Constraint Satisfaction Method to Search Patterns in the Data Taking into Account the Hierarchy of Features

Alexandr Zuenko
Institute of Informatics and Mathematical Modelling
Kola Science Center, the Russian Academy of Sciences
Apatity, Russia
Email: a.zuenko@ksc.ru

Olga Zuenko
Institute of Informatics and Mathematical Modelling
Kola Science Center, the Russian Academy of Sciences
Apatity, Russia
Email: o.zuenko@ksc.ru

***Abstract*—This article considers how the analysis of knowledge from subject domain ontology can help discover the most interesting and well-interpreted patterns in data. For this research area, the term "Pattern discovery guided by ontology" is used in the literature, and ontologies are considered as a means of semantic pruning of the search space. The analysis of ontologies in the pattern discovering can significantly reduce the enumeration of alternatives by pruning the search space, and also allows you to consider the elements of patterns at various levels of abstraction. The proposed approach to Data Mining is based on compact representation of the training sample using specialized matrix-like structures and the application of original inference procedures in these structures. Research lies at the intersection of such areas of artificial intelligence as data mining and semantic technologies for the representation and processing of information.**

***Keywords*—frequent pattern discovery, data mining, ontology, machine learning, constraint satisfaction problem, semantic technology**

I. Introduction

The presented research continues a series of works that deal with the application of the author's approach to solving Data Mining (DM) problems. Previously, the author's methods of clustering, closed pattern discovery, associative rule discovery were presented [1] as well as method that accelerates generating JSM-hypotheses in large databases [2]. The developed methods relate to methods of explainable Artificial Intelligent.

The article considers how the analysis of knowledge from subject domain ontology can help discover the most interesting and well-interpreted patterns in data by the end user. For this research area, the term "Pattern discovery guided by ontology" is used in the literature [3], and ontologies are considered as a means of semantic pruning of the search space. The analysis of ontologies in the pattern discovery can significantly reduce the enumeration of alternatives by pruning the search space, and also allows you to consider the elements of patterns at various levels of abstraction. Research lies at the intersection of such areas of artificial intelligence as DM and semantic technologies for the representation and processing of information.

In [2], a method was proposed for searching closed frequent patterns of the required type based on Constraint Programming Paradigm using the original representation of the training sample in the form of table constraints and the author's methods of constraint satisfaction. However, this method involves two stages in its implementation: 1) the stage of generating candidates for closed patterns and 2) the stage of verification of candidates and selection of those that satisfy the closeness property.

This publication presents research on the development of a method for closed frequent pattern discovery, taking into account hierarchy of features. The method is designed to quickly discover and enhance the interpretability of cause-and-effect relationships in multilevel descriptions of objects. Unlike the author's previous developments, the method avoids generating "redundant" nodes in search tree since it excludes the preliminary stage of generating candidates for the desired patterns.

II. Statement of the problem under discussion

We will provide information needed for further discussion [4]–[7].

As the initial information for the frequent pattern discovery problem is a *transactional database*, each row of which contains a *transaction identifier*, as well as a list of *transaction elements*. For example, analyzing purchases of goods in a store, the receipt number can be as a transaction identifier, and names of the purchased goods can act as elements. Also, the transactional database can be presented in the form of binary *object-feature table* where transactions are mapped to *objects*, and transaction elements are mapped to *features*. There is a "1" at the intersection of a row and a column in the table only if the object has this feature. A part of transactional database that is analyzed when pattern discovering we will refer to as a *training sample*.

A *pattern A* is any subset of features (elements). A *frequent pattern* is a set of features *A* that occurs at least in θ objects of the training sample. The θ is called a *frequency threshold*. The number of objects in which pattern *A* occurs is called an *absolute support* of pattern *A* and denoted *freq*(*A*).

It is usually necessary to find not all frequent patterns, but only those that have interesting features to the end user. Such patterns we will be refered to as *interesting*. Closed patterns are often considered as interesting because they can be used to express all other patterns. A closed set of features (pattern) is such that objects that have all these features at the same time do not have any other common features.

When generating interesting patterns, sometimes it is not enough to simply limit ourselves to searching for sets of features that meet the requirements for frequency and closeness, since there are too many such patterns and/or they are not well interpreted by the end user.

In this work, when discovering interesting patterns, an additional constraint is considered: features can be arranged in hierarchies. The inclusion of additional information about the grouping of elements in the form of hierarchies increases the interpretability of the DM results by the end user.

There are distinguishes of the representation of partonomies (based on relation "part–whole") and taxonomies (using relation "class–subclass") within the framework of the considered approach. They are explained below using the following example.

To illustrate the proposed method of pattern discovery taking into account hierarchy of features let's consider simplified example using the Figure 1. Let there be a set of hierarchically ordered features: the right tree corresponds to the taxonomy and the left one corresponds to the partonomy.

Now let's clarify which elements of hierarchies can be present in the records of the transactional database, that is, to form the initial description of transactions, and which can occur only in the generated patterns.

For a taxonomy, the transaction elements are leaf elements. In partonomy, each element can be included in the initial description of the transaction.

Table I shows an example of a fragment of a transaction database.

Table I
Fragment of the transaction database

Transaction number	Elements
1	Cisterns, Oil Spill
2	Pipeline D, Pipe Break
3	Pipeline B, Pipe Break
4	Pipelines, Pipe Break
5	Cisterns, Pipelines, Pipe Break, Oil Spill
6	Pipeline D, Cisterns, Pipe Break, Oil Spill

The information contained in a transactional database and the knowledge contained in hierarchies of features can be combined and presented as an object-feature table (Table II).

Each row of the object-feature table corresponds to a transaction with the same number. The "1"s in the table mark the transaction elements themselves, as well as those elements that need to be included in the transaction description based on the analysis of hierarchies of feature.

If an element of a certain taxonomy occurs in a transaction, then in the corresponding row of the object-feature table, the "1"s also mark those elements that are higher in the hierarchy than the one under consideration. For example, since elements *e* ("Oil Spill") is included in transaction №1 it automatically includes element *b* ("Emergency") describing a superclass of the concept "Oil Spill".

If an element of a certain partonomy occurs in a transaction, then in the corresponding row of the object-feature table, the "1"s also mark those elements that are lower in the hierarchy than the one under consideration. For example, since element c ("Pipelines") is included in transaction №4 elements g ("Pipeline D") and h ("Pipeline B") which are parts of the element "Pipelines" should also be included in the transaction.

In this case, a pattern will be considered $acceptable$ if any two elements of which are not connected by hierarchical relationships (either incomparable or belong to different hierarchies).

Let's set a value of minimal support θ=2 (i. e. two transactions). It is necessary to find all closed frequent patterns taking into account the given relations of hierarchy of features and a fragment of the transactional database. The following sections considerates solving such problems within the framework of the author's approach.

III. The proposed approach to Data Mining

Recall that the $Constraint\ Satisfaction\ Problem$ (CSP) is to find solutions for a $network\ of\ constraint$. The network of constraints is the following triple [8]–[10]: $\langle X, Dom, C\rangle$, where X is the set of variables, Dom are the domains of variables, Y are the constraints setting the permissible combinations of values of the variables. It is necessary to find such values of all variables that all constraints of network are satisfied.

The presented research uses the so-called table constraints. In addition to typical tables, $table\ constraints$ include $compressed\ tables$, $smart\ tables$, etc. [11]. These types of constraints differ in what is meant by a tuple of relation. For further explanation, only compressed tables will be used. Tuples of compressed tables contain sets as components. Similar structures are described in [12] and are called $finite\ predicate\ matrices$.

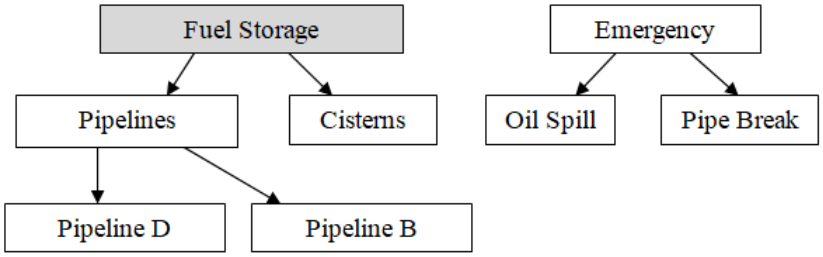

Figure 1. The simplified example of the subject domain ontology

Table II
Object-feature table

№	*a*	*b*	*c*	*d*	*e*	*f*	*g*	*h*
	Fuel Storage	Emergency	Pipelines	Cisterns	Oil Spill	Pipe Break	Pipeline D	Pipeline B
1		1		1	1			
2		1				1	1	
3		1				1		1
4		1	1			1	1	1
5		1	1	1	1	1	1	1
6		1		1	1	1	1	

Within the framework of the approach developed in the work, DM problems are stated and solved as table constraint satisfaction problems. The approach relies on the representation of the training sample in the form of specialized table constraints that allow for compact expression of n-ary relations, as well as on the use of author's procedures for inference on these structures.

As in previous studies [2], [3], *compressed tables of the $D-type$* are used to model the training sample, which contain two attributes in their schema: attribute X, which corresponds to the objects of the training sample, and attribute Y, which describes the features of objects. So for the object-feature table under consideration (Table II) the corresponding compressed table of the D-type will be as follows:

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \begin{array}{|cc|} X & Y \\ \{1,2,3,4,5,6\} & \{b,c,d,e,f,g,h\} \\ \{1,2,3,4,5,6\} & \{c,d,e,f,g,h\} \\ \{4,5\} & \{b,d,e,f,g,h\} \\ \{1,5,6\} & \{b,c,e,f,g,h\} \\ \{1,5,6\} & \{b,c,d,f,g,h\} \\ \{2,3,4,5,6\} & \{b,c,d,e,g,h\} \\ \{2,4,5,6\} & \{b,c,d,e,f,h\} \\ \{3,4,5\} & \{b,c,d,e,f,g\} \end{array} \quad (1)$$

Here, the feature a is excluded from consideration in advance, since it does not occur in any transaction.

Each row of the compressed table can be interpreted as the following implication $(Y = m_k) \to (X \in O_k)$ (if feature m_k is under consideration, the set of objects that possess it is equal to O_k). For example, the following logical expression corresponds to the third row:

$$\begin{aligned} &(X \in \{1,5,6\}) \vee (Y \in \{b,c,e,f,g,h\}) = \\ &= \overline{(Y \in \{b,c,e,f,g,h\})} \to (X \in \{1,5,6\}) = \\ &= (Y \in \{d\}) \to (X \in \{1,5,6\}) = \\ &= (Y = d) \to (X \in \{1,5,6\}) \end{aligned} \quad (2)$$

This expression means the following: "The feature d is possessed by objects from the set {1, 5, 6}".

In order to search for closed frequent patterns within the framework of the proposed approach, the author's methods of inference on table constraints and methods of branching the search tree are used.

The author's methods of inference on tables of the D-type are implemented using rules 1-7:

Statement 1 (**S1**). If at least one row of a compressed table of the D-type is empty (contains all empty components) then the table is empty.

Statement 2 (**S2**). If all the components of an attribute are empty then this attribute can be deleted from the compressed table (all components of the corresponding columns are deleted).

Statement 3 (**S3**). If there is a row in the compressed table that contains single non-empty component then all values in domain that are not included in this component are deleted from the corresponding domain.

Statement 4 (**S4**). If a row of a compressed table of the D-type contains at least one full component, it is deleted.

Statement 5 (**S5**). If a component of a compressed table of the D-type contains a value that does not belong to the corresponding domain then this value is deleted from the component.

Statement 6 (**S6**). The components of a compressed table of the D-type corresponding to a variable X with cardinality below a certain threshold θ are replaced by empty components.

Statement 7 (**S7**). If a cardinality of the domain of a variable X is below defined threshold θ then a solution for the constraint satisfaction problem does not exist.

The statements 1-5 are used for closed pattern discovery. The last two statements are used for pruning infrequent patterns.

Taking into account the hierarchy relations on a set of features, in fact, requires the analysis of another additional type of constraints and, accordingly, the improvement of the mechanism of reasoning on constraints. The following section also discusses the specifics of the implementation of the search tree branching procedure for the case under consideration [2].

IV. The developed method and its illustration

The proposed method consists in the implementation of the following stages:

Stage 1. Representation of the training sample in the form of a special type of table constraints – compressed tables of the D-type, with the exception of those elements that have support below the specified one. Each row of the compressed table can be mapped to some feature m_k.

Stage 2. Formation of a binary tree based on a backtracking depth first search. The essence of this procedure is to select at each step of the search a certain feature m_k and form two subtrees: a) the left one, which serves to discover patterns containing this feature (an arc labeled m_k inputs to the left descendant node); b) the right one, to discover patterns that do not contain this feature the (arc with the label $\setminus m_k$ is aimed at the right descendant node). The feature m_k is selected among those features that have not yet participated in the selection and are included in the current domain of the variable Y. After selecting the descendant node, reduction procedures are used to discard obviously unpromising branches of the search tree, allowing to reduce a compressed table of the D-type characterizing the ancestor node to a table of a smaller dimension, excluding the "redundant" rows, columns, component values, attribute values from the domains of variables X and Y. The feature m_k is selected basing on the following heuristic: in the compressed table of the D-type obtained after applying the rules for reducing table constraints in the previous step, the row with the highest cardinality of the component X is selected. In addition to the rules (**S1**-**S7**) discussed above, a specialized rule is applied – statement 8, which analyzes the hierarchy relations on features.

Statement 8 (**S8**). If it is assumed that some feature m_k must necessarily be included as an element in the desired pattern p, then all elements m_j that are above and below in hierarchy (taxonomy, partonomy) than the one under consideration should be excluded from consideration (from the domain of attribute Y).

Stage 3. Listing patterns based on the analysis of the nodes of the search tree. To list patterns nodes are analyzed, the input arcs of which have a label that does not contain symbol "\". These nodes correspond to interesting patterns one by one. Listing patterns can be carried out during the construction of the search tree, rather than at the end of this procedure.

Unlike the methods "Apriori" and Eclat [4], [13], the proposed method implements a search tree traversal not in width, but in depth. The developed method uses a training sample representation similar to the TID representation in Eclat. As in the method "Close by one" [14], in the author's method, the main component is the logical inference procedure, which makes it possible to calculate its closure for a given set of features, but the proposed method reduces calculations by eliminating duplicate actions. Unlike the FP-Growth algorithm, the considered method makes it easy to integrate additional constraints on the required type of pattern and use them to reduce the search space. This makes it similar to various Apriori modifications.

Now let's return to our end-to-end example. The decision tree in this example built according to heuristic and constraint propagation rules considered above is shown below in Figure 2.

Let's demonstrate how pattern discovery is implemented taking as an example one of the branches of the search tree. So, initial table of the D-type is described by formula (1).

Let's first select feature b («Emergency») since it has maximum support. This means that component X of the first row is selected, but since this component matches the current domain there is no reducing the domain of X. According to **S8**, the features e («Oil Spill») and f («Pipe Break») are excluded from consideration, since they are child nodes of the node b in the taxonomy. We have the following reminder of the initial compressed table of the D-type:

$$\begin{array}{c} \\ 2 \\ 3 \\ 6 \\ 7 \end{array} \begin{array}{c} X \\ \{1,2,3,4,5,6\} \\ \{4,5\} \\ \{1,5,6\} \\ \{2,4,5,6\} \\ \{3,4,5\} \end{array} \begin{array}{c} Y \\ \{\boldsymbol{b},c,d,g,h\} \\ \{b,d,g,h\} \\ \{b,c,g,h\} \\ \{b,c,d,h\} \\ \{b,c,d,g\} \end{array} \quad (3)$$

Here and below, the features that form the desired patterns are highlighted in bold.

In the obtained compressed table of the D-type, the row 1 is eliminated based on statement **S4**, and a "tuning" to a new domain of the variable Y: $\{\boldsymbol{b},c,d,g,h\}$ is performed using **S5** and **S4**, and rows 4 and 5 are excluded from consideration. At this step, the pattern $[\{1,2,3,4,5,6\},\{b\}]$ is obtained.

Now the feature g is selected, and according to **S8** the feature c is excluded from consideration. The current domain of the variable Y becomes equal to the set $\{\boldsymbol{b}, d, \boldsymbol{g}, h\}$, and the domain of the variable X becomes equal to the set $\{2, 4, 5, 6\}$, i.e. to the component X of the sixth row. After "tuning" to new domains (statements **S5** and **S4**) we obtain the remainder:

$$\begin{array}{cc} & \begin{array}{cc} X & Y \end{array} \\ \begin{array}{c} \\ 3 \\ 7 \end{array} \left] \begin{array}{cc} \{2,4,5,6\} & \{\boldsymbol{b}, d, \boldsymbol{g}, h\} \\ \{5,6\} & \{b, g, h\} \\ \{4,5\} & \{b, d, g\} \end{array} \right[\end{array} \quad (4)$$

Here, row 6 is excluded due to reducing the domain of X, and row 2 is excluded due to narrowing the domain of Y. At this search step the pattern $[\{2, 4, 5, 6\}, \{b, g\}]$ has been discovered.

Next we select the feature d. This leads to narrowing the domain of the variable X to a set $\{5, 6\}$. As a result of "tuning" the compressed table to a new variable domain using statements **S5** and **S4**, row 3 is excluded from consideration, and in row 7 the component becomes equal to a single-element set, while the support threshold is two. After applying **S6** and **S3**, the domain of the variable Y is narrowed to a set $\{b, d, g\}$, and row 7 is eliminated according to **S4**.

At this step we obtain pattern $[\{5, 6\}, \{b, d, g\}]$. All rows of the compressed table have been deleted, which indicates that the study of the branch of the search tree has been completed. As a result of traversal of this branch of the search tree, three patterns were discovered.

Let's list all the closed frequent patterns discovered as a result of the application of the developed method:

$$\begin{gathered} [\{1,2,3,4,5,6\}, \{b\}], [\{2,3,4,5,6\}, \{f\}], \\ [\{2,4,5,6\}, \{b,g\}], [\{3,4,5\}, \{b,h\}], \\ [\{1,5,6\}, \{b,d\}], [\{4,5\}, \{b,c\}], \\ [\{2,4,5,6\}, \{f,g\}], [\{3,4,5\}, \{f,h\}], \\ [\{1,5,6\}, \{d,e\}], [\{4,5\}, \{c,f\}], \\ [\{5,6\}, \{b,d,g\}], [\{4,5\}, \{b,g,h\}], \\ [\{4,5\}, \{f,g,h\}], [\{5,6\}, \{d,e,f,g\}] \end{gathered} \quad (5)$$

Conclusion

The constraint programming paradigm is often used to solve complex combinatorial search problems, which include most DM problems. Within the framework of the proposed original approach, DM problems are proposed to be solved as table constraint satisfaction problems. To represent the training sample, it is proposed to use specialized table constraints – compressed tables of the D-type.

The proposed method of closed frequent pattern discovery taking into account subject domain ontology relies on the procedure of constructing a binary search tree, which provides interesting patterns without the preliminary stage of generating candidates for the desired patterns. Within the framework of the designed approach, it is quite easy to take into account additional requirements for type of discovered patterns. For each type of constraints on the type of pattern, appropriate rules of reduction of the search space are developed. In the presented study, an additional constraint is the requirement for a hierarchical ordering of features.

Constraints of hierarchy of feature are processed by specialized procedures for search space reduction. The application of the method makes it possible at each step of the search to reduce the existing problem to a problem of significantly smaller dimension, which eventually reduces the immediacy of the exponential catastrophe problem. In comparison with analogs using logical inference, the method allows to exclude repetitions of actions when calculating closures on sets of features.

Acknowledgment

The work was carried out within the framework of the current research topic «Methods and information technologies for monitoring and management of regional critical infrastructures in the Arctic zone of the Russian Federation» (registration number FMEZ-2025-0054).

References

[1] A.A. Zuenko, O.N. Zuenko, "Finding dependencies in data based on methods of satisfying table constraints." Ontology of designing. 2023, 13(3), pp. 392-404.

[2] A.A. Zuenko, "Applying methods to satisfy table constraints for modeling reasoning of the JSM-type," Proceedings of Voronezh State University. Series: Systems Analysis and Information Technologies, vol. 4, 2024, pp. 116–128.

[3] Y. Traore et al., "Discovering frequent patterns guided by an ontology." Revue Africaine de la Recherche en Informatique et Mathématiques Appliquées. 2016, vol. 25, pp. 75-92.

[4] R. Agrawal, T. Imielinski, A. Swami, "Mining association rules between sets of items in large databases," Proceedings of the ACM SIGMOD Conf. on Management of Data. Washington, 1993. pp. 207–216.

[5] A. Hien et al., "A relaxation-based approach for mining diverse closed patterns." Hutter, F., Kersting, K., Lijffijt, J., Valera, I. eds. ECML PKDD 2020. LNCS (LNAI), vol. 12457, Springer, Cham, 2021, pp. 36–54.

[6] W. Song, W. Ye, P. Fournier-Viger, "Mining sequential patterns with flexible constraints from MOOC data." Applied Intelligence, 2022, vol. 52, pp. 16458–16474.

[7] X. Liu, X. Niu, P. Fournier-Viger, "Fast Top-K association rule mining using rule generation property pruning." Applied Intelligence, 2021, vol. 51, pp. 2077–2093.

[8] S. Russel, P. Norvig, "Artificial Intelligence: A Modern Approach." 3rd ed, Prentice Hall, 2010, 1132 p.

[9] H. Verhaeghe et al., "Learning optimal decision trees using constraint programming." Constraints, vol. 12, 2020, pp. 226–250.

[10] H. Simonis ed., "CP 2020. Proceedings of LNCS, vol. 12333. Springer, Cham, 2020.

[11] J.-M. Mairy, Y. Deville, C. Lecoutre, "The smart table constraint." L. Michel ed. CPAIOR 2015. LNCS, vol. 9075, pp. 271–287. Springer, Cham, 2015.

[12] A. Zakrevskij, "Integrated Model of Inductive-Deductive Inference Based on Finite Predicates and Implicative Regularities." Diagnostic Test Approaches to Machine Learning and Commonsense Reasoning Systems, IGI Global, 2013, pp.1-12.

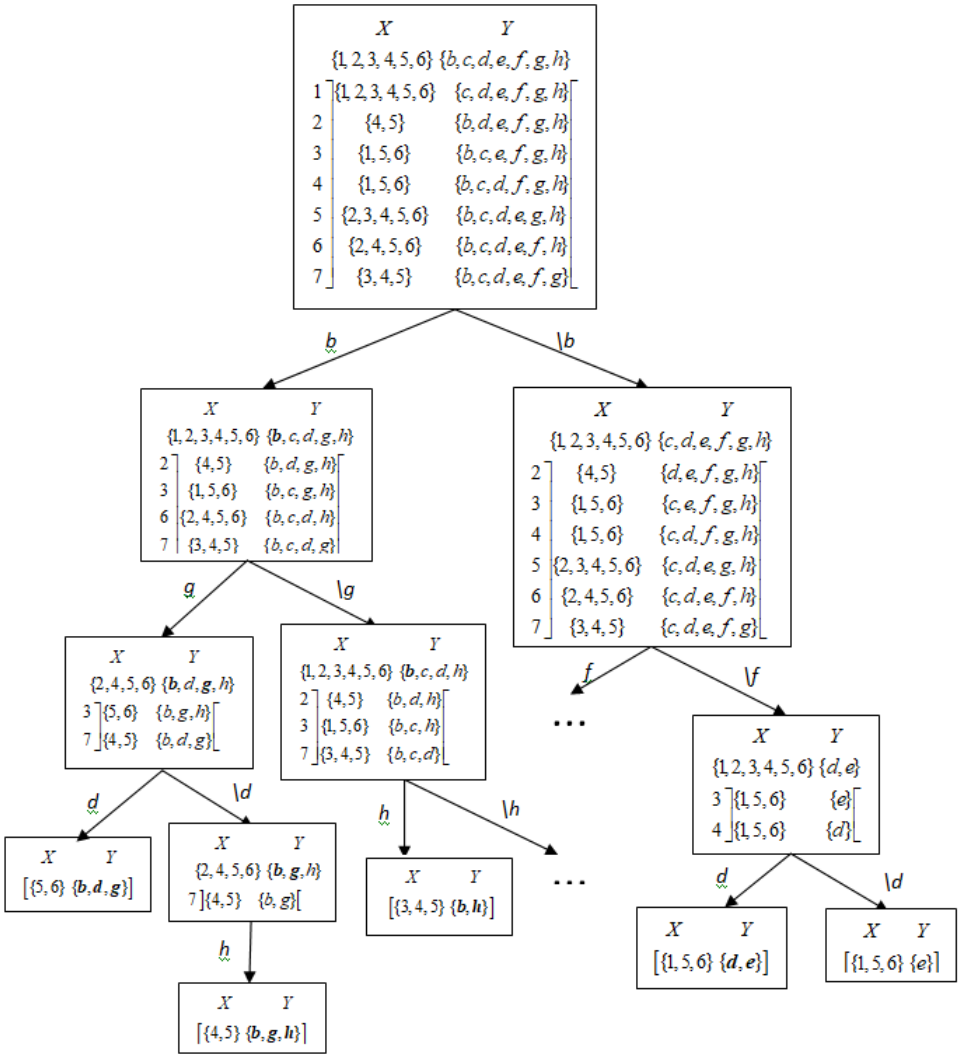

Figure 2. A decision tree based on the proposed method

[13] R. Agrawal, R. Srikant, "Fast algorithms for mining association rules in large databases." Proceedings of the 20th International Conference on Very Large Data Bases. VLDB, Santiago, Chile, 1994, pp. 487-499.

[14] S.O. Kuznetsov, S.A. Obiedkov, "Comparing performance of algorithms for generating concept lattices." Journal of Experimental and Theoretical Artificial Intelligence, 2002, pp. 1-28.

МЕТОД УДОВЛЕТВОРЕНИЯ ОГРАНИЧЕНИЙ ДЛЯ ВЫЯВЛЕНИЯ ПАТТЕРНОВ В ДАННЫХ С УЧЕТОМ ИЕРАРХИЙ ПРИЗНАКОВ

Зуенко А. А., Зуенко О. Н.

В статье рассматривается, каким образом анализ онтологии предметной области способен помочь в выявлении интересных и хорошо интерпретируемых паттернов в данных. Для этой области исследований в литературе используется термин "Поиск паттернов, управляемый онтологиями", а онтологии рассматриваются в качестве средства семантической обрезки пространства поиска. Анализ онтологий при поиске паттернов позволяет существенно сократить перебор вариантов за счет редукции пространства поиска, а также рассматривать элементы паттернов на различных уровнях абстракции. Предложенный подход к интеллектуальному анализу данных основывается на компактном представлении обучающей выборки с помощью специализированных матрицеподобных структур и применении оригинальных процедур вывода на данных структурах. Исследования лежат на стыке таких направлений искусственного интеллекта как интеллектуальный анализ данных и семантические технологии представления и обработки информации.

Received 21.03.2025