RESEARCH PAPER

# CKGSE: A Prototype Search Engine for Chinese Knowledge Graphs

**Xiaxia Wang, Tengteng Lin, Weiqing Luo, Gong Cheng† & Yuzhong Qu**

State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China

**Keywords:** Knowledge graph; Search engine; Snippet generation; Dataset profiling; Browsing

Citation: Wang, X.X., et al.: CKGSE: A prototype search engine for Chinese knowledge graphs. Data Intelligence 4(1), 41-65 (2022). doi: 10.1162/dint_a_00118
Received: October 30, 2021; Revised: December 23, 2021; Accepted: January 19, 2022



## ABSTRACT

Nowadays, with increasing open knowledge graphs (KGs) being published on the Web, users depend on open data portals and search engines to find KGs. However, existing systems provide search services and present results with only metadata while ignoring the contents of KGs, i.e., triples. It brings difficulty for users' comprehension and relevance judgement. To overcome the limitation of metadata, in this paper we propose a content-based search engine for open KGs named CKGSE. Our system provides keyword search, KG snippet generation, KG profiling and browsing, all based on KGs' detailed, informative contents rather than their brief, limited metadata. To evaluate its usability, we implement a prototype with Chinese KGs crawled from OpenKG.CN and report some preliminary results and findings.

## 1. INTRODUCTION

Reusing existing data, especially knowledge graphs (KGs), saves duplicate human labors, thus being important in scientific research and application development. In recent years, lots of academic and industrial efforts have been paid to constructing reusable KGs, especially in specific domains such as e-commerce, biomedicine and education. As a result, many KGs have been increasingly published on the Web as reusable resources [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11]. This motivates the development of data sharing platforms, from the early Datahub (since 2006) and European Data Portal, to hundreds of data portals around the world [12]. Among all the resources available at data portals such as the 256 portals recorded

† Corresponding author: Gong Cheng (Email: gcheng@nju.edu.cn; ORCID: 0000-0003-3539-7776).

in [12], KGs form an important part. For example, Data.Gov[①] has indexed over 10K KGs by November 2021. Linked Open Data Cloud[②] has indexed 1,301 KGs. They have made a promising start for the user to easily provide or obtain KG resources. Furthermore, to assist the user in efficiently finding KGs and judging their relevance, recent research efforts have proposed various systems, from general search engines such as Google Dataset Search [13], to specialized systems like LODAtlas [14] for KG-centric data. OpenKG.CN is a popular platform for Chinese open KGs with keyword-based search service.

**Motivation.** Although the above systems [13, 14] provide search services for the user's convenience, they rely on *metadata*, which are meta-level annotations attached to each KG from its provider including the authorship and license information. Generally, metadata annotations are high-level descriptions but contain no details underlying KG content. Relying on metadata brings about two limitations of existing systems. **Limitation 1**: Given a keyword query referring to the KG's *content* (i.e., triples), such as an entity, class or property, they cannot effectively find the target KG. Actually this kind of queries are common in practice. According to an analysis [15], over 60% queries for KGs contain keywords referring to the content. **Limitation 2**: For search result presentation, metadata cannot provide close-up views of the underlying KG content. Indeed, many user interactive activities involved in relevance judgment depend on the content. In [16], users' real data needs are summarized into ten categories, and most of them such as exploration and analysis are mainly focused on the data content. Besides, existing analysis of KG search process [17] and search result presentation [18] show that they rely heavily on the KG content. In [17], the search process is divided into four steps, and the comprehension of the underlying KG content in the data handling step is crucial to the effectiveness of search. In [18], some features for characterizing a KG such as representative elements and instance-level statistics are based on the content. To sum up, the utility of metadata is limited [15, 19], while the KG content should be incorporated into the search process.

**Our Attempts.** Given the two limitations of existing systems, a content-based search engine for KGs is needed. However, as a new attempt in this direction, its practicability remains unknown. In this paper, as a preliminary effort to build and evaluate a content-based search system for open KGs, we present CKGSE, short for **C**hinese **KG S**earch **E**ngine. It has four components. **To address Limitation 1**, *KG Crawling and Storage* obtains KGs and their metadata using CKAN API, and then parses and stores the KGs locally. *Content-Based Keyword Search* parses keyword queries, and retrieves and ranks relevant KGs based on an inverted index containing both metadata and content fields. **To address Limitation 2**, for each search result, *Content-Based Snippet Generation* extracts a sub-KG to justify its query relevance. *Content Profiling and Browsing* provides detailed information about a KG, including a quality profile, statistical, abstractive and extractive summaries, and we also provide a faceted browsing panel for the user to explore the original KG. To evaluate the practicability of CKGSE, we implement a prototype[③] based on Chinese KGs collected from OpenKG.CN. Our contributions are summarized as follows.

① https://www.data.gov/
② https://lod-cloud.net/
③ http://ws.nju.edu.cn/CKGSE

- We propose CKGSE, as one of the first content-based search engines for open KGs;
- We present and discuss experimental results about the practicability of such a system.

**Outline.** In the rest of this paper, related work is discussed in Section 2. Section 3 and Section 4 introduce an overview of CKGSE and its detailed implementation, respectively. Experimental results are presented in Section 5. Section 6 concludes the paper with future work.

This article extends our previous work [20] in six aspects, including new system components, updated implementation, new user study, and more comprehensive research background.

- We revised our design and implementation of the inverted index in content-based keyword search component in Section 4.2.
- We added query-relevant indexed fields to the content-based snippet generation component in Section 4.3.
- We added the quality profile with six quality metrics to the content profiling component in Section 4.4.
- We added three visualized tag clouds to the statistical summary of the content profiling component in Section 4.4.
- We added a user study to verify the helpfulness of CKGSE by comparing it with OpenKG.CN in Section 5.3.
- We extended related work about KG summarization and snippet generation in Section 2.3.

We have accordingly updated our online system with new components and implementation.

## 2. RELATED WORK

We present some related work of our system from the following three aspects. Firstly, Section 2.1 provides some background about existing KG search systems and techniques. Secondly, since an important part of a KG search system is to present each result KG to the user, Section 2.2 reviews the methods for profiling a KG. Besides, as our system CKGSE focuses on the KG content, some content-based summarization methods for illustrating and exemplifying a KG are discussed in Section 2.3.

### *2.1 Open KG Portals and Search Engines*

Various open data portals are available nowadays [12]. They generally rely on metadata annotation under specific vocabularies such as W3C DCAT[④] to collect and manage KG resources. For example, European Data Portal [21] uses the title and description values in the metadata of each resource for deduplication. Google Dataset Search [13] identifies each data resource by the metadata index without considering the content, as it aims at navigating the user to the original webpage of each data resource. Some KG-centric

④ https://www.w3.org/TR/vocab-dcat-2/

systems also depend much on metadata, such as LODAtlas [14], providing metadata-based faceted filters for the user to select KGs.

As metadata-based systems cannot provide close-up views of the underlying KG content, they tend to be affected by the unguaranteed quality of metadata. To overcome their limitation, CKGSE takes KG content into consideration. Firstly, to support keyword search over content, for each KG we incorporate content elements into the inverted index. Secondly, to facilitate relevance judgment of search results, for each KG we extract a query-biased snippet as illustration. Thirdly, to provide a close-up view for each KG, we use various content-based profiles and exploration methods to ease comprehension.

### 2.2 KG Profiling

KG profiling aims at interpreting a KG from various descriptive aspects [17, 18, 22]. In [18], by surveying literature over the past two decades, a taxonomy was proposed to categorize profiling features into seven kinds, most of which are metadata-related such as Licensing and Provenance. The qualitative category consists of metrics for assessing the usability of a resource [12, 23]. The Statistical category includes data element counts and distributions, such as the number of instantiated classes or properties in a KG [24]. The General category contains methods to select representative elements from the KG, like structural summaries [25, 26, 27, 28] and pattern mining [29, 30].

Benefiting from existing fruitful research efforts, CKGSE incorporates several profiling techniques. In addition to metadata, CKGSE evaluates each KG with a set of qualitative metrics as a quality profile, provides element level statistical summaries, mines frequent entity description patterns (EDPs) as its abstractive summary [31, 32], and illustrates its content with an extractive summary [33, 34].

### 2.3 KG Summarization and Snippet Generation

KG summarization is to distill a small significant part from the original large KG, to accomplish specific tasks such as saving storage cost or answering queries efficiently. Lots of research attention has been paid to generating abstractive summaries for KGs [25]. They focus on graph structures and aggregate the frequent or common sub-structures as a summary. For example, entities can be aggregated by EDP-based similarity [29, 30] or by common multi-hop neighborhood [26, 35]. This kind of aggregation can also be hierarchical [28]. In [36], a trade-off between summary size and restoration accuracy was discussed. Complementary to abstractive summaries, KG snippet generation methods extract a representative sub-graph from the original KG to exemplify the content. Depending on the application, KG snippets can be either query-relevant [31, 32, 37] or query-independent [33, 34].

To provide a user with diversified views of the KG content, CKGSE implements both abstractive KG summaries [31] to show representative patterns, and extractive snippets [34] to illustrate the underlying data content. Besides, as snippets can be query-relevant, CKGSE also presents snippets [37] on the search results page, to exemplify the query relevance for each KG.

## 3. SYSTEM OVERVIEW

Figure 1 presents the user interaction and system architecture of CKGSE, consisting of four main components. Given a query submitted by the user, Content-Based Keyword Search firstly parses the query, then retrieves relevant KGs and ranks them as a list. To exemplify the relevance of each KG in the list, Content-Based Snippet Generation presents not only query-relevant metadata information but also an extractive sub-KG as snippet in the search results page. When a KG is selected by the user, Content Profiling and Browsing presents its profiled detailed information with supports for content exploration. In the back end, KG Crawling and Storage collects, parses and stores all the KGs with their metadata.

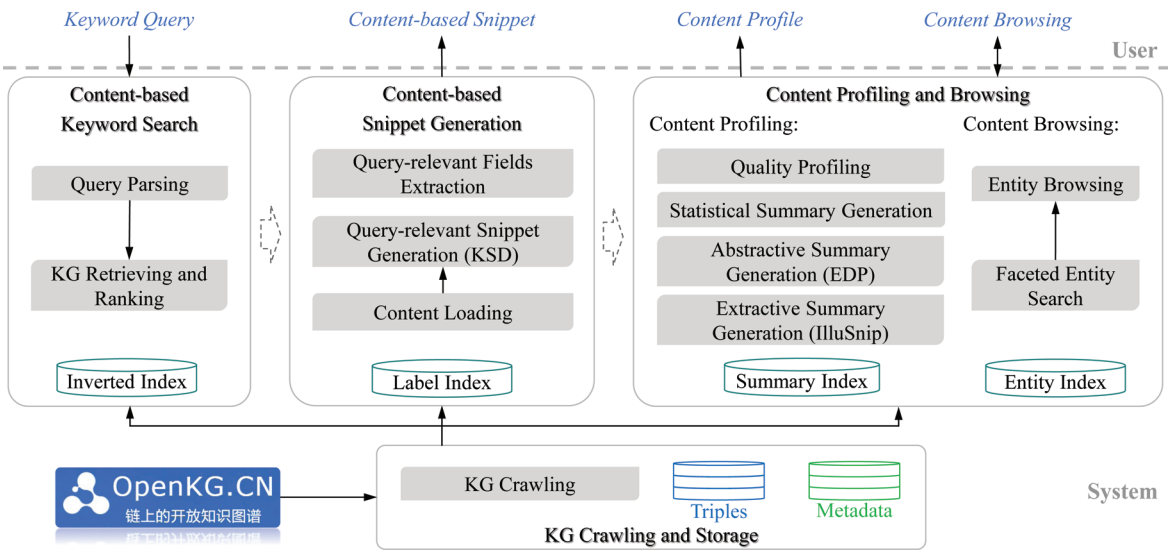


**Figure 1.** Overview of CKGSE: User interaction and system components.

**KG Crawling and Storage.** First of all, to collect KG resources in an offline process, CKGSE retrieves and stores all the available metadata records from OpenKG.CN. Then following the download links in the records, all the accessible KG dump files are downloaded, parsed and stored in a local database. Based on that, four indexes are built to support downstream tasks in other components, whose details will be introduced in Section 4.1.

**Content-based Keyword Search.** In this component, any input keyword query is firstly parsed by segmenting into (Chinese) words, with stop words being removed. Then the parsed query is searched over the content-based inverted index to obtain top-ranked KGs according to a relevance scoring function. The inverted index contains multiple fields of both metadata and content to support keyword match on any of them, where each field has a boost factor for the relevance scoring. Details will be presented in Section 4.2.

**Content-based Snippet Generation.** For each KG in the ranked results list, a query-relevant snippet containing metadata and content information is presented to the user. As the first part, we present the

indexed fields that match the query and highlight the matched keywords. To further exemplify the query relevance of the KG content with the graph structure, in the second part an extractive content snippet is online generated. After loading all the triples of the KG into memory, the generation method KSD [37] iteratively extracts a sub-KG considering query relevance and content representativeness with the support of a label index. Finally, the output content snippet contains $k$ top-ranked triples where $k$ is a pre-defined size limit, presented as a node-link diagram along with the metadata snippet on the search results page. An example of the content-based snippet is presented in Figure 6. Details will be presented in Section 4.3.

**Content Profiling and Browsing.** For each KG selected by the user, its profile is presented, including a qualify profile and three content-based summaries at different levels. The KG quality profile [23] is a set of content-based quantitative metrics to evaluate the intrinsic quality and usability of the KG such as availability and understandability. The statistical summary contains counts, distributions, and cloud representations of KG elements including classes, properties and entities. The abstractive summary contains the most frequent entity description patterns (EDPs) [31, 32] instantiated in the KG. The extractive summary is an optimal sub-KG of $k'$ triples generated by IlluSnip [33, 34] in terms of content representativeness, where $k'$ is a pre-defined size limit. All the quality metrics and summaries are pre-computed and indexed in an offline process. In addition to the profile, supported by an entity index, all the entities in the KG can be explored in a faceted manner, i.e., by filtering entities with their classes and properties. By selecting any filtered entity, all the triples describing it can be browsed. Details of the methods will be presented in Section 4.4. Examples of content profiling and browsing are given in Section 5.

## 4. SYSTEM IMPLEMENTATION

Now we introduce the detailed implementation of CKGSE by components.

### *4.1 KG Crawling and Storage*

This component crawls all the metadata and KGs from OpenKG.CN, and stores them in a local database.

**KG Crawling.** Through the CKAN API, CKGSE first retrieves all available metadata records of the resources from OpenKG.CN. Among the obtained records, 40 resources are identified as KGs by formats, such as RDF/XML, N-Triple, Turtle and JSON-LD, while the others are non-KG resources which are not our focus. Then by accessing the download links in metadata, all KG dump files are downloaded, parsed and stored in a local MySQL database.

**Metadata.** Metadata of all the KGs are stored as a table, where each row identifies a KG, and each column represents a metadata field instantiated in the CKAN vocabulary such as `Title`, `Author`, and `License`, most of which have textual values. We notice that incorrect and incomplete field values are relatively common, since they are freely submitted by KG publishers.

**Triples.** For each KG, all its dump files are parsed using Apache Jena 3.8.0. For some of the KGs that have more than one dump files, a triple-level deduplication is conducted before storing them into the database. The triples of each KG are stored in a table, with three columns identifying the subject, predicate and object. Besides, each RDF term in the KG is labeled with a human-readable textual form, including the local name, the value of `rdfs:label` property, and the textual form of literals. The term-label map is stored in database and will be used in downstream components.

We report all the storage costs of CKGSE in Table 5.

### 4.2 Content-based Keyword Search

This component parses the given keyword query, then retrieves and ranks relevant KGs with the support of an inverted index.

**Inverted Index.** To support keyword matching on both KG metadata and contents, an inverted index is created with eight fields, as presented in Table 1. Four of them are manually selected from existing metadata, as they are descriptive and likely to be matched to query keywords. For KG contents which are not considered in existing systems, following the RDF schema of classes and properties, we divide all the RDF terms in each KG into the four categories, and index all of them by their textual forms.

**Table 1.** Metadata and content fields used in the inverted index.

| Category | Fields |
|---|---|
| Metadata | `Title`, `Description`, `Author`, `Tags` |
| Content | `Classes`, `Properties`, `Entities`, `Literals` |

For all the eight fields, we assign each of them with a boost factor between 0 and 1 when being aggregated into a relevance result score. At the current stage, the boost factors are manually tuned, which could be adjusted in the future with more user query logs. We use Apache Lucene 7.5.0 to construct the index, and report the construction time cost in Table 4.

**Query Parsing.** We implement the keyword search component with Apache Lucene 7.5.0. Since it only provides basic query analyzers which cannot effectively conduct Chinese word segmentation, we incorporate IK analyzer[9] which is an open-source tool for Chinese word segmentation.

**KG Retrieving and Ranking.** The keywords parsed from the query are then searched on the inverted index, with OR as the default Boolean operator between keywords. CKGSE adopts a multi-field query parser based on Lucene to retrieve relevant KGs over all the eight fields with BM25 scoring function, then combines all the scores with the assigned boost factor of each field, and normalizes it as the overall relevance score. The KGs are ranked by the overall relevance scores, and top-ranked ones are returned.

[9] https://code.google.com/archive/p/ik-analyzer/

### 4.3 Content-based Snippet Generation

Existing systems simply list the metadata information to illustrate each returned KG, providing limited help for relevance judgement. Distinguished from existing systems, CKGSE provides a query-relevant snippet for each returned KG including both metadata and content parts, to facilitate relevance judgement.

**Query-relevant Fields Extraction.** As the first part of the query-relevant snippet, the indexed fields being matched to any keywords are presented in a flattened manner as key-value pairs. Following conventional Web search engines, CKGSE highlights all the matched parts in each field as suggestions. An example is presented in Figure 6.

**Label Index.** To support the query-relevant content snippet generation, a label index is constructed for each KG. This label index records a map from each word to the IRIs in the KG whose textual form contains the word. It is used to measure the query relevance of each triple by the number of keywords contained by its textual form, i.e., contained by the textual form of its subject, predicate, or object.

**Content Loading.** As a preprocess of content snippet generation, all triples of each KG in the results list are extracted from the database and loaded into memory. Based on the Label Index, the query relevance of each triple is computed in the loading process, as well as other content representativeness measures such as relative frequencies of the class or property instantiated in the triple.

**Query-relevant Snippet Generation.** CKGSE adopts KSD [37] to extract a snippet from the original KG content. By regarding each triple as an element set containing query keywords, classes, properties, and entities, and assigning a weight of importance to each element, KSD formulates snippet generation as a weighted maximum coverage problem. It aims at selecting at most $k$ triples to maximize the total weight of covered elements. We implement it with a greedy strategy. In each iteration, a triple containing the largest weight of uncovered elements is selected until reaching the pre-defined size limit $k$ or all elements are covered. Here we set $k = 10$. The details of KSD are introduced in [37].

### 4.4 Content Profiling

For each KG selected by the user, besides presenting the metadata information as existing systems do (as shown in Figure 7), this component presents an offline computed profile to illustrate its content, including a quality profile, a statistical summary, an abstractive summary, and an extractive summary.

**Quality Profile.** The quality profile aims to present a set of quantitative quality assessments as signals from both metadata and content, to facilitate the user with judgements. Since the quality metrics for KGs have been studied extensively [23], in CKGSE we select and implement six metrics that are relatively close to our search contexts, without any need of external resources. As shown in Table 2, three of them are relevant to metadata. Availability measures to what extent the KG resources can be obtained. Licensing represents whether or not the KG is under a specific license. Timeliness indicates whether the KG has been recently updated or not. The other three metrics are related to the content. Intra-KG interlinking stands for

the proportion of non-isolated entities in the KG. Inter-KG interlinking indicates how often the entities in the KG are linked with entities in other KGs. Understandability represents if most entities in the KG have a human-readable textual form. An example of the quality profile is shown in Figure 8. For each KG, all quality metrics are offline-computed and stored in the database.

**Table 2.** Metrics in the quality profile.

| Category | Name | Metrics |
|---|---|---|
| Metadata | Availability | (# successfully downloaded and parsed dump files)/(# provided dump files) |
| | Licensing | 1 (if with a license) or 0 (if without a license) |
| | Timeliness | (last updated – created)/(present – created) |
| Content | Intra-KG interlinking | (# non-isolated entities)/(# all entities) |
| | Inter-KG interlinking | (# triples with property `owl:sameAs`)/(# all triples) |
| | Understandability | (# entities with `rdfs:label`)/(# all entities) |

**Statistical Summary Generation.** The statistical summary consists of overall statistics of the selected KG, and close-up views to different kinds of elements contained in the KG. The basic statistics includes counts such as the number of triples and entities, which are presented to the user as a table, as shown in Figure 9. In addition to them, as summaries of the KG elements, for classes and properties we implement their distribution by relative frequencies as a pie chart, as shown in Figure 10. For top-ranked entities by the PageRank score computed on the original KG, we present them as the central content of the KG. Besides, we visualize three tag clouds for the classes, properties and entities, where classes and properties are weighted by frequencies and entities are weighted by the PageRank scores. An example entity tag cloud is shown in Figure 11.

**Abstractive Summary Generation.** Motivated by pattern mining techniques, CKGSE incorporates the frequent entity description patterns (EDPs) [31, 32] into the KG profiling component. In a KG, each entity is described by a set of classes and properties, i.e., schema-level elements. Each EDP retains a common description pattern shared among a set of entities. Therefore, frequent EDPs can be regarded as a pattern-level abstractive summary. Each of them consists of a set of forward properties, backward properties and classes that describe a set of entities. In the profile page, each EDP is presented as a node-link diagram as in Figure 12.

**Extractive Summary Generation.** Complementary to the abstractive schema-level summary represented by frequent EDPs, CKGSE also incorporates an extractive summary to directly exemplify the KG content. To extract a connected sub-KG with the most content representativeness, IlluSnip [33, 34] formulates the selection of triples as a combinatorial optimization problem. It defines content representativeness as the coverage of the most frequently instantiated classes, properties, and the most important entities with the highest PageRank scores. For each KG, such a selected sub-KG can be viewed as an extractive summary. A greedy algorithm is applied in IlluSnip to generate a sub-KG containing at most $k'$ triples. We set $k' = 20$. The result is presented as a node-link diagram on the profile page as shown in Figure 13.

**Summary Index.** The statistical, abstractive and extractive summaries for each KG are all offline computed and stored in a summary index.

### 4.5 Content Browsing

Beyond the metadata and profile page, CKGSE also enables the user to interactively explore the selected KG by selecting and viewing entities.

**Entity Index.** For each KG, an entity index is built to support efficient filtering of entities, which contains two parts to map the classes and properties to their entity instances. This index is also implemented using Lucene.

**Faceted Entity Search.** As shown in Figure 14, supported by the entity index, the user is provided with a panel to choose classes and properties instantiated in the KG. Then the selected classes and properties are used as a filter with AND as the Boolean operator, to filter entities in the KG. Filtered entities are returned as a list, and each of them can be browsed by clicking.

**Entity Browsing.** For each filtered entity, all the triples describing it are retrieved by CKGSE and presented as a node-link diagram. It depicts the neighborhood of this entity in the original KG. Further, by switching between entities in this diagram, the user is able to explore any part of the KG according to the interest.

## 5. EXPERIMENTS

We implemented a prototype of CKGSE on an Intel Xeon E7-4820 (2.0GHz) with 100GB memory for JVM. Our experiments mainly focused on evaluating the practicability of CKGSE. We also presented a case study by comparing CKGSE with the search service provided by the current version of OpenKG.CN, to show the usefulness of the unique features of CKGSE.

### 5.1 Practicability Analysis

#### 5.1.1 KG Crawling and Storage

Table 3 shows the statistics of the 40 KGs crawled from OpenKG.CN. The size and schema vary greatly among them. Some KGs are very large, and the largest KG has a dump file size more than 28GB, being further parsed into more than 150 million triples. Most KGs have more than 1 dump files, and the largest number of dump files of a single KG reached 46. Among the 185 available dump file records, a total of 178 were successfully downloaded, parsed and stored.

**Table 3.** Statistics about the KGs.

| Dump Files (MB) | | # Triples | | # Classes | | # Properties | | # Entities | |
|---|---|---|---|---|---|---|---|---|---|
| median | max | median | max | median | max | median | max | median | max |
| 23 | 28,929 | 68,399 | 151,976,069 | 7 | 20,096 | 23 | 40,077 | 93,268 | 303,952,138 |

The time complexity of the parsing process is O(#Triples). As shown in Table 4, parsing all the 40 KGs affordably finished in one day. Among them, 10% (4/40) were very large and parsed with more than one hour. We observed that the largest KG© reached 8 hours for the parsing process. Based on the parsed KGs, the other indexes were also finished in acceptable time, where larger KGs generally cost longer index time. Therefore, the overall time cost for parsing and indexing the KGs is acceptable in practice.

**Table 4.** Run-time of offline computation (hours).

| Parsing | Inverted Index | Label Index | Quality Profile | Summary Index | Entity Index |
|---|---|---|---|---|---|
| 20.27 | 5.10 | 2.52 | 1.62 | 34.35 | 33.21 |

The space complexity of all the indexes is O(#Triples). Table 5 presents the disk use. In practice, the total size of the triple store and all the indexes is smaller than the size of the original dump files, showing the disk-use efficiency and practicability of CKGSE.

**Table 5.** Disk use of the dump files, parsed triples, and indexes (MB).

| Dump Files | Triple Store | Inverted Index | Label Index | Summary Index | Entity Index |
|---|---|---|---|---|---|
| 54,133 | 35,756 | 1,816 | 9,708 | 165 | 1,129 |

The above analysis of the disk use demonstrates the practicability of CKGSE, and the run-time of the indexes is acceptable in practice. Meanwhile, further optimizations such as parallel indexing should be applied to improve the performances in the future, especially on large KGs.

### *5.1.2 Content-based Keyword Search*

As presented in Table 4, CKGSE spent 5.1 hours building an inverted index for all the 40 KGs to support keyword search over both metadata and KG contents. Note that we did not build them in parallel, which would otherwise be much faster. According to Table 5, the inverted index only takes 1.8GB which is relatively small and affordable.

### *5.1.3 Content-based Snippet Generation*

In addition to the indexed fields, CKGSE also spent 2.5 hours constructing a label index to support the query-relevant snippet generation as shown in Table 4. About half of the whole time, i.e., 1.1 hours, were

© http://www.openkg.cn/dataset/zhishi-me-dump

cost by the largest KG. For the result index which takes 9.7GB (Table 5), about half of it is for the largest KG.

The time complexity of KSD is O($k$·#Triples), where $k$ is a pre-defined size limit. To evaluate the performance of online snippet generation by KSD, we created 10 keyword queries containing 1-5 keywords. Then we retrieved the top-5 relevant KGs for each query. We recorded the run-time of KSD for each of the 50 retrieved KGs. Figure 2 shows the complementary cumulative distribution of the run-rime over all these KGs. The median run-time is only 1 second, while for 12% (6/50) KGs the run-time exceeded 10 seconds. It suggests that though KSD is fast enough for most KGs, further optimization is still needed, especially for large KGs.

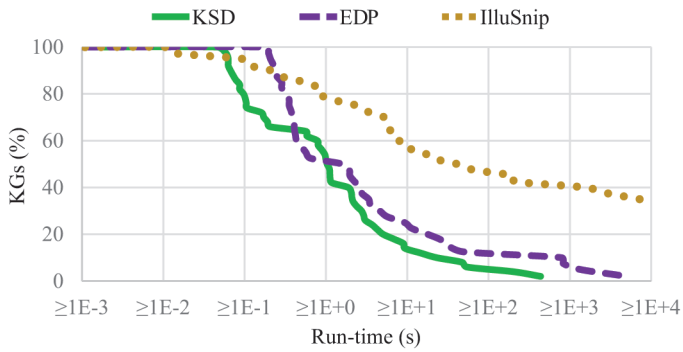


**Figure 2.** Complementary cumulative distribution of run-time.

*5.1.4 Content Profiling*

As shown in Table 4, the content profiling component has two major time cost, for the quality profile, and the summary index.

For the quality profiles, each metric related to metadata (i.e., Availability, Licensing and Timeliness) has a time complexity of O(1), while each metric related to content (i.e., intra-KG interlinking, inter-KG interlinking and understandability) requires a time complexity of O(#Triples). CKGSE spent 1.6 hours computing the values of 6 metrics for all the KGs, which is relatively short.

Figure 3 shows the complementary cumulative distribution of the quality scores over the 40 KGs. According to the three quality metrics for metadata, for most of the KGs, their dump files are available for downloading and parsing. For the licensing score, all the KGs from OpenKG.CN have specific licenses which were recorded in metadata, thus each of them having the licensing score of 1 (for this reason we do not specify its cumulative distribution in the figure). The timeliness score measures to what extent the KG is up-to-date after it was created. However, over half of the KGs were never updated after submission, or their updated time was missing in metadata, thus the timeliness score being undefined and regarded as zero.

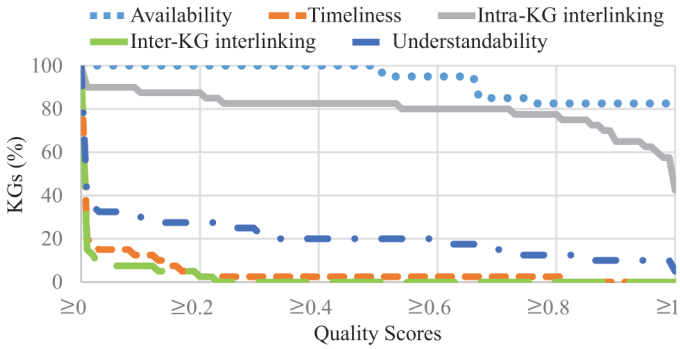


**Figure 3.** Complementary cumulative distribution of quality scores.

For the quality scores about the content, as suggested by the relatively high average intra-KG interlinking score, in most of the KGs, entities are usually linked to others instead of being isolated. On the contrary, inter-KG interlinking scores are commonly low for these KGs. For the understandability, less than half of the KGs use `rdfs:label` to describe entities with human-readable tags, though we observed that some KGs have other properties with similar meaning, such as `http://cndbpedia/ontology/`实体名称(entity name). As this kind of properties vary among different KGs, we did not take them into consideration.

The summary index contains the statistics of elements, the results of abstractive summary and extractive summary for all 40 KGs. It only uses 165 MB for storage as in Table 5 but cost 34 hours for computation as in Table 4. Among the 34 hours, CKGSE spent about 3 hours preparing the statistical summaries, including 2 hours for computing PageRank to identify central entities. For the rest of the time, most was spent on generating the extractive summaries using IlluSnip. We used an anytime version of IlluSnip [34], and allowed it to iteratively find a better summary within a maximum of 2 hours for a single KG. If needed, one could adjust to a smaller time limit to trade between the result quality and the generation time. In our experiments, the median run-time of IlluSnip was 31 seconds. By comparison, generating abstractive summaries (i.e., EDP) was much faster, even being comparable to KSD, as shown in Figure 2.

#### *5.1.5 Content Browsing*

CKGSE spent 33 hours creating an entity index to support faceted entity search as shown in Table 4, although the index was as small as 1.1 GB upon completion according to Table 5. We observed that there is still much room for improving the performance of our trivial implementation of this index, such as using better index structures and/or more efficient algorithms.

### *5.2 Case Study*

We compared the performance of CKGSE with OpenKG.CN (assessed on October 28, 2021) by a case study.

### *5.2.1 Keyword Search*

As shown in Figure 4, given the query "哈利波特人物关系"("relationships between characters in *Harry Potter*"), both OpenKG.CN and CKGSE can successfully find the target KG, since both keywords in the query can be matched by the metadata of this KG.


(a) OpenKG.CN (b) CKGSE

**Figure 4.** Search results pages of OpenKG.CN and CKGSE with regard to the query "哈利波特人物关系" ("relationships between characters in *Harry Potter*"). Each returned entry is a KG retrieved by the query in the top input box.

However, for the query "格兰芬多人物关系"("relationships between characters about Gryffindor"), in which "格兰芬多" ("Gryffindor") refers to an entity but not contained in the metadata of the target KG, only CKGSE found this KG as shown in Figure 5. Thanks to the content-based keyword search whose inverted index covers both the metadata and KG content, CKGSE is distinguished from existing systems.


(a) OpenKG.CN (b) CKGSE

**Figure 5.** Search results pages of OpenKG.CN and CKGSE with regard to the query "格兰芬多人物关系" ("relationships between characters about Gryffindor"). Each returned entry is a KG retrieved by the query in the top input box.

On the results page of CKGSE, each returned KG is presented with indexed fields including metadata and content ones. In each field the query-relevant words are highlighted in red, as shown in Figure 4(b) and Figure 5(b).

### *5.2.2 Query-relevant Snippet Generation*

On the search results pages, OpenKG.CN and other existing systems only show some metadata for each top-ranked KG. More than that, CKGSE gives a query-relevant snippet including both indexed fields and an extracted sub-KG, as shown in Figure 6.

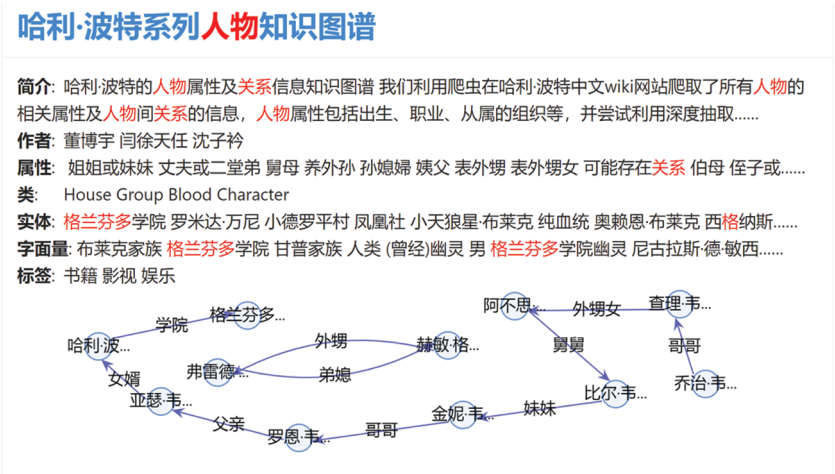


**Figure 6.** Query-relevant snippet. The title of the KG is "*Harry Potter* Character Relationship". The upper part of the figure provides query-related indexed fields and the lower part of this figure shows an extracted sub-KG.

The generation of this sub-KG is biased towards the keyword query, e.g., containing the "格兰芬多" "Gryffindor" entity mentioned in the query. Therefore, it can help the user quickly and accurately judge the relevance of the underlying KG to the query even before browsing its full content which could be a time-consuming process.

### *5.2.3 Profiling and Browsing*

When a KG is selected, in addition to the metadata information usually shown by OpenKG.CN and other existing systems as in Figure 7, CKGSE further presents a quality profile and content summaries for the KG.

Currently, the quality profile of each KG is presented as a table of metric values in CKGSE. As shown in Figure 8, each metric corresponds to an aspect relevant to search. Not only as quality signals for the user, this quality profile could also advise the search process. As a future direction, we will consider incorporating the quality metrics into the KGs ranking function.

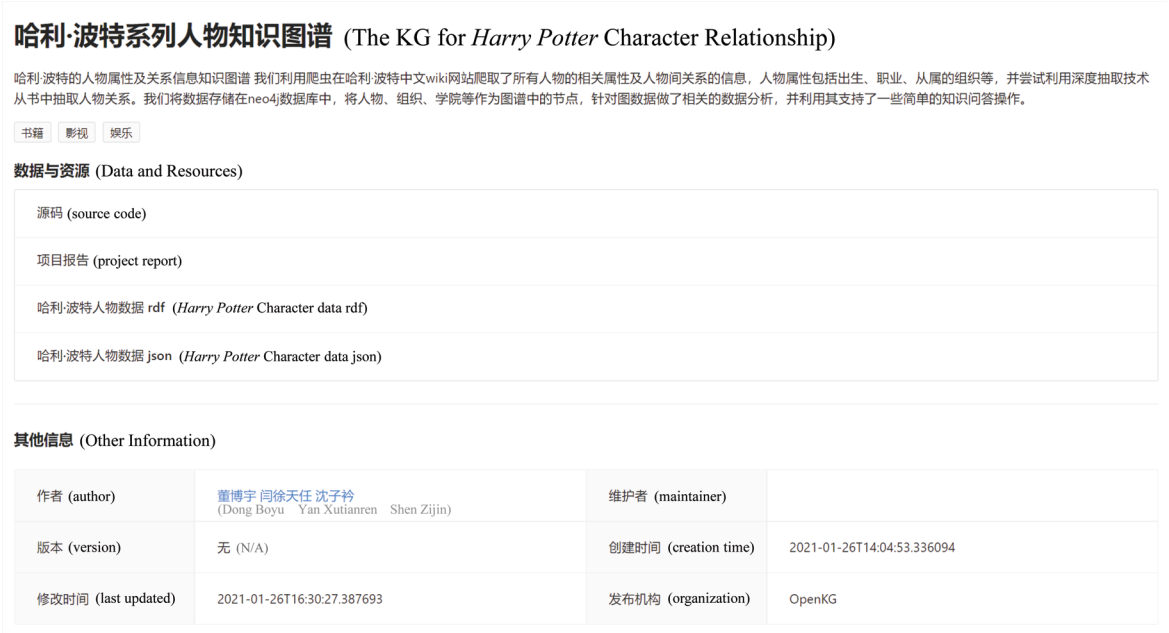


**Figure 7.** Metadata information.


**Figure 8.** Quality profile.

The statistical summary consists of basic statistics of the KG such as triple count as shown in Figure 9, and element-level content distributions with visualizations. Figure 10 shows the distribution of all properties instantiated in the KG, visualized as a pie chart. An entity tag cloud is visualized in Figure 11 ranked by the PageRank scores computed over the KG. Such a statistical summary provides the user with a brief overview of the KG content.


| 统计量 (Statistics) | 数量 (Count) |
|---|---|
| 三元组 (triple) | 7585 |
| 类 (class) | 4 |
| 属性 (property) | 159 |
| 实体 (entity) | 11201 |
| 字面量 (literal) | 1920 |
| sameAs链接 (sameAs link) | 0 |

**Figure 9.** Statistical summary: basic statistics.

| 属性 (Property) | 实例数量 (Count) |
|---|---|
| 从属 (provenance) | 957 |
| type | 956 |
| name | 956 |
| 物种 (species) | 660 |
| 性别 (gender) | 642 |
| 出生 (birth) | 456 |
| 职业 (profession) | 416 |
| 血统 (lineage) | 287 |
| 头发颜色 (hair color) | 198 |
| 逝世 (death) | 197 |

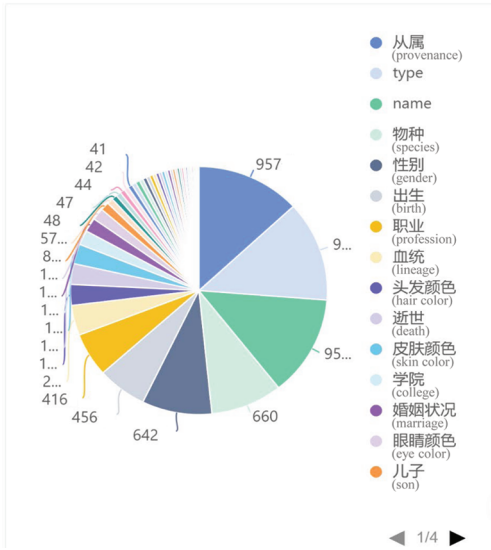


**Figure 10.** Statistical summary: property distribution.

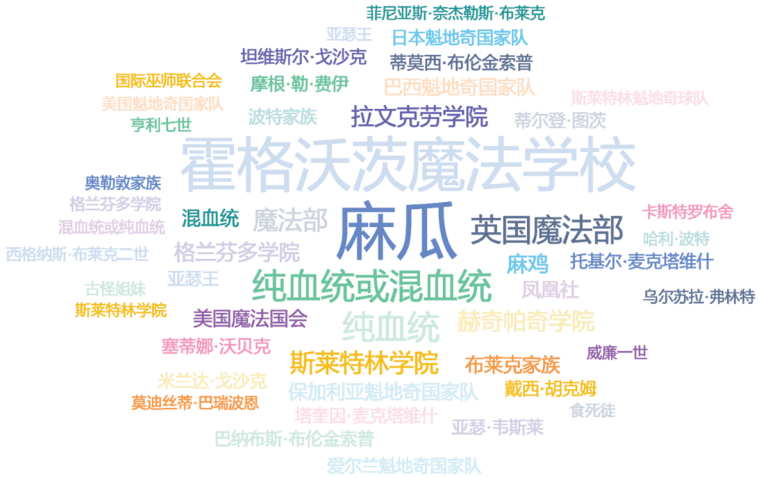


**Figure 11.** Statistical summary: entity tag cloud. Each tag represents an entity, ranked by its PageRank score.

The abstractive summary in Figure 12 describes the KG on the pattern level, by presenting the most frequent EDPs in the KG to show how entities in the KG are described, i.e., by which combinations of classes and properties.

| 实体描述模式 (Entity Description Pattern) | 实体数量 (Count) |
|---|---|
| EDP-0 | 262 |
| EDP-1 | 44 |
| EDP-2 | 30 |
| EDP-3 | 19 |
| EDP-4 | 17 |
| EDP-5 | 13 |
| EDP-6 | 12 |

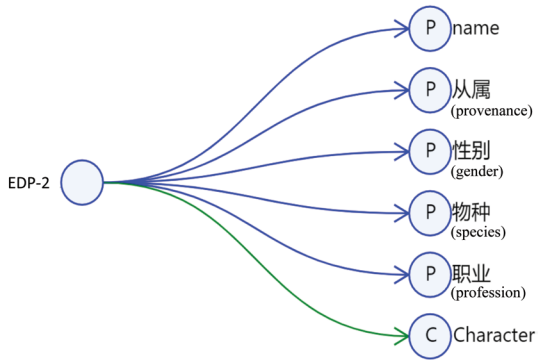


**Figure 12.** Abstractive summary (EDP).

The extractive summary in Figure 13 presents an extracted sub-KG which is different from the snippet on the search results page. The sub-KG here is query-independent but illustrates the most frequent classes and properties in the KG with a few concrete entities and triples. Compared with metadata and statistics, our content summaries provide a distinguishing closer-up view of the KG content, thus assisting the user in comprehending the KG and further judging its relevance before downloading it.

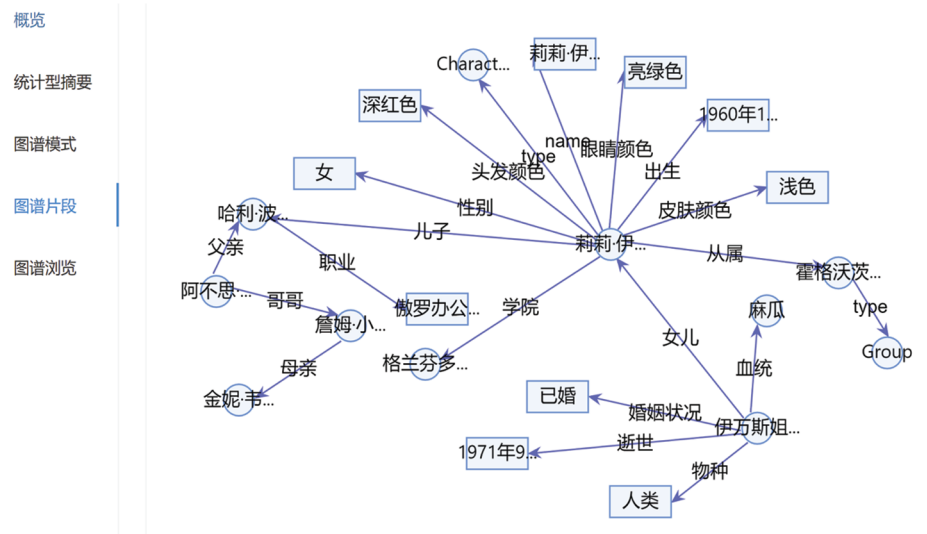


**Figure 13.** Extractive summary (IlluSnip). The extractive summary is visualized as a node-link diagram, where each directed edge represents a triple.

Last but not least, as shown in Figure 14, CKGSE allows the user to interactively browse entities in the KG. The user can select classes and properties to filter entities. For each filtered entity, all its triples are visualized as a node-link diagram. With this simple yet effective browsing interface, for many users they do not need any other tools for KG browsing but can easily investigate the KG content.

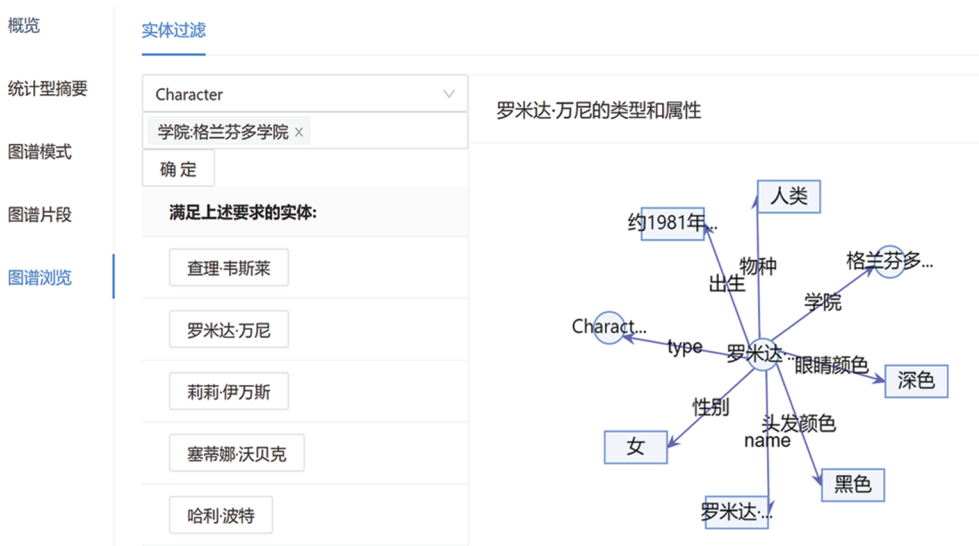


**Figure 14.** Content browsing. The entities are filtered in the left panel and the triples of each filtered entity are visualized on the right of the figure.

### 5.3 User Study

In addition to the practicability analysis and case study, we also conducted a user study to verify the usefulness of CKGSE for search result relevance judgement and KG content comprehension by comparing it with OpenKG.CN. We recruited 20 students majoring in computer science via a mailing list. All of them have necessary background knowledge and research experiences with KG.

#### 5.3.1 Design and Process

Following the general KG search process, each participant was firstly required to form a specific data need to find some target KG. Then the participant was allowed to search using OpenKG.CN and CKGSE separately for the target KG. After viewing the returned results and judging the relevance of each KG, the participant was invited to rate the usefulness of the two systems separately on a 1–5 scale, including how useful the system was in (1) search result relevance judgement, and (2) KG content comprehension. We also asked the participants to select the most useful system component for the two aspects (1) and (2).

#### 5.3.2 Result Analysis

The results of user-rated usefulness are summarized in Table 6. Paired two-sample *t*-test showed that CKGSE received significantly ($p < 0.01$) higher ratings than OpenKG.CN both on search result relevance judgement and KG content comprehension. Most of the participants (65%) gave higher ratings to CKGSE in helping them judge the KG's relevance, and all of them agreed that CKGSE performed better than OpenKG.CN in helping them understand the main content of the KG.

**Table 6.** User-rated usefulness for search result relevance judgement and KG content comprehension.

| | Search result relevance judgement | KG content comprehension |
|---|---|---|
| OpenKG.CN | 3.45 ± 0.84 | 2.15 ± 0.77 |
| CKGSE | 4.15 ± 0.35 | 4.35 ± 0.64 |
| | Proportion | Proportion |
| OpenKG.CN > CKGSE | 10% | 0% |
| OpenKG.CN = CKGSE | 25% | 0% |
| OpenKG.CN < CKGSE | 65% | 100% |

According to the most useful components selected by the participants, 10 (50%) participants who rated 3 or higher for OpenKG.CN in search result relevance judgement generally relied on the title and description in metadata to filter out irrelevant KGs. For CKGSE, all the 20 (100%) participants rated over 3 for relevance judgement. Apart from the title and description, 10 (50%) participants also selected the query-relevant snippet to be especially useful for their judgement. For helping the user understand the KG's main content, OpenKG.CN received relatively low ratings with an average of 2.15, since it could not provide any detailed KG elements or patterns to exemplify the content. Compared to OpenKG.CN, CKGSE was given much better ratings of usefulness for comprehension with an average of 4.35. The participants selected the

statistical summary (50%) and the content browsing component (25%) to be most helpful for them to quickly know about the representative KG elements.

We also interviewed the participants about their likes and dislikes of the two systems. Most participants (85%) preferred CKGSE for it provided more and detailed views to the KG, while some of them also mentioned several limitations, such as the snippet generation could be fastened, and some visualization methods for summaries could be improved.

## 6. CONCLUSION

In this paper we present CKGSE, one of the first content-based search engines for open KGs. Complementary to existing systems only considering the metadata of KGs, by incorporating fields of content-level elements such as classes and properties into the inverted index, CKGSE can handle queries referring to the KG content. To facilitate the user with relevance judgement, CKGSE provides a query-relevant snippet for each KG on the search results page. Apart from metadata information, CKGSE uses a quality profile, content-based summaries, and browsing capabilities to comprehensively present the user with a closer-up view to the KG. We implement a prototype with KGs crawled from OpenKG.CN. Our preliminary experimental results demonstrate the practicability and usability of such a new paradigm for KG search, though the system performance could be further optimized.

Our experiments also uncover some limitations of CKGSE that we should address in the future. First, we will particularly focus on improving the efficiency of processing large KGs, and improve the efficiency of browsing and presenting large KGs by entity summarization techniques [38, 39]. Besides, according to the feedbacks from users, we will improve the overall system performance and design better visualization methods for each component in the future.

## ACKNOWLEDGEMENTS

This work was supported by the Nantional Science Foundation of Chnia (No. 62072224).

## REFERENCES

[1] Deng, C., et al.: GAKG: A multimodal geoscience academic knowledge graph. In: The 30th ACM International Conference on Information and Knowledge Management (CIKM 2021), pp. 4445–4454 (2021)
[2] Dsouza, A., et al.: Worldkg: A world-scale geographic knowledge graph. In: The 30th ACM International Conference on Information and Knowledge Management (CIKM 2021), pp. 4475–4484 (2021)
[3] Schindler, D., et al.: Somesci- A 5 star open data gold standard knowledge graph of software mentions in scientific articles. In: The 30th ACM International Conference on Information and Knowledge Management (CIKM 2021), pp. 4574–4583 (2021)
[4] Shen, Y., et al.: CKGG: A Chinese knowledge graph for high-school geography education and beyond. In: 2021 International Semantic Web Conference, pp. 429–445 (2021)

[5] Larmande, P., Todorov, K.: Agrold: A knowledge graph for the plant sciences. In: ISWC 2021: International Semantic Web Conference, pp. 496–510 (2021)
[6] Dimitrov, D., et al.: Tweetscov19 - A knowledge base of semantically annotated tweets about the COVID-19 pandemic. In: The 29th ACM International Conference on Information and Knowledge Management (CIKM2020), pp. 2991–2998 (2020)
[7] Walsh, B., Mohamed, S.K., Novácek, V.: Biokg: A knowledge graph for relational learning on biological data. In: The 29th ACM International Conference on Information and Knowledge Management (CIKM2020), pp. 3173–3180 (2020)
[8] Dessì, D., et al.: AI-KG: An automatically generated knowledge graph of artificial intelligence. In: 2020 International Semantic Web Conference (Part II), pp. 127–143 (2020)
[9] McCusker, J.P., et al.: Nanomine: A knowledge graph for nanocomposite materials science. In: 2020 International Semantic Web Conference (Part II), pp. 144–159 (2020)
[10] Michel, F., et al.: Covid-on-the-web: Knowledge graph and services to advance COVID-19 research. In: 2020 International Semantic Web Conference (Part II), pp. 294–310 (2020)
[11] Steenwinckel, B., et al.: Facilitating the analysis of COVID-19 literature through a knowledge graph. In: 2020 International Semantic Web Conference (Part II), pp. 344–357 (2020)
[12] Neumaier, S., Umbrich, J., Polleres, A.: Automated quality assessment of metadata across open data portals. ACM Journal of Data and Information Quality 8(1), Article No. 2 (2016)
[13] Brickley, D., Burgess, M., Noy, N.F.: Google dataset search: Building a search engine for datasets in an open web ecosystem. In: The World Wide Web Conference (WWW '19), pp. 1365–1375 (2019)
[14] Pietriga, E., et al.: Browsing linked data catalogs with LODAtlas. In: 2018 International Semantic Web Conference (Part II), pp. 137–153 (2018)
[15] Chen, J., et al.: Towards more usable dataset search: From query characterization to snippet generation. In: The 28th ACM International Conference on Information and Knowledge Management (CIKM2019), pp. 2445–2448 (2019)
[16] Degbelo, A.: Open data user needs: A preliminary synthesis. In: WWW '20: Companion Proceedings of the Web Conference 2020, pp. 834–839 (2020)
[17] Chapman, A., et al.: Dataset search: A survey. The VLDB Journal 29, 251–272 (2020)
[18] Ellefi, M.B., et al.: RDF dataset profiling—a survey of features, methods, vocabularies and applications. Semantic Web 9(5), 677–705 (2018)
[19] Wang, X., et al.: A framework for evaluating snippet generation for dataset search. In: 2019 International Semantic Web Conference (Part I), pp. 680–697 (2019)
[20] Wang, X., et al.: Content-based open knowledge graph search: A preliminary study with openkg.cn. In: China Conference on Knowledge Graph and Semantic Computing (CCKS 2021), pp. 104–115 (2021)
[21] Dutkowski, S., Schramm, A.: Duplicate evaluation - position paper by Fraunhofer Fokus. In: SDSVoc 2016, pp. 1–4 (2016)
[22] Koesten, L., et al.: Everything you always wanted to know about a dataset: Studies in data summarisation. International Journal of Human-Computer Studies 135, Article No. 102367 (2020)
[23] Zaveri, A., et al.: Quality assessment for linked data: A survey. Semantic Web 7(1), 63–93 (2016)
[24] Auer, S., et al.: LODStats – An extensible framework for high-performance dataset analytics. In: Knowledge Engineering and Knowledge Management (EKAW 2012), pp. 353–362 (2012)
[25] Cebiric, S., et al.: Summarizing semantic graphs: A survey. The VLDB Journal 28(3), 295–327 (2019)
[26] Song, Q., et al.: Mining summaries for knowledge graph search. IEEE Transactions on Knowledge and Data Engineering 30(10), 1887–1900 (2018)

[27] Khatchadourian, S., Consens, M.P.: ExpLOD: Summary-based exploration of interlinking and RDF usage in the linked open data cloud. In: Knowledge Engineering and Knowledge Management (EKAW 2010, Part II), pp. 272–287 (2010)
[28] Cheng, G., Jin, C., Qu, Y.: HIEDS: A generic and efficient approach to hierarchical dataset summarization. In: The International Joint Conference on Artificial Intelligence (IJCAI 2016), pp. 3705–3711 (2016)
[29] Zneika, M., et al.: RDF graph summarization based on approximate patterns. In: International Workshop on Information Search, Integration, and Personalization (ISIP 2015), pp. 69–87 (2015)
[30] Zneika, M., et al.: Summarizing linked data RDF graphs using approximate graph pattern mining. In: The 19th International Conference on Extending Database Technology (EDBT 2016), pp. 684–685 (2016)
[31] Wang, X., et al.: BANDAR: Benchmarking snippet generation algorithms for (RDF) dataset search. IEEE Transactions on Knowledge and Data Engineering (2021). Available at: https://ieeexplore.ieee.org/document/9477056. Accessed 20 January 2021
[32] Wang, X., et al.: PCSG: Pattern-coverage snippet generation for RDF datasets. In: 2021 International Semantic Web Conference, pp. 3–20 (2021)
[33] Cheng, G., et al.: Generating illustrative snippets for open data on the Web. In: Proceedings of the Tenth ACM International Conference on Web Search and Data Mining (WSDM 2017), pp. 151–159 (2017)
[34] Liu, D., et al.: Fast and practical snippet generation for RDF datasets. ACM Transactions on the Web13(4), Article No. 19 (2019)
[35] Tian, Y., Hankins, R.A., Patel, J.M.: Efficient aggregation for graph summarization. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data (SIGMOD 2008), pp. 567–580 (2008)
[36] Campinas, S., Delbru, R., Tummarello, G.: Efficiency and precision trade-offs in graph summary algorithms. In: Proceedings of the 17th International Database Engineering & Applications Symposium (IDEAS 2013), pp. 38–47 (2013)
[37] Wang, X., Cheng, G., Kharlamov, E.: Towards multi-facet snippets for dataset search. In: PROFLILES & SemEx 2019, pp. 1–6 (2019)
[38] Liu, Q., et al.: Entity summarization: State of the art and future challenges. Journal of Web Semantics 69, Article No. 100647 (2021)
[39] Liu, Q., et al.: Entity summarization with user feedback. In: European Semantic Web Conference (ESWC 2020), pp. 376–392 (2020)

## AUTHOR BIOGRAPHY

**Xiaxia Wang** is a M.S. student at the Department of Computer Science and Technology, Nanjing University. She received her B.S. degree in Information and Computing Science from Nanjing University of Aeronautics and Astronautics in 2019. Her current research interests include data summarization and semantic search. She has published papers in journals like *IEEE Transactions on Knowledge and Data Engineering*, and conferences including International Semantic Web Conference (ISWC) and ACM International Conference on Information and Knowledge Management (CIKM).

**Tengteng Lin** is a M.S. student at the Department of Computer Science and Technology, Nanjing University. She received her B.S. degree in Software Engineering from Shandong University in 2020. Her research interests include dataset retrieval and ranking.

**Weiqing Luo** is a B.S. student at the Department of Computer Science and Technology, Nanjing University. His research interests include dataset retrieval and ranking.

**Gong Cheng** is an Associate Professor at the Department of Computer Science and Technology, Nanjing University. He received his Ph.D. degree in Computer Software and Theory from Southeast University in 2010. His research interests include Semantic Web and knowledge graph, in particular, knowledge graph search and question answering. He has published more than 50 papers in major venues in these areas such as journals like *IEEE Transactions on Knowledge and Data Engineering*, and conferences including the World Wide Web Conference (WWW) and International Semantic Web Conference (ISWC).

**Yuzhong Qu** is a Professor at the Department of Computer Science and Technology, Nanjing University. He received his Ph.D. degree in Computer Software from Nanjing University in 1995. His research interests include Semantic Web, question answering, and novel software technology for the Web. He has published more than 80 papers in major venues in these areas such as journals like *Journal of Web Semantics* and conferences including the World Wide Web Conference (WWW) and International Semantic Web Conference (ISWC).