Keyword Search Procedure Using Fuzzy Matching to Detect Ambiguity in Expert Formulations in Knowledge Bases of Decision Support Systems

Vitaliy Tsyganok[a,b], Mykhailo Dubok[b] and Olha Tsyhanok[c]

[a] *Faculty of Information Technology Taras Shevchenko National University of Kyiv, Bohdana Havrylyshyna Street, 24, Kyiv, 04116, Ukraine*
[b] *Institute for Information Recording of National Academy of Sciences of Ukraine, Mykola Shpak Street, 2, Kyiv, 03113, Ukraine*
[c] *Department of Foreign Philology and Translation Kyiv National University of Trade and Economics, Kyoto Street, 19, Kyiv, 02156, Ukraine*

Abstract

Decision support systems use complex weakly structured system models, whose components are formulations provided by experts in a natural language. For adequate construction of models of such systems, it is important that formulations are understood the same way by different participants in group expertise, otherwise the model will not reflect the knowledge of the team of experts sufficiently correct. Because any natural language is characterized by ambiguity, measures should be taken to detect and, if possible, remove it at the stage of providing an expert formulation. For certain languages, including English, German and Ukrainian, there is a list of keywords that indicate the potential for ambiguity. Some of these keywords are variable parts of speech, so exact matching alone cannot ensure that all keywords for ambiguity detection are identified in a formulation. The use of search via fuzzy matching makes it possible to identify keywords for ambiguity detection in a non-basic form. Having tested the proposed method, the use of the search procedure in the list of keywords via fuzzy matching was able to increase the recall to maximum, which means that all paragraphs de facto containing keywords for ambiguity detection are covered using the proposed method. It has absolute precision when using keyword search to detect ambiguity, which is possible due to a known set of words that are used in search via fuzzy matching and the use of information about the part of speech and grammatical categories. The absolute precision means that no odd paragraph that does not contain keywords for ambiguity detection was covered. This increases the probability to detect text formulations that are potentially ambiguous. Since the procedure of search via fuzzy matching is much more resource-intensive than search via exact matching, the paper presents ways to increase the speed of the proposed algorithm without compromising parameters of the ambiguity detection in expert formulations.

Keywords

Decision support system, expert formulation, ambiguity detection, keywords used for ambiguity detection, fuzzy matching

1. Introduction: Analysis of the Problem Situation

Decision support (DS) tasks are typical of the so-called weakly structured subject domains [1-3], which are complex systems. Usually, DS is carried out on the basis of pre-built models of these systems. Figure 1 shows a simplified diagram, which, in addition to the main properties of a weakly

II International Scientific Symposium «Intelligent Solutions» IntSol-2021, September 28–30, 2021, Kyiv-Uzhhorod, Ukraine
EMAIL: tsyganok@ipri.kiev.ua (A. 1); midubok@gmail.com (A. 2); olzyg@ukr.net (A. 3);
ORCID: 0000-0002-0821-4877 (A. 1); 0000-0001-5313-4844 (A. 2); 0000-0002-0009-6562 (A. 3);


structured complex system, shows the relationship between the construction of models of such systems and the need to use expert estimation in solving DS problems.

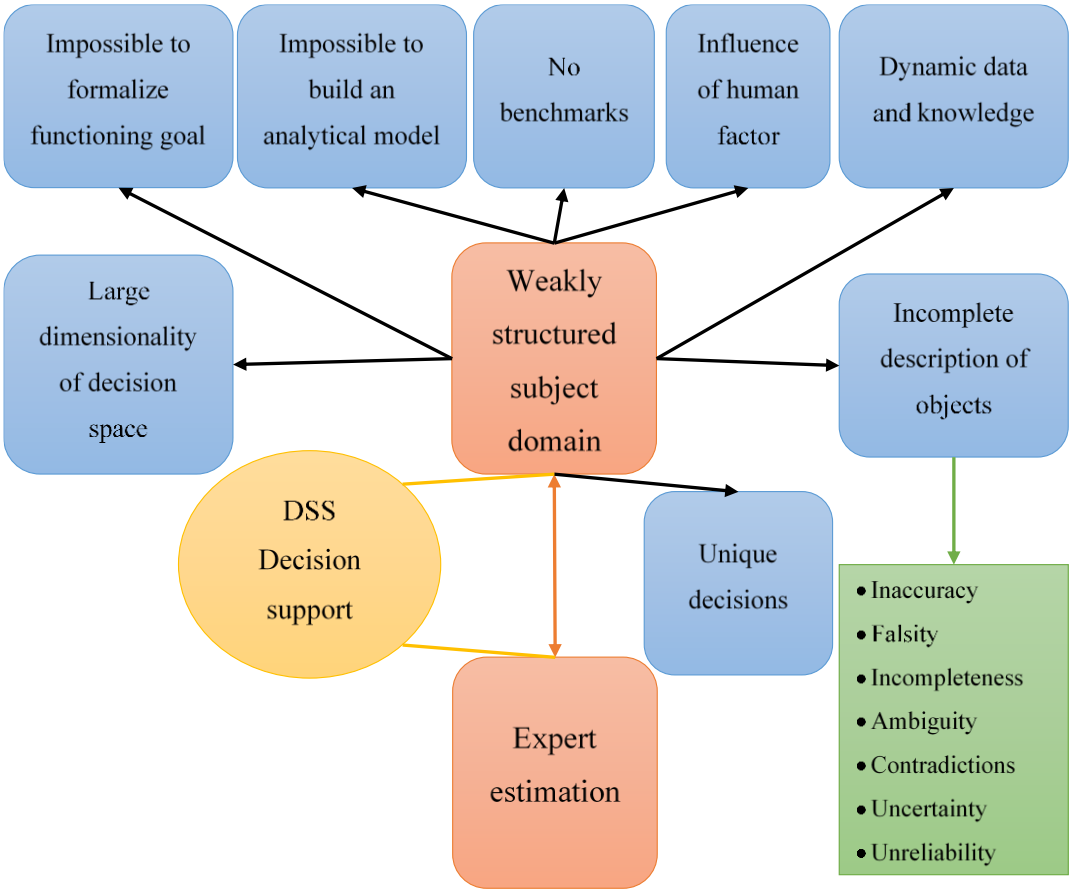

Figure 1: The main properties of complex systems in which expert DS is used

Tasks related to weakly structured complex systems are characterized by the following main features:

- *the impossibility of formalizing the functioning goal* of a system (the goal of non-man-made systems, such as environmental, natural, social, administrative, etc., is usually their performance in general or maintaining certain parameters within specified limits, but it is, as a rule, impossible to formalize such a goal due to the number of factors that affect its functioning, and the complexity and incomprehensibility of the links between these factors);
- *the impossibility to build an analytical model* of the subject domain (due to the lack of a formalized functioning goal of the system and the complexity of the relationships between system components, it is not possible to build a function whose optimization would provide the best mode of operation);
- *lack of optimality* (due to the lack of an objective optimization function, it is possible to optimize only certain factors, which, in the general case, does not lead to the optimal functioning of the whole system);
- *incomplete description of objects* in the subject domain, which is associated with inaccuracy, incompleteness, uncertainty and unreliability of available information about objects;
- *no benchmarks* for evaluating objects (because, for the most part, one deals with so-called intangible/immeasurable factors/criteria);
- *the uniqueness* of the problem to be solved and the impossibility of repeating the decision-making process (because objects in weakly structured subject domains are unique and the process of solving real problems and transferring them to other objects requires high costs or is simply impossible);

- *dynamism* is due to the fact that the structure and functioning of the object changes over time, i.e. the object evolves (therefore, the tasks in such systems must be adaptive, able to change when the object changes);
- the influence of *the human factor* (since the objects of control or elements of the system can be people who have free will, and predicting their behavior is often impossible, because people act in the system, taking into account their personal goals and interests, so when modeling the subject domain, it is difficult to take into account human behavior);

In addition, non-formalized tasks can be characterized by the following features:

- ambiguity, incompleteness, contradictions and falsity of the initial data, knowledge about the problem subject domain and about the specific problem to be solved;
- large dimensionality of decision space, which leads to a fairly significant search when searching for a solution;
- dynamic data and knowledge.

All the above circumstances do not allow the use of simulation modeling approaches, focused on the use of quantitative objective estimates, for decision-making [4].

The class of problems solved with the help of DS tools is quite wide and is constantly expanding. This class includes the following areas of human activity: industry, energy, defense, commerce, banks, transport, economic management, medicine, sustainable development, automation, informatization, although the class of tasks is not limited to only these areas. In the conditions of constant increase of responsibility for decision-making in various areas of human life, the volume of application of decision support systems (DSS), developed on the basis of the corresponding DS technologies, constantly grows worldwide.

In fact, due to the above features the problems in weakly structured subject domains cannot be fully solved without the involvement of experts – specialists who could use their competence (knowledge, experience, intuition) to build adequate models of such complex systems and, further, evaluate solutions on the basis of these models.

It should be noted that among the so-called "No"-factors – incomplete description of objects in the model there is a factor of ambiguity, which this study aims to reduce.

Content ambiguity can have a negative impact on the quality of representing information in text form in various areas of human life. If textual information provides knowledge, the importance of unambiguous interpretation dramatically increases many times over. This is especially true in the field of DS, where at the present stage the "cost" of the wrong decision is constantly growing and can be invaluable.

Text expert formulations are used in the construction of models of subject domains in DSSs. DSSs are used to help a decision maker (DM) make informed decisions. Such systems, based on the application of methods [5–8], models and technologies [9], taking into account dozens and hundreds of factors, criteria, goals and their interrelationships, provide recommendations for solving problems related to decision-making.

Typically, groups of experts are involved in building a subject domain model in the form of a DSS knowledge base [9]. While building a model, experts provide their individual formulations of goals, criteria, factors, etc. Currently, when building a model, it is important to maintain the compliance of the knowledge available to each expert with the knowledge included in the existing model. This correspondence largely depends on the clear understanding of expert formulations and on the impossibility of different interpretations of formulations. Misunderstandings and misinterpretations of certain expert formulations by other experts and knowledge engineers can lead to significant inconsistencies between the constructed subject domain model and the knowledge of experts and, consequently, to reduction of the quality of recommendations provided by a DSS based on this model.

In other words, how clearly expert formulations are formulated affects how they are understood by other experts, and this affects compliance, adequacy of the constructed model of the subject domain. Therefore, it is necessary to detect and decrease text ambiguity. In decision-making support, ambiguities in expert formulations pose threats of misinterpretation and thus reduction of adequacy of subject domain models. After all, incorrectly interpreted formulation can lead not only to a misunderstanding of a part of the expert formulation, but also to the incorrect formation of the structure of the hierarchy of goals. Such errors affect the quality parameters of subject domain models

– the compliance of the model to the study area, the adequacy of the model, etc., and this leads to a decrease in the quality of recommendations generated by a DSS based on such constructed models.

However, there are still cases of misinterpretation even within one field. For example, if there is a main goal – to increase the company's profit, and the subgoal is to improve job satisfaction, the expert formulation can be as follows: "provide a lounge for departments A and B." This can be interpreted as "provide a common lounge" or "provide one lounge for each department". If misinterpreted, the cost of implementing this step and possibly the impact of this step on achieving the main goal will be wrong. Another example of an expert formulation: "a qualified manager choice." The statement can be interpreted as "a choice made by a qualified manager" or "a choice of a qualified manager among the applicants." Such examples are not rare, and therefore, the urgent task is to reduce the ambiguity of expert formulations in computer DSSs.

According to the above, the detection of ambiguity immediately upon the introduction of a new expert formulation makes it possible to create more adequate models of subject domains.

To reduce the ambiguity of expert formulations in computer decision support systems, there are automatic and non-automatic ways to process it. The automatic way (ambiguity resolution) is not acceptable because it does not reduce ambiguity, but provides only one of the possible meanings of a text formulation. The non-automatic way is represented by four different techniques: ambiguity avoidance (when instructions for writing texts are provided), ambiguity prevention (when the writing of text in a fixed format is regulated), ambiguity detection (automatic detection in the written text is carried out with the subsequent notification of the expert on existence of ambiguity in the text) and ambiguity correction (semi-automatic means of correcting a text formulation are used, which interact with experts who provide a formulation) [10]. Given the drawbacks of ambiguity avoidance, prevention and correction and the danger of leaving ambiguity unhandled [11], [12], ambiguity detection was evaluated as the most promising technique of the non-automatic way.

To detect ambiguity in expert formulations, one can utilize a list of keywords that indicate potential ambiguity, compiled by Gleich et al. and often used for detection of textual ambiguity [13]. The list includes words listed in the Ambiguity Handbook ("acceptable", "or", "include", etc.) [14], as well as words added by the compilers of the list ("they", "all", "many", etc.). These keywords are used to detect ambiguities at the lexical level. For example, if the sentence contains the word "otherwise", the sentence will be marked as ambiguous, as the formulation in the sentence may apply to many cases, as in the formulation "Otherwise the system should display an error message". This list is universal and is not formed separately for each subject domain. This is based on the thesis that certain words a priori introduce ambiguity. The list's compilers themselves ascertained its effectiveness in detecting ambiguity in German-language texts. The application of this approach has been tested by translation keywords into German. The effectiveness of using this keyword list for ambiguity detection is almost not reduced after translating its keywords into Ukrainian, except for the keywords "до" ("before" and "until", can relate to place or time, be a part of a fixed phrase or be required by a verb), "по" ("through", has too many occurrences irrelevant to time), "перед" ("before", can relate either to time or place) [15].

Since a lot of the keywords in the list belong to variable parts of speech, at least in Ukrainian, the use of search via exact matching alone does not make it possible to detect keywords used in non-basic form, such as in another case, number, person, tense, etc. This necessitates the use search via fuzzy matching. Using such a search different word forms of one word can be associated to a common base form.

In inflected languages, such as Ukrainian, Russian, partly German, the grammatical information of a word belonging to a variable part of speech is concentrated in the inflection [16]. This theoretical knowledge allows one to focus on endings (inflections) when working with such languages. Quasi-inflections (mimic endings) can be less than, equal to or greater than the grammatical inflection. They are obtained not from theoretical information, but by practical experiments, during which it is found out what volume of the word, starting from the end, is the minimum necessary to precisely obtain the appropriate base form. Therefore, a quasi-inflection can also include the whole word, for example, for the word «жодна» (feminine "none") only the quasi-inflection "жодна" can unmistakably indicate the part of speech "займенник" (pronoun), since any smaller quasi-inflection will also include the numeral «одна» (feminine "one") and will lead to a wrong match with the numeral «один» (masculine "one") and not the pronoun «жоден» or «жодний» (2 variants of a masculine "none").

In previous studies, the effectiveness of search via fuzzy matching was practically tested using quasi-inflections in order to obtain correct part-of-speech markup. The 98.70% accuracy of part-of-speech tagging was obtained using this method which is one of the best results among such methods [17].

Information about a certain part of speech and grammatical categories (number, gender, etc.) can be used to avoid false positive matches with keywords. For example, the word "дорогу" which can be a noun ("road") or an adjective ("expensive") in an expert formulation will not match the dictionary word "дорого", which is an adverb ("expansively").

During the analysis of the quality of the method of detecting ambiguity of textual formulations, it is necessary to first investigate the degree of recall increase (finding expert formulations containing keywords for ambiguity detection) and determine the precision of the method which shows whether the base keyword is correct for the word form in the expert formulation when using fuzzy matching. This will allow to use the keyword list for early detection of ambiguity in an expert formulation and to notify an expert about ambiguity. Another important task is to study the impact of the use of fuzzy matching when searching for the base form on performance and to study ways to speed up the search procedure without reducing search recall and the ambiguity detection method's precision.

2. Discussion

Directly in the field of decision support systems, the issue of ambiguity of expert formulations has previously been raised [15], [17].

The detection of text ambiguity has been studied, for example, in the field of software requirements [13], [18]. Since most requirements are written in natural language [19], they often contain inaccuracies that, if misinterpreted, lead to errors in software development. Similar to expert formulations in building a subject domain model, the later an error in software requirements is identified, the more difficult it is to correct, especially if the implementation of misinterpreted software requirements is already present as a part of the program. Ambiguity detection is used for early detection of inaccuracies in the development process.

From the work of Gleich et al. [13], which lists keywords used for ambiguity detection, it is possible to obtain confirmation that automatic ambiguity detection as a promising technique of ambiguity reduction [17]. During the translation where each English keyword in the list was matched with a set of semantically equivalent words in Ukrainian, it was found that most keywords have a clear set of equivalents in Ukrainian, except for a small number of words in Ukrainian that themselves have several different meanings. In the latter case, some keyword meanings cannot indicate ambiguity, or require additional rules for elimination of a significant number of irrelevant results [15]. For example, by translating the English keyword "until" into the Ukrainian equivalent "до", a large number of formulations containing the word " до " are recorded, but most of them relate to place, not time.

Having applied the translated list of 17 categories of keywords for ambiguity detection (a total of 75 keywords) to the Ukrainian-language corpus (a part of which is expert formulations, each of which is a separate paragraph), amounting to more than 137 thousand words, 6983 paragraphs were found by searching via exact matching. These paragraphs contain at least one keyword from one category of keywords for ambiguity detection. The paragraphs were detected as follows: if the formulation contains 2 or more keywords from the same category, the formulation was logged, i.e. was written by the program in a text file with the detected formulations, only once. If the formulation contains 2 or more keywords from different categories, the formulation was logged in each of these categories. This implementation is motivated by the goal – to obtain unambiguous expert formulation. Achieving the goal implies obtaining expert formulations that contain keywords, rather than just a complete list of keywords. At the same time, after writing the formulation the expert needs to get the information about all categories for which the expert formulation is detected as ambiguous. Thus, having written the formulation, the expert will understand why the formulation is recognized as ambiguous and will react or ignore the warnings at one's own discretion.

However, analysis shows that a lot of expert formulations containing keywords in a non-base form (for example, in the genitive case) can be omitted. The result is a reduction in recall. In order to cover

all word forms of variable keywords and get all potentially ambiguous formulations, one needs to apply search via fuzzy matching.

When searching via fuzzy matching, approximate string matching (or fuzzy string searching) is often used, measured by the number of primitive operations (insertion, deletion, substitution; also, sometimes primitive operations include transposition) that have to be performed to make the two strings match exactly [20]. The disadvantage of this is the problem of formal match of unrelated words, such as "мив" ("washed") and "мир" ("peace"), "справи" ("affairs") and "вправи" ("exercises"). This problem can be solved by creating rules according to which words can match only if they meet certain conditions, namely, the correspondence of the parts of speech of both words that are being checked and the correspondence of their number, gender categories.

An alternative to counting the number of primitive operations is proposed – the use of only gradual truncation of letters, starting from the end of a word [15], [17]. In this way, the total number of primitive operations is also known, but no distinction is made between them. However, the order or priority of fuzzy match checks, which can be set according to one's needs, is crucial in this case. For example, if the priority is the precision of fuzzy matching, then the checks for the matching of two words start from the smallest truncation (1 character from the end of the text word or keyword) and go gradually to the largest truncation (several characters from the end of the text word and keyword). Otherwise, if the priority is the speed of obtaining the result, rather than precision, checks for the matching of two words can be placed, starting with a larger truncation. The method of truncation from the end of a word was chosen as the most promising for inflectional languages, because it is mostly the last letters that indicate grammatical categories. For analytical languages, the hypothetical effectiveness of this method is lower, because the grammatical information in these languages is concentrated in auxiliary words, not inflections. In other words, fuzzy matching via truncation is suitable to inflectional languages (Ukrainian, Russian) and those that are partly inflectional (German).

Research objective – improvement of the reliability and the adequacy of models on the basis of which decision support is provided by automatic detection with the possibility of further reduction of ambiguity in textual formulations of experts via searching with fuzzy matching of words among the formed set of keywords that are typical in detection of ambiguity. The paper provides an analysis of the recall, precision and ways to optimize the procedure for the use of fuzzy matching of words when searching in a set of keywords that are indicators of ambiguity.

3. Formal Statement of the Problem

What is given: $W = \{w_i\}, i = (1..n)$ – a set of words in a text formulation, where *n* is the number of individual words in the formulation that do not match each other when writing.

$A = \{a_{jk}\}, j = (1..c), k = (1..m_c)$ – a set of keywords formed for a particular language, indicators of ambiguity, which are divided into *c* categories, where each category has m_c keywords.

i, j, k – integer indices.

Needed to define: Mapping $f : W \to A$, where $\exists a_{jk} : f(w_i) = a_{jk}] \Rightarrow \exists$ ambiguity in a text formulation given by *W*.

3.1. The Proposed Method

The paper proposes to implement a search via fuzzy matching of the basic form of keywords for ambiguity detection, which had been also used to identify part-of-speech classes using analysis by rules and a dictionary [17]. Since the part of speech is also used to check matches of formulation words with keywords, keywords and their part of speech have to be added to the dictionary which is further used by search via fuzzy matching.

Because it would require a lot of time and resources (corpora containing expert formulations) that can be unavailable, the method was tested on only one inflectional language (Ukrainian) but should be suitable to other inflectional languages such as German and Russian after translating the keywords into the respective language. In the translated list, all keywords for ambiguity detection belonging to variable parts of speech (noun, adjective, numeral, pronoun, verb, participle) were given a special

symbol "◦". The special symbol indicates whether search via fuzzy matching has to be applied if there is no result after searching via exact matching.

Keywords are divided into 17 categories, each divided into small groups of 1 to 5 words according to whether they are different Ukrainian-language versions of a single English-language keyword. However, this organization is not mandatory: keywords can be divided into equal groups or processed individually. Categories provide a textual explanation of why a keyword that falls into a certain category is ambiguous. In addition, in the proposed method, categories are used to optimize performance, because they in a certain way segment the overall list of keywords. This allows one to perform certain massive filtering of all keywords in a certain category.

4. Validation of the Method

To check the recall and precision of fuzzy matching of words in expert formulations, the Ukrainian-language corpus mentioned in the Discussion section, which consists of 7,389 paragraphs and a part of is expert formulations, each of which is a separate paragraph, was used again.

The corpus contains 16599 keywords for ambiguity detection and their forms which can be found in 4141 paragraphs that have at least one keyword in any form from any ambiguity category. To estimate the recall and precision, ambiguity categories have not been taken into account which means that one paragraph is listed only once despite even having keywords from multiple ambiguity categories.

Without listing all word forms of keywords that belong to variable parts of speech, exact matching is able to find 4010 paragraphs which provides 96.84% recall. In order to cover all possible keyword forms, one can enumerate all 423 keyword forms in the list and use exact matching or use the original list of keywords and apply search via fuzzy matching. The advantage for the latter approach is the ability to cover all forms of new keywords by adding only base forms to the list.

704 fuzzy matches of keywords for ambiguity detection were found in the Ukrainian-language corpus. After removing duplicates, 120 unique forms in expert formulations were obtained. Having grouped matches by common base forms, fuzzy matches with 19 variable keywords were found: "відповідний" ("appropriate", 12 word forms), "включати" ("include", 4 word forms), "всі" ("all", 8 word forms), "достатній" ("sufficient", 5 word forms), "ефективний" ("efficient", 12 word forms, 1 of which is a qualitative adjective in the comparative degree of comparison), "єдиний" ("only", 7 word forms), "кожний"/"кожен" ("each"/"every"/"everybody", 10 word forms), "легкий" ("easy", 2 word forms), "містити" ("include", 2 word forms), "наступний" ("next", 5 word forms), "однаковий" ("even", 7 word forms), "попередній" ("previous", 6 word forms), "прийнятний" ("acceptable", 3 word forms), "рівний" ("even", 10 word forms), "справедливий" ("even", 11 word forms), "суттєвий" ("essential", 4 word forms), "точний" ("accurate", 2 word forms), "усі" ("all", 7 word forms) and "швидкий" ("fast", 3 word forms).

The base form was correctly detected in all fuzzy matches, ensuring 100.00% precision. It is worthy of note that absolute precision was received after optimization steps, some of which help reject false results. Searching in the Ukrainian-language corpus with fuzzy matching results in an increase in recall to 100.00%: all of the paragraphs that contain keywords are covered.

Ambiguity categories were neglected in determining recall and precision in order not to overestimate the impact of search via fuzzy matching by simply stacking already listed paragraphs in each category. The use of fuzzy matching decreases performance because it requires more computing resources compared to the use of only exact matching. To research this impact, tests were conducted using different configurations of check in the program's code, but one device – an Asus X75VC laptop: dual-core Intel Core i5-3230M (2.6 GHz), 8 GB of RAM (1600 MHz DDR3), SATA SSD. The tests were conducted on the material of one Ukrainian-language corpus. The speed of using only exact matching – an average of 50 seconds – is taken as a benchmark. The performance of all subsequent tests is given with rounding to seconds and as the average of all tests conducted using a given configuration.

To estimate the performance, ambiguity categories have been taken into account. Therefore, the same paragraph can be listed in different ambiguity categories provided it contains keywords from multiple ambiguity categories. When applying fuzzy matching without restrictions, the number of

detected results is the largest – 8492. At the same time, 2.86% of fuzzy matches are false. The speed drops to a minimum of 44 minutes 7 seconds, which increases time costs by 52.94 times compared to using only exact matching. Below are given configurations to minimize the performance loss. The paper provides names for key configurations, which have a large difference between them, based on Greek alphabet letters' names. The key configurations are given in Table 1.

By adding a prerequisite that the current keyword category must contain at least 1 variable keyword, the number of results found is reduced to 8264, but only false matches are lost, as there should be no fuzzy match for words of an invariable part of speech. The speed is significantly higher – 14 minutes 36 seconds (17.52 times longer than exact matching).

Adding the condition that the word in the formulation should be absent in the dictionary, the number of results is reduced to 8260, but only 4 false results are lost, because words that are used in the base form should not have a fuzzy match. Speed accelerates to 9 minutes 55 seconds (11.9 times longer than exact matching). Hereinafter, we will call this configuration "Alpha".

By adding the condition that the part of speech of the word in the formulation and the part of speech of the word in the dictionary must match, the number of results is reduced to 8249 due to the elimination of erroneous results. The speed is accelerated to 7 minutes 38 seconds (9.16 times longer than exact matching). For all subsequent configurations, the number of results remains the same – 8249. Adding to fuzzy match checks, where at least 2 characters are cut off from the word in the formulation, the condition that a word in a formulation must be longer than 2 characters, the program speeds up to 7 minutes 36 seconds (9.12 times longer than exact matching). Let us call this configuration "Beta". By adding the condition that the word in the formulation must have more than 3 characters, to fuzzy matching checks, where at least 3 characters are cut off from the word in the formulation, the speed is accelerated to 7 minutes 34 seconds (9.08 times longer than exact matching).

By skipping words in the formulation that are 0 characters long after removing punctuation, the program speeds up to 7 minutes 32 seconds (9.04 times longer than exact matching). We named this configuration "Gamma".

Adding the condition that the words in the dictionary must be longer than 1 character, the speed is accelerated to 7 minutes 29 seconds (8.98 times longer than exact matching).

By moving the filtering of words longer than 2 characters from the fuzzy matching algorithm to the method that uses the algorithm, the speed is accelerated to 7 minutes 25 seconds (8.9 times longer than exact matching). The improvement is due to the reduction of the number of filters, because in the algorithm it had to be done before every check. That is, instead of a lot of identical checks in the fuzzy matching algorithm, only one check is performed before the algorithm starts.

By adding a check to see if the first letter of a word in the formulation matches the first letter of any keyword, the speed is significantly accelerated to 2 minutes 51 seconds (3.42 times longer than exact matching). We named this configuration "Delta".

By creating a character variable that stores the first letter of the word in the formulation and replacing all references to the word as an array of characters, in which one must constantly get the first element, with the character variable, the speed is accelerated to 2 minutes 37 seconds (3.14 times longer than exact matching).

By adding the condition that only variable keywords should be checked for fuzzy matches, the performance is accelerated to 2 minutes 36 seconds (3.12 times longer than exact matching). Hereinafter, we will call this configuration "Epsilon".

After removing formulation words, the part of speech of which could not be determined by analysis, the speed is accelerated to 2 minutes 35 seconds (3.1 times longer than exact matching).

By adding the condition that the word in the formulation must have more than 2 characters, and removing similar checks from the fuzzy matching algorithm, the speed is accelerated to 2 minutes 32 seconds, which is only 3.04 times longer than using only exact matching. Let us call this final configuration "Zeta". It allows for the best performance among fuzzy match search configurations while maintaining the absolute recall and precision (see Table 1).

While being representative in the performance aspect, ambiguity categories can be misleading in terms of recall and precision because of stacked paragraphs which increase the impact of search via fuzzy matching. Therefore, it may be more objective to consider time costs relative to the number of detected paragraphs (see Figure 2).

Table 1
Key configurations

Configuration	Average performance (seconds)	Results (paragraphs)	Precision	Recall
Exact matching	50	6983	100.00%	84.65%
Fuzzy match: no restrictions	2647	8492	97.14%	100.00%
Alpha	595	8260	99.87%	100.00%
Beta	456	8249	100.00%	100.00%
Gamma	452	8249	100.00%	100.00%
Delta	171	8249	100.00%	100.00%
Epsilon	156	8249	100.00%	100.00%
Zeta	152	8249	100.00%	100.00%

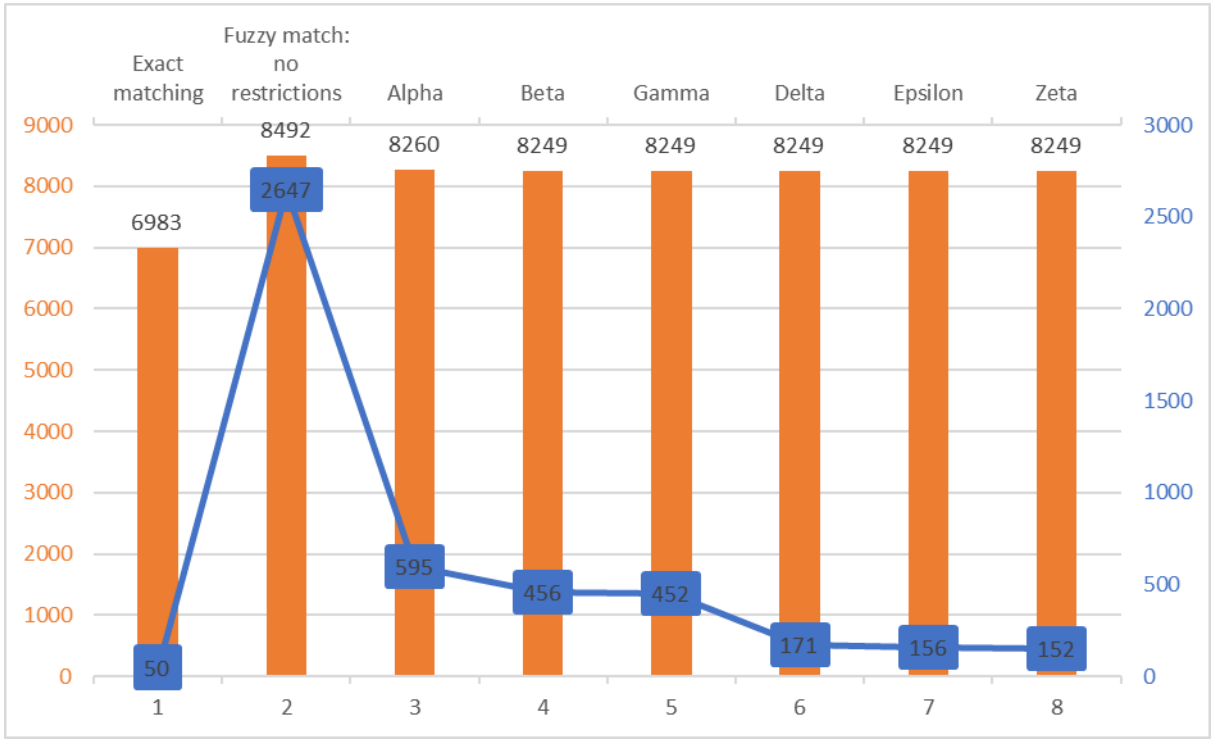

Figure 2: Time costs (blue line) relative to the number of results (orange bars) among different key configurations

Taking into account ambiguity categories helped to test search via fuzzy matching in partially unknown environment where it is hard to predict whether each paragraph will be listed only once or more times and thus take more time to process or quickly skipped because the paragraph contains no keywords.

5. Conclusions

Having tested the proposed method on a Ukrainian corpus, the use of the search procedure in the list of keywords via fuzzy matching was able to increase the recall to maximum, which means that all paragraphs de facto containing keywords for ambiguity detection are covered using the proposed method. It has absolute precision when using keyword search to detect ambiguity, which is possible

due to a known set of words that are used in search via fuzzy matching and the use of information about the part of speech and grammatical categories. The absolute precision means that no odd paragraph that does not contain keywords for ambiguity detection was covered. The increase in recall and precision was reached at the cost of performance. Time costs can be minimized by implementing optimization.

6. References

[1] A. Gorry, M.S. Scott-Morton, A Framework for Information Systems, Sloan Management Review, 13, 1, Fall 1971, pp. 56–79.

[2] N. Althuizen, Analogical Reasoning as a Decision Support Principle for Weakly-Structured Marketing Problems, 2006.

[3] D.A. Pospelov, Situational management. Theory and practice, Nauka, Moscow, 1986.

[4] O.I. Larichev, The Science and Art of Decision Making, Nauka, Moscow, 1979.

[5] V. Tsyganok, S. Kadenko, O. Andriichuk, P. Roik, Combinatorial Method for Aggregation of Incomplete Group Judgments, in: Proceedings of 2018 IEEE First International Conference on System Analysis & Intelligent Computing (SAIC), Igor Sikorsky Kyiv Polytechnic Institute, Kyiv, 2018, pp. 25–30 https://doi.org/10.1109/SAIC.2018.8516768

[6] A. Orlov, Management Decision-Making Methods: Textbook, KNORUS, Moscow, 2018.

[7] O. Mulesa, Methods of considering the subjective character of input data in voting, Eastern-European Journal of Enterprise Technologies, 1(3), (2015), 20–25.

[8] V.V. Tsyganok, S.V. Kadenko, O.V. Andriichuk, Simulation of Expert Judgements for Testing the Methods of Information Processing in Decision-Making Support Systems, Journal of Automation and Information Sciences 43(12), (2011), 21–32.

[9] T. Saaty, Decision Making for Leaders; the Analytical Hierarchy Process for Decisions in a Complex World, Wadsworth, Belmont, Calif., 1982.

[10] R. Alomari, H. Elazhary, Implementation of a Formal Software Requirements Ambiguity Prevention Tool, International Journal of Advanced Computer Science and Applications 9(8) (2018), 424–432 https://doi.org/10.14569/IJACSA.2018.090854

[11] S. Winkler, Ambiguity: Language and Communication, 2015.

[12] R.W. Shuy, Deceptive Ambiguity by Police and Prosecutors, 2017. URL: https://books.google.com.ua/books?id=zF0vDwAAQBAJ&lpg=PP1&ots=vicuMYw8cW&dq=Deceptive%20ambiguity%20by%20police%20and%20prosecutors&lr&hl=uk&pg=PP1#v=onepage&q

[13] B. Gleich, O. Creighton, L. Kof, Ambiguity Detection: Towards a Tool Explaining Ambiguity Sources, in: R. Wieringa, A. Persson (eds.) REFSQ 2010. LNCS, vol. 6182, Springer, Heidelberg, pp. 218–232, 2010 https://doi.org/10.1007/978-3-642-14192-8_20

[14] D.M. Berry, E. Kamsties, M.M. Krieger, From Contract Drafting to Software Specification: Linguistic Sources of Ambiguity – a Handbook, 2003. URL: https://cs.uwaterloo.ca/~dberry/handbook/ambiguityHandbook.pdf

[15] M.Y. Dubok, Keywords for Detecting Ambiguity of Expert Formulations, in: Proceedings of 2021 annual scientific and technical conference "Data Recording, Storage and Processing", Institute for Information Recording of National Academy of Sciences of Ukraine, Kyiv, 2021, pp. 127–129.

[16] M.S. Maučec, Z. Kačič, B. Horvat, Modelling Highly Inflected Languages, Information Sciences, Volume 166, Issues 1–4, 2004, pp. 249-269, https://doi.org/10.1016/j.ins.2003.12.004.

[17] M.Y. Dubok, V.V. Tsyganok, Quasi-inflection-based Part-of-Speech Tagging Method, Journal "Data Recording, Storage and Processing" 22(3) (2020) 96–106.

[18] S.J. Körner, T. Brumm, RESI - A Natural Language Specification Improver, 2009. URL: http://www.ipd.kit.edu/tichy/uploads/publikationen/217/ICSC2009.pdf

[19] L. Mich, M. Franch, P. Novi Inverardi, Market research on requirements analysis using linguistic tools, Requirements Engineering 9 (2004) 40–56 https://doi.org/10.1007/s00766-003-0179-8

[20] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, 2nd ed., MIT Press, Cambridge, Massachusetts London, 2001.