IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 24, 2019, accepted July 24, 2019, date of publication August 12, 2019, date of current version August 23, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2934747

A Novel Large-scale Chinese Encyclopedia Knowledge Parallel Refining Method Based on MapReduce

TING WANG[1], JIE LI[1], JIALE GUO[1], AND JINGYAO XIE[2]
[1]School of Management and Engineering, Capital University of Economics and Business, Beijing 100070, China
[2]State Grid Beijing Electric Power Company, Beijing 100031, China

Corresponding author: Ting Wang (wangting@cueb.edu.cn)

This work was supported in part by the Scientific Research Project of Beijing Municipal Education Commission (General Social Science Project) under Grant SM201910038010, and in part by the Youth Excellent Teachers Grant of Capital University of Economics and Business under Grant 23491854840429.

ABSTRACT The open collaborative characteristics of online encyclopedia and the large number of ambiguity phenomena in the encyclopedia entry lead to inappropriate classification of plenty of Infobox knowledge triples of entries, which requires for refining and denoising of large-scale knowledge to improve the precision of Knowledge Base (KB). The enormous amount of triples in the KBs will cause excessive serial computing time expenditure by knowledge denoising and disambiguation processing. Existing knowledge refinement and disambiguation techniques have limitations in terms of scalability and time-efficient. There is still few typical research on the parallel processing of knowledge refinement in distributed environment. Therefore, this paper proposes a novel parallel algorithm for Chinese large-scale knowledge refinement based on MapReduce to further improve the overall system computing speed through parallel optimization for serial algorithm. Based on the original serial refining algorithm which can enhance the precision of encyclopedia-oriented KBs, results show that the novel parallel denoising algorithm proposed in this paper can further provide the system with good scalability and high speedup.

INDEX TERMS Knowledge base, knowledge refinement, similarity computing, parallel computing, MapReduce.

I. INTRODUCTION

Knowledge Base (KB) building for machine readable and understandable is the important way to achieve the vision of ''Web of Data'' [1]. As KBs are used for handling semantic query and reasoning on the web, the intelligent application system built on the basis of KBs can provide more accurate and comprehensive information for users.

A. MOTIVATION

Chinese is the native language of the world's largest population. As China's economy develops, Chinese is increasingly being used as a second language in the world. With the development of Semantic Web, large-scale KBs described in Chinese have been constructed and shared on the web [2]. Unfortunately, due to the improper classification of massive Infobox knowledge triples caused by the open-collaborative of online encyclopedia and the plenty of ambiguity between entries, there are a large amount of noise knowledge existing in Chinese online encyclopedia KBs [3], [4]. In order to avoid the precision of KBs from being lowered by noise knowledge, it is believed that refining and denosing the noise triples can provide the KBs with high quality and precision.

Unfortunately, with the continuously increasing of knowledge and RDF triples on the online encyclopedia, the enormous quantity of triples in KB will lead to excessive serial computing time expenditure of algorithms. Dealing with the tremendous amount of triples in the process of refining the KBs on a single machine environment will inevitably lead to the unacceptable time expenditures and non-scalability of system. Therefore, in order to make our system scalable and high efficiency, on the basis of the serial knowledge denosing algorithm, we further improve it to a MapReduce-based parallel algorithm and provide our system with scalability in distributed environment.

The associate editor coordinating the review of this article and approving it for publication was Mansoor Ahmed.

B. CONTRIBUTIONS

The main contributions of this paper are summarized as follows:

i) With BaiduBaike as the target KB for refinement, Hudong as the referenced KB, the initial similarity computing of Chinese knowledge triples is conducted by integrating the Edit-Distance [35] and Chinese TongYiCiCiLin similarity algorithm [14]. The initial similarity is defined as the mass of triple in the encyclopedia knowledge data field.

Because there are both the literal and semantic similarity between the Chinese knowledge triples in Semantic Web environment, so it is one-sided and inaccurate to only adopt one method. Thus, after comparing two algorithm results, the larger value is selected for accumulation to the initial similarity result of the BaiduBaike triple.

ii) Because the enormous amount of triples in KB will lead to excessive serial computing time expenditure brought by initial similarity computing tasks. Therefore, in order to archive the scalability of our system, a MapReduce-based method for parallel optimization of the initial similarity algorithm is proposed in this paper.

For example, while computing the initial similarity value of the Infobox triple of BaiduBaike entry ''Li Na'', the number of Infobox knowledge triple involved in the data field of BaiduBaike ''People'' top-class is about 200,000, and the number of Infobox knowledge triple contained in Hudong ''People'' top-class is about 300,000. If the initial similarity serial computing of full pairing mapping is conducted on this scale, the executing times of similarity algorithm will be up to as $(20 \times 10^4) \times (30 \times 10^4) = 6 \times 10^{10}$ times (about 6 billion times). Through preliminary testing, such scale requires for the continuous operation of an ordinary PC (2.4GHz 4-cores CPU, 4G memory) for about 264 hours. The time expenditure is obviously unacceptable.

iii) In order to further improve the precision of KB, an Infobox knowledge triple-constructed nuclear field-like potential function is involved by means of initial similarity value and semantic distance between tags of entry to compute the target similarity value of the triples.

Specifically, we introduce a piecewise function based on semantic distance between triple's tags and embed it into the nuclear field-like potential function. The piecewise function is used for punishing the improper classified triples so as to optimize and decrease its initial similarity value in its triples set. Finally, the processes of re-ranking and deleting of lower-ranked knowledge triples are carried out based on the target similarity value for the purpose of refining a large number of improper classification and ambiguity of triples in Chinese encyclopedia KBs.

II. RELATED WORK

With the development of the Internet and the advent of online encyclopedia such as Wikipedia[1], BaiduBaike[2] and Hudong[3], knowledge creations on the web are increasingly emerging in open and collaborative patterns. The prosperous development of online encyclopedia provides a good data source for knowledge discovery in open-domain. It is available to build the open-domain KBs based on multi-domain knowledge published on the online encyclopedia. The wiki systems support for automatic construction of large-scale KB, as it is available to support semantic query and reasoning by RDF triples graph. Hence, the automatic building of KBs based on online encyclopedia and online encyclopedia-oriented knowledge discovery are attracting more and more attentions from researchers.

Some typical works, such as: YAGO [5], DBpedia [6] and Freebase are regarded as extremely representative large-scale semantic KBs built on the basis of Wikipedia system and Semantic Web technology specifications, they extract knowledge from structured information of the wiki page through specific wikis system middleware. The hierarchical relation in open classification system of encyclopedia is confirmed as the ''is-a'' relationship between concepts. At the same time, the Infobox on the entry web page contains enormous knowledge triples. DBpedia is jointly developed by the Free University of Berlin and the University of Leipzig which can automatically extract, transform and generate semantic data from a dataset released on the Internet (e.g., Wikipedia) and establish a semantic KB, add statements of facts and properties of relations in an open way, thus far exceeding the traditional models of manually built in efficiency and scale. Kylin [7], [8] system develop by Fei Wu and S. Weld et al. not only concentrates on the structured information appearing on wiki encyclopedia entry page but also tries to extract knowledge triples from unstructured texts in Wikipedia. DBpedia has been proved as a successful structured KB and has linked semantic data in many different domains as the Hub for Linked Open Data (LOD) [9], thus forming a grand-scale LOD Web.

Most of the existing semantic datasets are based on English expression, while the research and release of Chinese KBs and LOD are still at the starting stage. Google has released retrieval service for Freebase, and officially proposed the concept of ''Knowledge Graph'' (KG) in 2012.

With the advent of the era of Big Data and KG, different kinds of knowledge have emerged as a flood. With the extremely fast growth of the number of entries in Chinese encyclopedia and the scale of KB, plenty of relevant research results have also come into being. Chen *et al.* [10] propose to automatically extract well-formed training samples by using the property-value pair information in Chinese encyclopedia Infobox, thus extracting large-scale knowledge triples from unstructured texts in encyclopedia based on statistical learning model. Twelve effective features of Chinese Wikipedia tag are designed from the perspectives of lexical, grammatical and structural aspects to predict and extract ''is-a'' relationship by Li *et al.* [11], in which the Skip-gram model is

[1]http://www.wikipedia.org
[2]http://baike.baidu.com/
[3]http://www.hudong.com/

used for training word embedding, mining and description of semantic relationships in the plain-text. The ''is-a'' relation inference method based on language pattern, heuristic rules and association rules mining is proposed. In the end, a large-scale Chinese classification system construction algorithm is proposed to design and implement the Chinese classification system query system: CTCS2, so as to meet the requirements of semantic query. Wang *et al.* [4] propose a new approach of automatic building for domain ontology based on machine learning algorithm, and by which the large-scale e-Gov ontology is built automatically. In the first stage, rough mapping from thesaurus to ontology is carried out, and domain rough ontology is formed according to the conversion rules. In the second stage, the structured knowledge and rough ontology in open encyclopedia are automatically merged and expanded to form the domain ontology with rich semantic information. As mentioned above, the existing KB building methods based on semi-automatically or automatically extract knowledge mainly depend on the semi-structured and structured information published on the web. However, the efficiency and precision of KBs still need to be further improved and optimized.

In the past few years, research work on Chinese LOD based on Chinese large-scale KB has been carried out rapidly. Wang *et al.* [3] propose to extract the hierarchical relation between concepts based on classification system of Chinese encyclopedia, to obtain concept properties and encyclopedia entries on web pages containing Infobox. Finally, a Chinese encyclopedia KB has been built and a co-reference relationship with DBpedia based on simple keywords matching strategy has been established. Niu *et al.* [12] conduct semantic integration of BaiduBaike, Hudong and Chinese Wikipedia[4], and developed an instance-level Chinese LOD application system. Wang *et al.* [13] propose a multi-source KB entity alignment algorithm based on online semantic tag, which can align Chinese encyclopedia entities by comprehensive usage of attribute tags, category tags and unstructured text keywords. After being tested in experiments, the algorithm is proved to solve the multi-source KB entity alignment problem, and can maintain a good recall of up to 55% under the precision of up to 95%. It can satisfy the demands of KB entity alignment after being applied to practical system. Wang *et al.* [2] propose a Schema-Level oriented large-scale Chinese LOD building model which firstly simplified and compressed the large-scale Chinese ontology mapping scale based on quasi-nuclear force potential function with improved fusion concept similarity and dissimilarity. Secondly, similarity measurement has been conducted on combined concepts by introducing sequence alignment algorithm [14].

Furthermore, KB acan also provide ancillary support for research work in relevant fields. For example, massive named entities in KB and relationship between them can be used as large-scale knowledge corpus in natural language processing (NLP) field. In addition, research work on classical problems such as semantic relevance measurement, word sense disambiguation, and co-reference relationship recognition between entities have been widely carried out based on this [15]. Microsoft and Google have detected its potential value, acquired PowerSet (a Wikipedia-based natural language search engine) and MetaWeb respectively, and improved their own search engine based on the acquired technology. Richard [16] propose a KB management system according to the fact that some existing intelligent support systems cannot provide sufficient verification tests, thus providing a complete life cycle environment for the development and verification of business rules and expert systems.

Nowadays, towards the development of new technologies, there is a trend in different areas of Web 3.0 that allow analyzing very large-scale of datasets efficiently. In recent years, in order to guarantee the scalability of systems, parallel computing technology (e.g., MapReduce framework) has been widely applied to some typical areas in Semantic Web based on distributed environment, such as: large-scale RDF triples store, inference, query, change detection, and LOD.

Rohloff and Schantz [17] design SHARD system based on Hadoop, which stores RDF in HDFS and supports the storage of large-scale semantic data. In order to improve the efficiency of large-scale semantic data query, all queries is converted to MapReduce jobs, which can ensure the fault tolerance and scalability of the system. While querying the RDF graph, the minimum decomposition unit of HadoopRDF [18] is also the triple pattern, and it also uses MapReduce to divide RDF triples into multiple small files based on predicates. S2X [19] firstly matches all triple patterns in the query, then gradually discards some intermediate results by iteration calculating, and finally joins the remaining matched results. Virtuoso [20] processes RDF graph queries based on relational databases, transforms SPARQL into SQL, and obtains matching results through join operations of RDB tables. S2RDF [21] introduce a relational partitioning schema for RDF data called ExtVP that uses a semi-join based preprocessing, akin to the concept of Join Indices in relational databases, to efficiently minimize query input size regardless of its pattern shape and diameter based on Spark [22]. Peng *et al.* [23] use a distributed framework to perform SPARQL queries based on gStore. In the local computing stage, each node in the cluster matches the complete query, and then in the assemble stage, a large number of local matching results are sent to the central node for join operations so as to get the final results. Shao *et al.* [24] proposes RDF graph storage system based on MPI [25]: Trinity, which is totally based on distributed memory. Trinity.RDF [26] store RDF data in its native graph form, so it can support random walks and reachability on RDF graphs as well. TriAD [27] use asynchronous MPI to execute SPARQL, which establishes 6 indexes of SPO. RDF triples are partitioned into these indexes, and a locality-based RDF graph summarization is used to speed up queries. The Jena-HBase system [28] is developed by using the Jena interface to meet the needs of

[4]http://zh.wikipedia.org/

large-scale RDF data processing. The system uses HBase to manage and solve the storage problem of large-scale RDF data. Wang *et al.* [29] proposed a query processor based on MapReduce framework: SDec, which has high query efficiency for SPARQL BGP on large-scale RDF graph; in addition, it further improves the efficiency of query algorithm by postponing Cartesian product operation.

Ahn *et al.* [30] propose G-Diff, a Map key grouping algorithm based on MapReduce for RDF change detection. It groups triples by URIs during Map phase and sends the triples to a particular Reduce task rather than multiple Reduce tasks. Also based on distributed environment and MapReduce framework, Lee *et al.* [31] propose a similarity-based RDF change detection system which can match blank nodes and produce smaller delta results.

Based on virtual documents and MapReduce techniques, Zhang *et al.* [32] propose a 3-stage approach which significantly reduces the run time of ontology matching process. They use a word-weight-based partition method to calculate similarities between entities in the reducers. VDoc+ [33] is an improved word-weight based partitioning method based on Ref. [32], so as to make it more flexible to fit different preferences on efficiency and effectiveness. Torre-Bastida *et al.* [34] implement the Map-Reduce processing framework and context-based ontology matching techniques so as to discover the maximum number of possible relationships between entities within different data sources in an computationally efficient fashion.

Unfortunately, to the best of our knowledge, existing knowledge refinement and disambiguation techniques have limitations in terms of scalability and time-efficient. There is still few typical research on the parallel processing method of encyclopedia knowledge refinement and disambiguation in distributed environment.

III. BACKGROUND

A. KNOWLEDGE TRIPLES: INFOBOX PARSERING FROM THE STRUCTURED-INFORMATION ON ENTRY'S WEB PAGE

The 11 top-classes in BaiduBaike and Hudong refer to the classification tree of both of them, namely: People, Sports, Life, Culture, Science, Economy, History, Society, Geography, Nature, and Art. According to the classification system of encyclopedia, each entry can belong to one or more classification concepts(classes). Obviously, these entries are instances of the classification concept.

The structured-information of each entry appearing on entry's web page is called as: Infobox, which can be directly parsed into a set of knowledge triples, that is to say, all triples in this set belong to the instance of entry.

The triple defined in this paper is constituted by subject, predicate and object, respectively. Knowledge triple is denoted by $< S, P, O >$, in which subject, predicate and object are denoted as S, as P, and O, separately. This knowledge acquisition process can be automated by parsing the entry's HTML pages of Encyclopedia.

For example, the entry card of entry: ''Qian Xuesen'' belongs to unstructured-information, where contains a factual statement in its plain-text description: ''Qian Xuesen (December 11, 1911 - October 31, 2009), was born in Shanghai, and his native place is Hangzhou, Zhejiang Province.'' The knowledge contained in it: ''Born in Shanghai'' is expressed in the Infobox on this entry's page as a record: ''Birthplace: Shanghai''. Therefore, the structured-information appearing in the Infobox can be directly extracted and transformed into triple form: < Qian Xuesen, birthplace, Shanghai>. The construction process of Chinese encyclopedia KBs is shown in Fig.1 [4].

B. THE SEMANTIC TAGS OF TRIPLES

The semantic tags discussed in this paper are divided into two parts:

i) The one part is the encyclopedia entries' tags labeled by its open collaborative editors. Especially in the BaiduBaike, there are two kinds of tags for entries: classification tag and property tag. These two kinds of tags are not distinguished on the web page of entries. Therefore, entry's tags of BaiduBaike can be used to describe both the classification and the properties of entries.

ii) The other part is the misclassified-tags labeled by previous work because of the ambiguity caused by a large number of encyclopedia entries which have the same name but different meanings. These tags are named as ambiguous tag. The ambiguous categorization of entries will eventually result in all Infobox triples contained in the entries being labeled with the ambiguous tags.

In this paper, tags in the two parts mentioned above are collectively called as semantic tag. They form a set T of semantic tags, in which each tag appearing in classification tree of Chinese encyclopedias. There are two possible cases where noise triples would be generated. Below we will take the BaiduBaike's entry ''Verona'' for example to explain:

i) Incorrect categorization may occur when entries are categorized according to entry's tags, that is: if an entry contains a tag describing its properties, then all the triples of this entry will be inappropriately classified under the concept of the classification tree corresponding to the property tag.

For example, the ''Sports'' tag is used to characterize the properties of entry: ''Verona''(a famous football club in Italy) rather than to declare the classification to which it belongs.

ii) The inappropriate categorization of these ambiguity entries will result in the ambiguous categorization of all the Infobox knowledge triples contained in the entry. Besides, because of open collaborative editing mode of online encyclopedia, it should be noted that the encyclopedia tags of the first part of the entry itself may also contain some improperly categorized tags by the entry's editors, which will also lead to inappropriate categorization of the triples.

For example, the entry ''Verona'' has five homonymous terms, these ambiguity entries belong to five different Encyclopedia sub-classes, i.e. Scenic Spots, Geography, Italy, Football and Sports. Therefore, when the previous work

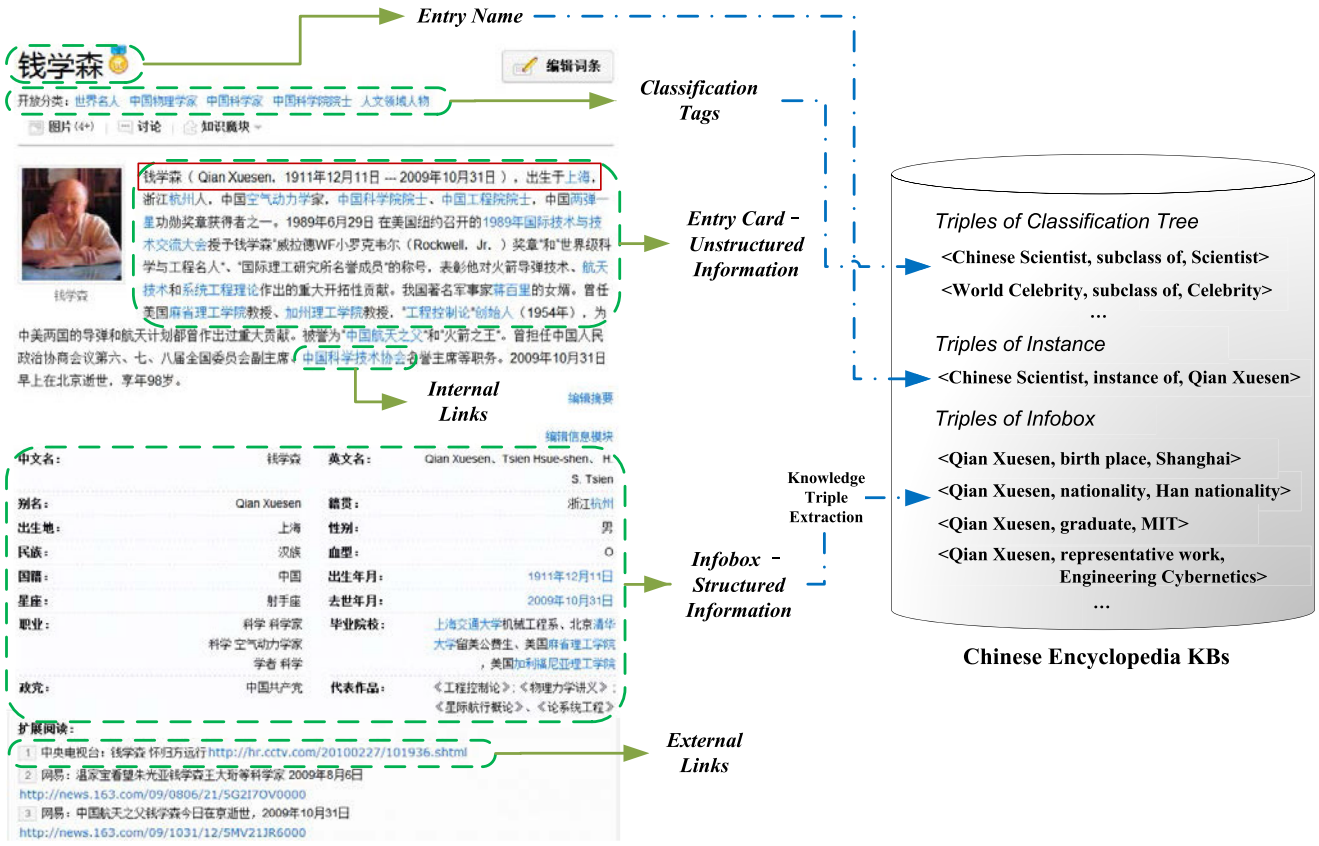

FIGURE 1. The construction process of Chinese encyclopedia KBs.

system classifies the entry, it will cause the ambiguity of the entry itself, which will lead to all the ambiguity entries being classified to every sub-class in the set T of semantic tags. That is, causing the following results:

Verona(Scenic Spots) → Scenic Spots, Geography, Italy, Football, Sports

Verona(Geography) → Scenic Spots, Geography, Italy, Football, Sports

Verona(Italy) → Scenic Spots, Geography, Italy, Football, Sports

Verona(Football) → Scenic Spots, Geography, Italy, Football, Sports

Verona(Sports) → Scenic Spots, Geography, Italy, Football, Sports

The word in brackets indicates the correct classification of each entry with the same name. ''→'' means ''be classified''.

All of the problems mentioned above will bring a large number of noise and error information to the large-scale open-domain KB, and finally degrade KB's precision.

It should be noted that all semantic tags of triples do not necessarily all exist in the encyclopedia classification tree. In this paper, we do not consider the tags do not exist in the encyclopedia classification tree.

The method proposed in this paper tries to solve the problems mentioned above, namely the noise challenge in Chinese KB. We label all the classification concepts in set T of an entry in the form of semantic tags (including correct or incorrect classification, property, and ambiguous tags) onto each triple of this entry.

IV. OVERALL DESIGN

In general, the whole triple refinement process is systematically divided into the following two stages:

i) First Stage: Initial similarity computing of triples by using multi-strategy integrated similarity algorithm combining with cross-language universal Edit-Distance with Chinese TongYiCiCiLin.

In this paper, we mainly parallel the process of the first stage based on MapReduce.

ii) Second Stage: Initial similarity correction by using the triple-constructed nuclear field-like potential function to compute the target similarity of triple for the final purpose of knowledge denoising and refinement.

A. FIRST STAGE: SERIAL ALGORITHM OF THE INITIAL SIMILARITY COMPUTING OF INFOBOX TRIPLES

The initial similarity computing in this paper is conducted between triple sets in BaiduBaike sub-class and Hudong top-class. For example, for initial similarity computing of BaiduBaike sub-class ''Currency'', the triple initial similarity computing together with Hudong top-class ''Economy'' is required as ''currency'' sub-class belongs to ''Economy''

TABLE 1. Example of a TongYiCiCiLin code of "China".

Code Bit	1	2	3	4	5	6	7	8
Sub-Code	D	i	0	2	A	0	3	= or # or @
Meaning	Broad heading	Middle heading	Small heading		Word group	Atomic word group		Synonymous /unequal /isolated
Layer	1	2	3		4	5		

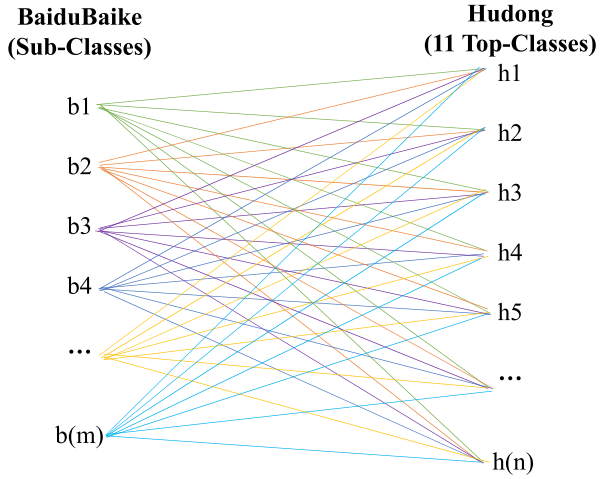

FIGURE 2. The mapping relations for initial similarity computing.

top-class in BaiduBaike. Therefore, mapping of all the triples in the "Economy" top-class in Hudong is required for initial similarity computing and accumulation.

The specific algorithm is shown as follows:

Suppose b_i, h_j as any triple in the triple set B of a BaiduBaike sub-class and any triple in the triple set H of Hudong top-class. B and H contain m and n knowledge triples, respectively.

Suppose $< b_s, b_p, b_o >$, $< h_s, h_p, h_o >$ as knowledge triples of BaiduBaike and Hudong in the form of $< S, P, O >$ respectively. The mapping computing relations are shown in Fig.2.

Thus, the method of computing the initial similarity S_b_i of triple b_i is expressed in Eq. (1):

$$S\_b_i = \sum_{j=1}^{n} \mathrm{SIM}\left(b_i \cdot h_j\right) = b_i \cdot h_1 + b_i \cdot h_2 + \cdots + b_i \cdot h_n \quad (1)$$

Specifically, compute the two similarity values based on the Edit-Distance algorithm and TongYiCiCiLin algorithm at the same time of initial similarity computing; conduct complementary integration of the similarity value of the Edit-Distance and the similarity value of TongYiCiCiLin according to the preset method to finally obtain first-stage initial similarity of the triple.

1) EDIT-DISTANCE SIMILARITY COMPUTING

Suppose any triple in the triple set B of a BaiduBaike sub-class as $b_i =< b_{is}, b_{ip}, b_{io} >$, any triple in set H of Hudong top-class corresponding to set B as $h_j =< h_{js}, h_{jp}, h_{jo} >$, the Edit-Distance similarity between the subjects in the triples b_i and h_j can be given by Eq.(2). The similarity value between predicates and objects can be obtained in a similar way.

$$\mathrm{SIM_E}\left(b_{is}, h_{js}\right) = \frac{1}{1 + \frac{|\mathrm{STEP}(b_{is}, h_{js})|}{\max(\mathrm{len}(b_{is}), \mathrm{len}(h_{js}))}} \quad (2)$$

where, $|\ \mathrm{STEP}(b_{is}, h_{js})|$ is the required edit-operation times to make b_{is} and h_{js} equal. The length of characters b_{is} and h_{js} can be denoted as $\mathrm{len}(b_{is})$ and $\mathrm{len}(h_{js})$.

2) TongYiCiCiLin SIMILARITY COMPUTING

"TongYiCiCiLin" edited by Mei [36] in 1983 is a Chinese synonym dictionary with the original aim at providing more synonymous expressions and translation work. This dictionary contains not only the synonym of an entry but also a certain number of entries of the same kind, namely, related entries in a broad sense. In the Chinese synonym dictionary TongYiCiCiLin, each vocabulary after coding is organized in a tree structure in a hierarchical relation with five layers from top to bottom. There are code identifiers in corresponding levels respectively, which are arranged from left to right to constitute CiLin code of lexical unit. Each node in the tree represents a concept, and the connotative semantic correlation between entries will increase with the increase of layer. The Chinese words co-reference relationship identification can actually be abstracted as the issue of identifying Chinese synonyms. We actually adopted the extended version in this invention, namely: HIT TongYiCiCiLin (extended)[5], as the dictionary lexicon for the TongYiCiCiLin similarity computing [14]. It is the largest dictionary of Chinese synonyms at present. It contains the largest number of Chinese synonyms.

Taking the lexical unit "中国(China)" as an example (TongYiCiCiLin code: "Di02A03="), its TongYiCiCiLin code format as indicated in Table 1.

Firstly, TongYiCiCiLin code parsing of subject, predicate and object in the triple according to the structural characteristics of TongYiCiCiLin for sub-code extraction from the first to the fifth layer and comparison from the sub-code in the first layer. In case of different sub-codes, give the corresponding similarity weight of the mapping according to the appeared layer. The deeper layer of the sub-codes appear, the higher the similarity weight, otherwise the lower. The semantic similarity between the triples can be obtained by the TongYiCiCiLin similarity algorithm: $\mathrm{SIM_T}$. In this case, $\mathrm{SIM_T}$ is adopted to

[5]http://ir.hit.edu.cn/demo/ltp/Sharing_Plan.htm

explain with the similarity computing between the subjects in two triples as an example. The similarity computing between the predicates and the objects can be obtained in a similar way, as shown in Eq. (3):

$$\mathrm{SIM_T}\left(b_{is}, h_{js}\right) = \lambda \times \frac{L_n}{|L|} \times cos\left(N_T \times \frac{\pi}{180}\right) \times \left(\frac{N_T - D + 1}{N_T}\right) \quad (3)$$

To adjust the parameter semantic correlation factors, thus controlling the possible similarity degree of lexical units in branches at different layers, $\lambda \in (0,1)$. When λ is set to 0.9, our method can achieve the best overall performance. Because TongYiCiCiLin tree consists of 5 layers altogether, so $L = \{1, 2, 3, 4, 5\}$. $|L|$ is the number of elements in the set L and equals to 5 in this system. Setting $\forall L_n \in L$, L_n is the n-th layer's number where different sub-codes appearing between b_{is} and h_{js}. N_T is the total number of nodes of b_{is} and h_{js} on the branch of the n-th layer. D is the code-distance at b_{is} and h_{js}'s branch.

3) INTERLINKINGVALUE - THE TRIPLE INITIAL SIMILARITY ALGORITHM

Considering the semantic complementarity of $\mathrm{SIM_E}$ and $\mathrm{SIM_T}$ algorithms, complementary integration of the similarity results of these two algorithms is conducted. The maximum value is selected:

$$b_{is} \cdot h_{js} = \max\left(\mathrm{SIM_E}\left(b_{is}, h_{js}\right), \mathrm{SIM_T}\left(b_{is}, h_{js}\right)\right) \quad (4)$$

Here, the similarity computing method for $b_{ip} \cdot h_{jp}$ and $b_{io} \cdot h_{jo}$ can be obtained in a similar way.

InterlinkingValue: The initial similarity algorithm of any triple b_i in the triple set B of BaiduBaike sub-class is shown in Eq.(5):

$$S\_b_i = \sum_{j=1}^{n} \begin{pmatrix} 0.3 \times \max\left(\mathrm{SIM_E}\left(b_{is}, h_{js}\right), \mathrm{SIM_T}\left(b_{is}, h_{js}\right)\right) \\ +0.5 \times \max\left(\mathrm{SIM_E}\left(b_{ip}, h_{jp}\right), \mathrm{SIM_T}\left(b_{ip}, h_{jp}\right)\right) \\ +0.2 \times \max\left(\mathrm{SIM_E}\left(b_{io}, h_{jo}\right), \mathrm{SIM_T}\left(b_{io}, h_{jo}\right)\right) \end{pmatrix} \quad (5)$$

Among them, 0.3, 0.5 and 0.2 represent the weight coefficients of the subject similarity, predicate similarity and object similarity during the initial similarity computing of the whole triple b_i, which can be adjusted according to the target effect.

The knowledge triple initial similarity corresponding to triple set B of a BaiduBaike sub-class can be obtained after calculating.

B. PARALLEL OPTIMIZATION FOR INTERLINKINGVALUE ALGORITHM

1) DATA DEPENDENCE ANALYSIS - PROVE THE CORRECTNESS OF SERIAL ALGORITHM PARALLELIZATION

Before starting the parallel optimization of serial algorithm, we firstly investigated and analyzed data dependence relationship of the algorithm in the executing process, then used it as the theoretical basis for the parallel optimization of InterlinkingValue algorithm.

$b_{1s}, b_{1p}, b_{1o}, h_{1s}, h_{1p}, h_{1o}$
$b_{1s}, b_{1p}, b_{1o}, h_{2s}, h_{2p}, h_{2o}$
……
$b_{1s}, b_{1p}, b_{1o}, h_{ns}, h_{np}, h_{no}$
$b_{2s}, b_{2p}, b_{2o}, h_{1s}, h_{1p}, h_{1o}$
$b_{2s}, b_{2p}, b_{2o}, h_{2s}, h_{2p}, h_{2o}$
……
$b_{2s}, b_{2p}, b_{2o}, h_{ns}, h_{np}, h_{no}$
……
$b_{is}, b_{ip}, b_{io}, h_{1s}, h_{1p}, h_{1o}$
$b_{is}, b_{ip}, b_{io}, h_{2s}, h_{2p}, h_{2o}$
……
$b_{is}, b_{ip}, b_{io}, h_{js}, h_{jp}, h_{jo}$
……
$b_{ms}, b_{mp}, b_{mo}, h_{ns}, h_{np}, h_{no}$

FIGURE 3. Description of six-tuple file format.

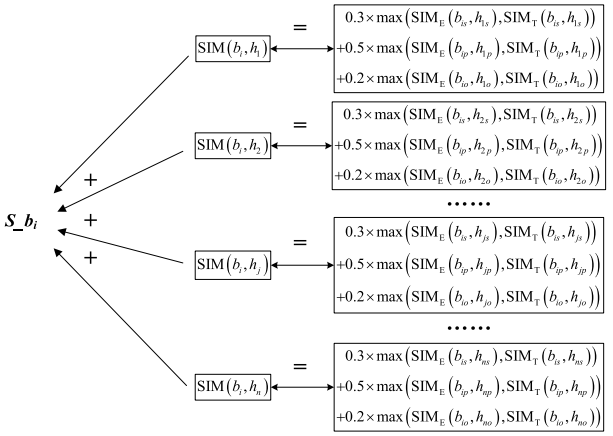

FIGURE 4. Data dependence analysis diagram.

To fully use the function of HDFS Architecture for Big Data storage and the parallel computing model of MapReduce by-line read-in, we conduct merge processing of the pending knowledge sets according to the InterlinkingValue algorithm. Merge the corresponding two knowledge sets into a dataset of six-tuple in the form of $< b_s, b_p, b_o, h_s, h_p, h_o >$ according to the knowledge triple mapping computing relationship as shown in Fig.2. Among all, the triple of BaiduBaike knowledge concentration triples is in the first three columns of each line. For example, the triples set of BaiduBaike sub-class "Currency" can be merged with the triples set of Hudong top-class "Economy". Thus, the finally obtained input data concentration in the form of six-tuple according to the InterlinkingValue algorithm should contain six-tuples totally in $m \times n$ lines, as shown in Fig.3:

Verify the correctness of the parallelization process of serial algorithm according to Eq. (5). The data dependence analysis results are shown in Fig.4.

It can be seen that there exists a dependence relationship between S_b_i and SIM(b_i, h_1), …, SIM(b_i, h_j), …, SIM(b_i, h_n). Thus, S_b_i cannot be executed with SIM(b_i, h_1), …, SIM(b_i, h_j), …, SIM(b_i, h_n) for parallel operation, namely, serial execution can only be performed between S_b_i

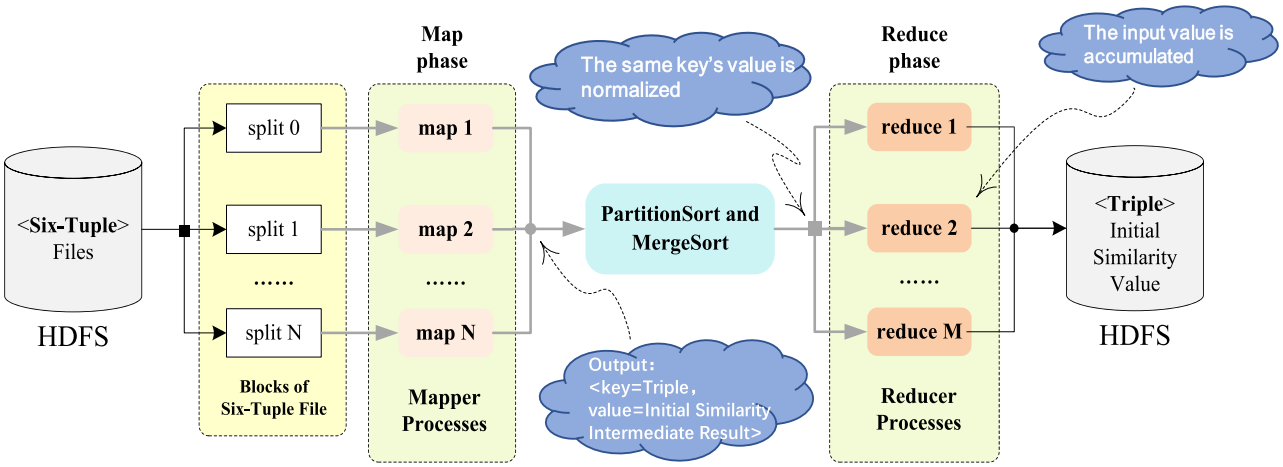

FIGURE 5. Parallel initial similarity computing based on MapReduce framework.

and SIM(b_i, h_1), …, SIM(b_i, h_j), …, SIM(b_i, h_n) as the value of S_b_i can only be computed after computing the value of SIM(b_i, h_1), …, SIM(b_i, h_j), …, SIM(b_i, h_n). In other words, S_b_i depends on SIM(b_i, h_1), …, SIM(b_i, h_n).

However, there exists no dependence relationship between SIM(b_i, h_1), …, SIM(b_i, h_j), …, SIM(b_i, h_n). Thus, parallel computing can be realized, and SIM(b_i, h_1), …, SIM(b_i, h_j), …, SIM(b_i, h_n) corresponds to each line in any six-tuple file. In other words, there exists no data dependence relationship between any random two lines. Thus, it is available for segmentation of datasets between any random lines to obtain blocks for next-step parallel processing of each split, thus finally obtaining the operating results completely consistent with InterlinkingValue serial algorithm.

We confirm by proving that there exists no data dependence relationship between the six-tuples participating in the initial similarity algorithm operation, which serves as a theoretical basis for us to carry out the parallelization transformation of the original serial algorithm. Parallel executed programs will automatically conduct Mergesort operation at the Shuffle stage on the Reduce phase in the computing-intensive environment. This algorithm is available for parallel optimization of InterlinkingValue serial algorithm at the first stage based on MapReduce [40] architecture.

2) PARALLEL IMPLEMENTATION OF INTERLINKINGVALUE ALGORITHM ON FIRST STAGE BASED ON MAPREDUCE

Upload files to HDFS, read each six-tuple successively by using the characteristic of by-line read-in at the Map phase to implement *map*() method.

It only requires to implement *reduce*() method and accumulate the output values at the Map side to further obtain the initial similarity value of the triple. Finally, the output value at the Map side is used as the input value at the Reduce side, and the accumulated values are saved in HDFS as the output value at the Reduce side, as shown in Fig.5.

To sum up, Parallel_InterlinkingValue – pseudo code of the initial similarity parallel algorithm based on MapReduce is shown as follows:

Parallel_InterlinkingValue(B_H)

Map method:

Input: six-tuple file B_H merging triple set B of a BaiduBaike sub-class and the corresponding triple set H of Hudong top-class

Output: triple of triple set B of a BaiduBaike sub-class and its initial similarity value

1. Obtain the six-tuple sets in a file and convert it into the file Split for the input Map process
2. *for each* line$_j$ in six-tuple set B_H
// Call the Map method once for each line: *map*(*key*, *value*)
// Parsing the six-tuple on line j
3. b_{is}, b_{ip}, b_{io}, h_{js}, h_{jp}, h_{jo} $\leftarrow$ *value*$_j$.split(" \t")
// Generate the new key
4. $b_i \leftarrow b_{is} + b_{ip} + b_{io}$
5. $\text{SIM}\_b_ih_j$
$\leftarrow 0.3 \times \max(\text{SIM}_\text{E}(b_{is}, h_{js}), \text{SIM}_\text{T}(b_{is}, h_{js}))$
$+0.5 \times \max(\text{SIM}_\text{E}(b_{ip}, h_{jp}), \text{SIM}_\text{T}(b_{ip}, h_{jp}))$
$+0.2 \times \max(\text{SIM}_\text{E}(b_{io}, h_{jo}), \text{SIM}_\text{T}(b_{io}, h_{jo}))$
6. *out*(b_i, SIM$_b_ih_j$)
7. end *map* // The Map method ends
8. end *for*

C. ADD SEMANTIC DISTANCE TO TAGS OF ENTRY'S INFOBOX KNOWLEDGE TRIPLES

Add a semantic distance to all the entry's tags for each knowledge triple. If the tag and the sub-class of the triplet b_i belong to the same concept, it is a center tag with a distance value of 0. We calculate the semantic distance between tags is the distance of other tags from the center of the circle in the same set of concept sets. We stipulate that if the current tag belongs to the sub-tree of the current top-class, the current tag must be the parent-class or sub-class concept of the center tag. If the current tag is the direct parent or sub-class concept of the center tag, the semantic distance is 1. Because the maximum depth of classification tree of BaiduBaike and Hudong are

Reduce method:
1. *for each* line$_i$
// Call the method of reduce once for each line in the intermediate result set: *reduce* (*key*, *values*)
2. $S\_b_i \leftarrow 0$
3. *for each* S_value in *values*
4. $S\_b_i \leftarrow S\_b_i + S\_value$
5. end *for*
6. *out*(key, S_b_i)
7. end *reduce*
8. end *for*

both equal to 2, so the maximum semantic distance between tags is 6.

For example, a triple <Suzhou River, Chinese name, Suzhou River> belongs to the class: ''River'', because the entry: ''Suzhou River'' has a tag: ''River'', but it also has an ambiguous tag: ''Song''. ''Geography'' is a top-class and also the direct parent-class of ''River'', so the semantic distance between ''Geography'' and ''River'' is 1. Class: ''Leisure'' is sub-class of the top-class: ''Life'', while ''Song'' is sub-class of ''Leisure'', so the semantic path between ''River'' and ''Song'' can be expressed as:

River → Geography →*ROOT*→ Life → Leisure → Song

So the distance between tags: ''Song'' and ''River'' is 5.

D. A METHOD OF TARGET SIMILARITY COMPUTING OF INFOBOX KNOWLEDGE TRIPLES BASED ON THE IMPROVED DATA FIELD

It should be noted that the theory of data field is proposed based on the field theory in physics [37]. It is a description method of finalizing into field theory by abstracting the mutual relation between data in the number domain space as the issue of mutual effect between material particles. The theory which expresses the interaction relationship between different data through the potential function can manifest the distribution characteristics of data, conduct clustering partition of dataset according to the equipotential line structure in the data field.

The triples' ''potential function'' is introduced while computing the triple target similarity computing.

Assume f as the data field produced by data in D, and the function $f_X(Y)$ as its potential function. Where, $X \in D$, $Y \in \Omega$. It indicates the potential value generated by the data element X at Y. $f_X(Y)$ must satisfy the following conditions:

(1) $f_X(Y)$ is a continuous, smooth, bounded function;
(2) $f_X(Y)$ has isotropy;
(3) $f_X(Y)$ is a decreasing function about distance $||X\text{-}Y||$.
As $||X\text{-}Y|| = 0, f_X(Y)$ gets the maximum value.
As $||X\text{-}Y|| \to \infty, f_X(Y) \to 0$.

In this paper, a commonly used potential function is enumerated as following.

Nuclear field-like potential function:

$$f_X(Y) = m \times e^{-\left(\frac{\|X-Y\|}{\sigma}\right)^k} \tag{6}$$

where, $m \geq 0$ represents the influence intensity of X on Y, which can be interpreted as the mass of X. $\sigma \geq 0$ is called the influence factor, which determines the scope of influence of the element. The potential function value increases with the increase of σ. In this paper, we set $\sigma = 10$, $k = 2$ in order to make the semantic distance have a greater impact on the target similarity value.

In the following description, we assume the tag set of a certain triple b_i as T, then $T = (t_1, t_2, t_3 \ldots\ldots t_n)$, $n > 0$ represents the number of tag, and t_i represents its center tag. It should be noted that, as common-sense knowledge, the entry's classification tags mentioned in the invention all belong to the concept set in BaiduBaike open classification system. In this paper, the shortest path length (semantic distance) between two tags is $d = ||t_i\text{-}t_j||$. Thus, field majority function expression of the interaction between the center tag t_i of the triple b_i and the other tag t_j is:

$$f_{t_j}(b_i) = S\_b_i(t_j) \times e^{-\left(\frac{\|t_i - t_j\|}{10}\right)^2} \tag{7}$$

where S_b_i represents the initial similarity of the triple b_i associated with the sub-class concept t_j subject to different top-classes corresponding to the tag.

Tags not belonging to the current top-class are punished in this paper for the purpose of weakening the initial similarity of triples containing tags with far semantic distance for secondary ranking based on the target similarity. The target triples ranking behind will be removed from the KB. Therefore, the field majority computing based on the tags carried by the triple is required to obtain target similarity, which is the result of the initial similarity correction. The improved and optimized piecewise function (8) for punishing the improper classified triples is shown as follows:

$$\varphi_{t_j}(b_i) = \begin{cases} +f_{t_j}(b_i), & d \in (0, 3] \\ -f_{t_j}(b_i), & d \in (3, 6] \end{cases} \tag{8}$$

Namely, tag t_j belongs to the top-class of the center tag t_i.

Namely, tag t_j does not belong to the top-class of the center tag t_i.

Where, $\varphi_{t_j}(b_i)$ is the piecewise function we built. The plus-minus sign represents whether the current tag t_j and the center tag t_i belong to the same top-class. If yes, it indicates that the tag has a positive acting force on the triple b_i; if no, it indicates that the tag has an opposite acting force on the triple b_i. The final BaiduBaike triple's target similarity computing is shown in Eq. (9):

$$F\_b_i = S\_b_i(t_i) + \sum_{j=1}^{n} \varphi_{t_j}(b_i) \tag{9}$$

After the sorting of the initial and the target similarity set of the triples processed by the data field according to the similarity from large to small, it can be seen that this method has good effect on rank descending of incorrect triples, thus more conductive to KB denoising and refinement.

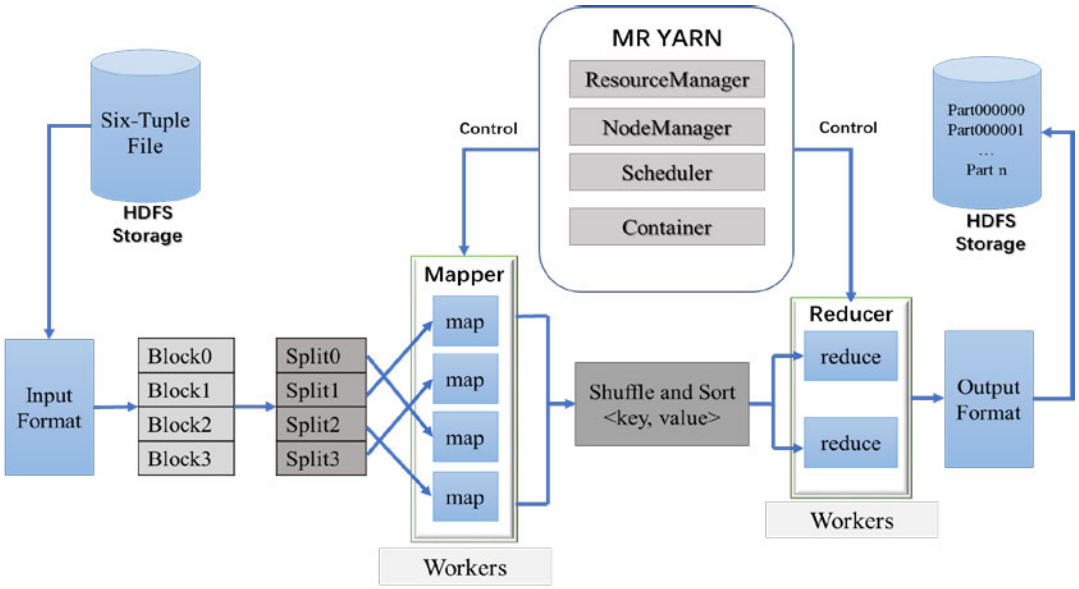

FIGURE 6. HDFS-MapReduce Architecture.

E. KNOWLEDGE REFINEMENT ACCORDING TO THE TARGET SIMILARITY OF INFOBOX KNOWLEDGE TRIPLE

After the improved data field algorithm processing of the initial similarity, descending sorting according to the target similarity value is performed. For the triple target similarity set of a sub-class, the latter 41% of triples are selected according to the idea of golden section point to obtain the final denoised and refined knowledge set.

V. EXPERIMENTAL ENVIRONMENT - HADOOP BIG DATA PROCESSING PLATFORM

A. OVERALL ARCHITECTURE

The parallel execution architecture of InterlinkingValue algorithm based on HDFS and MapReduce model is shown in Fig.6:

Hadoop [38] is a distributed computing infrastructure developed by the Apache Foundation characterized with a series of features such as high scalability, high efficiency, high fault tolerance, and low cost, which allow users to easily develop and run applications of large-scale data processing on Hadoop.

1) HDFS

Hadoop Distributed File System (HDFS) [39] as a distributed file system suitable for running on general purpose hardware is not only high fault tolerant but also suitable for application to cheap machines. Good support for general users, with high HDFS access data throughput, it is also very suitable for large-scale datasets.

2) MAPREDUCE

Hadoop supports MapReduce programming model invented by Google Inc. [40], which can solve level of Big Data problems by using clusters. There are two independent steps in this model, both of which can be user-defined through configuration or Java API in the program.

Map: The input datasets will be read and converted in the form of $<key, value>$. In this step, parallel processing will be performed on input data.

Reduce: Values with the same key will be integrated and aggregated.

3) YARN

Hadoop2.x, i.e., MapReduce upgrade, is called MR v2 or YARN. Hadoop1.x job scheduler is mainly responsible for resource management and job scheduling. YARN has broken through many limitations on this basis, such as the addition of a node manager and a container. The node manager as a slave service co-works with the resource manager and accepts requests from resource manager to allocate containers to applications. It is also responsible for monitoring and reporting the resource utilization and health status between nodes to further stabilize HDFS with the strong support of YARN.

B. ENVIRONMENT CONFIGURATION

1) HARDWARE CONFIGURATION

In this paper, our Hadoop platform consists of 9 high performance servers (including 1 master-node and 8 slave-nodes): DELL Power Edge R730 (Physical Machine). In the 9-nodes cluster, each server is equipped with Xeon E5-2640 v4 CPU, 2.4 GHz, 64GB memory and 2TB hard disk. Each slave-node can provide 8 vCores for MapReduce jobs, so theoretically our cluster can provide concurrent execution efficiency of up to 64 map processes.

2) SOFTWARE CONFIGURATION

Operation system: 9 servers are installed with Ubuntu 16.04 64-bit Linux operating system;

TABLE 2. Configuration information of Hadoop.

No.	Configuration file name	Configuration content
1	hdfs-site.xml	Properties of configuration HDFS such as block-size, etc.
2	core-site.xml	Core of configuration Hadoop system and general properties
3	mapred-site.xml	Configuration resource manager, task scheduling, MapReduce memory request, JVM size
4	yarn-site.xml	Configuration container resource memory requests, the number of CPU cores, etc.

```
<configuration>
  <property>
      <name>mapreduce.architecture.name</name>
      <value>yarn</value>
  </property>
  <property>
      <name>mapred.child.java.opts</name>
      <value>-Xmx64000m</value>
  </property>
</configuration>
```

FIGURE 7. Fragment of mapred-site.xml.

Hadoop version: Hadoop 2.9.1
Development Tools: JDK1.8 and Eclipse Neon

3) HADOOP CLUSTER CONFIGURATION

9 Servers in Hadoop cluster are connected by 1000Mbps Ethernet Switch, configure IP addresses of master-slave nodes in /etc/hosts of Ubuntu OS to build Hadoop platform, configure all master-slave nodes in the cluster are in the same LAN.

C. RELEVANT PARAMETER SETTING OF HDFS AND MAPREDUCE

For better using of HDFS and MapReduce (MR v2) framework, this paper conducts parameter setting of Hadoop configuration files, mainly including four files: hdfs-site.xml, core-site.xml, mapred-site.xml and yarn-site.xml. Relevant Hadoop configuration file information involved in this paper is shown in Table 2:

The input block-size involved in this paper is determined by the number of Map process in parallel execution. Thus, obtain the following Eq. (10):

$$block\_size = \frac{file\_size}{map} \tag{10}$$

Now conduct following necessary configuration for mapred-site.xml and yarn-site.xml, shown in Fig.7 and Fig.8.

In which, *mapreduce.task.io.sort.factor* property is set as 100. This parameter is the memory size of map output sort, 10 as the default value; *Mapred.child.Java.opts* property is the size of JVM. For maximal utility of JVM, it is set as 64000M in this paper.

The *yarn.nodemanager.resource.memory-mb* property represents the total physical memory that can be used by the YARN architecture on the node. As the configurated memory of each server is 64G, 65536M is set to fully use the node memory and to adjust the uniform distribution of map process; the *yarn.nodemanager.vmem-pmem-ratio* property represents the ratio of physical memory to virtual memory,

```
<configuration>
  <property>
      <name>yarn.resourcemanager.hostname</name>
      <value>s1</value>
  </property>
  <property>
      <name>yarn.nodemanager.aux-services</name>
      <value>mapreduce_shuffle</value>
  </property>
  <property>
      <name>yarn.nodemanager.resource.memory-mb</name>
      <value>65536</value>
  </property>
  <property>
      <name>yarn.nodemanager.vmem-pmem-ratio</name>
      <value>4</value>
  </property>
</configuration>
```

FIGURE 8. Fragment of yarn-site.xml.

2.1 as the default value. To prevent memory overflow, the value is set as 4 in this paper.

VI. EXPERIMENT RESULTS AND ANALYSIS

With Chinese online encyclopedia open encyclopedia KB as the experimental data source in this paper, the crawler toolkit HTMLParser is used for crawling and parsing the Infobox structured-information contained on entry's web page. In this paper, the strategy for building KBs of BaiduBaike and Hudong is based on the methods reported in Ref. [3]. There are totally 1,698,149 and 2,161,616 Infobox triples extracted from entries' web page of BaiduBaike and Hudong respectively. Then, the large-scale Chinese open-domain KBs are established.

Then, the crawled data can be divided into 11 top-class sets, each of which contains sub-class concepts and each sub-class concept includes corresponding triples. The detailed BaiduBaike KB is shown in Table 3.

A. PRESENTATION ON PRECISION IMPROVEMENT BY TWO STAGES OF ORIGINAL SERIAL REFINEMENT ALGORITHM

Next, parallel execution of InterlinkingValue algorithm will be performed, and the serial execution results of the original algorithm will be used as the basis for evaluating the correctness of the results, the running time and executing efficiency of the algorithm obtained by parallel implementation of the algorithm proposed in this paper are used for evaluating. To verify the speedup of the Parallel_InterlinkingValue algorithm, every server is configured with parameters in accordance with the configuration of the 9-nodes Hadoop cluster.

TABLE 3. Information of BaiduBaike KB.

11 top- classes	Sub-class	The number of triples	The number of correctly classified triples	The number of improper classified triples
Geography	Landform	143	120	23
	River	810	587	223
	Lake	490	399	91
Economy	Bank	848	622	226
People	Teacher	4769	1773	2996
Society	Politics and law	5977	5537	440
Life	Scenic spot	3498	3137	361
	Food	3591	3112	479
Sports	Competition	93	81	12
Culture	Calligraphy and paintings	2130	1568	562
	Prose	1522	383	1139
	Language	857	136	721
Art	Band	663	396	267
	Architecture	3222	2385	837
	Collectible	426	405	21
Nature	Earthquake	475	316	159
	Constellation	755	638	117

1) COMPARISON OF RANKING DECLINE OF INAPPROPRIATELY CLASSIFIED TRIPLES ON TWO STAGES

Based on the partition of processing phases of parallel refining algorithm in this paper, the refined results are also divided into two stages for presentation.

i) First Stage: Sorting each triples set in descending order according to the initial similarity value running by proposed Parallel_InterlinkingValue algorithm.

ii) Second Stage: Sorting every triple set in descending order according to the target similarity value.

After data field processing of triple tags, sorting of the obtained second-stage target similarity set and the first-stage initial similarity set by descending order is performed respectively, and then the changes on the number of improper classified triples in the latter 41% of the respective sub-class triple sets are compared respectively. As shown in Table 4, the number of triples of improper classified in the latter 41% of most sub-class triples set increased after the data field processing, and showed significant effect. The most significant is sub-class ''Politics and law'', with the number of improper classified triples increased by 347. Meanwhile, the number of triples of improper classified in the sub-class triples set ''Bank'' and ''Architecture'' decreased and the reduced degree is relatively small. It could be negligible considering that the total number of improper classified triples has increased by 761.

2) COMPARISON OF PRECISION IMPROVED EFFECT ON TWO STAGES

Precision is an important indicator to evaluate information retrieval effect.

Among the data at these two stages, the latter 41% of the sub-class triples are deleted firstly, and then *P*-values of remaining triples can be calculated for staged comparison.

Table 5 is made for listing the *P*-values of each sub-class triple set running by proposed parallel refining algorithm at two stages. The *P*-values at the first and second stage are compared with state-of-the-art Chinese encyclopedia KBs construction methods: Ref. [3] and [4]. There is comparability between our method and Ref. [4] because knowledge triples are both refined in the second stage of each system but different algorithms are used. Specifically, the latter 41% of triples in each sub-classes triple set are also deleted according to the descending order of triples' TF-IDF values.

It can be seen that the average *P*-values obtained by TF-IDF algorithm [4] is better than that by Ref. [3]. This is because most triples which can reflect the Infobox characteristics of their concept achieve a high ranking, so as to easily eliminate the noise triples which rank relatively lower. So it can be proved that as long as we use appropriate intelligent algorithms to refine knowledge, we will get KBs with high quality and precision. But the *P*-values of parallel refining algorithm is higher than TF-IDF algorithm [4], it is mainly because TF-IDF algorithm can only compute the frequency of appearing of Infobox mechanically and ignore its semantic features completely. Therefore, some randomness will be involved into the results of TF-IDF algorithm. By comparison, our method not only introduces the TongYiCiCiLin (extended version) as the semantic dictionary but also introduces the tags of Infobox triples as their semantic feature, so the improved nuclear field-like potential function will have more rationality and stability in the re-ranking and re-clustering process of triples.

The average *P*-values based on the parallel refining algorithm has been increased state-of-the-art methods: Ref. [3] and Ref. [4] by about 5.72% and 4.16%respectively.

B. PARALLEL_INTERLINKINGVALUE - PERFORMANCE EVALUATING ON PARALLEL ALGORITHM OF FIRST-STAGE: INITIAL SIMILARITY COMPUTING

The knowledge set to be processed will be stored in the form of plain text. 17 sub-class triple sets in BaiduBaike

TABLE 4. Changes in the number of triples of improper classified in the latter 41% of sub-class before and after data field.

Hudong Top-classes	BaiduBaike Sub-classes	Before data field	After data field	Difference value
Geography	Landform	15	17	+2
	River	112	128	+16
	Lake	45	54	+9
Economy	Bank	123	115	-8
People	Teacher	1288	1300	+12
Society	Politics and law	6	353	+347
Life	Scenic spot	159	214	+55
	Food	163	271	+108
Sports	Competition	2	12	+10
Culture	Calligraphy and paintings	435	517	+82
	Prose	256	312	+56
	Language	300	338	+38
Art	Band	126	143	+17
	Architecture	306	301	-5
	Collectible	7	9	+2
Nature	Earthquake	90	98	+8
	Constellation	72	84	+12
Total		3505	4266	+761

TABLE 5. *P*-values of the latter 41% deleted at two stages.

Hudong Top-classes	BaiduBaike Sub-classes	P-value				*P*-value increases
		Ref. [3]	Ref. [4]	First Stage	Second Stage	
Geography	Landform	83.92%	90.10%	90.52%	92.89%	+8.97%, +2.79%
	River	65.80%	64.78%	65.47%	68.82%	+3.02%, +4.04%
	Lake	81.43%	80.88%	84.09%	87.20%	+5.77%, +6.32%
Economy	Bank	73.35%	75.63%	79.41%	77.81%	+4.47%, +2.18%
People	Teacher	38.82%	41.10%	42.09%	42.52%	+3.70%, +1.42%
Society	Politics and law	92.64%	93.80%	87.69%	97.53%	+4.89%, +3.73%
Life	Scenic spot	89.68%	92.45%	90.21%	92.88%	+3.20%, +0.43%
	Food	86.66%	88.74%	85.09%	90.18%	+3.52%, +1.44%
Sports	Competition	87.10%	84.96%	81.78%	100.00%	+12.90%, +15.04%
Culture	Calligraphy and paintings	25.16%	23.42%	21.60%	30.73%	+5.57%, +7.31%
	Prose	73.62%	74.44%	75.65%	80.11%	+6.49%, +5.67%
	Language	15.87%	20.74%	16.74%	24.25%	+8.41%, +3.51%
Art	Band	59.73%	64.59%	63.95%	68.30%	+8.57%, +3.71%
	Architecture	74.02%	75.81%	72.07%	71.80%	-2.22%, -4.01%
	Collectible	95.07%	95.13%	94.43%	95.23%	+0.16%, +0.10%
Music	Earthquake	66.53%	70.23%	75.41%	78.23%	+11.71%, +8.00%
	Constellation	84.50%	83.55%	89.90%	92.59%	+8.09%, +9.04%
Average		70.23%	71.79%	71.53%	75.95%	+5.72%, +4.16%

are used as the experimental dataset of the parallel refining algorithm. Hudong contains 11 top-classes. In this paper, there are 9 top-classes in Hudong corresponding to 17 sub-classes of BaiduBaike for semantic similarity computing. The mapping relationship and the number of Infobox knowledge triples in each set are shown in Table 6.

The number of similarity computing time is $m \times n \times 3$, m and n are the number of knowledge triple in the dataset of BaiduBaike and Hudong, respectively. 3 represents that each initial similarity computing must be performed once on $< S$, P, $O >$ in the triple respectively. Thus, the total number of computing needs to multiplied by 3, $b_{is} \cdot h_{js}$, $b_{ip} \cdot h_{jp}$ and $b_{io} \cdot h_{jo}$ represent the every operation.

For better use of the MapReduce architecture, this paper sets once-call *Combiner*() method between map and reduce. The *Combiner*() method can conduct Reduce process once in advance on the Map side, which can reduce the intermediate result of the mapTask output, thus reducing the amount of remote copy data volume of each reduceTask, eventually reflected as the shortened executing time of the mapTask and reduceTask. As key is used for value accumulation computing in this paper, it is available to set and call *Combiner*() method.

For better verification of the parallelism of the algorithm, speedup of program parallel execution is mainly used in this paper for measurement. The size of time expenditure is mainly reflected in the number of Map process. It requires adjusting the size of each block according to the number of Map processes started as one block is allocated to one Map task. In each task, we set the number of Reduce process by using *setNumReduceTasks*() method to half of the corresponding number of Map process.

TABLE 6. Mapping relationship between BaiduBaike and Hudong.

Task Number	Sub-classes of BaiduBaike	Top-classes of Hudong	Number of triples in BaiduBaike	Number of triples in Hudong	Number of six-tuples in input file (billion rows)	Number of similarity computing (billion times)
1	Landform	Geography	143	455182	0.6509	1.9527
2	River		810		3.6870	11.0609
3	Lake		490		2.2304	6.6911
4	Bank	Economic	848	129312	1.0966	3.2897
5	Scenic spot	Life	3498	567438	19.8490	59.5469
6	Food		3591		20.3767	61.1301
7	Calligraphy and paintings	Culture	1522	277476	4.2232	12.6696
8	Language		857		2.3780	7.1339
9	Prose		2130		5.9102	17.7207
10	Band	Art	663	164656	1.0917	3.2750
11	Architecture		3222		5.3052	15.9156
12	Collectible		426		0.7014	2.1043
13	Competition	Sports	93	44371	0.0413	0.1238
14	Teacher	People	4768	512014	24.4128	73.2385
15	Politics and law	Social	5978	734776	43.9249	131.7747
16	Earthquake	Nature	975	1642294	16.0124	48.0371
17	Constellation		628		10.3136	30.9408

TABLE 7. Comparison of first-stage interlinkingvalue algorithm running time.

Task number	Initial similarity computing tasks	Serial executing time(s) (1 map 1 reduce)	Number of Maps Parallel executing time(s)				
			8 maps 4 reduces	16 maps 8 reduces	32 maps 16 reduces	64 maps 32 reduces	128 MB (number of maps, 32 reduces)
1	Landform - Geography	462	65	42	23	13	19(46 maps)
2	River - Geography	3416	463	253	138	75	100(241 maps)
3	Lake - Geography	1985	276	151	83	45	47(150 maps)
4	Bank - Economic	1071	153	84	46	25	27(83 maps)
5	Teacher - People	23269	3383	1853	1008	554	560(481 maps)
6	Scenic spot - Life	39564	6809	4337	2380	1258	1359(1401 maps)
7	Food - Life	20508	2933	1805	875	476	532(1530 maps)
8	Calligraphy and paintings - Culture	3466	473	259	141	76	83(280 maps)
9	Language - Culture	1777	253	139	76	42	87(154 maps)
10	Prose - Culture	5771	771	420	228	124	157(397 maps)
11	Politics and law - Social	1857	249	136	75	40	50(121 maps)
12	Band - Art	957	135	75	41	22	24(71 maps)
13	Architecture - Art	5536	756	413	225	122	172(381 maps)
14	Collectible - Art	1752	232	124	67	37	38(102 maps)
15	Competition - Sports	317	48	27	15	8	87(4 maps)
16	Earthquake - Nature	7893	1063	579	315	171	173(39 maps)
17	Constellation - Nature	8057	1070	547	294	163	164(183 maps)

1) TIME EFFICIENCY COMPARISON

As analyzed above, it is available to control the number of the starting Map process and the distribution of Map process according to the block_size and the memory request size in the configuration of YARN. Considering the limited hardware resources, this paper can set 1 to 64 Map processes on the 9-nodes Hadoop cluster for evaluating the parallel time expenditure of the Interlinking-Value algorithm and the executing efficiency of the parallel algorithm. While conducting experiment, we set up each Map process only responsible for one block. Thus, it requires setting up the size of each block according to the number of Map process in face of or different computing tasks. Please refer to Eq. (10) for computing method. Time results of the Parallel_InterlinkingValue algorithm are shown in Table 7.

In the manner of Bar Graph, Fig.9 and Fig.10 show the comparing of the executing time of totally 17 computing tasks running by parallel computing in different numbers of Map processes by using Chinese knowledge refinement method proposed in this paper. It can be seen that the executing time of Parallel_InterlinkingValue algorithm can shorten with the growth of computing resources.

And it also indicates that the parallel algorithm proposed in this paper can effective reduce the program operating time and enhance program efficiency. With the growth of computing resources, it is expected to realize the maximization of the executing efficiency of the parallel algorithm.

2) PARALLEL EFFICIENCY - SPEEDUP

It can be known from *Amdahl's Law*that the speedup of parallel execution of each initial similarity computing task is

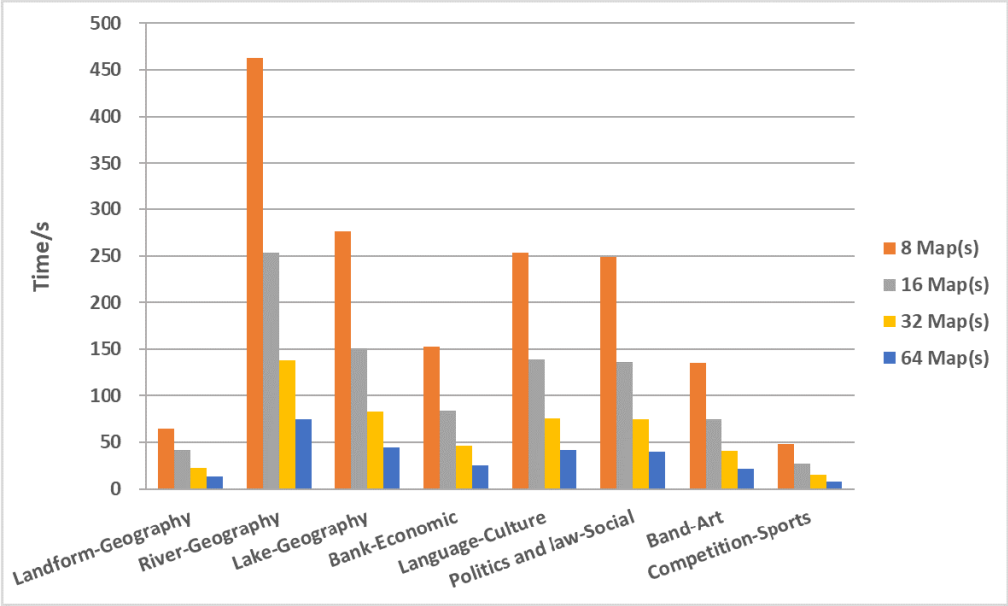

FIGURE 9. Bar graph of comparing the executing time of Parallel_InterlinkingValue of computing tasks- Part I.

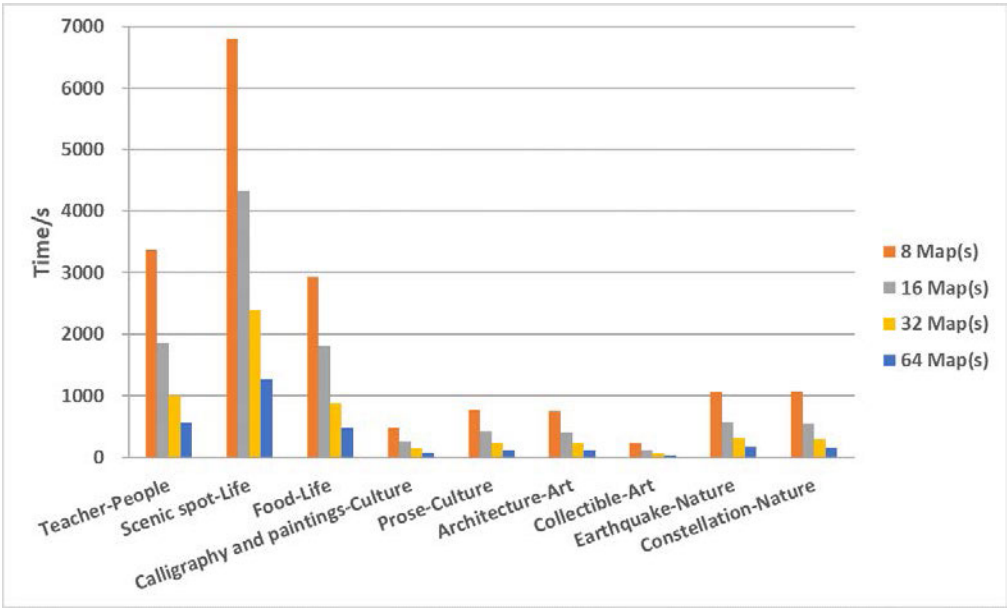

FIGURE 10. Bar graph of comparing the executing time of Parallel_InterlinkingValue of computing tasks- Part II.

shown in Eq. (10):

$$Speedup = \frac{Serial\ executing\ time}{Parallel\ executing\ time} \tag{11}$$

The finally computed and aggregated speedup results of all tasks are shown in Table 8.

As indicated in Table 8 that when the number of Map processes is 8, the average execution efficiency of Parallel_InterlinkingValue algorithm is 7.13 times higher than that of serial algorithm. When the number of Map processes is 16, the speedup of the Parallel_InterlinkingValue algorithm is nearly 13. When the number of Map processes is 32, the efficiency of parallel algorithm is nearly 24 times higher than that of serial algorithm on average. When the number of map processes reaches 64, the average efficiency of Parallel_InterlinkingValue algorithm reaches the highest, that is, the average speedup is close to 43. Analyzing the causes of this phenomenon, we think that with the increasing number of Map and Reduce processes, the costs of communication between processes, competing resources and the time cost caused by Partition-Sort and MergeSort operations (Shuffle process) required by Reducer will gradually increase, which will gradually slow down the growth trend of the speedup. But on the whole, the Parallel_InterlinkingValue algorithm has good scalability.

TABLE 8. Speedup.

Task number	Initial similarity computing tasks	Speedup				
		8 Map, 4 Reduces	16 Maps 8 Reduces	32 Maps 16 Reduces	64 Maps 32 Reduces	128 MB (number of maps, 32 reduces)
1	Landform - Geography	7.11	11.00	20.09	35.54	24.32(46 maps)
2	River - Geography	7.38	13.50	24.75	45.55	34.16(241 maps)
3	Lake - Geography	7.19	13.15	23.92	44.11	42.23(150 maps)
4	Bank - Economic	7.00	12.75	23.28	42.84	39.67(83 maps)
5	Teacher - People	6.88	12.56	23.08	42.00	41.55(481 maps)
6	Scenic spot - Life	5.81	9.12	16.62	31.45	29.11(1401 maps)
7	Food - Life	6.99	11.36	23.44	43.08	38.55(1530 maps)
8	Calligraphy and paintings - Culture	7.33	13.38	24.58	45.61	41.76(280 maps)
9	Language - Culture	7.02	12.78	23.38	42.31	20.43(154 maps)
10	Prose - Culture	7.49	13.74	25.31	46.54	36.76(397 maps)
11	Politics and law - Social	7.46	13.65	24.76	46.43	37.14(121 maps)
12	Band - Art	7.09	12.76	23.34	43.50	39.88(71 maps)
13	Architecture - Art	7.32	13.40	24.60	45.38	32.19(381 maps)
14	Collectible - Art	7.55	14.13	26.15	47.35	46.11(102 maps)
15	Competition - Sports	6.60	11.74	21.13	39.63	3.64(4 maps)
16	Earthquake - Nature	7.43	13.63	25.06	46.16	25.10(39 maps)
17	Constellation - Nature	7.53	14.73	27.40	49.43	49.13(183 maps)
	Average	7.13	12.79	23.58	43.35	34.22

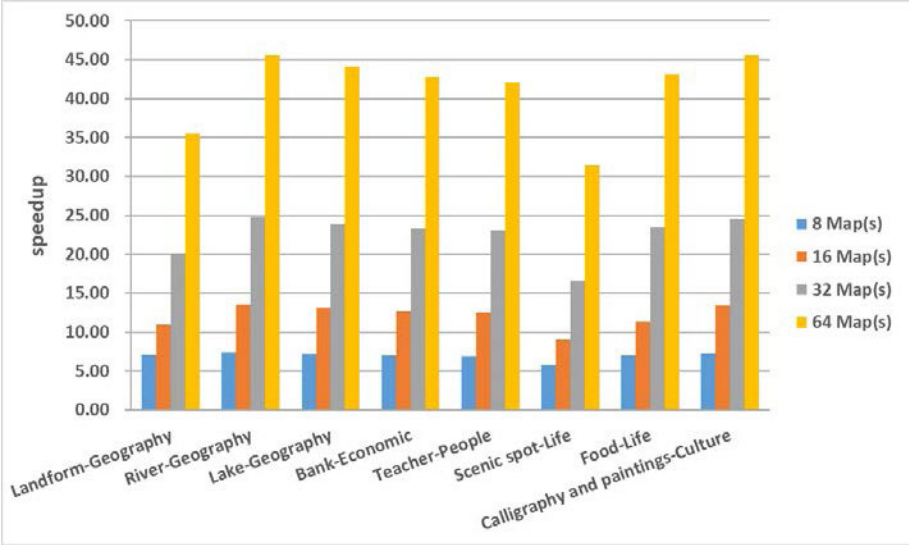

FIGURE 11. Bar graph of comparing the speedup – Part I.

We also tested the speedup of the Parallel_Interlinking Value algorithm in the case of HDFS default block-size: 128MB. However, the average speedup obtained in this experiment(34.22) will be less than that when 64 Maps are executed(43.35). This is because there are only 64 Maps can be executed concurrently in our 9-nodes Hadoop cluster (at present, the total number of vCores in the cluster is 64). Unfortunately, the number of Map processes initiated by each task in the case of default block-size is generally greater than 64, so Yarn is required to repeatedly start and allocate a large number of redundant Map processes to execute, which leads to more time waste besides Shuffle operations. Therefore, under the existing conditions, we think that the Parallel_InterlinkingValue algorithm can archive the best speedup when totally 64 Maps are run, which equal to the total number of vCores on slave-nodes in Hadoop cluster. We regard the execution time of running 1 map process as equal to that of the serial InterlinkingValue program.

For clarity, we also illustrate the Fig.11 and Fig.12 in the manner of Bar Graph for showing the speedup of totally 17 computing tasks obtained by the proposed parallel computing algorithm in different numbers of Map processes. It can be seen that the speedup of parallel algorithm would increase with the growth of computing resources. In allowable equipment and ideal condition, it is expected to obtain the peak of speedup and maximization of parallel efficiency, which also indicates that the proposed Parallel_InterlinkingValue algorithm is feasible and effective.

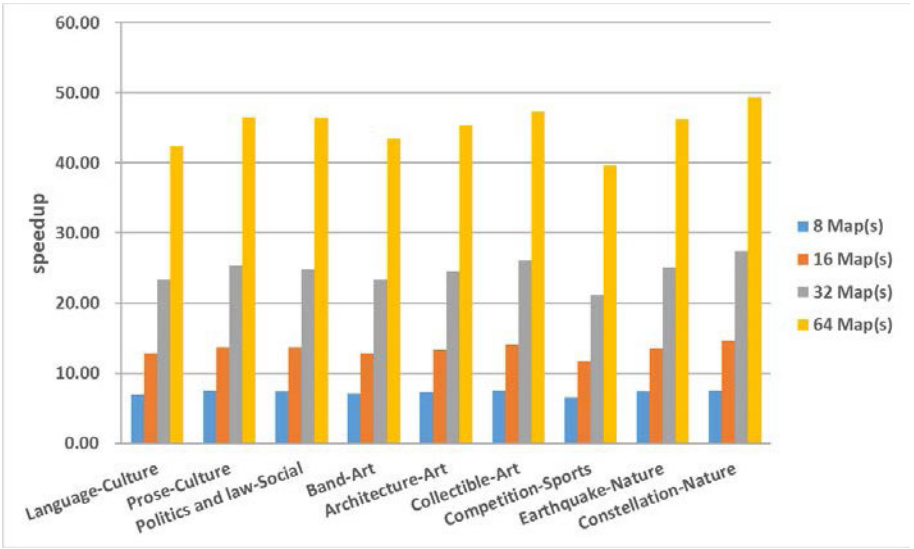

FIGURE 12. Bar graph of comparing the speedup – Part II.

VII. CONCLUSION

Dealing with the tremendous amount of triples in the process of refining and denosing the KBs on a single machine environment will inevitably lead to the unacceptable time expenditures and non-scalability of system. In order to make the algorithm scalable and high efficiency, this paper proposes a parallel algorithm for knowledge similarity computing and refining knowledge triples of large-scale Chinese encyclopedia based on HDFS and MapReduce, and then analyses results with various experimentation.

Based on the 9-nodes Hadoop cluster environment, performance evaluating and experimental verification of the proposed parallel refining algorithm has been conducted on the real world datasets. Results show that the proposed parallel algorithm can reduce the computing time, provide our denoising system with good scalability and improve the computing efficiency to a greater degree.

ACKNOWLEDGEMENTS

Furthermore, we are also appreciated our college for providing us 9 high-performance servers to run our MapReduce jobs.

REFERENCES

[1] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic Web," *Sci. Amer.*, vol. 284, no. 5, pp. 28–37, 2001.

[2] T. Wang, "Aligning the large-scale ontologies on schema-level for weaving Chinese linked open data," in *Cluster Computing*. Berlin, Germany: Springer, 2018, pp. 1–16. doi: 10.1007/s10586-018-1732-z.

[3] Z.-C. Wang, Z.-G. Wang, J.-Z. Li, and J. Z. Pan, "Knowledge extraction from Chinese wiki encyclopedias," *J. Zhejiang Univ. Sci. C*, vol. 13, no. 4, pp. 268–280, 2012.

[4] T. Wang, H. Gu, Z. Wu, and J. Gao, "Multi-source knowledge integration based on machine learning algorithms for domain ontology," in *Neural Computing Applications*. Berlin, Germany: Springer, 2018, pp. 1–11. doi: 10.1007/s00521-018-3806-5.

[5] F. M. Suchanek, G. Kasneci, and G. Weikum, "YAGO: A large ontology from Wikipedia and WordNet," *J. Web Semantics*, vol. 6, no. 3, pp. 203–217, 2008.

[6] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann, "DBpedia—A crystallization point for the Web of data," *J. Web Semantics*, vol. 7, no. 3, pp. 154–165, 2009.

[7] F. Wu and D. S. Weld, "Automatically refining the wikipedia infobox ontology," in *Proc. 17th Int. Conf. World Wide Web*, 2008, pp. 635–644.

[8] F. Wu and D. S. Weld, "Autonomously semantifying wikipedia," in *Proc. 16th ACM Conf. Conf. Inf. Knowl. Manage.*, 2007, pp. 41–50.

[9] C. Bizer, T. Heath, and T. Berners-Lee, "Linked data: The story so far," *Int. J. Semantic Web Inf. Syst.*, vol. 5, no. 3, pp. 1–22, 2009.

[10] Y. Chen, L. Chen, and K. Xu, "Learning Chinese entity attributes from online encyclopedia," in *Proc. Asia–Pacific Web Conf.* Berlin, Germany: Springer, 2012, pp. 179–186.

[11] J. Li, C. Wang, X. He, R. Zhang, and M. Gao, "User generated content oriented Chinese taxonomy construction," in *Proc. Asia–Pacific Web Conf.* Berlin, Germany: Springer, 2015, pp. 623–634.

[12] X. Niu, X. Sun, H. Wang, S. Rong, G. Qi, and Y. Yu, "Zhishi.me—Weaving Chinese linking open data," in *The Semantic Web ISWC*, L. Aroyo, C. Welty, H. Alani, J. Taylor, A. Bernstein, L. Kagal, N. Noy, and E. Blomqvist, Eds. Berlin, Germany: Springer, 2011, pp. 205–220.

[13] X. Wang, K. Liu, S. He, S. Liu, Y. Zhang, and J. Zhao, "Multi-source knowledge bases entity alignment by leveraging semantic tags," *Jisuanji Xuebao/Chin. J. Comput.*, vol. 40, no. 3, pp. 701–711, 2017.

[14] T. Wang, T. Xu, Z. Tang, and Y. Todo, "TongSACOM: A tongyicicilin and sequence alignment-based ontology mapping model for Chinese linked open data," *IEICE Trans. Inf. Syst.*, vol. 100, no. 6, pp. 1251–1261, 2017.

[15] O. Medelyan, D. Milne, C. Legg, and I. H. Witten, "Mining meaning from Wikipedia," *Int. J. Hum.-Comput. Stud.*, vol. 67, no. 9, pp. 716–754, 2009.

[16] R. C. Hicks, "Knowledge base management systems-tools for creating verified intelligent systems," *Knowl.-Based Syst.*, vol. 16, no. 3, pp. 165–171, 2003.

[17] K. Rohloff and R. E. Schantz, "High-performance, massively scalable distributed systems using the MapReduce software framework: The SHARD triple-store," in *Proc. Program. Support Innov. Emerg. Distrib. Appl. (PSI EtA)*, New York, NY, USA, 2010, Art. no. 4.

[18] M. Husain, J. McGlothlin, M. M. Masud, L. Khan, and B. M. Thuraisingham, "Heuristics-based query processing for large RDF graphs using cloud computing," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 9, pp. 1312–1327, Sep. 2011.

[19] A. Schätzle, M. Przyjaciel-Zablocki, T. Berberich, and G. Lausen, "S2X: Graph-parallel querying of RDF with graphX," in *Biomedical Data Management and Graph Online Querying*. Berlin, Germany: Springer, 2015, pp. 155–168.

[20] O. Erling and I. Mikhailov, "Virtuoso: RDF support in a native RDBMS," in *Semantic Web Information Management*. Berlin, Germany: Springer, 2010, pp. 501–519.

[21] A. Schätzle, M. Przyjaciel-Zablocki, S. Skilevic, and G. Lausen, "S2RDF: RDF querying with SPARQL on spark," *Proc. VLDB Endowment*, vol. 9, no. 10, pp. 804–815, 2016.

[22] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets," in *Proc. HotCloud*, 2010, p. 10.

[23] P. Peng, L. Zou, M. T. Özsu, L. Chen, and D. Zhao, "Processing SPARQL queries over distributed RDF graphs," *VLDB J. Int. J. Very Large Data Bases*, vol. 25, no. 2, pp. 243–268, 2016.

[24] B. Shao, H. Wang, and Y. Li, "Trinity: A distributed graph engine on a memory cloud," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2013, pp. 505–516.

[25] E. Gabriel, G. E. Fagg, G. Bosilca, T. Angskun, J. J. Dongarra, J. M. Squyres, V. Sahay, P. Kambadur, B. Barrett, A. Lumsdaine, R. H. Castain, D. J. Daniel, R. L. Graham, and T. S. Woodall, "Open MPI: Goals, concept, and design of a next generation MPI implementation," in *Proc. Eur. Parallel Virtual Mach. Message Passing Interface Users Group Meeting*. Berlin, Germany: Springer, 2004, pp. 97–104.

[26] K. Zeng, J. Yang, H. Wang, B. Shao, and Z. Wang, "A distributed graph engine for Web scale RDF data," *Proc. VLDB Endowment*, vol. 6, no. 4, pp. 265–276, 2013.

[27] S. Gurajada, S. Seufert, I. Miliaraki, and M. Theobald, "TriAD: A distributed shared-nothing RDF engine based on asynchronous message passing," in *Proc. SIGMOD Int. Conf. Manage. Data*, 2014, pp. 289–300.

[28] V. Khadilkar, M. Kantarcioglu, B. Thuraisingham, and P. Castagna, "Jena-HBase: A distributed, scalable and efficient RDF triple store," in *Proc. 11th Int. Semantic Web Conf. Posters Demonstrations Track ISWC-PD*, 2012, pp. 85–88.

[29] X. Wang, Q. Xu, L.-L. Chai, Y.-J. Yang, and Y.-P. Chai, "Efficient distributed query processing on large scale RDF graph data," (in Chinese), *Ruan Jian Xue Bao/J. Softw.*, vol. 30, no. 3, pp. 498–514, 2019.

[30] J. Ahn, D.-H. Im, J.-H. Eom, N. Zong, and H.-G. Kim, "G-Diff: A grouping algorithm for RDF change detection on MapReduce," in *Proc. Joint Int. Semantic Technol. Conf.* Berlin, Germany: Springer, 2014, pp. 230–235.

[31] T. Lee, D.-H. Im, and J. Won, "Similarity-based change detection for RDF in MapReduce," *Proc. Comput. Sci.*, vol. 91, pp. 789–797, 2016.

[32] H. Zhang, W. Hu, and Y. Qu, "Constructing virtual documents for ontology matching using MapReduce," in *Proc. Joint Int. Semantic Technol. Conf.* Berlin, Germany: Springer, 2011, pp. 48–63.

[33] H. Zhang, W. Hu, and Y.-Z. Qu, "VDoc+: A virtual document based approach for matching large ontologies using MapReduce," *J. Zhejiang Univ. Sci. C*, vol. 13, no. 4, pp. 257–267, 2012.

[34] A. I. Torre-Bastida, E. Villar-Rodriguez, J. Del Ser, D. Camacho, and M. Gonzalez-Rodriguez, "On interlinking linked data sources by using ontology matching techniques and the Map-Reduce framework," in *Proc. Int. Conf. Intell. Data Eng. Automated Learn.* Berlin, Germany: Springer, 2014, pp. 53–60.

[35] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Sov. Phys.-Dokl.*, vol. 10, no. 8, pp. 707–710, 1966.

[36] J.-J. Mei, *TongyiciCilin. Shanghai Lexicographical*. Beijing, China: Publishing House, 1983.

[37] D. Li and Y. Du, *Artificial Intelligence With Uncertainty*. Beijing, China: National Defence Industry Press, 2005.

[38] T. White and H. T. D. Guide, *Hadoop: The Definitive Guide*. Newton, MA, USA: O'Reilly Media, 2012.

[39] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The Hadoop distributed file system," in *Proc. MSST*, vol. 10, 2010, pp. 1–10.

[40] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, 2008.

TING WANG received the B.S. and Ph.D. degrees in computer science and technology from the Beijing University of Technology, in 2008 and 2014, respectively. In 2012, he was a Research Student with the Institute of Automation, Chinese Academy of Sciences, Beijing, China. He is currently an Associate Professor with the School of Management and Engineering, Capital University of Economics and Business. His research interests include semantic web, knowledge discovery, and big data. He is also a member of the Chinese Information Processing Society of China (CIPSC).

JIE LI is currently pursuing the bachelor's degree in computer science and technology with the Capital University of Economics and Business. His research interests include big data and semantic web. In 2017, he was awarded Excellent Cadre Award and Social Work Scholarship. He also won the National Encouragement Scholarship, the Second Prize for Excellence, the Three Good Students Award for Excellence, and the Research Scholarship, in 2018.

JIALE GUO is currently pursuing the bachelor's degree in computer science and technology with the Capital University of Economics and Business. His research interests include big data and semantic web. In 2018, he won the Excellent Cadre Award, the Social Work Scholarship, and the Research Scholarship.

JINGYAO XIE received the B.S. and M.D. degree in computer science and technology from the Beijing University of Aeronautics and Astronautics, in 2011 and 2014, respectively. She is currently an Engineer with State Grid Beijing Electric Power Company. Her research interests include data mining and virtualization technology.