Integration of Ontological Case-Based Reasoning with Principal Component Analysis: Application to the IT Support Service

© Tatiana Avdeenko © Anastasiia Timofeeva © Ekaterina Makarova
Novosibirsk State Technical University,
Novosibirsk, Russia
avdeenko@corp.nstu.ru a.timofeeva@corp.nstu.ru katmc@yandex.ru

Abstract. In present paper we propose an original approach to the indexing of cases by ontology concepts, as a result of which the special semantic data matrix is generated. The elements of this matrix are semantic links between cases and terminal concepts of the ontology. This matrix contains knowledge about the most stable, non-trivial relationships between the ontology concepts that determine the most frequently used cases. To identify these groups of concepts we propose and approve an approach based on modification of the principal component analysis with use of combination of polychoric correlations and correlation ratio. Interpretation of the loadings matrix on the principal components allows us to identify groups of interrelated concepts from different hierarchical branches of the ontology. Thus, problems that are at the junction of different concepts can be identified. The proposed method is implemented in the knowledge management system for IT support service.

Keywords: case-based reasoning, ontology, principal component analysis, polychoric correlation.

1 Introduction

Maintenance (support) of the software is the process of improving, optimizing and correcting software defects after putting it into operation. Software maintenance is one of the phases of the software life cycle. In the course of maintenance changes are made to the program in order to correct the defects discovered during the use, as well as to add new functionality increasing the usability and applicability of the software.

There are two different points of view on the terms "software maintenance" and "software support". The first one considers these two terms as synonyms. We hold the opposite view on this issue, when there is a difference between these concepts. Maintenance of the software is executed by a maintainer who can be both the external organization or the organization, which uses the software (department or a separate employee). Support is provided exclusively by employees of the department of the organization that uses the software. They are less qualified specialists than maintainers.

To implement the stage of software maintenance in organizations there appear IT departments containing the staff of analysts, programmers, consultants, most of whose work consists of consulting support of the users. Typically, several maintenance lines are distinguished, differing, on the one hand, with the experience and qualifications of IT support specialists, on the other hand, the burden on consultants. On the zero-line (call-center, information center, hotline) consultants have not very much experience, but a very large flow of telephone calls from customers.

users, the IT consultant has to determine the scope of the problem, to analyze the primary information and, using personal experience and (or) reference materials, to formulate the answer to the question. Our analysis shows that the average time taken to make a decision by a novice consultant and an experienced specialist differs 2-4 times with the same complexity of the problem. At the same time, the use of even very simple means of recording and extracting knowledge about solving similar problems in the past (handwritten, text editor, spreadsheet editor, etc.) makes it possible to bring the effectiveness of a novice consultant closer to the effectiveness of the experienced analyst. Thus, it seems promising to build a knowledge management system that helps to accumulate, systematize, integrate and effectively use the experience of analysts to solve IT problems of employees of the organization.

The most important component of the knowledge management system is the knowledge representation model, as well as the mechanism that allows this knowledge to be extracted and adapted to the solution of the required problem. It seems to us that Case-Based Reasoning (CBR) is best suited for solving the problems of IT users than Rule-Based Reasoning (RBR) [1]. First, cases are the most natural way to write down the experience of already made decisions, implementation of the system is reduced to the identification of essential features describing the case. Second, identical or nearly identical user's problems are very common, especially if the organization has many branches. Third, it is almost impossible to build static rule-based model in an extremely rapidly changing IT field, when very often

Proceedings of the XX International Conference "Data Analytics and Management in Data Intensive Domains" (DAMDID/RCDL'2018), Moscow, Russia, October 9-12, 2018

new products and releases come out, interfaces and functionality change. And, finally, what is the most important for the dynamic IT field, CBR-systems can be self-learning, thus, it is possible to obtain new cases and even rules from the case base.

At the same time there are essential shortcomings of traditional CBR. The major one reveals itself when the number of cases accumulated in the knowledge base becomes great. The large case base results in reduced system performance. It is difficult to determine good criteria for indexing and comparison of cases.

To overcome the disadvantages of traditional CBR, it has been widely integrated with other methods in various application domains [2,3]. Some systems (ADIOP, CADRE, CADSYN, CHARADE, COMPOSER, IDIOM, JULIA) integrated CBR with constraint satisfaction problem (CSP) algorithm. Some systems (ANAPRON, AUGUSTE, CAMPER, CABARET, GREBE, GYMEL and SAXEX) combined CBR with rule-based reasoning (RBR) approach. It is worth to noting that the first prototype of the system, integrating CBR with RBR was CABARET system [4]. In [5] it is proposed possible connection of CBR with RBR and its application to the financial domain implemented in prototype system MARS. Various types of coupling models involving combinations of CBR and RBR such as sequential processing, co-processing and embedded processing are described in [6]. CBR can be combined with fuzzy logic in fruitful ways in order to handle imprecision. A usual approach is the incorporation of fuzzy logic into a CBR system in order to improve CBR aspects [7-10]. In [11] combinations of CBR with other intelligent methods are considered.

Ontologies facilitate knowledge sharing and reuse. They can provide an explicit conceptualization describing data semantics and ensuring common understanding of the domain knowledge. To enhance the case retrieval and case adaptation, in [12] it was created the domain ontology in the field of railroad accidents from which cases are instantiated in the case base and operational ontology in the form of decision rules. In [13] integration of CBR with domain ontology is applied for Fault Diagnosis of steam turbine. In [14] jCOLIBRI (Cases and Ontology Libraries Integration for Building Reasoning Infrastructures) is proposed to create knowledge-intensive and domain-independent CBR architecture. In [15] ontology-oriented CBR approach is presented for trainings adaptive delivery.

Despite the fact that there is a significant number of papers concerning integration of CBR with other intelligent methods, and even with the ontologies, only very few papers consider its application for the IT consultation problem. For example, in paper [16], the representation of the IT application domain in the form of ontology was used to improve the semantic search for documents based on the indexing of documents by the ontology concepts in comparison with the usual indexing by keywords. However this paper does not use possibilities if CBR in order to apply past information for solving current problems.

In this paper we propose an original approach to the organization of the knowledge based on the integration of the case base with the domain ontology. As a result of such integration we obtain a semantic matrix, the application to which methods of data analysis allows us to improve the procedure for retrieving relevant cases for solving IT user's problems.

The paper is organized as follows. In Section 2, we describe the most important features determining the structure of case base for the IT support field. We consider the problems We accumulated the cases of IT problems arising from users working in the personnel and accounting departments of the commercial company, although similar problems can also be experienced by IT users of non-profit companies, universities, etc. In section 3 we describe the ontology of concepts to which the cases in the IT support field could be referred. In section 4 the proposed mechanism for the integration of cases with the ontology concepts and obtaining the semantic matrix "case–terminal" are presented. In Section 5 the modification of principal component analysis is given and its application to the semantic matrix allowing to identify groups of interrelated concepts and to interpret them. In section 6 we give conclusion.

2 The structure of case base

CBR is an approach that allows to solve a new problem by using or adapting a solution previously taken in a similar situation. In CBR method the knowledge base consists of cases forming a case base. A case is a description of a problem or situation in conjunction with a detailed enumeration of actions taken in this situation to solve the problem. When a new situation is considered, the system finds a similar case in the knowledge base as an analog of the problem being solved and tries to use the solution of the found case. If necessary, a close case is adapted to the current situation. After applying the solution obtained from CBR to the current problem, the results are analyzed, then a new case is added to the case base for its use in the future. Thus, CBR-method includes four stages that form the so-called CBR-cycle, or the 4R cycle (Retrieve, Reuse, Revise, Retain) [17].

Case-based reasoning (CBR) literature defines the process of building case base as a hard and time-consuming task. In [18] methods are presented that can be used to build the initial case base including the steps taken in order to make sure that the quality of the initial case set is appropriate. The case should include the following elements: description of the situation with the help of attributes; the decision that was made in this situation; the result of applying the solution.

When developing a case structure for describing the problems of IT users, the *description of the situation* should contain, if possible, all the information that is necessary to achieve the goal, i.e. choosing the most appropriate solution. The more detailed the expert will describe the current problem, the faster the answer will be found. Quite often, users form a request very briefly, for example: "There was a problem in the personnel order." Here it is not clear in which order an error occurred, because the order number is not specified, and it is not specified which kind of problem arose. To clarify the issue the time is wasted, and the solution will be given to the user not immediately, but after a while.

The *decision that was made* contains: a set of operations that must be performed to obtain successful result, i.e. for the decision of a question of the user. The description of the solution may include links to other cases, text information, an attached document with an instruction, and so on. The *result of applying the solution* is the feedback that occurs when the solution is applied to the current situation.

The cases can be represented in various ways. It is necessary to choose a case representation model based on the overall objectives of the system. The main problems when presenting a case are: the choice of information that should be included in the description of the case, the search for a convenient case structure and the organization of a knowledge base for optimal and efficient search.

We propose a hierarchical structure of the case in the field of IT support, which is specified using the *Precedent* class. The purpose of this class is to create the most complete structure for the information about the cases for counseling (solving the user problem), and also to establish a connection with the domain ontology. This class includes three groups of properties - *Main*, *Changes* and *Files*, whose purpose is structurally and meaningfully to divide the information included in the description of the case (see Figure 1).

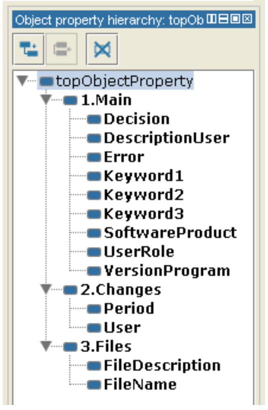
```
Object property hierarchy: topOb
▼ topObjectProperty
  ▼ 1.Main
      Decision
      DescriptionUser
      Error
      Keyword1
      Keyword2
      Keyword3
      SoftwareProduct
      UserRole
      VersionProgram
  ▼ 2.Changes
      Period
      User
  ▼ 3.Files
      FileDescription
      FileName
```

Figure 1 The sructure of *Precedent* class

The Main property has the following subordinate properties:

– *Decision* - a complete description of the sequence of actions (technology) to solve the problem;

– *DescriptonUser* - information about the problem that the user informs the consultant when formulating the request;

– *Error* - technical error that can be solved only by reprogramming (filled or not);

– *Keyword 1 ... 3* - one or more attributes for the concepts of the domain that characterize the problem. With these attributes the case is related with the ontology;

– *SoftwareProduct* - software product where a user error occurred is made as a selection from the list (1C, Axapta, etc.);

– *UserRole* - user can be a human resources officer, an accountant, a timekeeper, a chief accountant, a deputy chief accountant, an auditor, etc. The functionality that can be used to solve the problem depends on the user's role;

– *VersionProgram* - release or version of the software product. Software products are constantly updated, the developers fix bugs, therefore, before answering the user's question, it is necessary to understand which release the user is working on.

The *Changes* property of the *Precedent* class is useful for the case where several consultants work with the case base. You can always understand who changed the case and when. This attribute has the following subordinate properties:

– *Period* - the date and time when the case was created, or changes were made;

– *User* - the name of the user who has made the change.

The *Files* property has the following subordinate properties:

– *FilesDescription* - a brief description of the file;

– *FileName* - the path to the file attached to the case. This can be a file with the error that occurs in this request, or a file with a troubleshooting guide.

The proposed structure of the case, which was described above, has necessary completeness and non-redundancy, since it specifies the main characteristics of the user's request: user description, error, a set of keywords, software product, software version, user role and, finally, the decision of the user problem. The consultant gives a professional description that characterizes the user's problem. The case also contains information about making changes to the case: the date when the changes were made, by whom they were made, so that it is possible to analyze the changes made. One can attach a file to the case which contains instructions for solving the problem, or user errors that can be attached to the case. This information is sufficient to solve the user's problem and quickly find a suitable precedent.

A set of *Keyword 1..3* properties is reserved to establish relations from the Case to the concepts of the Domain Ontology described in the following section. These relationships allow to organize efficient retrieval of cases being relevant to the current problems.

3 Domain Ontology in the IT support field

The concepts of the IT support field (relevant to personnel and accounting departments user's problems) are organizes in the form of ontology. Ontology is a formal explicit description of the concepts and the relations between them. The ontology can be represented by the following tuple

$$O = \langle C, R, S, T \rangle,$$

where $C = \{c_i \mid i = \overline{1,n}\}$ is a set of classes (concepts) describing the basic notions of the domain;

$R = \{r_i \mid i = \overline{1,m}\}$ is a set of binary relations between the classes, $R \subseteq C \times C$, $R = \{R_{ISA}\} \cup \{R_{ASS}\}$, R_{ISA} is an antisymmetric, transitive, non-reflexive hierarchy relation; R_{ASS} is an associative relationship used to establish a link from the case to the ontology; $S = \{s_i \mid i = \overline{1,k}\}$ is a set of class properties; T is a set, which determines the vocabulary of the domain concepts, built on a set of basic terms (a set of ontology classes) $B = \{b_i \mid i = \overline{1,n}\}$. The structure of the class is defined as

$$c = \left\langle Name, (is - a\ \ c_{parent}), (s_1,...s_{n(c)}) \right\rangle,$$

where $c, c_{parent} \in C$ are the ontology classes connected by the hierarchy relation R_{ISA}, $s_i \in S$ are the class slots, $Name_c \in B$ is the class name being the base term of the vocabulary T. Taxonomy of classes is formed by means of indicating the relation «is-a» and the name of the parent c_{parent} in the descendant class. Terminal concepts that have no descendants will be called terminals.

Ontology was created in the Ontology Editor Protégé 4.2 which is free software. The ontologies built in this editor are exported to many formats, this software has an open and easily extensible architecture. A fragment of the hierarchy of the top-level concepts, which are direct descendants, is shown in Figure 2.

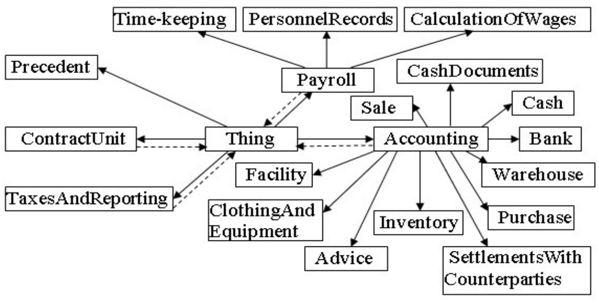

Figure 2 Ontological graph of top-level concepts

The main classes of the top-level ontology are *Precedent* (class for cases instances), *Accounting*, *Payroll* and *ContractUnit*. The *Accounting* concept describes the main subsections of accounting. Accounting is an orderly system for collecting, recording and summarizing information in monetary terms about property, liabilities of organizations and their movement through continuous, continuous and documented accounting of all business transactions.

Accounting forms a taxonomy, which is formed by twelve subordinate concepts. The *Bank* and *Cash* concepts reflect the conduct of transactions with cash. The Concept *Sale* reflects the design of operations for the sales of goods and services to customers, this concept is one of the main for the conduct of the enterprise. The Concept *Purchase* is designed to take into account the conduct of transactions for the purchase of goods and services from suppliers. The *Warehouse* concept reflects the accounting of the movement of materials in the warehouse, etc. These concepts help to express the meaning of questions that from users. For example, the question of the user "In the receipt of goods, the rate in the nomenclature is shown without VAT, why?". It is advisable to relate this case to the *Purchase* concept.

The concept *Payroll* describes the basic subsections of the taxonomy "Calculations with the staff". In this taxonomy, the tasks of automating the activities of both managers who make decisions on the salary of staff and accountants of salaries are being solved. Users can have various questions related to these concepts. For example, a human resource officer may have the following questions: "When creating an employee, there is a mistake that an individual already exists, what should I do?", "How do I make a sick list?" These questions can be related to the *PersonnelRecords* concept.

The *ContractUnit* concept describes the main subsections of the subject area "Contractual Block". This block is intended to automation of work in the sphere of registration and conducting contracts of counterparts. For example, a contractor may have the following questions: "How to put the contract into effect?", "Why is there no accrual under the contract?" related to the *ContractCounterparties* concept.

The hierarchy of concepts contains 71 concepts of the *Payroll* taxonomy, 82 concepts of the *Accounting* taxonomy and 11 concepts of the *ContractUnit* taxonomy.

4 Integration of Case Base with the Domain Ontology

In the conventional CBR method, the measure of closeness (distance) in a multidimensional space defined by the case features is used to retrieve cases. However, not necessarily the closest case is the most relevant in the semantic terms. Therefore it seems promising to make a comparison between the current situation and cases, assessing the degree of their connection with the concepts of ontology. Thus, closeness of cases to each other is estimated by the degree of semantic closeness of the concepts associated with these cases. To achieve that it is necessary to determine the semantic links of the newly introduced cases with the ontology concepts at the stage of creating the initial case base.

The link of the instances of the *Precedent* class with the ontology concepts is established by setting the associative relation R_{ASS} for the *Keywords* property group for *Precedent* class that has type D_{class}. Specifying the type D_{class} for each of the I properties *Keywords* involves specifying an additional argument – the associated ontology concept. If, for example, the i-th slot of the group *Keywords* has the type D_{class} with the associated class C_i, then as slot values when creating the class *Precedent* instances we can use the classes of the transitive closure $Tr(C_i)$ of the concept C_i including $C_i = C_i^{(0)}$ and all its subclasses below in the hierarchy: $Tr(C_i) = \{C_i = C_i^{(0)}\} \bigcup ISA(C_i^{(0)})$, where

$ISA(C^{(0)}) = \bigcup_{l=1}^{L}\{C^{(l)} \in C \mid \exists R_{ISA}(C^{(l-1)}, C^{(l)})\}$, L being the maximum depth of the class c_i descendants. Here the classes $Precedent$ and C_i are connected by the associative relation $R_{ASS}(Precedent, C_i)$.

Establishing the connection of a specific case with the ontology, the analyst chooses concepts that are semantically closest to the case. It can be either terminal, the most specified concepts, and non-terminal (intermediate) concepts that have a more general meaning. It should be emphasized that in our approach we allow setting several links for one case with different ontology concepts. This expands the expressive possibilities of the approach and can be usedwhen the problem arises at the junction of several concepts, and its adequate description requires consideration of this interdisciplinary character.

Let in addition to the concept name C_i, the weight value v_i, $0 \le v_i \le 1$, $\sum_{i=1}^{I} v_i = 1$, is given as an attribute for the i-th slot of the group $Keywords$ establishing the strength of the relation between the case and the ontology concept. The more is the weight v_i, the closer by the meaning the case is to the corresponding concept of the application domain.Let wehave J terminals in the ontology, and each terminal kw_j, $j = \overline{1,J}$ corresponds to the weight w_j, $j = \overline{1,J}$, $\sum_{j=1}^{J} w_j = 1$, that can be computed from the weights v_i for the cases and the weights of the hierarchy relations in the ontology. The procedure for forming a vector of weight coefficients w_j, $j = \overline{1,J}$, for the terminals kw_j, $j = \overline{1,J}$, can be presented as follows. Suppose that considered case is related to the concepts C_1, $C_2, ..., C_I$.

1. First we assign $w_j = 0, \forall j = \overline{1,J}$.

2. Second, cycle for all concepts $C_i, i = \overline{1,I}$ connected with the case:

- if C_i is a terminal concept ($kw_j = C_i = C_i^{(0)}$), then $w_j = w_j + v_i$;
- if C_i is not a terminal concept, i.e. terminal concept kw_j is the L- level descendant of the intermediate concept C_i, $kw_j = C_i^{(L)}$, then $w_j = w_j + v_i \cdot \prod_{l=1}^{L} v_i^{(l)}$, where $v_i^{(l)}$ is the weight of the hierarchical relation $R_{ISA}(c_i^{(l-1)}, c_i^{(l)})$ from the concept parent $C_i^{(l-1)}$ to the child concept $C_i^{(l)}$ on the way from the concept C_i connected with the case instance to the terminal concept kw_j. The weights of concepts being descendants to the one parent in the ontology are considered to be the same.In principle, if the descendants of a certain parent have unequal influence on the parent concept, then it is possible to introduce weight coefficients into the taxonomy. To do this, each concept with a parent is added an attribute – the weight of the concept. In present version of the ontology it is assumed that all the children of the same parent have the same weight, equal to $1/G$, where G is the number of children of the given parent.

Thus, all the cases stored in the case base are connected with the ontology concepts. Each concept is included into the case representation with a weight calculated on the basis of the associative relationships between the case and the ontology concepts. As a result we obtain semantic matrix with the values of weights w_j, $j = \overline{1,J}$, for each case in the case base being the instances of the $Precedent$ class. The number of rows of the semantic matrix is equal to the number of cases, and the number of columns is equal to J – the number of the ontology terminal concepts. One can further apply data mining and machine learning methods to the semantic matrix extracting knowledge from data. In the next section we propose application of the principal component analysis to this data.

5 Modification of Principal Component Analysis for Grouping Ontology Concepts

Despite the fact that the concepts are carefully organized into the ontology by a domain specialist, the IT problems of the users are often arise at the junction of various concepts. Therefore, the cases often refer to different hierarchical branches of the ontology. The application of methods for grouping the concepts could identify the most frequent combination of concepts describing the user's problems.

To group similar concepts, we apply the principal component analysis. However, the values of weight coefficients, which show the semantic connection between concepts and cases, take a limited number of rational values as a result of multiplication of simple fractions. Thus, the original data are discrete. The standard principal component analysis uses a correlation matrix consisting of Pearson's correlation coefficients, which are based on the assumption of a multidimensional normal distribution of variables. In our case this assumption is violated.

It is more correct to use special correlation measures for discrete variables, in particular, polychoric correlations. They have several advantages over the standard Pearson's correlation coefficient. First, they allow a better recovering of the theoretical model by means of factor analysis [19]. Secondly, they are a measure of monotonous dependence, that is, they allow us to reveal nonlinear relationship. Third, due to the fact that only the order of the values is taken into account, not the interval between them, polychoric correlations are more robust to outliers.

However, they have a number of drawbacks. First, estimation of polychoric correlation is based on the optimization procedure and uses the values of bivariate normal distribution function, so the calculation is rather slow with a large number of categories. To solve this problem, we developed an algorithms described in [20].

Second, the definition of polychoric correlation is based on the assumption of a joint normal distribution of latent variables [21]. To overcome this limitation, one can use skewed distributions and distributions with heavy "tails" [22]. In particular, in [23] generalizations of the polychoric correlation were proposed to improve flexibility. For this purpose bivariate Student and generalized lambda distributions were used allowing to increase the number of cases in which the data are consistent with distributional assumptions.

Finally, third, it was found that with a certain structure of the contingency table, polychoric correlation erroneously indicates a strong relationship. This is a particular problem for sparse frequency tables with a large number of zero values. This problem arose in the course of analysis of the semantic connection between concepts and precedents. For a number of concepts, the structure of contingency tables containing the frequency d_{ij} of the fact that the semantic connection for the first concept was assigned to the i-th category, and for the second to the j-th category, was reduced to the form presented in Table 1, where v_1, v_2 are the weights for the first and second concepts.

Table 1 Two-way contingency table

Semantic correspondence	v_2=0, no correspondence	v_2>0, some correspondence
v_1=0, no correspondence	$d_{11}\neq 0$	$d_{1k}\neq 0$
v_1>0, some correspondence	$d_{k1}\neq 0$	$d_{kl}=0,\ \forall\, k,\ l\neq 1$

In this case, the polychoric correlation is equal to -1, which indicates a strong negative relationship. From the Table 1 it can be clearly seen that this problem corresponds to the situation where there are no cases associated with two selected concepts, but a lot of cases not related to either one or the other. Logically, this correlation must be zero. Thus, the polychoric correlation is erroneous.

In order to avoid such problem situations when calculating the correlation matrix, it is proposed to replace the polychoric coefficient by the correlation ratio, which is actively used in factor analysis of mixed data (FAMD) [24].Thus, for grouping similar concepts of ontology a method is proposed, which consists of the following steps.

*Step 1.*Calculation of polychoric correlations ρ.

*Step 2.*Identification of problem situations by frequency tables, as well as by the values of the polychoric correlations close to –1.

Step 3. Replacement of polychoric correlations in the problem situations, revealed at the step 2, by the values of the correlation ratios η, calculated as the mean between $\eta_{Y|X}$ and $\eta_{X|Y}$, taken with sign(ρ).

Step 4. Based on the resulting correlation matrix consisting of polychoric correlations and correlation ratios, the implementation of the principal component analysis, the calculation of loadings on the principal components and the extraction of interrelated concepts.

With the use of this method, five principal components were extracted. It allows to present concepts of the domain ontology in a space of small dimension. The loadings on the principal components are presented in Table 1. Their absolute values reflect the closeness of relationship between the concepts and the principal components. The advantages of using the proposed approach in comparison with the standard one (calculation of the Pearson's correlation) should consist in increasing the percentage of variance of concepts explained by the extracted components. So, with the standard approach, the five extracted components sum up only 38,9% of the initial variation of the concepts, whereas the proposed approach allows to explain 55,1% of the variance. As a result, it allows us to break down the concepts into a smaller number of groups, the interconnections within which are closer.

The obtained results can be interpreted from the point of view of IT consulting practice. The concepts, combined the first principal component, reflect the most common user errors in the calculations. If there is an incorrect calculation, then as a rule the error arises either in the incorrect formulation of vacation or sick leave, and the problem with the time-keeping. At the same time, problems with vacation and sick leave can lead to the errors in reporting on taxes (*2-NDFL* and / or *6-NDFL*). Reports on personal income tax are also interrelated, if there is an error or a question on one report, then the second one most likely will also have an error. The second group of concepts deals with problems in personnel reporting. If there is a question on the admission / dismissal orders, there will be a problem with personnel reporting, and vice versa, if there is an error in the report, then it is worth checking the personnel orders (admission, dismissal).

The concept *Recalculation* is connected with the third principal component. When recalculating, as a rule, users forget to remake taxes, so there are errors in taxes, insurance payment and wirings as a consequence.

Wirings also fell into the fourth group. The problem with wirings also arises when the calculation is incorrectly. These are interdisciplinary issues. *Calculation* and *Payment at the average wage* are mutually exclusive types, that is, at the same time a sick leave (payment at the average wage) and calculation (salary payment) cannot meet together, this is a mistake. So, the user needs to make changes.

The fifth principal component associates with *Calculation prepayment*, *Calculation of deductions* and *Salary*. In the payment documents, it is always necessary to check the calculation of deductions, so that everything is reflected correctly in the 6-NDFL statements. Also through salary payment documents a prepayment is formed. The prepayment is usually a fixed amount, sometimes as half of the salary, and then in the payment document deductions are reflected. But such questions are rare.

Thus, concepts are combined into the groups by how often the errors occur when working with the software products. The first group of concepts is associated with the most frequently encountered user's problems, since the calculation errors are usually more frequent. The second most popular are the problems with personnel documents (the errors of the second group). The problems with taxes and the average wage are not very frequent operations, this part is fairly well implemented in the programs. So, there are fewer questions connected with this group of concepts. The prepayment, deductions and salary are, as a rule, the most recent operations in the general list of all operations, and if everything was done correctly in the previous steps, there are very few errors associated with this group.

As a result, concepts from different hierarchical branches of the ontology were grouped.

Table 2 Loadings on principal components and cumulative percentage of explained variance

Concepts	Principal components				
	1	2	3	4	5
Order on admission		0,664			
Order of dismissal		0,573			
Vacation	-0,629				
Sick leave	-0,514				
Time-keeping	-0,549				
Reporting		0,806			
Calculation prepayment					0,573
Calculation				0,748	
Payment at the average wage				-0,543	
Calculation of deductions					0,476
Salary					0,576
Recalculation			-0,607		
2-NDFL	0,838				
6-NDFL	0,727				
Insurance payment			-0,677		
Other taxes			-0,491		
Wirings			-0,461	0,415	
Cumulative explained variance, %	**14,7**	**27,0**	**37,7**	**46,6**	**55,1**

6 Conclusion

Thus, we proposed an original approach to the indexing of cases through integration with the ontology concepts, as a result of which the semantic matrix "case-terminal" is generated. The elements of this matrix are calculated on the basis of the initial assignment of the weights to the relationships of cases with the ontology concepts, and the subsequent "descending" of the weights to the lowest level (terminal concepts) of the hierarchy. This numerical matrix contains the knowledge about the most stable, non-trivial relationships between the ontology concepts that determine semantics of the frequently used cases.

To identify groups of interrelated concepts we proposed modification of principal component analysis. Its main difference from the standard method is that instead of Pearson correlation coefficients combination of polychoric correlations and the correlation ratio is used. It allows to increase the percentage of variance of concepts explained by the principal components. Interpretation of the matrix of loadings on the principal components allows us to identify groups of interrelated concepts from different hierarchical branches of the ontology. Thus, problems that are at the junction of different concepts can be identified. The latter can be used for the intelligent help for the user what additional concepts (in addition to the one already selected) to choose for the link with the current case (user problem).

According to the users of the IT support department, after the introduction of the knowledge management system, user satisfaction increased by 15% in average. User satisfaction was measured as an integrated indicator, which includes both the quality of problem solving and the time during which the user received a response from the support service.

One of the directions for further research involves the introduction of knowledge domains based on the extending the ontology for the users of other departments of the organization, and, accordingly, the accumulation of cases in these domains. Another direction involves using principal components to structure the case base using clustering methods. Standard clustering algorithms are sensitive to noise in the data [25], so the reduction in the dimension is often used for preliminary data processing. According to the results of [26], this allows increasing the classification accuracy. In addition, the statistical efficiency of using the principal component analysis should consist in increasing the stability of clustering results.

Acknowledgments. The reported study was funded by

Russian Ministry of Education and Science, according to the research project No. 2.2327.2017/4.6.

References

[1] Watson, I., and Marir, F.: Case-based reasoning: A review. The Knowledge Engineering Review 9(4), 327-354 (1994)

[2] Marling, C., Sqalli, M., Rissland, E., Hector, M.A. and Aha, D.: Case-based reasoning integrations. AI Magazine 23(1), 69-86 (2002)

[3] Yang, H.L. and Wang, C.S.: Two stages of case-based reasoning - Integrating genetic algorithm with data mining mechanism. Expert Systems with Applications 35, 262–272 (2008)

[4] Rissland, E.L. and Skala, D.B.: Combining case-based and rule-based reasoning: A heuristic approach. In: Eleventh International Joint Conference on Artificial Intelligence, pp. 524-530. IJCAI-89, Detroit (1989)

[5] Dutta, S., Bonissone P.P.: Integrating case- and rule-based reasoning. International Journal of Approximate Reasoning 8(3), 163-203 (1993)

[6] Prentzas, J., Hatzilygeroudis, I.: Categorizing Approaches Combining Rule-Based and Case-Based Reasoning. Expert Systems 24, 97-122 (2007)

[7] Cheetam, W., Shiu, S.C.K., Weber, R.O.: Soft Case-Based Reasoning. Knowledge Engineering Review 20, 267-269 (2006)

[8] Pal, S.K., Shiu, S.C.K.: Foundations of Soft Case-Based Reasoning. John Wiley (2004)

[9] Avdeenko, T., Makarova, E.: Integration of case-based and rule-based reasoningthrough fuzzy inference in decision support systems. Procedia Computer Science 103, 447-453 (2017)

[10] Avdeenko, T.V., Makarova, E.S.: Acquisition of knowledge in the form of fuzzy rules for cases classification. Lecture Notes in Computer Science 10387, 536-544 (2017)

[11] Prentzas, J. and Hatzilygeroudis, I.: Combinations of Case-Based Reasoning with Other Intelligent Methods. CEUR Workshop Proceedings 375, 55-58 (2008)

[12] Maalel, A., Mejri, L., Hadj-Mabrouk, H., Ben,Ghézela H.: Towards a Case-Based Reasoning Approach Based on Ontologies Application to Railroad Accidents, In: Proceeding of Third International Conference of Data and Knowledge Engineering (ICDKE), 48-55 (2012)

[13] Dendani-Hadiby, N., Khadir, M.T.: A Case based Reasoning System based on Domain Ontology for Fault Diagnosis of Steam Turbines. International Journal of Hybrid Information Technology 5(3), 89-103 (2012)

[14] Recio-Garía, J., and Díaz-Agudo, B.: Ontology based CBR with jCOLIBR. In: Applications and Innovations in Intelligent Systems XIV, pp. 149-162. Springer (2007)

[15] Mansouri, D., Hamdi-Cherif, A.: Ontology-oriented case-based reasoning (CBR) approach for trainings adaptive delivery. Recent Researches in Computer Science. In: Proc. of the 15th WSEAS International Conference on Computers, Corfu Island, Greece, July 15-17, pp. 328-333 (2011)

[16] Shanavas N, Asokan S.: Ontology-Based Document Mining System for IT Support Service. In: Procedia Computer Science. International Conference on Information and Communication Technologies (ICICT 2014), 46, 329-336 (2015)

[17] Aamodt, A., Plaza, E.: Case–Based Reasoning: foundational issues, methodological variations, and system approaches. AI Communications 7(1), 39-59 (1994)

[18] Wienhofen, L.W.M., Mathisen, B.M.: Defining the Initial Case-Base for a CBR Operator Support System in Digital Finishing: A Methodological Knowledge Acquisition Approach. Lecture Notes in Computer Science 9969 (1), 430-444 (2016)

[19] Holgado–Tello, F.P., Chacón–Moscoso, S., Barbero–García, I., Vila–Abad, E.: Polychoric versus Pearson correlations in exploratory and confirmatory factor analysis of ordinal variables. Quality & Quantity 44, 153-166 (2010)

[20] Timofeeva, A. Y.: Data type detection for choosing an appropriate correlation coefficient in the bivariate case. CEUR Workshop Proceedings 1837, 188-194 (2017)

[21] Olsson, U.: Maximum Likelihood Estimation of the Polychoric Correlation Coefficient. Psychometrica 44, 443-460 (1979)

[22] Uebersax, J.S., Grove,W.M.:A Latent Trait Finite Mixture Model for the Analysis of Rating Agreement. Biometrics 49, 823-835 (1993)

[23] Timofeeva, A. Y., Khailenko, E.A.: Generalizations of the polychoric correlation approach for analyzing survey data. In: Proc. 2016 11th International Forum on Strategic Technology, IFOST 2016, pp. 254-258 (2016)

[24] Pagès, J.: Multiple Factor Analysis by Example Using R. London, Chapman & Hall/CRC The R Series (2014)

[25] Balcan, M.F., Liang, Y.,Gupta P.: Robust hierarchical clustering. The Journal of Machine Learning Research 15, 3831-71 (2014)

[26] Ding, C., Xiaofeng H.: K-means clustering via principal component analysis. In: Proc. of the twenty-first international conference on Machine learning. ACM (2004)