td>95</td>
</tr>
<tr>
<td>Scenic spot</td>
<td>214</td>
<td>153</td>
<td>292</td>
</tr>
<tr>
<td>Food</td>
<td>271</td>
<td>158</td>
<td>307</td>
</tr>
<tr>
<td>Teacher</td>
<td>1300</td>
<td>1205</td>
<td>1335</td>
</tr>
<tr>
<td>Politics</td>
<td>353</td>
<td>294</td>
<td>467</td>
</tr>
</tbody>
</table>

Total | 4266 | 3561 | 715 | +135 |

(16) 

\[ P-text{value} = \frac{number\ of\ outputted\ Y}{total\ number\ of\ outputs} \times 100\% \]

(17) 

\[ R-text{value} = \frac{number\ of\ outputted\ Y}{total\ number\ of\ Y\ in\ each\ dataset} \times 100\% \]

Similarly, the triples are sorted by the TF-IDF method in Ref. [9]. For a better and fair comparison, we also take the first 62% of the triples sorted by the TF-IDF and calculate the P-value and R-value to compare with it. The P-value and R-value obtained by the TriplePV algorithm proposed in this paper will be compared with the state-of-the-art methods Ref. [1, 9].

(1) Precision

Based on Eq.(16), calculate the P-value of 17 sub-categories datasets. The results are shown in Table 6 and Figure 10. Among them, the P-difference represents the change of P-value based on the TriplePV algorithm compared with other state-of-the-art methods, the positive sign (+) represents the rise, the negative sign (-) represents the decline, and the 0 represents unchanged.

The results reveal that the denoising effect based on the \( P_T(t_i) \) value is generally improved in the P-value compared with other state-of-the-art methods.
### Table 6

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Competition</td>
<td>84.96%</td>
<td>100.00%</td>
<td>100.00%</td>
<td>+15.04%</td>
<td>0.00%</td>
</tr>
<tr>
<td>Landform</td>
<td>90.10%</td>
<td>92.89%</td>
<td>100.00%</td>
<td>+9.90%</td>
<td>+7.11%</td>
</tr>
<tr>
<td>Collection</td>
<td>95.13%</td>
<td>95.23%</td>
<td>95.45%</td>
<td>+0.32%</td>
<td>+0.22%</td>
</tr>
<tr>
<td>Band</td>
<td>64.59%</td>
<td>68.30%</td>
<td>82.97%</td>
<td>+18.38%</td>
<td>+14.67%</td>
</tr>
<tr>
<td>Bank</td>
<td>75.63%</td>
<td>77.81%</td>
<td>83.05%</td>
<td>+7.42%</td>
<td>+5.24%</td>
</tr>
<tr>
<td>Lake</td>
<td>80.88%</td>
<td>87.20%</td>
<td>90.10%</td>
<td>+9.22%</td>
<td>+2.90%</td>
</tr>
<tr>
<td>Language</td>
<td>20.74%</td>
<td>24.25%</td>
<td>25.05%</td>
<td>+4.31%</td>
<td>+0.80%</td>
</tr>
<tr>
<td>River</td>
<td>64.78%</td>
<td>68.82%</td>
<td>84.06%</td>
<td>+19.28%</td>
<td>+15.24%</td>
</tr>
<tr>
<td>Paintings</td>
<td>23.42%</td>
<td>30.73%</td>
<td>36.06%</td>
<td>+12.64%</td>
<td>+5.33%</td>
</tr>
<tr>
<td>Architecture</td>
<td>75.81%</td>
<td>71.80%</td>
<td>74.36%</td>
<td>-1.45%</td>
<td>+2.56%</td>
</tr>
<tr>
<td>Poise</td>
<td>74.44%</td>
<td>80.11%</td>
<td>85.68%</td>
<td>+11.24%</td>
<td>+5.57%</td>
</tr>
<tr>
<td>Earthquake</td>
<td>70.23%</td>
<td>78.23%</td>
<td>86.39%</td>
<td>+16.16%</td>
<td>+8.16%</td>
</tr>
<tr>
<td>Constellation</td>
<td>83.55%</td>
<td>92.88%</td>
<td>95.30%</td>
<td>+11.75%</td>
<td>+2.71%</td>
</tr>
<tr>
<td>Scenic Spot</td>
<td>92.45%</td>
<td>92.88%</td>
<td>97.56%</td>
<td>+5.11%</td>
<td>+4.68%</td>
</tr>
<tr>
<td>Food</td>
<td>88.74%</td>
<td>90.18%</td>
<td>92.46%</td>
<td>+3.72%</td>
<td>+2.28%</td>
</tr>
<tr>
<td>Teacher</td>
<td>41.10%</td>
<td>42.52%</td>
<td>44.25%</td>
<td>+3.15%</td>
<td>+1.73%</td>
</tr>
<tr>
<td>Politics</td>
<td>90.80%</td>
<td>97.53%</td>
<td>97.57%</td>
<td>+3.77%</td>
<td>+0.04%</td>
</tr>
<tr>
<td>Average</td>
<td>71.79%</td>
<td>75.95%</td>
<td>80.61%</td>
<td>+8.82%</td>
<td>+4.66%</td>
</tr>
</tbody>
</table>

As shown in Table 4, it is revealed by our manually annotated results that the average Precision of the original 17 sub-categories datasets before refining is only 70.23%. It can be seen that the average P-value of our method is increased by about 10.38% compared with the original datasets [13] for refining. Compared with Ref. [9], although the sub-category “Architecture” are reduced by 1.45%, but the average value is still increased by about 8.82%. Compared with Ref. [1], the average value is also increased by about 4.66%. It can be concluded that after denoising, the accuracy of KB is further improved.

(2) Recall
Based on Eq.(17), calculate the R-value of 17 subcategories datasets. The results are shown in Table 7 and Figure 11. Among them, the R-difference represents the change of R-value based on the TriplePV algorithm compared with other state-of-the-art methods, the positive sign (+) represents the rise, the negative sign (-) represents the decline, and the 0 represents unchanged.

Table 7

<table>
<thead>
<tr>
<th>Sub-categories of BaiduBaike</th>
<th>R-value</th>
<th>R-value</th>
<th>R-difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ref.[9]</td>
<td>Ref.[1]</td>
<td>P_T(t_i)</td>
<td>Compared with Ref.[9]</td>
</tr>
<tr>
<td>Competition</td>
<td>69.57%</td>
<td>55.56%</td>
<td>69.51%</td>
</tr>
<tr>
<td>Collection</td>
<td>59.30%</td>
<td>64.17%</td>
<td>73.33%</td>
</tr>
<tr>
<td>Landform</td>
<td>63.71%</td>
<td>63.04%</td>
<td>81.00%</td>
</tr>
<tr>
<td>Bank</td>
<td>73.43%</td>
<td>64.63%</td>
<td>99.99%</td>
</tr>
<tr>
<td>Lake</td>
<td>61.33%</td>
<td>62.28%</td>
<td>67.91%</td>
</tr>
<tr>
<td>Language</td>
<td>61.16%</td>
<td>62.50%</td>
<td>72.97%</td>
</tr>
<tr>
<td>River</td>
<td>66.23%</td>
<td>62.39%</td>
<td>72.14%</td>
</tr>
<tr>
<td>Paintings</td>
<td>15.53%</td>
<td>50.00%</td>
<td>58.71%</td>
</tr>
<tr>
<td>Architecture</td>
<td>59.86%</td>
<td>57.99%</td>
<td>62.29%</td>
</tr>
<tr>
<td>Prose</td>
<td>72.37%</td>
<td>60.26%</td>
<td>72.13%</td>
</tr>
<tr>
<td>Earthquake</td>
<td>83.88%</td>
<td>67.52%</td>
<td>80.38%</td>
</tr>
<tr>
<td>Constellation</td>
<td>64.79%</td>
<td>62.85%</td>
<td>69.91%</td>
</tr>
<tr>
<td>Scenic spot</td>
<td>61.76%</td>
<td>59.03%</td>
<td>67.08%</td>
</tr>
<tr>
<td>Food</td>
<td>75.94%</td>
<td>75.99%</td>
<td>66.06%</td>
</tr>
<tr>
<td>Teacher</td>
<td>63.13%</td>
<td>62.72%</td>
<td>73.32%</td>
</tr>
<tr>
<td>Politics</td>
<td>33.60%</td>
<td>58.67%</td>
<td>66.70%</td>
</tr>
<tr>
<td>Average</td>
<td>63.39%</td>
<td>60.58%</td>
<td>72.91%</td>
</tr>
</tbody>
</table>

The results reveal that the denoising effect based on the $P_T(t_i)$ value is generally improved in the R-value compared with other state-of-the-art methods. The average R-value is increased by about 11.52% compared with the average value of Ref. [9]; compared with Ref. [1], the average R-value is increased by about 12.33%.

6.2.3. T-test analysis of paired samples of experimental results

In order to further and more scientifically explain the improvement of the method based on $P_T(t_i)$ value from other researches in Ref. [1, 9], we make the T-test analysis of paired samples for several groups of comparative results, and the results are shown in Table 8.

Table 8

<table>
<thead>
<tr>
<th>Paired Samples Test</th>
<th>Paired Differences</th>
<th>t</th>
<th>df</th>
<th>Sig.(2-tailed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>Std. Deviation</td>
<td>t</td>
<td>df</td>
<td>Sig.(2-tailed)</td>
</tr>
<tr>
<td>Lower</td>
<td>Upper</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pair 1</td>
<td>P3 - P1</td>
<td>0.097</td>
<td>0.062</td>
<td>0.018</td>
</tr>
<tr>
<td>Pair 2</td>
<td>R3 - R2</td>
<td>0.123</td>
<td>0.100</td>
<td>0.027</td>
</tr>
<tr>
<td>Pair 3</td>
<td>R3 - R1</td>
<td>0.115</td>
<td>0.201</td>
<td>0.049</td>
</tr>
<tr>
<td>Pair 4</td>
<td>P3 - P2</td>
<td>0.047</td>
<td>0.046</td>
<td>0.008</td>
</tr>
</tbody>
</table>

In Table 8, P1, P2 and P3 respectively represent the precision of the Ref. [1, 9] and the method based on $P_T(t_i)$ value. R1, R2 and R3 respectively represent the recall of the Ref. [1, 9] and the method based on $P_T(t_i)$ value. The comparison results of pairs 1 to 4 show that...
the significance level of the $P_T(t_i)$ method is less than 0.05 when compared with Ref. [1, 9] in precision and recall, namely, the denoising effect based on the $P_T(t_i)$ value is significantly improved compared with other state-of-the-art methods.

6.3. Experiment 2: Verification of parallelization

In order to verify the time effect of parallelization, a single-machine with the same configuration as the above cluster server is built in this paper to evaluate the serial execution time of the TripleES and TriplePV algorithms. Forethemore, the speedup, which is given by Eq. (18), is used to verify the parallel efficiency of the ES_Spark and PV_Spark algorithms.

$$Speedup = \frac{T_s}{T_n}$$

Where $T_s$ represents the execution time of serial algorithms and $T_n$ represents the parallel execution time executed by $n$ processes.

6.3.1. Effect evaluation on the parallel computing of semantic similarity $ES_{ai}$

In order to verify the optimization effect of parallel computing on the value of $ES_{ai}$, the following three methods are implemented as the comparative experiments:

- Method 1: reproduce the $ES_{ai}$ calculation based on MapReduce framework in Ref. [1];
- Method 2, SparkCART: parallel verification of ES_Spark algorithm based on cartesian operation proposed in Section 4.5.1;
- Method 3, SparkBCST: parallel verification of the improved ES_Spark algorithm based on broadcast mechanism proposed in Section 4.5.2.

(1) Running Time

According to the cluster configuration in Section 5.1, this paper sets 8 and 16 processes respectively to perform mapping tasks, the running results are shown in Table 9. Figure 12 and 13 are in the manner of Bar Graph of execution time according to Table 9.

The results reveal that while the same datasets are tested on both MapReduce and Spark cluster, with the increase of the number of processes, the execution time is greatly reduced. At the same time, with the same number of running processes, the execution time of SparkCART and SparkBCST methods are much faster than the MapReduce-based one in Ref. [1]. The specific reasons are as follows:

Generally, there are four main factors that can affect the computing efficiency of cluster: CPU, memory, network overhead and I/O. So the optimization strategies should be mainly as:

1) Improve CPU utilization;
2) To avoid out of memory (OOM);
3) Reduce network overhead;
4) Reduce I/O operations.

It is more resource-saving that ES_Spark algorithm reads the target and reference triples sets separately. While the input of Ref. [1] is a combined set in the form of six-tuple, the redundancy of this kind of dataset is very large, which will not only lead to increased I/O operations and network overhead during data exchange, but also a large amount of memory are occupied. Compared with MapReduce framework, which needs to write data to disk at map-end firstly, and then read data from disk at reduce-end, Spark not only has greater advantages in memory management, but also can reduce the transmission time of data exchange by the lazy of transformations between RDD operations and the caching for intermediate results etc., thus reducing network overhead and I/O operations. At the same time, Spark’s broadcast mechanism saves only one variable copy in an Executor, which reduces the memory consumption of tasks, so the SparkBCST has the fastest execution time.

(2) The speedup

The speedup of 17 denoising tasks in parallel computing are shown in Table 10 and Figure 14.

It can be seen that in the same dataset, with the increase of the number of processes, the speedup of the three methods are gradually expanding, but the speedup of SparkCART and SparkBCST method based on Spark is obviously increasing than that of MapReduce-based method [1]. Specifically, in 8 processes, the average speedup of method SparkBCST is increased by 1.6 compared with the Ref. [1], while compared with the method SparkCART, it improves 0.5. In 16 processes, compared with the Ref. [1], the average speedup of method SparkBCST is increased by 3.7, it improves 1.5 compared with the SparkCART.

(3) Analysis the advantages of algorithm SparkBCST by observing the garbage collection time

In order to further analyze the advantages of the SparkBCST algorithm proposed in this paper, by showing the garbage collection (GC) time of the JVM (Java virtual machine), the time when the running process pauses for garbage collection, we cleverly reflect the advantages, disadvantages and differences between the three methods in terms of time and space utiliza-
Table 9
Execution time of $E_{S_{ai}}$ computing

<table>
<thead>
<tr>
<th>Sub-categories of BaiduBaike</th>
<th>Serial time (s)</th>
<th>8 processes (s)</th>
<th>16 processes (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Spark CART</td>
<td>Spark BCST</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ref.[1]</td>
<td>Ref.[1]</td>
</tr>
<tr>
<td>Competition</td>
<td>710.99</td>
<td>112.83</td>
<td>102.80</td>
</tr>
<tr>
<td>Landform</td>
<td>8963.06</td>
<td>1447.89</td>
<td>1348.94</td>
</tr>
<tr>
<td>Collection</td>
<td>9114.99</td>
<td>1338.72</td>
<td>1337.48</td>
</tr>
<tr>
<td>Band</td>
<td>15400.69</td>
<td>2018.87</td>
<td>2122.95</td>
</tr>
<tr>
<td>Bank</td>
<td>13894.71</td>
<td>2443.70</td>
<td>2164.82</td>
</tr>
<tr>
<td>Lake</td>
<td>30711.38</td>
<td>6350.33</td>
<td>4283.89</td>
</tr>
<tr>
<td>Language</td>
<td>35168.36</td>
<td>6460.25</td>
<td>4807.21</td>
</tr>
<tr>
<td>River</td>
<td>47879.57</td>
<td>8010.23</td>
<td>6562.54</td>
</tr>
<tr>
<td>Paintings</td>
<td>53189.87</td>
<td>8280.75</td>
<td>7665.08</td>
</tr>
<tr>
<td>Architecture</td>
<td>30755.78</td>
<td>6425.24</td>
<td>4488.65</td>
</tr>
<tr>
<td>Prose</td>
<td>37272.78</td>
<td>6455.73</td>
<td>5289.12</td>
</tr>
<tr>
<td>Earthquake</td>
<td>18225.28</td>
<td>2837.70</td>
<td>2612.33</td>
</tr>
<tr>
<td>Constellation</td>
<td>32593.12</td>
<td>5140.34</td>
<td>4689.21</td>
</tr>
<tr>
<td>Scenic spot</td>
<td>50712.93</td>
<td>8940.78</td>
<td>6903.17</td>
</tr>
<tr>
<td>Food</td>
<td>60607.27</td>
<td>11620.43</td>
<td>8292.03</td>
</tr>
<tr>
<td>Teacher</td>
<td>61488.26</td>
<td>11745.99</td>
<td>8576.32</td>
</tr>
<tr>
<td>Politics</td>
<td>125214.55</td>
<td>22134.53</td>
<td>20334.53</td>
</tr>
</tbody>
</table>

Fig. 12. Comparison of execution time of $E_{S_{ai}}$ - part I
Fig. 13. Comparison of execution time of $ES_{mi}$ - part II

Fig. 14. Comparison of speedup of $ES_{mi}$ computing
Table 10  
<table>
<thead>
<tr>
<th>Sub-categories of BaiduBaike</th>
<th>8 processes</th>
<th>16 processes</th>
<th>SparkCT</th>
<th>SparkBD</th>
<th>SparkCT</th>
<th>SparkBD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Competition</td>
<td>6.80</td>
<td>6.93</td>
<td>7.19</td>
<td>12.21</td>
<td>13.62</td>
<td>15.74</td>
</tr>
<tr>
<td>Landform</td>
<td>6.19</td>
<td>6.44</td>
<td>7.50</td>
<td>10.96</td>
<td>13.40</td>
<td>15.10</td>
</tr>
<tr>
<td>Collection</td>
<td>6.81</td>
<td>6.82</td>
<td>7.33</td>
<td>12.87</td>
<td>13.99</td>
<td>15.30</td>
</tr>
<tr>
<td>Band</td>
<td>7.63</td>
<td>6.96</td>
<td>7.51</td>
<td>13.38</td>
<td>14.06</td>
<td>15.57</td>
</tr>
<tr>
<td>Lake</td>
<td>4.84</td>
<td>7.17</td>
<td>7.59</td>
<td>10.95</td>
<td>14.52</td>
<td>15.90</td>
</tr>
<tr>
<td>Language</td>
<td>5.44</td>
<td>7.32</td>
<td>7.73</td>
<td>12.90</td>
<td>14.54</td>
<td>15.37</td>
</tr>
<tr>
<td>River</td>
<td>5.08</td>
<td>7.10</td>
<td>7.71</td>
<td>13.44</td>
<td>14.51</td>
<td>15.46</td>
</tr>
<tr>
<td>Paintings</td>
<td>6.42</td>
<td>6.94</td>
<td>7.60</td>
<td>12.37</td>
<td>13.95</td>
<td>15.87</td>
</tr>
<tr>
<td>Architecture</td>
<td>4.59</td>
<td>6.45</td>
<td>7.63</td>
<td>13.05</td>
<td>14.26</td>
<td>15.49</td>
</tr>
<tr>
<td>Prose</td>
<td>5.77</td>
<td>7.05</td>
<td>7.49</td>
<td>11.26</td>
<td>13.04</td>
<td>15.60</td>
</tr>
<tr>
<td>Earthquake</td>
<td>6.42</td>
<td>6.98</td>
<td>7.66</td>
<td>9.38</td>
<td>14.06</td>
<td>15.57</td>
</tr>
<tr>
<td>Constellation</td>
<td>6.34</td>
<td>6.95</td>
<td>7.54</td>
<td>13.62</td>
<td>14.13</td>
<td>15.30</td>
</tr>
<tr>
<td>Scenic spot</td>
<td>5.67</td>
<td>7.35</td>
<td>7.92</td>
<td>12.05</td>
<td>14.13</td>
<td>15.72</td>
</tr>
<tr>
<td>Food</td>
<td>5.22</td>
<td>7.31</td>
<td>7.60</td>
<td>12.29</td>
<td>14.21</td>
<td>15.48</td>
</tr>
<tr>
<td>Teacher</td>
<td>5.23</td>
<td>7.17</td>
<td>7.66</td>
<td>10.22</td>
<td>14.42</td>
<td>15.43</td>
</tr>
<tr>
<td>Politics</td>
<td>5.56</td>
<td>6.16</td>
<td>7.52</td>
<td>12.11</td>
<td>14.16</td>
<td>15.04</td>
</tr>
<tr>
<td>Average</td>
<td>5.02</td>
<td>6.40</td>
<td>7.52</td>
<td>11.44</td>
<td>13.13</td>
<td>15.52</td>
</tr>
</tbody>
</table>

Table 11  
<table>
<thead>
<tr>
<th>Sub-categories of BaiduBaike</th>
<th>GC time (s)</th>
<th>Ref.[1]</th>
<th>SparkCT</th>
<th>SparkBD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Competition</td>
<td>0.358</td>
<td>0.335</td>
<td>0.326</td>
<td></td>
</tr>
<tr>
<td>Landform</td>
<td>9.130</td>
<td>1.370</td>
<td>0.615</td>
<td></td>
</tr>
<tr>
<td>collection</td>
<td>7.944</td>
<td>2.656</td>
<td>0.483</td>
<td></td>
</tr>
<tr>
<td>Band</td>
<td>13.407</td>
<td>2.131</td>
<td>0.428</td>
<td></td>
</tr>
<tr>
<td>Bank</td>
<td>8.692</td>
<td>2.035</td>
<td>0.360</td>
<td></td>
</tr>
<tr>
<td>Lake</td>
<td>21.797</td>
<td>12.673</td>
<td>2.523</td>
<td></td>
</tr>
<tr>
<td>Language</td>
<td>22.868</td>
<td>9.652</td>
<td>1.176</td>
<td></td>
</tr>
<tr>
<td>River</td>
<td>20.359</td>
<td>8.309</td>
<td>0.882</td>
<td></td>
</tr>
<tr>
<td>Paintings</td>
<td>28.755</td>
<td>12.538</td>
<td>0.961</td>
<td></td>
</tr>
<tr>
<td>Architecture</td>
<td>18.139</td>
<td>7.827</td>
<td>0.729</td>
<td></td>
</tr>
<tr>
<td>Prose</td>
<td>17.583</td>
<td>7.055</td>
<td>0.574</td>
<td></td>
</tr>
<tr>
<td>Earthquake</td>
<td>15.071</td>
<td>5.013</td>
<td>0.478</td>
<td></td>
</tr>
<tr>
<td>Constellation</td>
<td>31.340</td>
<td>7.884</td>
<td>2.836</td>
<td></td>
</tr>
<tr>
<td>Scenic spot</td>
<td>16.001</td>
<td>5.029</td>
<td>0.596</td>
<td></td>
</tr>
<tr>
<td>Food</td>
<td>46.044</td>
<td>6.648</td>
<td>1.324</td>
<td></td>
</tr>
<tr>
<td>Teacher</td>
<td>48.244</td>
<td>3.991</td>
<td>1.075</td>
<td></td>
</tr>
<tr>
<td>Politics</td>
<td>59.755</td>
<td>15.673</td>
<td>3.323</td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td>22.676</td>
<td>6.519</td>
<td>1.099</td>
<td></td>
</tr>
</tbody>
</table>

has the least memory overhead and better computing efficiency.

6.3.2. Effect evaluation on the parallel computing of the potential value - $P_{T}(t_i)$

Because the dataset of the target triple is relatively small, in order to avoid the decrease of the parallel efficiency caused by the time consumed to start Spark cluster, this paper starts to calculate the $P_{T}(t_i)$ in parallel after obtaining the $ES_{a_0}$, instead of restarting the cluster. The experiment is carried out by 2 and 4 processes respectively. Table 12 shows the execution time of the $P_{T}(t_i)$ computing, including the speedup according to Eq.(18).

Table 12  
<table>
<thead>
<tr>
<th>Sub-categories of BaiduBaike</th>
<th>Serial time</th>
<th>2 processes</th>
<th>4 processes</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>Competition</td>
<td>1.30</td>
<td>0.79</td>
<td>0.78</td>
<td>1.64</td>
</tr>
<tr>
<td>Landform</td>
<td>2.08</td>
<td>1.25</td>
<td>1.04</td>
<td>1.67</td>
</tr>
<tr>
<td>collection</td>
<td>5.74</td>
<td>3.30</td>
<td>2.15</td>
<td>2.74</td>
</tr>
<tr>
<td>Bank</td>
<td>7.47</td>
<td>4.42</td>
<td>3.38</td>
<td>2.16</td>
</tr>
<tr>
<td>Paintings</td>
<td>18.55</td>
<td>10.62</td>
<td>7.44</td>
<td>2.50</td>
</tr>
<tr>
<td>Architecture</td>
<td>28.57</td>
<td>16.24</td>
<td>11.69</td>
<td>2.44</td>
</tr>
<tr>
<td>Prose</td>
<td>25.07</td>
<td>14.62</td>
<td>11.64</td>
<td>2.15</td>
</tr>
<tr>
<td>Earthquake</td>
<td>4.57</td>
<td>2.60</td>
<td>2.16</td>
<td>2.21</td>
</tr>
<tr>
<td>Constellation</td>
<td>6.92</td>
<td>4.55</td>
<td>3.04</td>
<td>2.27</td>
</tr>
<tr>
<td>Scenic spot</td>
<td>40.56</td>
<td>24.55</td>
<td>17.10</td>
<td>2.37</td>
</tr>
<tr>
<td>Food</td>
<td>34.85</td>
<td>18.21</td>
<td>14.25</td>
<td>2.45</td>
</tr>
<tr>
<td>Teacher</td>
<td>71.20</td>
<td>36.52</td>
<td>27.32</td>
<td>2.51</td>
</tr>
<tr>
<td>Politics</td>
<td>163.01</td>
<td>34.38</td>
<td>25.86</td>
<td>2.44</td>
</tr>
<tr>
<td>Average</td>
<td>20.57</td>
<td>11.60</td>
<td>8.65</td>
<td>2.25</td>
</tr>
</tbody>
</table>

Figure 15 is a Bar-Line comparison graph of execution time-speedup according to Table 12. The results reveal that the execution time of parallel computing is almost the same as that of serial computing when the dataset is small, but with the increase of the datasets by the target triples set, the execution effect of parallel program is significantly enhanced, and the speedup is gradually increased. In addition, the average speedup reached 1.71 in 2 processes, but only 2.25 in 4 processes. It can be seen that the $P_{T}(t_i)$ computing of the target triples set is relatively small, which leads to a large proportion of the inherent overhead of network transmission and other aspects of the system. Therefore, with the increase of the number of processes, the parallel speedup based on Spark will reach its peak quickly (when 4 processes). At the same time, it can be concluded that the Spark will have a better effect when dealing with large-scale datasets.

7. Conclusion and Future Work

In this paper, based on the theory of data field, an algorithm based on tags set potential function is first proposed to denoising the large-scale online encyclopedia KB. In this process, due to the large-scale of the datasets in the KB, it is difficult to accept the execution time by the serial program. Therefore, based on the Spark cluster, this paper proposes the parallel denoising and optimization algorithm for the large-scale
Fig. 15. Comparison of $P_T(t_i)$ calculation

online encyclopedic KB. Experimental results reveal that the parallel denoising algorithm based on Spark is better than the state-of-the-art methods in precision, recall and time efficiency. Finally, we have fully realized the End-to-End parallelism of the serial denoising algorithm. Therefore, the parallel denoising algorithm proposed in this paper has good scalability and can meet the challenges brought by the explosive growth of online encyclopedia knowledge in future.

In the future work, we will consider how to more accurately mine the relations between encyclopedia triples and their semantic tags, and use more reasonable methods such as Word2vec algorithm and Jaccard similarity coefficient to improve the calculation accuracy of semantic similarity, and then continuously improve the accuracy of online encyclopedia KB.

Acknowledgements

This work was supported by the Scientic Research Project of Beijing Municipal Education Commission (General Social Science Project) [grant number SM201910038010].

References

[12] N. Xing, S. Xinruo, W. Haofen, R. Shu, Q. Guijin and Y. Yong, Zhishi.me – weaving Chinese linking open data., in: Interna-


