Metagraph Representation for Overcoming Limitations of Existing Knowledge Bases

Valery Terekhov, Yury Gapanyuk, Anton Kanev
Bauman Moscow State Technical University
Moscow, Russia
terekchow@bmstu.ru, gapyu@bmstu.ru, aikanev@bmstu.ru

Abstract—**Currently, knowledge bases are actively used in many applications, such as search engines, dialog assistants. Primarily, it happens due to the improvement of existing methods through the analysis of meaning. Knowledge bases have different knowledge representations. Semantic networks, frames, semantic web, production rules are used for this purpose. But existing methods of knowledge representation have limitations. Thus, RDF is a flat model, and production rules contradict each other. Therefore, it is proposed to apply metagraphs to combine the advantages of different representations. The paper describes the emergence feature of this method for complex knowledge representation. An analytical assessment of the search time is carried out for a concept in the knowledge base. The authors implemented the proposed method of representation in a knowledge base management system and carried out an experiment to evaluate its performance.**

I. Introduction

Various ways of representing knowledge have drawbacks that the authors propose to overcome by using the metagraph model for knowledge base. It is a version of semantic network extended with metagraph model and proposed representation of concepts based on their extension and intension. The purpose of the study is the description for the metagraph representation of knowledge as well as an analytical and experimental assessment for the time of concept search in the knowledge base.

A. Knowledge representations

Knowledge bases expand the capabilities of dialog assistants and search engines. They improve the quality of their work through the analysis of meaning. The knowledge base includes two components: knowledge representation and an inference engine [1]. The knowledge representation requires a specific format. For example, a semantic network, frames, production rules can be used.

The production model of knowledge representation is based on the application of rules. Rules have a disadvantage when they begin to contradict each other with a large number of rules. One of the ways to overcome this problem is fuzzy inference.

The semantic network [1, 2] is a kind of graph, the vertices of which are concepts, and the edges are the relations between them. An example of a semantic web is WordNet [3]. It describes concepts that are relevant to all subject areas. In WordNet [4], nouns are connected to each other by various types of relationships: synonyms, antonyms, hyperonyms, hyponyms, holonyms, and meronyms.

The frame representation appeared in the 80s. Frame elements contain slots, which in turn can be frames. Thus, frames are networked and can be inherited from each other. Recently, knowledge representation research has been actively pursued with the Semantic Web. And the results of frame languages turned out to be very useful because classifiers enable the network to evolve constantly.

B. Concepts

Knowledge representation is closely related to the term "concept." A concept is a semantic meaning, the content of a sign, also called a significat. The concepts are connected with each other by relations. The most common and the most important relationship between concepts is the generalization one. It shows that a concept is a kind or a subclass of another concept.

Thus, a semantic network with generalization concepts is a hierarchical structure, where at the very top are the most general concepts, and at the bottom are particular concepts. A generalization relation is a directed edge in a semantic web graph. It is directed from more specific to a more general concept.

Specialized languages, such as SPARQL [5], are often used to access the Semantic Web data. Examples of publicly available knowledge bases are Freebase, DBpedia, Wikidata. Their data can be used to create other knowledge bases. But they are mostly limited to data from the Wikipedia project and other public projects, and there are also problems with disambiguation analysis. An important advantage of the semantic network is its independence from the language, while the concepts of ontology have linguistic meanings.

C. Relation types

A hyponym is a private concept to another one in relation. A hyponym is the result of a logical operation of limitation, that is, the addition of a new feature to the concept content. A hyperonym is a concept with a more general meaning. A hyperonym is the result of a logical generalization operation, that is, the removal of some feature to obtain a more general concept. Some semantic networks can only be built from hyperonyms and hyponyms [1]. A holonym is a concept that denotes a whole concept that includes another one as a part.

Meronym is an integral part of the whole concept. It is called, in another way, a partonym.

With the help of the semantic web, two essential aspects of a concept can be identified. It has the intension and the extension. The extension is a set of concepts that are more specific to the considered one. The extension means the scope of a concept, a list of objects that can be designated by a given concept, their varieties. The intension is the content of a concept, a list of features that characterizes this concept and distinguishes it from the others. Moreover, the scope of the concept and its content are connected by the law of the inverse relationship. The more content, the less the volume of the concept and vice versa.

D. Complex networks

According to [6]: "a complex network is a graph (network) with non-trivial topological features – features that do not occur in simple networks such as lattices or random graphs but often occur in graphs modeling of real systems." The terms "complex network" and "complex graph" are often used synonymously.

According to [7]: "the term 'complex network,' or simply 'network,' usually refers to real systems while the term 'graph' is generally considered as the mathematical representation of a network." In this article, we also acknowledge these terms synonymously.

One of the most important types of such models is "complex networks with emergence." The term "emergence" is used in general system theory. The emergent element means a whole that cannot be separated into its component parts. As far as the authors know, currently, there are two "complex networks with emergence" models that exist: hypernetworks and metagraphs.

In this article, we will consider the features of using the metagraph model as a data model in text search tasks.

II. The Metagraph Model vs. the Hypernetwork Model

A. Hypernetwork model

This section briefly discusses the advantages of the metagraph model over the hypernetwork model based on our articles [8, 9].

The main element of the hypernetwork model is hypergraph. According to [10], the hypergraph may be defined as follows:

$$HG = \langle V, HE \rangle, v_i \in V, he_j \in HE, \tag{1}$$

where HG – hypergraph; V – set of hypergraph vertices; HE – set of non-empty subsets of V called hyperedges; v_i – hypergraph vertex; he_j – hypergraph hyperedge.

It should be emphasized that in accordance with (1), the hyperedge inclusion operation is not explicitly defined. Thus, the hypergraph itself is a near flat graph model that does not fully implement the emergence principle. However, the hypergraph model is the basis for a more complex hypernetwork model.

Consider the version of the hypernetwork model that was proposed by Professor Jeffrey Johnson in his monography [11]. Given the hypergraphs $PS \equiv WS_0, WS_1, WS_2, \ldots WS_K$. The hypergraph $PS \equiv WS_0$ is called the primary network. The hypergraph WS_i is called a secondary network of order i.

B. Hypersimplex

The main idea of Professor J. Johnson's variant of the hypernetwork model is the idea of hypersimplex (the term is adopted from polyhedral combinatorics). According to [11], a hypersimplex is an ordered set of vertices with an explicit n-ary relation, and a hypernetwork is a set of hypersimplices. In a hierarchical system, the hypersimplex combines k elements at level N (base) with one element at level $N+1$ (apex). Thus, hypersimplex establishes an emergence between two adjoining levels. An example of a hypernetwork is shown in Fig. 1.

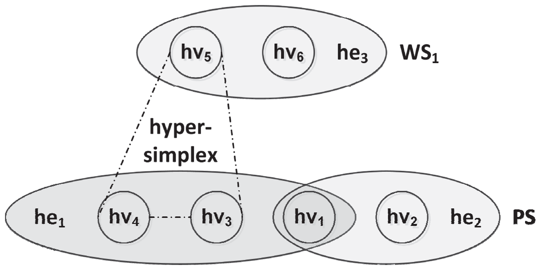

Fig. 1. The example of the hypernetwork

The primary network PS is formed by the vertices of hyperedges he_1 and he_2. The first level WS_1 of the secondary network is formed by the vertices of hyperedge he_3. The hypersimplex is emphasized with the dash-dotted line. The hypersimplex is formed by the base (vertices v_3 and v_4 of PS) and apex (vertex v_5 of WS_1). Unlike the relatively simple hypergraph model, the hypernetwork model is a full model with an emergence.

C. Metagraph description

Unlike the hypernetwork model, the metagraph model does not explicitly use hypergraphs. Let's consider the features of the metagraph model, based on articles [8, 9, 12]. Metagraph has four elements (2).

$$MG = \langle V, MV, E, ME \rangle, \tag{2}$$

where MG – metagraph; V – set of metagraph vertices; MV – set of metagraph metavertices; E – set of metagraph edges; ME – set of metagraph metaedges.

Metaedge is an optional element of the metagraph model aimed for process description and is not considered for data model purposes. Metagraph vertex is described by a set of attributes (3):

$$v_i = \{atr_k\}, v_i \in V, \tag{3}$$

where v_i – metagraph vertex; atr_k – attribute.

Metagraph edge (4) is described by a set of attributes, the source and destination vertices, and the edge direction flag:

$$e_i = \langle v_S, v_E, eo, \{atr_k\} \rangle, e_i \in E, eo = true \mid false, \quad (4)$$

where e_i – metagraph edge; v_S – source vertex (metavertex) of the edge; v_E – destination vertex (metavertex) of the edge; *eo* – edge direction flag (*eo=true* – directed edge, *eo=false* – undirected edge); atr_k – attribute.

The metagraph fragment is described as (5).

$$MG_i = \{ev_j\}, ev_j \in (V \cup E \cup MV \cup ME), \quad (5)$$

where MG_i – metagraph fragment; ev_j – an element that belongs to the union of vertices, edges, metavertices, and metaedges.

The metagraph metavertex has the following structure (6).

$$mv_i = \langle \{atr_k\}, MG_j \rangle, mv_i \in MV, \quad (6)$$

where mv_i – metagraph metavertex belongs to a set of metagraph metavertices *MV*; atr_k – attribute, MG_j – metagraph fragment.

The metavertex can be written as (7).

$$me_i = \langle vs, ve, \{atr_k\}, MG_j \rangle, me_i \in ME, \quad (7)$$

where me_i is a metaedge in a set of metaedges *ME*; *vs* and *ve* is the first and the second vertices of the metaedge, atr_k – attribute, MG_j – metagraph fragment.

D. Metagraph features

Metavertex, in addition to the attributes, includes a fragment of the metagraph. The presence of private attributes and connections for metavertex is a distinguishing feature of the metagraph. It makes the definition of metagraph emergent – metavertex may include a number of lower-level elements and, in turn, may be included in a number of higher-level elements. The example of the data metagraph (shown in Fig. 2) contains three metavertices: mv_1, mv_2, and mv_3.

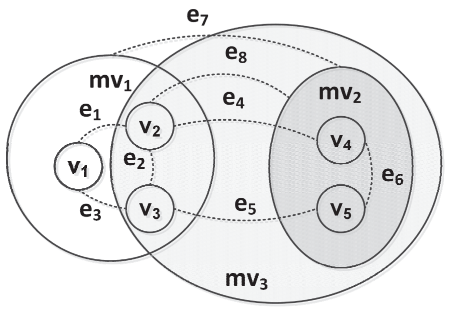

Fig. 2. The example of the metagraph

Metavertex mv_1 contains vertices v_1, v_2, v_3 and connecting them edges e_1, e_2, e_3. Metavertex mv_2 contains vertices v_4, v_5, and connecting them edge e_6. Edges e_4, e_5 are examples of edges connecting vertices v_2-v_4 and v_3-v_5 are contained in different metavertices mv_1 and mv_2. Edge e_7 is an example of the edge connecting metavertices mv_1 and mv_2. Edge e_8 is an example of the edge connecting vertex v_2 and metavertex mv_2. Metavertex mv_3 contains metavertex mv_2, vertices v_2, v_3, and edge e_2 from metavertex mv_1 and also edges e_4, e_5, e_8 showing the emergent nature of the metagraph structure.

E. Comparison metagraph and hypernetwork

Consider the differences between the hypernetwork and metagraph models. According to the definition of a hypernetwork, it is a layered description of graphs. It is assumed that the hypergraphs may be divided into homogeneous layers and then mapped with mappings or combined with hypersimplices. The metagraph approach is more flexible because it allows combining arbitrary elements that may be layered or not using metavertices.

Comparing the hypernetwork and metagraph models, we can make the following notes:

- Hypernetwork model may be considered as "horizontal" or layer-oriented. The emergence appears between adjoining levels using hypersimplices. The metagraph model may be considered as "vertical" or aspect-oriented. The emergence appears at any level using metavertices.
- In the hypernetwork model, the elements are organized using hypergraphs inside layers and using hypersimplices between layers. In the metagraph model, metavertices are used for organizing elements both inside layers and between layers. Hypersimplex may be considered as a particular case of metavertex.
- Metagraph model allows organizing the results of the previous structure. The fragments of the flat graph may be organized into metavertices, metavertices may be organized in higher-level metavertices, and so on. Metavertex organization is more flexible than hypersimplex organization because hypersimplex assumes base and apex usage, and metavertex may include general form graph.
- Metavertex may represent a separate aspect of the organization. The same fragment of a flat graph may be included in different metavertices whether these metavertices are used for modeling different aspects.

Thus, we can conclude that the metagraph model is more flexible than the hypernetwork model. At the same time, unlike the hypergraph model, the metagraph model is a complete graph model with emergence. Therefore, we use the metagraph model.

III. Metagraph knowledge base

A. Emergence feature

In this section, we will look at the advantages of the metagraph model as a data model. Knowledge-based approaches have limitations: looping and inconsistency. It is necessary to apply restrictions or use fuzzy inference to overcome this problem. The metagraph allows adding attribute weights to avoid this limitation.

Another advantage of metagraphs is the ability to consistently describe and detail previously added knowledge through the use of the emergence property [8, 9]. The RDF is characterized by a flat structure of triplets (Fig. 3), which leads to redundancy. It also loses information about emergence, that StatementID_4 and StatementID_5 are related. Emergence overcomes the limitations of the flat RDF model [13].

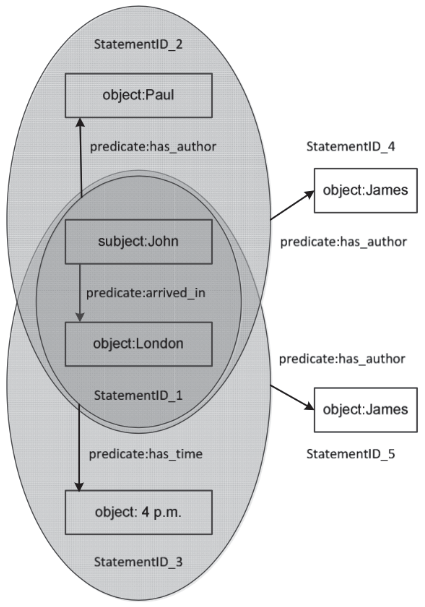

Fig. 3. The flat structure of RDF

This property also makes it possible to represent individual elements of the system in the form of modules connected by common data and having a more complex structure inside (Fig. 4), combining methods of soft computing and knowledge processing.

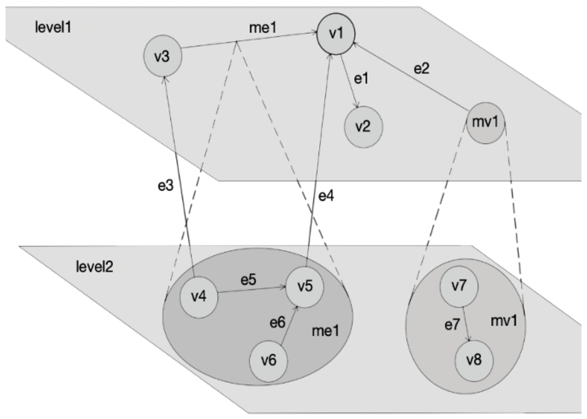

Fig. 4. Different levels of knowledge representation

The metagraph model turned out to be convenient for describing various existing artificial intelligence methods since it allows to describe a complex network using more homogeneous objects (metavertex) than a hypergraph (hypergraph and hypersimplex).

B. Concepts and objects

In [14], the authors suggested the system for the representation and processing of knowledge obtained from the text. The proposed representation is based on the concept of a semantic network extended with the help of metagraphs. Objects and classes of the real world are represented by vertices, and the relations between them are graph arcs. In this case, arcs can be different due to the use of metagraphs, where metavertices and metaedges have a nested structure. It contains abstract concepts and their special cases that personify concrete objects of the real world. These nodes are connected by many links that characterize the relationship between these objects.

A concept can be expressed in a text by one word, or it can be expressed by a sequence of words, for example, a noun with adjectives. Characteristic features for such a concept, which is not expressed in one word, are assumed to be a set of words with which it is designated in the text. Both nouns and adjectives are concepts in themselves, and the composite concept will be connected with them by relations.

A specific object is also considered as some concept of a specific class. An object can also have states that are more specific cases of a given object. Thus, the properties of an object are defined not as values of a particular type inherent in the object but some class to which the object belongs. The type of this property can be inferred from a more general class. Therefore, using one type of relationship and concepts, it is possible to determine the inheritance of concepts and properties of objects with their values.

The intension is concepts to which the relationship is directed from the considered, as well as parental concepts for these concepts. In this case, the intension includes not only generalization relations but also other types of relations that characterize the concept under consideration and distinguish it from the others.

The extension, in this case, is the concepts from which the generalization relationship proceeds to the considered one, as well as the children of them. The higher the concept is, the bigger the extension it has and the less characteristic features or intension it has.

C. Metagraph representation of knowledge

The metagraph representation of knowledge consists of concepts C that are metavertices and relations R that are metaedges. Using (2) it is written as (8). Concepts have a hierarchical representation where one concept is a subclass of another concept. The definition of this relationship is achieved by a hierarchical representation of metavertices, including several others.

$$Semantic = < C, R > \tag{8}$$

where C is a set of metavertices for concepts, R is a set of metaedges for relations.

All concept relationships from a set R are implemented using metaedges (7) that has an attribute t specifying the type of relationship between concepts (9). These metaedges have a weight attribute that indicates the validity of a given relationship in the conceptual model.

$$r = <\{attr\}, \{sem_r\}>, r \in R \tag{9}$$

where $\{attr\}$ is a set of attributes of a relation, $\{sem_r\}$ is a fragment of a metagraph for a given relation consisting of concepts, other relations, and connecting them edges.

A generalization type relation, consisting of one edge and two concepts, can be written as (10).

$$r = <\{t = g\}, \{< c_f, c_s, eo = true, \{k\} >\} > \tag{10}$$

$$c_f \in C, c_s \in C$$

where t is the type of relationship, c_f is the first concept of relationship, c_s is the second concept of relationship, $eo = true$ is the sign of the directional edge, k is the weight of the relation.

It is necessary to perform two types of operations: searching for concepts and adding them to the semantic representation of knowledge. If the metavertex is not found in the semantic graph, then it adds vertex and more general concepts hyperonyms and also connects them with metaedges. A hyperonym is a concept with a more general meaning, meaning a class of concepts, and a hyponym is a particular concept in relation to a more general one.

IV. Analytical Estimation

Search for a concept in the knowledge base begins with concepts that match certain words from the text. This process has a logarithmic dependence of time t_{base} on the number of general concepts $n_{general}$ (11).

$$t_{base}(concept_{general}) = \log(n_{general}) \tag{11}$$

If the text contains a compound concept of several words, then a search is carried out, taking into account the relationship from particular concepts to more general ones. And this depends linearly on the product of the number of general concepts in the particular ones and the number of connections in general concepts. The maximum length of a particular concept is designated as $m_{\max}$, the total number of general concepts corresponding to nouns as n_{noun}, and the number of general concepts corresponding to adjectives as n_{adj}. The maximum possible number of relations between a general concept and specific ones is equal to the number of other general concepts to the power of the maximum length of the general concept minus 1. One must be subtracted, since one of the entity's constituents has already been found at the first stage. Then the number of concepts in total $n_{concepts}$ with the maximum size of a particular concept in $m_{\max}$ is presented in (12).

$$n_{concepts} = \prod_{i=1}^{m_{\max}-1} (n_{adj} - i + 1) \cdot n_{noun} \tag{12}$$

The total search time for a compound concept is the sum of the search time for the first general concept, for example, a noun, and the time it takes to view its connections. Thus, the search time for a concept consisting of a sequence of nouns and adjectives $t_{base}(AdNn)$ can be estimated as (13) and (14) from the total number of words in texts n.

$$t_{base}(AdNn) = O(\log(n)) + \sum_{i=1}^{m_{\max}-1} O(n^i) \tag{13}$$

$$t_{base}(AdNn) = O(n_{adj}^{m_{\max}-1}) \tag{14}$$

If combinations of a noun phrase are considered with another noun phrase in the genitive case, then to search for such a concept, it is necessary to look through all the concepts n_{gen} connected by an association relationship with the second noun phrase (15).

$$n_{gen} = \prod_{i=1}^{m_{\max}-1} (n_{adj} - i + 1) \cdot n_{noun} \tag{15}$$

The search time for a concept corresponding to a noun phrase with a genitive case $t_{base}(Gen)$ is equal to the search time for one of the two components of a noun phrase and processing all of its relations $GenRl$ (16).

$$t_{base}(Gen) = t_{base}(AdNn) + t_{base}(GenRl) \tag{16}$$

Substituting the values for noun phrases and connections in the genitive case, it turns into (17) and (18).

$$t_{base}(Gen) = O(n_{adj}^{m_{\max}-1}) + O(n_{adj}^{m_{\max}-1} \cdot n_{noun}) \tag{17}$$

$$t_{base}(Gen) = O(n_{genConcept}^{m_{\max}}) \tag{18}$$

In this case, for any concept of a noun or adjective phrase, the search time $t_{base}(AdNnGen)$ is (19) and (20).

$$t_{base}(AdNnGen) = O(n_{adj}^{m_{\max}-1} \cdot n_{noun}) \tag{19}$$

$$t_{base}(AdNnGen) = O(n_{genConcept}^{m_{\max}}) \tag{20}$$

But $n_{concept}^{m_{\max}}$ corresponds to the maximum number of concepts in the knowledge base obtained from the text. This number cannot grow faster than the number of words in the text. Therefore, the search in the knowledge base has a linear dependence on the size of the input text data (21).

$$t_{base} = O(n) \tag{21}$$

V. Experiments

A. Methodology

The described metagraph method of knowledge representation was used to implement the knowledge base. The search time of concepts was analyzed to assess the performance of the knowledge base depending on the number of concepts. For a concept corresponding to a single word, the time is equal to the search for the concept itself, and for a compound one, given by a phrase, it is necessary to find more general concepts and then analyze their connections.

The knowledge base is filled with data from the text by an information extraction system using natural language processing. This information extraction system is based on previously developed parser [14] extended with new parsing rules and new structure of knowledge according to metagraph model. Relations weights were not calculated during this study and investigation of machine learning methods for that is a part of the future research.

Results were obtained on the Open Corpora dataset containing texts in various languages. Each text contains a number of concepts for knowledge base and order of texts for analysis is fixed. The measurement is carried out for 100 iterations with the same number of concepts in the knowledge base for averaging the access time to the knowledge base.

The function of getting a collection of text documents has been developed for this purpose. Ten iterations are performed with an increase of 45,000 concepts at each step. The result of the search time is analyzed depending on the size of the concept or the number of words with which it is specified in the text.

B. Results

The research results are presented in Table I. The number of analyzed concepts is presented in the first column of the table. The remaining columns show the average time of search for concepts in the knowledge base, depending on the number of concepts in it.

TABLE I. Time of concept search

Concepts	**Time of concept search in a knowledge base (ns)**					
	Average	***1***	***2***	***3***	***4***	***5***
45000	0,33	0,17	1,06	1,59	1,75	0,00
90000	0,43	0,17	1,58	2,29	2,92	3,33
135000	0,56	0,17	2,33	3,29	3,73	4,44
180000	0,68	0,17	2,98	4,19	5,26	6,36
225000	0,76	0,17	3,40	4,68	6,03	7,06
270000	0,85	0,17	3,99	5,14	8,78	6,11
315000	0,97	0,18	4,63	5,91	8,65	6,54
360000	1,12	0,18	5,48	6,71	8,25	7,78
405000	1,37	0,18	6,83	8,66	9,61	9,69
443681	1,42	0,18	7,24	8,92	11,87	7,58

The columns indicate the time separately for each group of concepts, consisting of 1, 2, 3, etc. words, as well as the average search time for all these concepts. The third column of Table 1 also allows estimating the execution time for searching for concepts that correspond to single words.

The search time increases linearly with the growth of the size of the knowledge base (Fig. 5). But the more words are used to define a concept, in other words, the more it has connections with other concepts, the longer it takes to search.

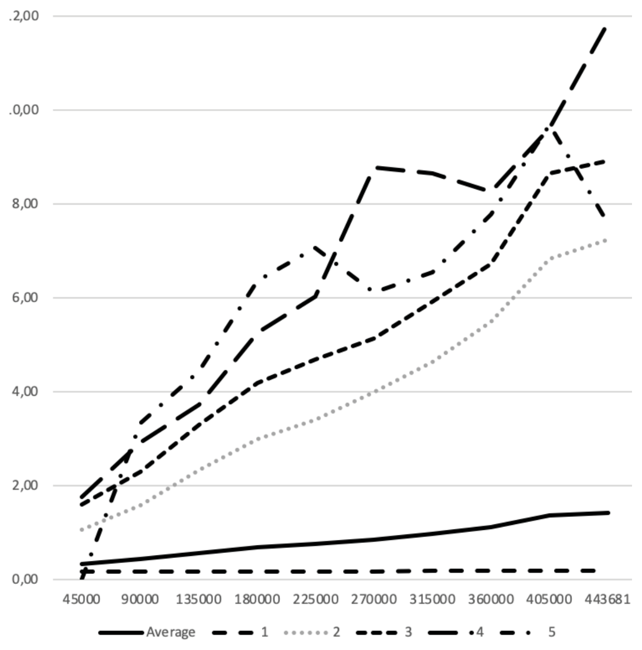

Fig. 5. The dependency of search time for the concept

Since the study used data for concepts from real texts, the number of concepts consisting of a larger number of words is less in texts, and they are less common. Therefore, there is less data for them and they are less representative. Also, for a specific concept, general concepts with a smaller number of links can be used, which will reduce the search time for this specific concept. These features are noticeable in the case of the number of words in a concept equal to 5. But for concepts of shorter length, the dependence of the search time on the complexity of the concept remains linear throughout the entire volume of the knowledge base.

VI. Conclusion

In the paper, the authors described the metagraph representation and implemented the knowledge base to overcome the limitations of existing methods of representing knowledge. The authors performed an analytical estimation of the time it took to search for a concept in the knowledge base, which coincided with the experimental estimate and showed a linear dependence. But for further improvement, it is necessary to optimize the process of finding a concept from linear to logarithmic dependency. Variation of a larger number of parameters and research on other datasets are expected for the future with the further development of the system and an increase in the number of types of analyzed relations. Future research plans also include more complex relationship types for description of dynamic processes and processing time taking into account object states and time relationships, the estimation of the required memory for knowledge base concepts, and exploring machine learning to find weights for relationships.

REFERENCES

[1] J. F. Sowa, Principles of Semantic Networks. Elsevier. 1991.

[2] S. Shapiro, *Encyclopedia of Artificial Intelligence*. Wiley, 1992.

[3] J. Zhong, H. Zhu, J. Li, and Y. Yu, "Conceptual Graph Matching for Semantic Search," *in Proc. Conceptual Structures: Integration and Interfaces*, 2002, pp. 92-106.

[4] M. Sussna, "Word sense disambiguation for free-text indexing Using a Massive Semantic Network," *in Proc. The Second International Conference on Information and Knowledge Management*, 1993, pp. 67-74.

[5] Q. Zhou, C. Wang, M. Xiong, H. Wang, and Y. Yu, "SPARK: Adapting Keyword Query to Semantic Search," *in Proc. International Semantic Web Conference Asian Semantic Web Conference (ISWC)*, 2007, pp. 694-707.

[6] B.S. Manoj, Abhishek Chakraborty, and Rahul Singh, *Complex Networks: A Networking and Signal Processing Perspective*. New York: Pearson, 2018.

[7] V. Chapela, R. Criado, S. Moral, and M. Romance, *Intentional Risk Management through Complex Networks Analysis.* SpringerBriefs in Optimization, Springer, 2015.

[8] V.M. Chernenkiy, Yu.E. Gapanyuk, G.I. Revunkov, Yu.T. Kaganov, Yu.S. Fedorenko, and S.V. Minakova, "Using metagraph approach for complex domains description," *in Selected Papers of the XIX International Conference on Data Analytics and Management in Data Intensive Domains (DAMDID/RCDL 2017)*, Oct. 2017, pp. 342-349.

[9] V.M. Chernenkiy, Yu.E. Gapanyuk, A.N. Nardid, A.V. Gushcha, and Yu.S. Fedorenko, "The Hybrid Multidimensional-Ontological Data Model Based on Metagraph Approach," *in Proc. Perspectives of System Informatics*, 2018, pp. 72–87.

[10] V.I. Voloshin. *Introduction to Graph and Hypergraph Theory.* Nova Science Publishers, Inc, 2009.

[11] J. Johnson, *Hypernetworks in the Science of Complex Systems*. London: Imperial College Press, 2013.

[12] V.M. Chernenkiy, Yu.E. Gapanyuk, A.N. Nardid, and N.D. Todosiev, "The Implementation of Metagraph Agents Based on Functional Reactive Programming," *in Proc. 26th Conference of Open Innovations Association (FRUCT)*, Yaroslavl, Russia, 2020, pp. 1-8, doi: 10.23919/FRUCT48808.2020.9087470.

[13] V. Chernenkiy, Y. Gapanyuk, A. Nardid, M. Skvortsova, A. Gushcha, Y. Fedorenko, and Picking R, "Using the metagraph approach for addressing RDF knowledge representation limitations," *in Proc. Internet Technologies and Applications (ITA)*, pp. 47-52.

[14] A. Kanev, S. Cunningham, and V. Terekhov, "Application of formal grammar in text mining and construction of an ontology," *in Proc. Internet Technologies and Applications (ITA)*, 2017, pp. 53-57.