Original Research Article: full paper

(2018), «EUREKA: Physics and Engineering»
Number 1

DEVELOPING OF THE RELATED DATA SEARCH LSA-BASED ALGORITHM AND ITS PROGRAMMED REALIZATION

Serhiy Mashkovskyi
Institute of Computer Technology
Open International University of Human Development "Ukraine"
23 Lvivska str., Kyiv, Ukraine, 03115
mssss@ukr.net

Abstract

In this article let's consider the theoretical basis of the data search in large data ordered arrays based on the context of the search request and tracking of semantic relationships. Also the first steps towards the practical implementation of this task are proposed. Simple program to check author's ideas has been developed. All the researches have been made with the VK (VKontakte) social network (http://vk.com). Internal API VK was used as retrieving data tool. The final results say that the VK's content has many opportunities to make them more useful and searchable, which means that it is possible to use this 'property' to create our own, more user-friendly way to search and get important data, in the first, for example, buying-selling information, from many kinds of data sources (official pages, users' profiles etc.). That feature never been presented (and probably won't) in other social networks like Facebook or Instagram.

Keywords: contextual search, social networking, text ontology, semantic search.

DOI: 10.21303/2461-4262.2018.00468

1. Introduction

Search engines that make search by keywords, provide access to billions of indexed web pages for thousands of users. Such phenomena as polysemy (one word has got several meanings) and synonymy (some words have got one meaning) increase the number of irrelevant results issued by any search engine.

In connection with ever-increasing number of sites there is also an increasing need for careful analysis of Internet documents content in order to minimize the opportunity to produce irrelevant results. Semantic Web technologies provide a way to solve this problem [1].

The aim of this research is examination of one of the existing technologies of semantic search and discussing of the possibility of adapting it to the using in social media as a large array of disordered data.

In this section, the author would like to describe some main papers and articles that made a big impact on the topic of the article formation.

First of all, the article [2], in which the author presents the idea of contextual search, defines the basic concepts and provides a simplified guide to action. Also the author provides an initial analysis of information resources for the search of linked data. Later the idea from this article will be called "the original algorithm". At the end of this article there is a postscript: "Some of the algorithm improvement ways are to improve the working with social networks...". The author of this article was interested in the author's proposal to expand the application area of "the original algorithm" and at the same time its improvement.

The authors of the article [3] investigated the problem that social networks have got two mutually exclusive characteristics which prevent search algorithms simultaneously from support of them. To solve this problem, the authors developed a framework that provides K-query in real time. This system is based on the rating function that includes relevance, social significance and the similarity of the test (i.e. if the same words are written in two different requests).

The authors of the article [4] propose the approach which addresses the issue of how the members of the enclosed space (groups) can find the shortest path to the desired object using only their closest contacts. The authors have modeled their studies within the real internal network of university students. It seems to us that the algorithm described in this article may be useful, for

example, if, in particular, for replacing its contacts (people) with the words (lexical search components).

The developed algorithm is based on a combination of the latent-semantic analysis, or LSA (described in [5, 6]) and the frequency analysis (described in [7], improved in [8]). These methods are most-common used to web data mining (as [9] says) by identifying and finding dependencies via 'understanding' the obtained data [10].

It's simpler way to complete the task in the fastest time than, for example, construction of a single semantic network based on the analysis of graphs of dependencies of lexemes of the text [11], or creating of a single semantic network based on the analysis of graphs of dependencies of lexemes of the text [12] using the Dice coefficient [13].

Moreover, a something similar to our project was developed and described in [14] – a recommender system that can automatically provide annotations to help user. The system could identify the topics discussed within article which is worked out by semantic approaches with Latent Semantic Analysis (LSA) and WordNet [15].

2. Materials and Methods

The algorithm will aim searching relevant data in the social network VK (VKontakte) that is based on context relationships between words.

One of the possible options for presenting a semantic representation is a structure consisting of "text facts" [16]. So, the idea of the algorithm is to track from where the specific account result is got. It may be, for example: a post of an average user, a post of the profile public page (for example, the post on the official IKEA page which has info for the word "bed"), a non-profile page post (entry from the page about repair of apartment where this word occurs).

There is in the two methods of analyzing big texts: literature holistic (h) and component or analytic methods (componential) (c) [17, 18]. Each methods has its own pros and contras [17, 19], and the holistic one more, so its idea to project a little.

Tracking is done by "reading" the name of the post source and then check if there are all keywords in the title in a predetermined word dictionary. If there are, this entry for further processing is leaved. Next, a number n is defined, let n=20. The next step is "reading" the text of all the entry and see if there are all the keywords in it. If there are, then we take 20 words to the right/left for each keyword. Then a neighborhood of (i–n, i+n) is formed, where i=a serial number of any search word in the all keywords array S, and see whether they are in the table of contexts. If the following keyword is closer than 20 words to the previous one, let's combine the "halves".

After all, the mechanism looks what words, how many times and for which keyword are included in the table of contexts. And on the calculation basis the user concludes what kind this recording is (an announcement of the sale of goods, a review or an ordinary journalistic post).

2. 1. The description and the explanation of the algorithm

The following is the basic idea of the search dependencies algorithm between data taken from [2]. The description is made in pseudocode because the concrete realization (programming language) will depend on available resources.

```
MainFunction ()
Query = enterTheData (keywords);
normalizeQuery (Query);                    (1)

    results = dataSearch (query);          (2)

remember the number of gotten results;

==>loop for each results from the array:
parseTheResult (results);                  (3)
```

wordsList = separateWordsFromTheQuery (source); (4)

indexesList = entryIndexesList (wordsList, source); (5)

neighborhoodWordsList = createTheNWL (wordsList, indexesList, radius); (6)

totalResult = union (totalResult, neighborhoodWordsList); (7)
<== end loop

sort (totalResult); (8)
displayResult();

if necessary, repeat the above is written to the same set, but with a different set of keywords; (9)

displayUpdatedResult(); (10)
--- end of MainFunction()

Explanation of the certain subfunctions:

1) *normalizeQuery* verifies and corrects any requests entered by a user, such as the elimination of insignificant words, punctuation marks and as well as the using of certain specific restrictions for VKontakte (e. g. "Safe Search").

2) *dataSearch* provides the standard search on request for VKontakte, but the results are submitted not to exit (for any user), but "inside" of the program, where they are further processed (e. g., leaving only the text of the post, cutting off the creation date, the name of the source and etc.), including an application-specific API VK (set of proprietary tools and functions for working with VKontakte social network using any third-party software) requests.

3) *parseTheResult*. The text of the particular post (the post result) selected in the previous step is processed. By analyzing the words in the text, the software tries to understand what this is about.

4) *separateWordsFromTheQuery* – all request keywords are selected from the text and given by the list.

5) *entryIndexesList* submits the plate of pairs – that is the list of entry indexes of every word from the request (4).

6) *createTheNWL* – for each index in (5) the number and the variety of keyword which are in likely given the neighborhood of the index.

7) *the union* – the points (2)–(6) are repeated for each search result and findings are incorporated into a single list, from which the duplicate lines are removed.

8) *sort* – records received in the list are sorted in alphabetical order for easier perception by the user.

2. 2. Using the results of the algorithm work

Processing results are displayed visually for a user as well as in the "pure" search VKontakte, but the results are divided into some groups – separately advertising, separately reviews (social network user accounts, where they describe goods).

The algorithm saves the list of the results for the word, in the form of table "a word (the term that it look for) – the number – where it is found (the list of these results containing the text of the word, in the form of links)." And then the next time when a user will search for another product, let's compare this table with the table for a new product. If there are the same lines, it means that there is a potentially useful link. The algorithm will derive the link to the product – linkholder (i.e. a product that has been sought the earliest), next to the results of a new search (something similar to the mechanism of contextual advertising). As a result, the user does not need to look for the original product again.

To simplify the work, the author will create the search mechanism only for the words-nouns as nouns uniquely characterize the object and require less grammatical and semantic relationships.

Accordingly, the thesaurus will also consist only of nouns and will be much shorter than for the same verbs. For example, due to the fact that separate certain item it is possible to make only a finite number of steps, let's have for the noun "bed" relatively small set of verbs ("break", "buy", "cover..."), while on the contrary, for the verb "tinker" let's find hundreds of nouns – names of objects that can be tinkered.

3. Experimental procedures

3. 1. The way of the algorithm realization

The place of use of the algorithm results is the VKontakte (VK) social network that contains a lot of randomly scattered data, which are also often duplicated. It seems that this is an "ideal" random simulation, allowing to choose freely research objects and provides their unique (relative originality) in large scale.

As for the dictionary of synonyms/thesaurus at the beginning of the work on the author's thesis the author is going to take the one used in Microsoft Word, and to highlight from there about 1000 random word-concepts (nouns) with pairs of matches and interpretations, and by using this "piece", to emulate search process.

A context table is a list of words (not all ones, but the most common ones; it is enough to test the model) which are mentally associated with a specific noun. Above all this table is needed to analyze whether another suitable search result is found on request (relevance). The author is going to establish it by combining the resources of several common dictionaries (e. g., from the company ABBYY). In order to expedite the work let's confine ourselves to 100 –150 of the most popular nouns in Russian language (The list is available in Wikipedia).

It is planned to receive data via API VK. Clearly, because every day millions of new records are created, and a quarter – is removed, the information eventually becomes obsolete, and the algorithm starts to give false results. Therefore, once a month the author is going to update the list of output data.

3. 2. Practical implementation of the algorithm

For the practical implementation of the developed algorithm, a software application has been written. To the date (September 2017), the first, the most significant and labor-intensive part of the work has been realized. This includes the search in the VK social network, obtaining data and their primary processing, and finding numerous links between words in the search results.

At this stage of development the program enables to connect to VK servers, to execute a search query and obtain results using the API VK and carry out their further processing (by means of the program itself).

After launching of the program, the user will be prompted to login by entering our credentials into the top left corner. After the successful logging in, the login form will change to the VK home page. Do not pay attention to it. Also, the value of the VK access token obtained in API VK will be displayed in the box on top. It was used by the author of the program during the debugging stage. It was decided not to remove this token. (**Fig. 1**).

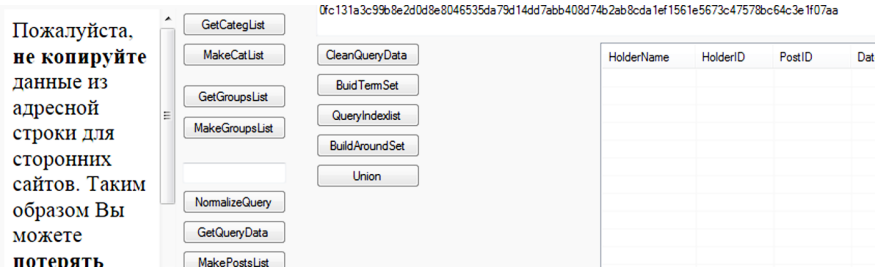


Fig. 1. Our VK token is displayed

Now the user can work with the program.

To the right of the login form, there are 4 buttons for receiving a list of categories of VK communities and a list of groups in each category, as well as for saving them in an external text file.

Just below there are some resources for work with a search query, and namely, a text box to enter the search query, a button to get the initial results of the query and also for saving them to a text file.

Also on the right, in the second "column" there are some buttons to work with the results of the query according to the algorithm outlined in this paper. The labels on the buttons match the English names of the algorithm stages.

On top there is a text box for debugging (following the correct connection with VK and the algorithm progress), where both, intermediate and final results are displayed.

The rest space in the program window is occupied by some lists, in which the results of queries to API VK are displayed in tabular form (**Fig. 2**).

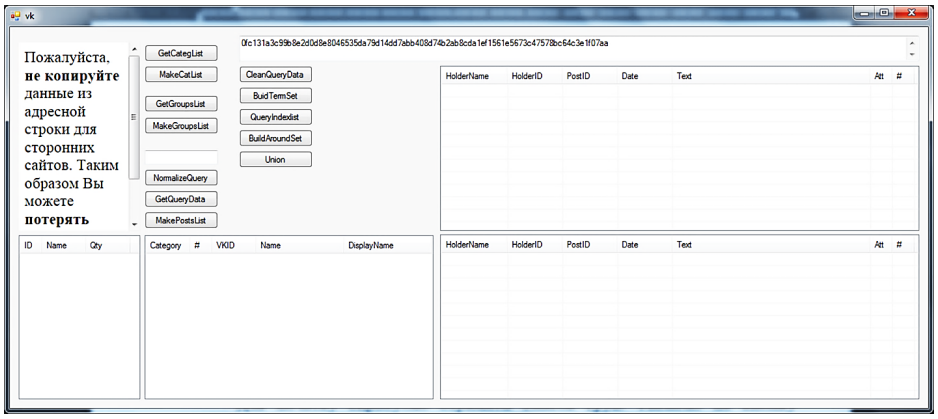
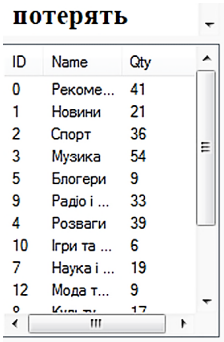


Fig. 2. Program interface at the starting point

For a start, let's try to obtain a list of groups, click the "GetCategList" button and, after waiting for a few seconds and get a list of communities categories (**Fig. 3**).


потерять

ID	Name	Qty
0	Рекоме...	41
1	Новини	21
2	Спорт	36
3	Музика	54
5	Блогери	9
9	Радіо і ...	33
4	Розваги	39
10	Ігри та ...	6
7	Наука і ...	19
12	Мода т...	9

Fig. 3. VK groups categories list

Then it is possible to obtain a list of specific communities in each category. It is done with the "GetGroupsList" button (**Fig. 4**).

Category	#	VKID	Name	DisplayName
0	1	97751087	Рамблер/	rambler
0	2	93640072	Synthposium	synthposium
0	3	18496184	СТС	ctc
0	4	23372133	Дорожное радио	dorognoe
0	5	15722194	Комсомольская пр...	kpru
0	6	123861...	Пацанки	pacankitv
0	7	112510...	Mash | Мэш	mash
0	8	128254...	Проект1917	project1917
0	9	149920...	Пабличные игры	publicgames
0	10	30315369	Роскосмос	roscosmos

Fig. 4. VK groups list

Data generation is complete. Now the user can start work directly with the algorithm itself. The first step is entering a search query – «жареные грибы» (“fried mushrooms”). Let’s click on the “NormalizeQuery” button to normalize and stem the entered request (**Fig. 5**).

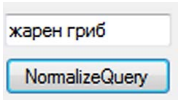


Fig. 5. Query normalization

After normalizing and stemming a query, it is possible to begin searching and getting the results (in fact, don’t even have to normalize it, but then the search results will be less accurate). The search is carried out with the “GetQueryData” button. The results will be listed on the top right.

The received texts of the search results should also be normalized (but not be stemmed). This is being done with the “CleanQueryData” button, which outputs normalized texts to the list at the bottom right (**Fig. 6**). It is essentially a copy of the top list, but with a changed part of the data.

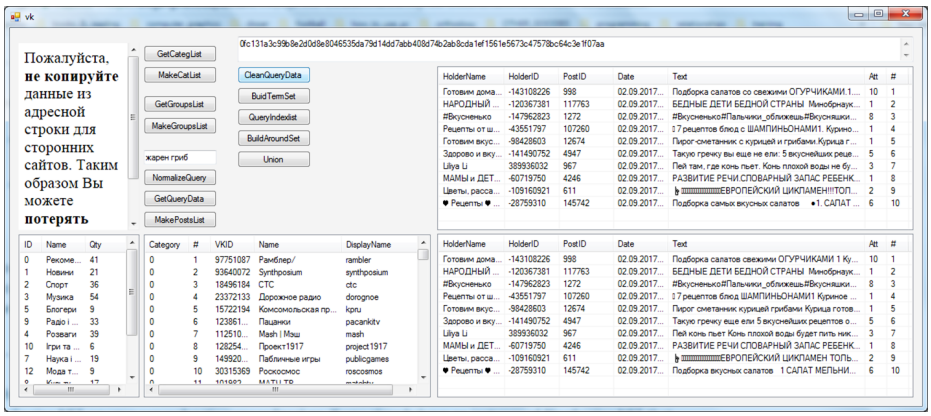


Fig. 6. The search results have been normalized now

Let’s use the “BuildTermSet” button to build a list of terms (individual words) from the text of each search result (the “post”). The result of the work (one of the intermediate stages of the algorithm) is displayed into the text box on the top. Such opportunity – viewing intermediate results – allows to monitor the progress of the algorithm, and if something goes wrong, stop and start over again. Also due to this feature, the user can notice certain posts which are intrusive for a user (**Fig. 7**).

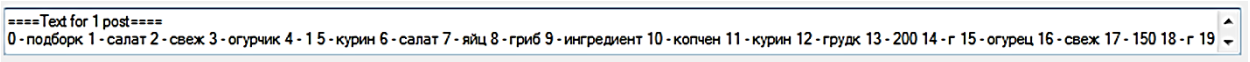


Fig. 7. Building a term set

The next step is creation of a list of indexes for the occurrence of each word from the query in the text of each post. This action is being done with the “QueryIndexList” button, and the results are displayed into the same text box, under the results of the previous step (**Fig. 8**).

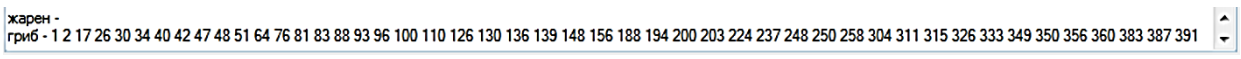


Fig. 8. Creating a list of the terms indexes

Construction of a neighborhood of each index and obtaining words which are included in each neighborhood, are performed with the “BuildAroundSet” button. The results are also displayed in the text box below the results of the previous step (**Fig. 9**).

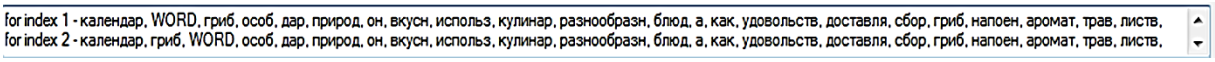


Fig. 9. Building an around set of each term

The union of intermediate neighborhoods and the counting of the number of entries are carried out with the "Union" button. The results are also displayed into the text box below the results of the previous step.

The results of the operation of this button are the final results of the algorithm (on the current developing stage) (**Fig. 10**).

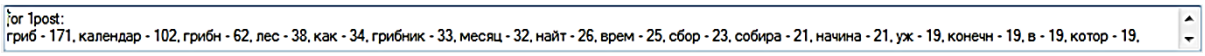


Fig. 10. Counting the final results

In addition, the results of obtaining data from API VK, that is, the content of the lists with the categories of groups, the communities in each category and the initial results of the search can be written to a text file and saved to the disk. These actions are performed by click the "MakeCatList", "MakeGroupsList", and "MakePostsList" buttons respectively. A message will be displayed if save each file successfully.

It should be noted that if the user's computer does not have an Internet connection and/or the ability to connect to the VK server, then all the steps described here will be impossible. Instead, a message will be posted at the very beginning of the work.

4. Results

The program has been written in C#. The operations of connecting to the social network VK and the primary processing of the search query are carried out using the Nemiro.OAuth and StemmersNet libraries, respectively.

During testing at this stage, the program showed good results in general. The percentage of totally relevant results was about 80 %.

The results were determined as follows: first, the number x was set – the maximum number of obtained results. Further, the text of each result was read by the researcher (the author of this article) and on the basis of what was written in the text, the conclusion was made, this result is known (that is, before the start of the algorithm's work) irrelevant, mostly relevant, irrelevant. For the example "fried mushrooms" taken above, a source containing a recipe for cooking fried mushrooms is considered relevant; a source containing a recipe for something just fried, but not mushrooms and/or only some dish of mushrooms, but not fried one, is considered to be partially relevant; and the result, which does not contain anything fried and not mushrooms recipes (or in which the search words from the query are used in another context) is considered irrelevant (for example, the text containing the phrase "smelled of something fried" and "mushrooms grew in the autumn forest") .

It should be borne in mind that the source text may contain, for example, two recipes, one of which corresponds to the criteria, and the other one is not. Or both recipes meet the criteria only partially (not all words in the query). In this case, this source is considered to be partially relevant.

The number of such results, which after their viewing (by man, and not by a computer) is considered adequate, let's denote as x_0. The percentage of such results is calculated as

$$x^* = \frac{x_0}{x} * 100. \qquad (11)$$

In the future, the execution of the algorithm begins, which determines precisely and more exactly what words from the query enter into each of the results, as well as which words are contained in the vicinity of each of the sources of the query. According to the received data method, all search results are sorted into four groups – the results containing all the words of the query containing several words of the query containing only 1 query word and those that do not contain

words from the query (let's call such kind of results "noise", their the appearance is due to the peculiarities of the search in VK). Let's denote the number of results in each group as a–d, respectively.

Then the percentage of successful results of the algorithm is calculated by the formula:

$$R = \frac{(a+b+c)\cdot x^*}{x}\cdot 100. \quad (12)$$

The result was rounded to hundreds.

In total during the development of the program, ten experiments were conducted with a different number of results obtained from search algorithm. The obtained data are shown in the **Table 1**.

Table 1

Results of the algorithm

No. of experiment	Quantity of results	From which				The share of adequate results, %	Relevance, %	
		Contains all the query words	Contains some words	Contains 1 word	Contains no words		Total	Internal VK search
1	200	127	50	21	2	84	83,16	77,4
2	116	62	14	31	9	73	67,34	71,8
3	280	146	52	50	32	77	68,20	83,7
4	256	139	66	42	9	73	70,43	66,8
5	300	84	169	47	0	82	82,00	86,3
6	300	105	93	72	30	72	64,80	77,6
7	300	165	85	43	7	70	68,37	88,05
8	324	127	99	98	0	76	76,00	78,7
9	320	49	214	37	20	74	69,38	76,9
10	20	6	5	5	4	81	64,80	68,1

In parallel with the work of the algorithm, at the same time, a search was carried out directly on the VK site – to compare the success of the author's work with the original search. The proportion of the relevant results of the internal VK search was estimated by the author manually (by reading the text of the results and reading the reading), and the value obtained is given in the last column of **Table 1.**

The further development of the program is being seen as implementation of the second part of the algorithm – the application of the tables with ontologies, synonyms, etc. to the found results, and to track the degree of dependence between data and the search in cross-community VK (based on the same algorithm, but with changed search domains and with a reduced sampling). The author also searches for ways to increase the relevance of the results.

5. Discussion

As mentioned above, the program demonstrated a percentage of results sufficient for initial usual work.

However, it should be taken into account that there are such search results (texts of entries in the VK) in which not all the words from the search query are present, but only a few or only one. For example, while requesting a "wooden carved bed", the results also took into account texts containing only the words "wooden bed" or "carved bed". If such "incomplete" texts are ignored, the relevance of the results will decrease to 65–70 %.

Such low percentage is due to the fact that there are a large number of objects (posts, communities, persons...) in the social network VK, which were created, firstly, artificially, and secondly, solely for spam, cheat number increase in attendance, etc. The texts of such objects usually contain a large meaningless set of words which are in part totally unbound, among which may be the ones specified in the search query.

Technically, such results do not differ from the "normal", and therefore the search is forced to take them into account.

There is a 100 % methodological problem – in an efficient search, there are always two contradictory tasks: increasing coverage in order to obtain the maximum amount of meaningful information and reducing the coverage to minimize the amount of noise information. And the hardest, as always, find the golden mean [20].

The developed program partially solves this problem, but one must also take into account the large semantic dispersion - a significant part of the VK texts are written using everyday vocabulary, slang words, etc., words that are clear only to their authors. Due to the peculiarities of machine work, such texts are thrown out, despite the fact that, from the human point of view, they can contain useful information.

6. Conclusions

Thus, the article affected semantic search topic and presented the general algorithm of its work. Some speculations about the possibility of its use for the large disordered data sets and the ways to improve it are shown. The program for approbation of the algorithm has been developed. The algorithm does not still allow fully to provide request support in free form, but the author will work on this.

Considering the operation of the algorithm, namely, the variants of the neighborhood of words, it is possible to find new (in comparison with previous studies, for example, [21]) emerging features of the semantics of the Russian language (in the future – any language for which this algorithm will be "translated"), which do not follow from generally accepted rules, and which can help future linguists.

Also, by processing the totals (the results obtained), it is possible to build a dependency tree on each step of the interaction with the search engine. Such tree, on each level of which the found connections will be stored, will help to answer the question of exactly how one object has a connection with another.

References

[1] Manning, C. D., Raghavan, P., Schutze, H. (2008). Introduction to Information Retrieval. Cambridge University Press. Available at: http://www.informationretrieval.org

[2] Glybovets, A. M. (2016). Alhorytm poshuku zviazkiv i zalezhnostey mizh danymy veb-storinok. Problems of programming, 1, 44–51.

[3] Li, Y., Bao, Z., Li, G., Tan, K.-L. (2015). Real time personalized search on social networks. 2015 IEEE 31st International Conference on Data Engineering, 639–651. doi: 10.1109/icde.2015.7113321

[4] Adamic, L., Adar, E. (2005). How to search a social network. Social Networks, 27 (3), 187–203. doi: 10.1016/j.socnet.2005.01.007

[5] Salamakha, O. (2015). Algoritm LSA dlya poiska pokhozhikh dokumentov. Available at: http://netpeak.net/ru/blog/algoritm-lsa-dlya-poiska-pohozhih-dokumentov/

[6] Basipov, A. A., Demich, O. V. (2012). Semanticheskii poisk: problemy i tekhnologii. Vestnik AGTU. Seriya: upravleniye, vychislitelnaya tekhnika i informatika, 1, 104–111.

[7] Panchenko, A. (2010). Postroyeniye semanticheskoy seti iz raznorodnyh dannyh. Moscow: MGTU im. Baumana. Available at: http://it-claim.ru/Persons/Panchenko/presentation2010_sept_final.pdf

[8] Voronoy, S. M. (2012). Obnovleniye ontologiy s pomoschyu semanticheskih setey tekstov na yestestvennom yazike. Informatsionnie upravlyayuschiye sistemy i kompyuterniy monitoring. Donetsk: DonNTU, 83–85.

[9] Glybovets, A. N., Glybovets, N. N., Prokoptsev, D. E., Sidorenko, M. O. (2013). Strukturirovannie dannie i semanticheskaya pautina: tekhnologii Wiki. Problems of programming, 1, 45–67.

[10] Glybovets, A. M., Glybovets, M. M., Polyakov, M. V. (2014). Intelektualni merezhi. Dnipropetrovsk: Nova Ideolohiya, 464.

[11] Naykhanova, L. V., Ayusheyeva, N. N., Khaptakhayeva, N. B. (2007). Postroyeniye semanticheskoy seti predmetnoy oblasti na osnove izvlecheniya znaniy iz nauchnogo teksta. Izvestiya vyschikh uchebnyh zavedeniy. Povolzhskiy region. Tekhnicheskiye nauki, 4, 51–61.

[12] Burakov, S. V., Tarsov, S. V. (2015). Kontekstno zavisimiy sposob poiska nechetkih dublikatov v relyatsionnikh bazah dannyh. Informatsionniye upravlyayuschie sistemy, 2, 76–81.

[13] Mazov, N. A. (1995). N-grammniye metodi obrabotki tekstovoy informatsii. Novosibirsk: OIGGM SO RAN, 180.

[14] Purwitasari, D., Yuniar, E., Yuhana, U. L., Siahaan, D. O. (2011). Ontology-based annotation recommender for learning material using contextual analysis. Proceedings of the IETEC'11 Conference,

Kuala Lumpur, Malaysia. Available at: http://ietec.apaqa.org/wp-content/uploads/IETEC-2011-Proceedings/papers/Conference%20Papers%20Refereed/Wendesday/WP4/WP4.4_59.pdf

[15] WordNet Search – 3.1. Available at: http://wordnetweb.princeton.edu/perl/webwn

[16] Leontyeva, N. N. (2002). K teorii avtomaticheskogo ponimaniya teksta. Chast 3. Semanticheskiy komponent. Lokalniy semanticheskiy analiz. Moscow: Izdatelstvo Moskovskogo universiteta, 49.

[17] Foltz, P. W., Gilliam, S., Kendall, S. (2000). Supporting Content-Based Feedback in On-Line Writing Evaluation with LSA. Interactive Learning Environments, 8 (2), 111–127. doi: 10.1076/1049-4820(200008)8:2;1-b;ft111

[18] Landauer, T. K., Laham, D., Rehder, B., Schreiner, M. E.; Shafto, M. G., Langley, P. (Eds.) (1997). How Well Can Passage Meaning be Derived without Using Word Order? A Comparison of Latent Semantic Analysis and Humans. Proceedings of the 19th Annual Meeting of the Cognitive Science Society, 412–417. Available at: http://lsa.colorado.edu/papers/cogsci97.pdf

[19] Gryshanova, I. Yu. (2016). Analitychniy oglyad metodiv ta zasobiv semsntichnoho poshuku v semantic web. Problems of programming, 1, 51–72.

[20] Uspenskii, I. V. Internet-marketing. Metody poiska informatsii. Biblioteka «Polka bukinista». Available at: http://polbu.ru/uspensky_inetmarketing/ch45_i.html

[21] Shelmanov, A. O. (2015). Issledovanie metodov avtomaticheskogo analiza tekstov i razrabotka integrirovannoy sistemy semantiko-sintaksicheskogo analiza. Moscow, 210. Available at: http://www.ipiran.ru/announce/dissertation_Shelmanov.pdf

APPLICATION OF FUZZY MATHEMATICS METHODS TO PROCESSING GEOMETRIC PARAMETERS OF DEGRADATION OF BUILDING STRUCTURES

Svitlana Terenchuk
Department of Architectural Structures
Kyiv National University of Civil Engineering and Architecture
31 Povitroflotskiy ave., Kyiv, Ukraine, 03680
terenchuksa@ukr.net

Bohdan Yeremenko
Department of information technology of design and applied mathematics
Kyiv National University of Civil Engineering and Architecture
31 Povitroflotskiy ave., Kyiv, Ukraine, 03680
erembm@ukr.net

Serhii Kartavykh
Department of information technology of design and applied mathematics
Kyiv National University of Civil Engineering and Architecture
31 Povitroflotskiy ave., Kyiv, Ukraine, 03680
snk07@ukr.net

Oleksandr Nasikovskyi
Development Investment Management
Limited Liability Company
8b Raisy Okipnoy str., Kyiv, Ukraine, 02002
avnukraina@gmail.com

Abstract

The aim of research is formalization of the expert experience, which is used in processing geometric parameters of building structure degradation, using fuzzy mathematics. Materials that are used to specify fuzzy models are contained in expert assessments and scientific and technical reports on the technical condition of buildings. The information contained in the reports and assessments