PSQS: Parallel Semantic Querying Service for Self-describing File Formats

Chenxu Niu[1], Wei Zhang[2], Suren Byna[3], Yong Chen[1]
[1]Texas Tech University, [2]Lawrence Berkeley National Laboratory, [3]Ohio State University
chenxu.niu@ttu.edu, zhangwei217245@lbl.gov, byna.1@osu.edu, yong.chen@ttu.edu

2023 IEEE International Conference on Big Data (BigData) | 979-8-3503-2445-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/BigData59044.2023.10386205

***Abstract*—Finding relevant datasets can be a time-consuming and challenging task, especially for self-describing file formats. Current solutions use either exact or partial keyword matching approaches to extract and process metadata queries, but they fail to capture semantic relationships between the metadata content and query keywords. To address this challenge, we introduce PSQS, a novel parallel semantic search method for self-describing files. The method leverages parallel processing and kv2vec semantic similarity measures to retrieve semantically relevant data efficiently. Our evaluation against existing metadata search solutions shows that PSQS offers a new, efficient and effective semantic search functionality for various fields where large self-describing files are used, such as scientific data management, leading to more accurate and efficient data retrieval.**

***Index Terms*—Semantic Query, Metadata Management, Parallel Metadata Search**

I. Introduction

Many scientific applications using high-performance computing (HPC) systems nowadays need to deal with massive amounts of experimental, observational, and simulation datasets in self-describing data formats, such as HDF5 [1], ASDF [2],netCDF [3], Zarr [4] and ADIOS-BP [5]. The primary advantage of self-describing file formats is that they are designed to allow scientists to store data and metadata describing the data together using attributes. The metadata in self-describing files provides detailed descriptive information about the internal data objects. Metadata can be accessed as a collection of attributes that are in the form of key-value pairs. Each attribute key-value pair consists of a key representing the attribute name and a value representing the metadata value.

While the metadata can be stored in self-describing files, a capability to search metadata is critical to find the data scientists require for their analysis. Searching for scientific datasets in a principled way has been researched for decades. With the internal metadata, the problem of searching over self-describing files can be achieved by performing metadata search over the attributes. Over the years, a series of metadata search solutions were proposed to address the self-describing dataset search problem [6]–[10]. In these dataset search solutions, search functionality was achieved by conducting exact or partial *lexical* match between the metadata attributes and queries.

The capability of searching metadata using lexical matching methods is a crucial step. However, critically required functionalities in searching for desired scientific data do not stop there. An important requirement in the scientific discovery process is finding semantically relevant datasets. For instance, in climate datasets, "precipitation", "rain", and "precip" are used to refer to the same measurement, but variables in datasets are named differently, which can be called "semantically synonymous attributes" issue. With this issue, searching for lexical matching of these datasets can result in any one of those, but not all unless a user is aware of both attributes. Our preliminary study on 50 real-world scientific datasets shows that more than 14% of attribute names have other synonyms in different datasets.

Due to the specific structure of the metadata in current self-describing files, such approaches often need scientists to navigate through dataset schema and understand the metadata attributes to be able to query and search interested datasets. It also results in a barrier of preventing scientists to mine a large number of datasets as these datasets often have inconsistent metadata attributes and naming schemas. If users fail to find the desired self-describing datasets, they may spend much more inordinate time just manually locating the attributes or rely on external unnecessary resources or repositories to assist, even before starting the experiments.

Semantic search to understand a searcher's intent has been an extensively studied topic for web documents [11]–[14]. The attention paid to the semantic search in web documents inspires our main research objective in this investigation: *Can we move beyond lexical matching and improve scientific dataset search capability by incorporating semantic matching for self-describing file formats?* Admittedly, due to the lack of semantic cues in the metadata attributes of self-describing file formats, the semantic solutions in information retrieval and semantic web are not directly applicable (or too complex in many cases), even though these semantic search solutions have been well studied for years. On one hand, the techniques in information retrieval are mainly about building linked data based on domain-specific definition of ontology and RDF [11], [12], [15]. However, most scientific datasets lack the linkage information and the ontology in self-describing files. On the other hand, the techniques in NLP (natural language processing) [16], [17] are designed to capture the grammatical and linguistic characteristics from the contextual sentences, paragraphs or articles out of datasets. The metadata attributes in self-describing data formats are in the form of key-value pairs, not in phrases or sentences. Thus, it is indeed challenging to build a semantic metadata search solution for self-describing files. Recently, a novel semantic method called `kv2vec` [18]

has been proposed to convert key-value pairs into embeddings. It can be utilized to map attributes to semantic embedding.

In this study, based on our previous work, we introduce PSQS - a parallel semantic querying service for self-describing file formats. It aims to capture features from key-value pairs and represent them by semantic vectors. By introducing a recurrent neural network with long short-term memory, we are able to achieve the goal effectively. We have built a prototype of `kv2vec` and conducted extensive experimental tests on metadata attributes of real-world scientific datasets. The results indicate that, with the new `kv2vec` method, the chance of finding semantically relevant key-value pairs is much higher than the existing methods.

The contributions of this research study are:

- We identify limitations of existing methods in representing key-value pairs as semantic vectors.
- We introduce a new distributed representation method, `kv2vec`, for key-value pairs. Based on similarity measures of cosine distance between two vectors, we can measure the similarity of two key-value pairs and perform metadata queries on scientific datasets.
- We develop a prototype implementation of `kv2vec` and test out the implementation to validate our design. The evaluations show that, as compared to most existing solutions, `kv2vec` achieves desired accuracy while being efficient.

II. Background and Motivation

Semantic search methods have been proven to be efficient and effective in enlarging the search coverage and providing relevant datasets at semantic level [12], [13], [19]–[21], which, we consider, is critically needed in improving search capability for self-describing files to discover datasets of interest. In self-describing files, metadata attributes provide detailed descriptions for the data. Therefore, it is natural to consider capturing the semantics or semantic relationships from the attributes.

Capturing semantics from metadata attributes is crucial to semantic dataset search over self-describing data files, but challenges abound, even though the existing semantic search solutions with either entity-based, document-based or context-based are abundant and well studied. First, it is not feasible to apply entity-based semantic search solutions. In the case of self-describing data formats, the metadata attributes may only describe partial properties of a data object, and a data object may not even refer to a real-world entity. Thus, it is hard, if not impossible, to represent multiple self-describing datasets as entities for further processing.

Entity-oriented search solutions, including RDF query engines, are designed to to provide semantic search capabilities by capturing typed relationships and building interconnection for real-world objects. Typical examples can be seen from Swoogle [12], Linked Geo Data [22], Nordlys [20] and Wukong [23]. They are often used to process and publish data online in the form of entities, properties, literals, and, most importantly, links to other resources. The main idea is to identify entities in the textual query or content and to match them to their counterparts in a third-party resource to build data linkage. These links facilitate search and exploration of a global decentralized data space, similar to browsing and navigation on the web. But they may not be directly applicable for scientific datasets since it is difficult to distinguish entities based on attributes and to capture linkage for building RDF relations. For instance, Linked Geo Data [22] uses the public comprehensive spatial data collection to achieve spatial data interconnection. It consists of more than 3 billion entities which map to real-world cities, blocks, streets and other geo elements. Wukong [23] is an efficient distributed graph-based RDF search engine. In general, the entity-oriented search solutions need valid information about the mapping relationship between the indexed data and the real-world entities.

Document-based semantic search solutions are not directly applicable to self-describing data files either. In self-describing data formats, each metadata attribute is named distinctively and the metadata attribute values are mostly distinctive. It is meaningless to capture the statistical relationship between semantics and attribute occurrences, not to mention that each metadata attribute provides little semantics. In addition, it is not feasible to apply context-based approaches to self-describing data files either. The metadata attributes in the self-describing data formats are in the form of key-value pairs, not well-formed sentences or paragraphs. The context of each word in the metadata is disconnected or even missing.

In summary, due to the lack of semantic cues in self-describing data formats, it is highly challenging, if not impossible, to apply existing semantic search solutions to self-describing data formats directly. There is an imperative need to design and develop a methodology to achieve semantic search specifically for self-describing file formats.

III. Methodology

In this section, we will first introduce a Parallel Semantic Querying Service (or PSQS) method for self-describing data formats. We first present an overview of our design, and then introduce semantic index construction process based on `kv2vec` method. After that, we introduce how our method processes and responds to semantic queries using the semantic index.

A. Overview of Our Design

As shown in Figure 1, the fundamental idea of our proposed `PSQS` method is to use `kv2vec` to convert key-value pairs into semantic vectors, instead of using word averaging or sentence extension algorithms. Considering the word order (and linguistic structure in general) and relationships between the keys and values could be important, as many key-value pairs contain multi-word content such as the description of a dataset. For all metadata attribute key-value pairs in a data file, our method yields a cluster of attribute-semantic vectors (namely, a semantic vector cluster) to represent the data file semantically. Given a collection of self-describing data files, we build semantic index by building the semantic vector clusters and finding their centroids for all data files

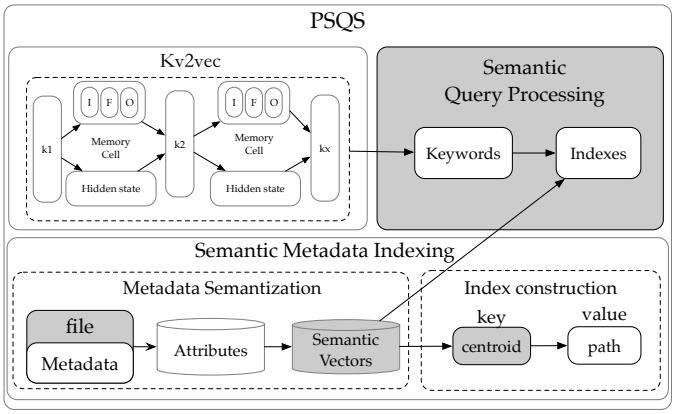

Figure 1: Overview of PSQS workflow. Our method contains three main components. The first component is the kv2vec method to capture the semantics between metadata attribute key-value pairs. The second component is a semantic metadata indexing module which builds indexes and achieves metadata attributes semantization in parallel. The third component is a semantic query processing module to process queries.

in the collection. In other words, the collection of all their centroids serve as the semantic index in our method for the given collection of data files.

In query processing module, by utilizing the same pre-trained dictionary, the query keyword or key-value pair is also converted into a semantic vector to determine the similarity (matching score) between the semantic vector and the vectors of each data object. We describe each component in detail below.

B. Pre-trained Dictionary for Key-value Pairs Based on kv2vec method

Before identifying the meaning of each key-value pair, we need to find out how to create word embeddings. All existing methods for generating distributed representations for words rely on a single assumption: a pre-existing dictionary for most or all words of the key-value pairs is available. Typically, these word representations are learned from a training dataset such as the Common Crawl web corpus (840 billion tokens, 2.2 million words) or Wikipedia 2014 (6 billion tokens, 400,000 words). However, this assumption does not always hold true for key-value pair words in real-world situations, as there may be domain-specific terms in scientific dataset metadata that are not in the pre-trained dictionary. Therefore, we must use a method to extend the dictionary to include these terms. In this article, we will demonstrate how to use retrofitting techniques to solve this problem.

In our kv2vec method, we propose the use of Recurrent Neural Networks (RNNs) with Long Short-Term Memory (LSTM) hidden units, commonly referred to as LSTM-RNNs. RNNs are a type of artificial network designed for sequential or time series data. They use the same task for each element in a sequence, and the output is dependent on previous computations. Like traditional feed-forward and Convolutional Neural Networks (CNNs), RNNs learn from training data. What sets them apart is their "memory", as they take into account prior inputs to influence the current input and output. Unlike traditional deep neural networks, which assume independence between inputs and outputs, the output of RNNs is dependent on previous elements in the sequence. In this case, the network will consider previous words when processing each individual word of a "key" or "value". This allows us to capture relationships and unify "key" words into meaningful phrases rather than unrelated words.

The kv2vec method comprises four distinct layers: an input layer, a word embedding layer, a composition layer, and an output layer. The input is each key-value pair, which is split into its "key" and "value" parts. Subsequently, all the words are extracted from both the "key" and "value" and matched with the pre-trained and retrofitted dictionary to create word vectors. Finally, these word vectors serve as inputs for the composition layer, which is implemented as a long short-term memory (LSTM) network, to generate the outputs. The use of RNN with LSTM in the kv2vec method provides efficiency in handling the vanishing gradient problem faced by traditional RNNs. LSTMs are specifically designed to tackle the issue of long-term dependency, and they have the ability to effectively remember information over extended periods of time, which is their default behavior. All recurrent neural networks have the form of a chain of repeating modules of neural network. In the standard RNNs, this repeating module will have a simple structure, such as a single tanh layer. It will combine all the words and convert them to one dense vector with the same dimension as the inputs. After the process of the composition layer, we can achieve the latest hidden state and the output vector to represent the entire key-value pair.

C. Parallel Semantic Index Construction and Query

Similarity query is the family of queries based on some similarity metrics, like top-k queries where the results are ranked by their similarity to the queried object and similarity join queries where records from two datasets are joined by their similarities between each other instead of by specific keys. While the computations of querying and indexing described in the previous subsection are independent, the I/O operations of reading/writing indexes on a file could require coordinations that are imposed by the underlying I/O library, such as NetCDF or HDF5. For instance, a file library may require data to be read or write collectively by all the processors participating in the file.

During the metadata semantization process, the metadata attributes are first extracted from the self-describing files. Each metadata attribute is then mapped to a semantic vector with the kv2vec generated during the metadata semantization process. PSQS performs a metadata scanning procedure when building a metadata index for the first time on a set of data files. To optimize for parallel environments in HPC systems, PSQS leverages parallelism from the index construction phase. In a parallel application with n processes, each process has a file counter (file_counter) that is incremented each time a file

is encountered during the scanning process. The decision on whether or not a file should be scanned can be determined using the equation:

$$process = file\_counter\%n \tag{1}$$

file_counter mod n = r, where r ranges from 0 to n-1. The equation process = r determines whether process r will scan the metadata of the encountered file or skip it and continue to the next one. By using this method, PSQS can take advantage of parallelism across different data files and prevent I/O conflicts that can occur when multiple processes access the same data file simultaneously.

After the recursive scan is completed, our method extracts the metadata from all data objects and stores the path of the metadata and the objects. Throughout the metadata semantization procedure, the metadata attributes are transformed into an attribute-semantic vector in a d-dimensional vector space using the $n \times d$ `kv2vec` previously trained in the pick the i-th row of the `kv2vec` to be the semantic vector of the attribute. Note that each semantic vector contains d elements.

Once the metadata is extracted and the metadata semantization is completed, the index constructor retrieves the semantic vectors of the metadata attributes and forms clusters of semantic vectors along with the corresponding data object and file paths to create the semantic index. In PSQS, the index contains: two path lists serving as mapping tables between object path and semantic vectors. We use this data structure to store both global file path and global object path mapping tables. In this study, we use the linked list and vectors to implement this functionality.

Evaluating the similarity accuracy of key-value pairs is a crucial task, yet it presents a challenge as finding a concrete and uniform metric is not straightforward. Our aim is to quantify how accurately the vector representations capture human-perceived similarity, and validate the distributional hypothesis that the meaning of key-value pairs is dependent on the context in which they occur. The cosine similarity is a commonly used evaluation tool and is incorporated in our methodology. This metric normalizes its scores based on the length of the vectors, making it scalable and computationally efficient. For clustering semantic vectors, we choose cosine similarity as our similarity measure to determine the similarity between two semantic vectors. As cosine similarity measures the angle between two vectors and hence can fit in both Euclidean and discrete versions of Euclidean spaces, we consider that the cosine similarity is well suited for measuring the difference between two vectors in multi-dimensional space.

$$\vec{c} = \frac{\sum_{i=1}^{num} \vec{a_i}}{num} \tag{2}$$

The centroid serves as the key of the semantic index and the value of the semantic index contains the object paths and the file paths of the corresponding files.

Indexes can be stored along with datasets as additional metadata for further processing and querying after they are created. It also follows the self-contained data management features of self-describing file formats. The index size is related to the number of metadata attributes. Based on the clustering mechanism, the file sizes of indexes are often in the range of 10MB to 100MB for a self-describing file with about 10k attributes, which is lightweight along with large-scale dataset. The individual indexes are beneficial for parallel applications because they are independent to each other. Note that querying and processing can be naturally parallelized based on index files.

IV. Evaluation

In this section, we evaluate the effectiveness of semantic query functionality of PSQS on self-describing data files. We contrast our approach with existing metadata search solution, including MIQS and MongoDB based solution.

A. Experiment Setup

The evaluation was performed on the Quanah cluster at Texas Tech University's High Performance Computing Center (HPCC). The Quanah cluster comprises 467 worker nodes, each with 36 cores, for a total of 16,812 cores, and is equipped with an Intel Omni-Path 100 Gbps fabric for MPI computing. Each node has dual-18-core Broadwell Xeon processors, 36 cores per node, and 192 GB of memory. The Quanah cluster has a peak performance of 485 Teraflops/s, as indicated by HPL benchmark results. It operates on a CentOS 7 Linux software environment managed by OpenHPC and uses the Lustre file system for home, work, and scratch spaces.

B. Datasets

The HDF5 file format is a typical example of a self-describing file format. For the evaluation, we gathered three sets of real-world HDF5 files: 1,987 from the National Snow and Ice Data Center (NSIDC) [24], 1,285 from the Sea Ice Altimetry Data Center (SIADC) [25], and 100 large-scale files from the Baryon Oscillation Spectroscopic Survey (BOSS) [26]. The file sizes in NSIDC and SIADC range from 0.8 MB to 2.5 MB, with the number of metadata attribute key-value pairs per file ranging from 2,000 to 5,000. For the BOSS datasets, the file sizes range from 500 MB to 1.35 GB, with the number of metadata attribute key-value pairs per file ranging from 2,000,000 to 3,700,000. We believe this collection of datasets is sufficient and diverse enough for our evaluation.

We conducted three series of experiments using three sets of scientific datasets. The first series of experiment is to evaluate metadata semantization on pre-trained dictionary. The second experiment focused on evaluating the overhead, including storage usage and pre-processing time. The third experiment aimed to assess semantic functionality, including query hit rate (as a percentage), recall and query performance with a MongoDB-based metadata search solution.

C. Evaluation of Search Functionality

Evaluating search functionality is the most important because it helps to assess the effectiveness and efficiency of

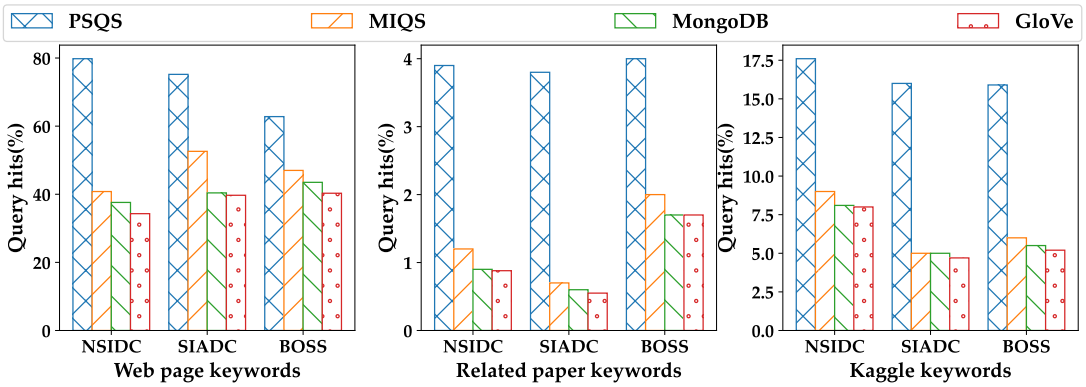

Figure 2: Query hits comparison of these four methods for three collections of real-world scientific datasets.

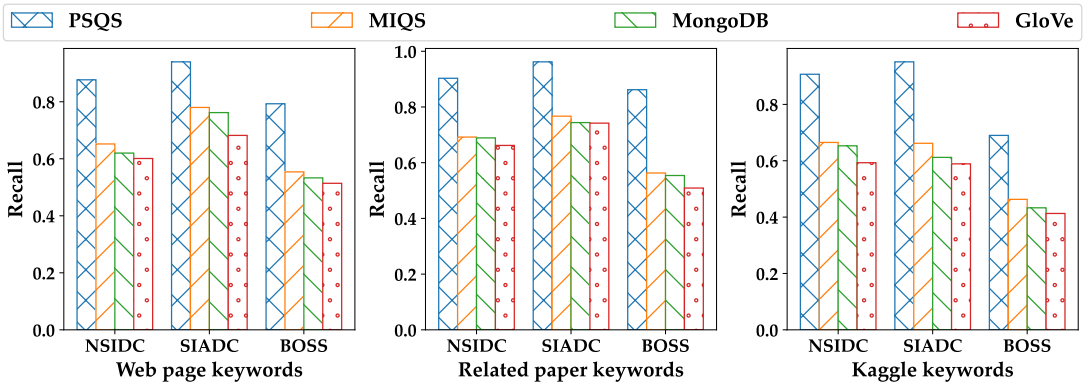

Figure 3: Recall comparison of these four methods for three collections of real-world scientific datasets.

a search system in finding relevant information. The two main metrics used for this evaluation are query hit rate and recall. Query hit rate measures the percentage of queries that return relevant results, while recall measures the proportion of relevant results that are retrieved by a query. These metrics help to determine the precision and recall of our search service, respectively, and provide insights into how well the service system is able to retrieve relevant information from large datasets. Overall, evaluating search functionality helps to ensure that a search system is able to deliver accurate and efficient results to researchers.

1) Query hits comparison: PSQS allows users to search for datasets of interest by querying metadata attributes. To test its effectiveness, we collected 10,000 key-value pairs as query inputs and compared it with other methods (MIQS, MongoDB-based search, and GloVe). To better compare them, we extracted all words from the content and randomly selected 1,000 words as keywords to perform queries with the other three methods for the query hits comparison. The cosine similarity between each metadata attribute and the query input was calculated, and the most similar result was returned. The results showed that PSQS matches 80% of the keywords, while the other methods only return less than 45%. The output of PSQS also includes results from the other methods, demonstrating its potential for real-world use cases.

The results of tests with two additional sets of datasets (related papers and Kaggle) were consistent with our findings. We also used publications that cited these datasets to evaluate the performance of PSQS. Random keywords were selected from these publications, and their semantic relevance was considered. Kaggle, a world-leading data science community, was searched using the name of these datasets. The results showed that PSQS outperformed the other methods (MIQS, MongoDB-based, and GloVe) by matching more than 15% of the NSIDC datasets, while the others only matched less than 10%. These results suggest that PSQS has better coverage and is more likely to return the desired datasets.

2) Recall comparison: The recall of the four methods (PSQS, MIQS, MongoDB-based, and GloVe) was computed based on the results of the previous experiment, as shown in Figure 3. The recall was computed using the same set of query keywords on all three datasets. The results indicate that PSQS outperforms the other methods, with a recall of over 80% for NSIDC datasets, while the others have a recall of roughly 60%. This trend was consistent for the other two datasets, demonstrating that PSQS has a superior ability to capture semantic relationships between queries and metadata attributes. Our method was further tested with various keywords from different sources on the three datasets and consistently outperformed the other solutions in terms of query

hits and recall. This confirms its efficiency in semantic search.

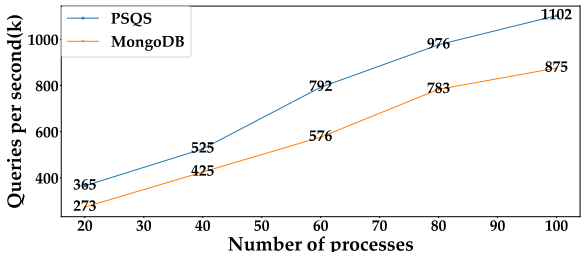

Figure 4: Query performance comparison of PSQS and MongoDB-based method

3) Query Performance: Query performance is obviously critical for metadata searches. Our method, which utilizes semantic indexes, provides an efficient metadata search with a consideration for semantics. Figure 4 shows the results of our comparison of query performance between our method and the MongoDB-based solution. Our PSQS method achieved a query throughput of 365k per second with 20 processes, while the MongoDB-based solution only managed 273k. The query throughput of our method increased linearly with the number of processes, reaching 1102k, without any reliance on external databases. In comparison, the MongoDB-based solution only increased its query throughput from 273k to 875k when 20 processes became 100 processes, resulting in a 25.94% lower average throughput. The PSQS method consistently delivered higher query throughput compared to the MongoDB-based metadata search solution.

D. Summary of Evaluation

Our evaluation highlighted the key advantages of our method, including its semantic querying capability, effective semantic index mechanism, and efficient query performance. The results confirmed that our method increases search coverage, locates semantically relevant results for metadata searches, and boasts impressive query performance.

V. Conclusion

In this paper, we present a novel parallel semantic querying service (PSQS) for self-describing file formats. It integrates key-value pair vectors into metadata search solutions and enables semantization of metadata through the `kv2vec` method, resulting in semantic indexes rather than relying on traditional lexical match-based solutions. Our extensive evaluations demonstrate that PSQS outperforms existing database management system-based metadata search solutions in the semantic search of self-describing files. To our knowledge, this is the first study to offer parallel semantic search for scientific datasets and has the potential to enhance data management functionality and improve the efficiency of searching and discovering scientific datasets through semantic meaning in the HPC community.

References

[1] M. Folk, A. Cheng, and K. Yates, "HDF5: A File Format and I/O Library for High Performance Computing Applications," in *Proceedings of supercomputing*, vol. 99, 1999, pp. 5–33.

[2] P. Greenfield, M. Droettboom, and E. Bray, "ASDF: A New Data Format for Astronomy," *Astronomy and Computing*, vol. 12, pp. 240–251, 2015.

[3] R. Rew, "NetCDF: An Interface For Scientific Data Access," *IEEE computer graphics and applications*, vol. 10, no. 4, pp. 76–82, 1990.

[4] F. Alted, M. Durant, S. Hoyer, J. Kirkham, A. Miles, M. Ratsimbazafy, M. Rocklin, V. Schut, A. Scopatz, and P. Goel, "Zarr," https://zarr.readthedocs.io, 2018.

[5] J. F. Lofstead and et al., "Flexible I/O and Integration for Scientific Codes through the Adaptable I/O System (ADIOS)."

[6] J. G. Institute, "The JGI Archive and Metadata Organizer(JAMO)," http://cs.lbl.gov/news-media/news/2013/new-metadata-organizer-streamlines-jgi-data-management, 2013.

[7] T. Craig E., E. Abdelilah, G. Dan *et al.*, "SPOT Suite," http://spot.nersc.gov/, 2013.

[8] H. Sim, Y. Kim, S. S. Vazhkudai, G. R. Vallée, S. Lim, and A. R. Butt, "Tagit: An Integrated Indexing and Search Service for File Systems," in *SC conference*, 2017, pp. 5:1–5:12.

[9] H. Tang, S. Byna, B. Dong, J. Liu, and Q. Koziol, "SoMeta: Scalable Object-centric Metadata Management for High Performance Computing," in *CLUSTER Conference*. IEEE, 2017, pp. 359–369.

[10] W. Zhang, S. Byna, H. Tang, B. Williams, and Y. Chen, "MIQS: Metadata Indexing and Querying Service for Self-describing File Formats," in *SC Conference*, 2019, pp. 1–24.

[11] S. Gupta and et al., "Karma: A System for Mapping Structured Sources Into the Semantic Web," in *Extended Semantic Web Conference*. Springer, 2012, pp. 430–434.

[12] L. Ding and et al., "Swoogle: A Search and Metadata Engine for the Semantic Web," in *Information and Knowledge Management Conference*, 2004, pp. 652–659.

[13] S. Zhang and K. Balog, "Ad Hoc Table Retrieval Using Semantic Similarity," in *World Wide Web Conference*, 2018, pp. 1553–1562.

[14] A. Chapman, E. Simperl, L. Koesten, G. Konstantinidis, L.-D. Ibáñez, E. Kacprzak, and P. Groth, "Dataset Search: A Survey," *The VLDB Journal*, vol. 29, no. 1, pp. 251–272, 2020.

[15] M. Aartsen and et al., "Search for Sources of High-Energy Neutrons with Four Years of Data from the IceTop Detector," *The Astrophysical Journal*, vol. 830, no. 2, p. 129, 2016.

[16] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global Vectors for Word Representation," in *EMNLP Conference*, 2014, pp. 1532–1543.

[17] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," in *ICLR Conference*, 2013, pp. 1532–1543.

[18] C. Niu, W. Zhang, S. Byna, and Y. Chen, "Kv2vec: A distributed representation method for key-value pairs from metadata attributes," in *2022 IEEE High Performance Extreme Computing Conference (HPEC)*. IEEE, 2022, pp. 1–7.

[19] D. Brickley, M. Burgess, and N. Noy, "Google Dataset Search: Building a Search Engine for Datasets in an Open Web Ecosystem," in *The World Wide Web Conference*, 2019, pp. 1365–1375.

[20] F. Hasibi and et al., "Nordlys: A Toolkit for Entity-Oriented and Semantic Search," in *SIGIR Conference on Research and Development in Information Retrieval*, 2017, pp. 1289–1292.

[21] W. Hua, Z. Wang, H. Wang, K. Zheng, and X. Zhou, "Short Text Understanding Through Lexical-Semantic Analysis," in *Data Engineering*. IEEE, 2015, pp. 495–506.

[22] C. Stadler, J. Lehmann, K. Höffner, and S. Auer, "Linkedgeodata: A core for a web of spatial open data," *Semantic Web*, vol. 3, no. 4, pp. 333–354, 2012.

[23] J. Shi, Y. Yao, R. Chen, H. Chen, and F. Li, "Fast and concurrent {RDF} queries with rdma-based distributed graph exploration," in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, 2016, pp. 317–332.

[24] "National Snow and Ice Data Center, a part of CIRES at the University of Colorado Boulder," https://nsidc.org/.

[25] "National Snow and Ice Data Center DAAC," https://www.earthdata.nasa.gov/eosdis/.

[26] D. J. Schlegel, M. Blanton, D. Eisenstein, B. Gillespie, J. Gunn, P. Harding, P. McDonald, R. Nichol, N. Padmanabhan, W. Percival *et al.*, "Sdss-iii: The baryon oscillation spectroscopic survey (boss)," in *Bulletin of the American Astronomical Society*, vol. 39, 2007, p. 966.