A Neural Network-Like Combinatorial Data Structure

Xenia A. Naidenova

Military Medical Academy, Saint-Petersburg, Russian Federation

ksennaidd@gmail.com

Abstract. A generalization of algorithm is proposed for implementing the well-known effective inductive method of constructing sets of cardinality (*q*+1) ((*q*+1)-sets) from their subsets of cardinality *q* ((*q*)-sets). A new neural network-like combinatorial data-knowledge structure supporting this algorithm is advanced. This structure can drastically increase the efficiency of inferring functional and implicative dependencies as like as association rules from a given dataset.

Keywords: Level-wise algorithm, Inferring dependencies from a dataset, Neural network-like data structure, Knowledge representation.

1. Introduction

Mining association rules from databases has attracted great interest because of its potentially very useful application. Mining association rules was firstly introduced by Agrawal et al. in [1], where an algorithm, called AIS (for Agrawal, Imielinski, and Swami), has been proposed. Another algorithm, called SETM (for Set Oriented Mining), has been introduced in [2]. This algorithm uses relational operations in a relational database environment.

The next step in solving the problem of inferring association rules has been done in [3-4], where the algorithms Apriori, AprioriTid, and AprioriHybrid have been presented. These algorithms use an effective inductive method of constructing sets of cardinality (q+1) ((q+1)-sets) from their subsets of cardinality q ((q)-sets). A (q+1)-set can be constructed if and only if there exist all its proper (q)-subsets. The same principle underlies the algorithm Titanic for generating key patterns [5] and the algorithm TANE for discovering functional and approximate dependencies [6]. The level-wise inductive method of (q+1)-sets' construction has also been proposed for inferring good diagnostic (classification) tests for a given classification or class of objects [7-9]. These tests serve as a basis for extracting functional dependences, implications, and association rules from a given dataset.

In all enumerated problems, the same algorithm deals with different sets of elements (items, attributes, values of attributes, transactions, indices of itemsets) and checks the different properties of generated subsets. These properties are "to be a frequent (large) itemset", "to be a key pattern", "to be a test for a given set of examples", "to be an irredundant set of attribute values", "to be a good test for a given set of examples”, and some others. If an obtained subset does not possess a required property, then it is deleted from consideration. This deletion reduces drastically the number of subsets to be built at all greater levels. In section 2, we introduce a Background algorithm solving the task of inferring all maximal subsets of set *S* (i.e., such subsets that cannot be extended) possessing a given PROPERTY. The set *S* can be interpreted depending on the context of a considered problem. This algorithm implements the level-wise inductive method of (q+1)-sets’ construction. In section 3, we consider some possible ways of increasing the efficiency of Background Algorithm. Finally, in Section 4, we propose a neural network-like combinatorial data structure for constructing (q+1)-sets from their q-subsets.

2. Background algorithm

The Background Algorithm. By $s_q = (i_1, i_2, \ldots, i_q)$, we denote a subset of *S*, containing *q* element of *S*. Let *S*(test-*q*) be the set of subsets $s = \{i_1, i_2, ..., i_q\}$, $q = 1, 2, ..., nt$, satisfying the PROPERTY. Here *nt* denotes the cardinality of *S*. We use an inductive rule for constructing $\{i_1, i_2, ..., i_{q+1}\}$ from $\{i_1, i_2, ..., i_q\}$, $q = 1, 2, ..., nt$-1. This rule relies on the following consideration: if the set $\{i_1, i_2, ..., i_{q+1}\}$ possesses the PROPERTY, then all its proper subsets must possess this PROPERTY too and, consequently, they must be in *S*(test-*q*). Thus the set $\{i_1, i_2, ..., i_{q+1}\}$ can be constructed if and only if *S*(test-*q*) contains all its proper subsets. Having constructed the set $s_{q+1} = \{i_1, i_2, ..., i_{q+1}\}$, we have to determine whether it possesses the PROPERTY or not. If not, s_{q+1} is deleted, otherwise s_{q+1} is inserted in *S*(test-(q+1)). The algorithm is over when it is impossible to construct any element for *S*(test-(q+1)).

Background Algorithm. Inferring all maximal (not extended) subsets of *S* possessing a given PROPERTY.

```
1. Input: $q = 1$, $S = \{1,2,\ldots, nt\}$, $S$(test-$q$) = $\{\{1\}, \{2\}, ..., \{nt\}\}$.

Output: the set $SMAX$ of all maximal subsets of $S$ possessing the PROPERTY.

2. $S_q$ := $S$(test-$q$);

3. While $||S_q|| \geq q + 1$ do

3.1 Generating $S\ (q + 1) = \{s =\{i_1, ..., i_{(q+1)}\}: (\forall\ j)\ (1 \leq j \leq q + 1)\ (i_1, ..., i_{(j-1)}, i_{(j+1)}, ..., i_{(q+1)}\} \in S_q\}$;

3.2 Generating $S$(test-$(q + 1)$) = $\{s = \{i_1, ..., i_{(q+1)}\}: (s \in S(q + 1))$ & (PROPERTY($s$)) = $true$)};

3.3 $S$(test-$q$) := $\{s = \{i_1, ..., i_q\}: (s \in S$(test-$q$)) & $((\forall\ s')(s' \in S$(test-$(q + 1)$) $s \not\subset s')\}$;
3.4. $q := q + 1$;
3.5. $max := q$;
end while
4. $TGOOD := \varnothing$;
5. While $q \le max$ do $SMAX := SMAX \cup \{s: s = \{i_1, ..., i_s\} \in S(\text{test-}q)\}$;
5.1 $q := q + 1$;
end while
end
```

The most important factor influencing on computational complexity of Background Algorithm is the method of inductive generating of q-sets in the level-wise manner. Generally, we use the following inductive rules, where SN is the family of sets S_q of cardinality equal to q, $q = 1, \ldots, nt$, $S_q \subseteq S = \{1, \ldots, nt\}$ and $C_S(q)$ denotes the number of combinations of S on q.

(1) $q = 1$, $q + 1 = 2$;
$s_q = \{i\}$, $s_{(q+1)} = \{i, j\}$, $(\forall j)$ $(i \neq j, \{j\} \in SN$;

(2) $q = 2$, $q + 1 = 3$;
$s_q = \{i, j\}$, $s_{(q+1)} = \{i, j, l\}$, where l different from i, j and such that there are in SN
a) two sets $s_1 = \{i, l\}$, $s_2 = \{j, l\}$ or
b) $s = \{l\}$;

(3) $q = 3$, $q + 1 = 4$;
$s_q = \{i, j, m\}$, $s_{(q+1)} = \{i, j, m, l\}$, where l different from i, j, m and such that there are in SN
a) three sets $s_1 = \{i, j, l\}$, $s_2 = \{i, m, l\}$, $s_3 = \{j, m, l\}$ or
b) three sets $s_1 = \{i, l\}$, $s_2 = \{j, l\}$, $s_3 = \{m, l\}$ or
c) $s = \{l\}$;
……
(q) q, $q + 1$;
$s_q = \{i_1, i_2, ..., i_q\}$, $s_{(q+1)} = \{i_1, i_2, ..., i_q, l\}$, where l different from $i_1, i_2, ..., i_q$ and such there are in SN
a) sets the number of which is equal to $C_S(q) = C_S(nt - q)$ and the cardinality of which is equal to q, such that $\{i_1, i_2, ..., i_{p-1}, i_{p+1}, ..., i_q, l\}\backslash\{i_p\}$ for all $p = 1, ..., q$ or
b) sets the number of which is equal to $C_S(q - 1) = C_S(nt - (q-1))$, the cardinality of which is equal to q-1, such that $\{i_1, i_2, ..., i_q, l\}\backslash\{i_{pi}, i_{pj}\}$ for all $\{p_i, p_j\} \subseteq \{1, ..., q\}$ or

c) sets the number of which is equal to $C_S(q - 2) = C_S(nt – (q-2))$, the cardinality of which is equal to q-2 , such that $\{i_1, i_2,...., i_q, l\}\backslash\{i_{pi}, i_{pj}, i_{pk}\}$ for all $\{p_i, p_j, p_k\}$, $\{p_i, p_j, p_k\} \subseteq \{1,2,..., q\}$ or

……

d) sets the number of which is equal to $C_S(1) = C_S(nt - 1)$, the cardinality of which is equal to 1, such that$\{l\}$, $l \notin \{i_1, i_2,..., i_q\}$.

The Background Algorithm has an essential disadvantage consisting in the necessity to generate all subsets of s in S_q, q = 1, 2,…, *qmax*. But it is possible constructing directly an element $s \in S_q$, $s = \{i_1, i_2,…, i_q\}$ without generating all of its subsets.

3. A structure of interconnected lists for implementing Background Algorithm

The inductive rules can be used not only for extending sets, but also for cutting off both the elements of S and the sets themselves containing these deleted elements. If element j enters in s_{q+1}, then it must enter in q proper subsets of s_{q+1}. If we observe that j enters in only one doublet (pair), then it cannot enter in any triplet. If we observe that j enters in only one triplet, then it cannot enter in any quadruplet and so on. If an element enters in two and only two doublets, it means that it can enter only in one triplet. If an element enters in three and only three doublets, it can enter in only one quadruplet.

These inductive reasoning are applicable to constructing triplets from doublets, quadruplets from triplets and so on. For instance, if a doublet enters in two and only two triplets, then it can enters in one quadruplet. If a triplet enters in two and only two quadruplets, then it can enter in only one set of five elements. The removal of a certain element (or set of elements) from the examination draws the removal of doublets, triplets, quadruplets,…) into which it enters.

Let us name the procedure for removal of elements and sets containing these elements the procedure of “winnowing”. It is convenient to realize this procedure with the use of a Matrix of Correspondences the columns of which are associated with elements of S, and the rows are associated with subsets of S. An entrance $\{i, j\}$ in this matrix equals 1, if index associated with column j enters in s associated with row i.

Consider the following example. The set $S = \{\{1\}, \{2\}, ..., \{14\}\}$.

Consider the Matrix of Correspondences (Table 1) between the 2-component subsets of S possessing a given PROPERTY (S(test-2)) and elements of S appearing in these subsets. In this matrix, the columns are ordered by increasing the number of subsets associated with the columns.

Element 9 enters in one and only one doublet, hence (9,11) cannot be included in any triplet. We can delete the corresponding column and row. We conclude also that set (9,11) cannot enter in any triplet.

Element 5 enters in two and only two doublets, hence it is included in only one triplet (1,5,12). Element 5 cannot be included in any quadruplet. We can delete the corresponding column and rows 2, 3.

Element 6 enters in three and only three doublets, hence it is included in only one quadruplet (4,6,8,11). Element 6 cannot be included in a subset of five indices. We can delete the corresponding column and rows 4, 5, 6.

By analogous reason, we conclude that collection (1,2,12,14) cannot be extended and we can delete the corresponding column and rows 7, 8, 9.

Note that all subsets (9,11), (1,5,12, (4,6,8,11), and (1,2,12,14) possess the PROPERTY.

Element 10 enters in three and only three doublets, hence it is included in only one quadruplet (2,3,8,10). This set does not possess the PROPERTY. In this case, we have to construct all the triplets with element 10. These triplets (2 8 10), (2 3 10), (3 8 10) do not possess the PROPERTY, it means that subsets (2,10), (3,10), (8,10) are maximal ones possessing the PROPERTY. Element 10 can be deleted together with rows 10, 11, 12. Currently, we have generated the following 7 subsets: (1,5,12), (4,6,8,11), (1,2,12,14), (2,3,8,10), (2,3,10), (2,8,10), (3,8,10). Table 2 shows the reduced Matrix of Correspondences.

Table 1. The Matrix of Correspondences for the S(test-2)

Subset	9	5	6	14	10	1	11	8	12	3	7	4	2
(9,11)	1						1						
(1,5)		1				1							
(5,12)		1							1				
(4,6)			1									1	
(6,8)			1					1					
(6,11)			1				1						
(1,14)				1		1							
(2,14)				1									1
(12,14)				1					1				
(2,10)					1								1
(3,10)					1					1			
(8,10)					1			1					
(1,2)						1							1
(1,4)						1						1	
(1,7)						1					1		
(1,12)						1			1				
(3,11)							1			1			
(4,11)							1					1	
(7,11)							1				1		
(8,11)							1	1					
(2,8)								1					1
(3,8)								1		1			
(4,8)								1				1	
(7,8)								1			1		
(2,12)									1				1
(3,12)									1	1			

(4,12)									1			1	
(7,12)									1		1		
(2,3)										1			1
(3,4)										1		1	
(3,7)										1	1		
(2,7)											1		1
(4,7)											1	1	
(2,4)												1	1

Table 2. The reduced Matrix of Correspondences (Reduction 1)

Subset	1	11	8	12	3	7	4	2
(1,2)	1							1
(1,4)	1						1	
(1,7)	1					1		
(1,12)	1			1				
(3,11)		1			1			
(4,11)		1					1	
(7,11)		1				1		
(8,11)		1	1					
(2,8)			1					1
(3,8)			1		1			
(4,8)			1				1	
(7,8)			1			1		
(2,12)				1				1
(3,12)				1	1			
(4,12)				1			1	
(7,12)				1		1		
(2,3)					1			1
(3,4)					1		1	
(3,7)					1	1		
(2,7)						1		1
(4,7)						1	1	
(2,4)							1	1

Element 1 enters in 4 doublets. In this case, we construct the following triplets including element 1: (1,2,4), (1,2,7), (1,2,12), (1,4,7), (1,4,12), (1,7,12). Only two triplets possess the PROPERTY: (1,4,7) and (1, 2, 12). We conclude that element 1 cannot be included in any quadruplet possessing the PROPERTY; hence it can be deleted from consideration with rows 13, 14, 15, 16. Since (1, 2, 12) $\subseteq$ (1, 2, 12, 14)), we conclude that subset (1,4,7) corresponds to maximal subset possessing the PROPERTY but subset (1, 2, 12) is not maximal with respect to the PROPERTY.

Analogously, the consideration of **element 11** leads to constructing the following subsets: (3,4,11), (3,7,11), (3,8,11), (4,7,11), (4,8,11), (7,8,11) from which only

(7,8,11) and (4,8,11) possess the PROPERTY. We conclude that element 11 cannot be included in any quadruplet possessing the PROPERTY; hence it can be deleted from consideration with rows 17, 18, 19, 20. We also conclude that subset (7,8,11) is maximal with respect to the PROPERTY, but (4, 8, 11) does not.

Currently, we have constructed 7 + 12 = 19 subsets. Table 3 shows the reduced Matrix of Correspondences.

With **element 8**, the following subsets can be constructed: (2,3,8), (2,4,8), (2,7,8), (3,4,8), (3,7,8), (4,7,8). But only (2,7,8) possesses the PROPERTY. We conclude that it is maximal with respect to the PROPERTY. Element 8 can be deleted together with rows 21, 22, 23, 24.

For **element 12**, the following triplets can be constructed: (2,3,12), (2,4,12), (2,7,12), (3,4,12), (3,7,12), (4,7,12). Only (3,7,12), (4,7,12) possess the PROPERTY. Since element 12 cannot be included in any quadruplet possessing the PROPERTY, we conclude that (3,7,12), (4,7,12) are maximal with respect to the PROPERTY. Element 12 can be deleted together with rows 25, 26, 27, 28.

Currently, we have constructed 19 + 12 = 31 subsets. Table 4 shows the reduced Correspondent Matrix. In this table, (2,3,4,7), the union of all remaining subsets, possesses the PROPERTY, hence the process of generating subsets is over.

Table 3. The reduced Matrix of Correspondences (Reduction 2)

Subset	8	12	3	7	4	2
(2,8)	1					1
(3,8)	1		1			
(4,8)	1				1	
(7,8)	1			1		
(2,12)		1				1
(3,12)		1	1			
(4,12)		1			1	
(7,12)		1		1		
(2,3)			1			1
(3,4)			1		1	
(3,7)			1	1		
(2,7)				1		1
(4,7)				1	1	
(2,4)					1	1

Table 4. The reduced Matrix of Correspondences (Reduction 3)

Subset	3	7	4	2
(2,3)	1			1
(3,4)	1		1	
(3,7)	1	1		
(2,7)		1		1
(4,7)		1	1	

(2,4)			1	1

Currently, we have constructed 31 + 1 = 32 subsets. Without the procedure of winnowing, it is necessary in Background Algorithm to form 91 + 38 + 3 = 91 +41 = 132 subsets, where 91 doublets, 38 triplets, and 3 quadruplets. The application of winnowing reduced the total quantity of considered subsets to 123: 91 + 32 = 123.

4. A special combinatorial network for implementing the Background Algorithm

The idea of the following algorithm is based on the functioning of a combinatory network structure, whose elements correspond to subsets of a finite set *S* generated in the algorithm. These elements are located in the network along the layers, so that each *q* - layer consists of the elements corresponding to subsets the cardinality of which is equal to *q*. All the elements of *q*–layer have the same number *q* of inputs or connections with the elements of previous (*q* – 1)–level. Each element "is excited" only if all the elements of previous layer connected with it are active. The weight of connection going from the excited element is taken as equal to 1; the weight of connection going from the unexcited element is taken as equal to 0. An element of *q*–layer is activated if and only if the sum of weights of its inputs is equal to *q*. The possible number N_q of elements (nodes) at each layer is known in advance as the number of combinations of *S* on *q*. In the process of the functioning of the network the number of its nodes can only diminish.

An advantage of this network consists in the fact that its functioning does not require the complex techniques for changing the weights of connections and it is not necessary to organize the process of constructing *q* -sets from their (*q* -1)-subsets. The nodes of network can be interpreted depending on a problem to be solved. The assigned properties can be checked via different attached procedures.

If an activated node does not possess the assigned property, then it is excluded from the network by setting to 0 all connections going from it to the nodes of above layer. Non-activated node does not require checking whether it possesses the PROPERTY or not. The work of this combinatorial network consists of the following steps:

Step 1. The setting of the first layer nodes of network to active state, the weights of connections leading to the second layer nodes are set equal to 1;

For each level beginning with the second one:

Step 2. The excitation of nodes, if they were not active and all their incoming traffic (links) have the weight equal to 1; checking the assigned property for the activated nodes of this layer;

Step 3. If the assigned property of node is not satisfied, then all the outgoing connections of this node are established to 0. If the assigned property of node is satisfied, then its outgoing connections are set to be equal to 1;

Step 4. The propagation of "excitation" to the nodes of the following higher layer (with respect to the current one) and the passage to analyzing the following layer;

Step 5. “The readout” of the active nodes not connected with above lying active nodes. Such nodes correspond to maximal (not extended) subsets possessing a given property.

In Fig. 1, all the nodes of two first levels are activated but nodes {4,10}, {7,10}, {1,8}, and {1,10} do not possess the given property and they have no active outgoing links. At the third level, only two nodes are activated among which node {4,7,8} does not possess the given property. As a result, we have two nodes corresponding to maximal subsets possessing the given property: {8,10}, {1,4,7}. In the process of network activating, only 12 nodes have been checked and 14 ones did not require to be checked.

Apparently, we can see that the size of network may be a problem if the data is large. But the decomposition of the main problem into sub-problems drastically diminishes the memory size of Background Algorithm. A subproblem is determined by a subnetwork generated by a node of the network.

Generally, the main advantages of combinatorial network are the following ones:

1. The size of network is computed in advance;
2. It is possible to decompose network into autonomic fragments;
3. Different fragments of network can be joined via common nodes;
4. The states of nodes can be established by the use of attached procedures.

This combinatorial network can be used for solving many problems of data mining such that finding frequent patterns, association rule mining, discovering functional dependencies and some others. The application of neural network models for these problems is a new field for investigating. We can refer the readers only to one work in this direction related to optical neural network model used for mining frequent itemsets in large databases [10]. The optical neural network model proposes the most optimized approach with only one database scan and parallel computation of frequent patterns.

5. Conclusion

In this paper, we describe an algorithm, called Background Algorithm, based on the method of mathematical induction. This algorithm is applicable to inferring many kinds of dependencies from a given dataset, for example, functional, implicative dependencies, and association rules. We discussed also the possible ways of increasing the efficiency of the Background Algorithm. For implementation of this algorithm, we proposed a neural network-like combinatorial structure of data an advantage of which consists in the fact that its functioning does not require the complex techniques for changing the weights of connections. The nodes of network can be interpreted depending on a problem to be solved. The assigned properties of nodes can be checked via different attached procedures.

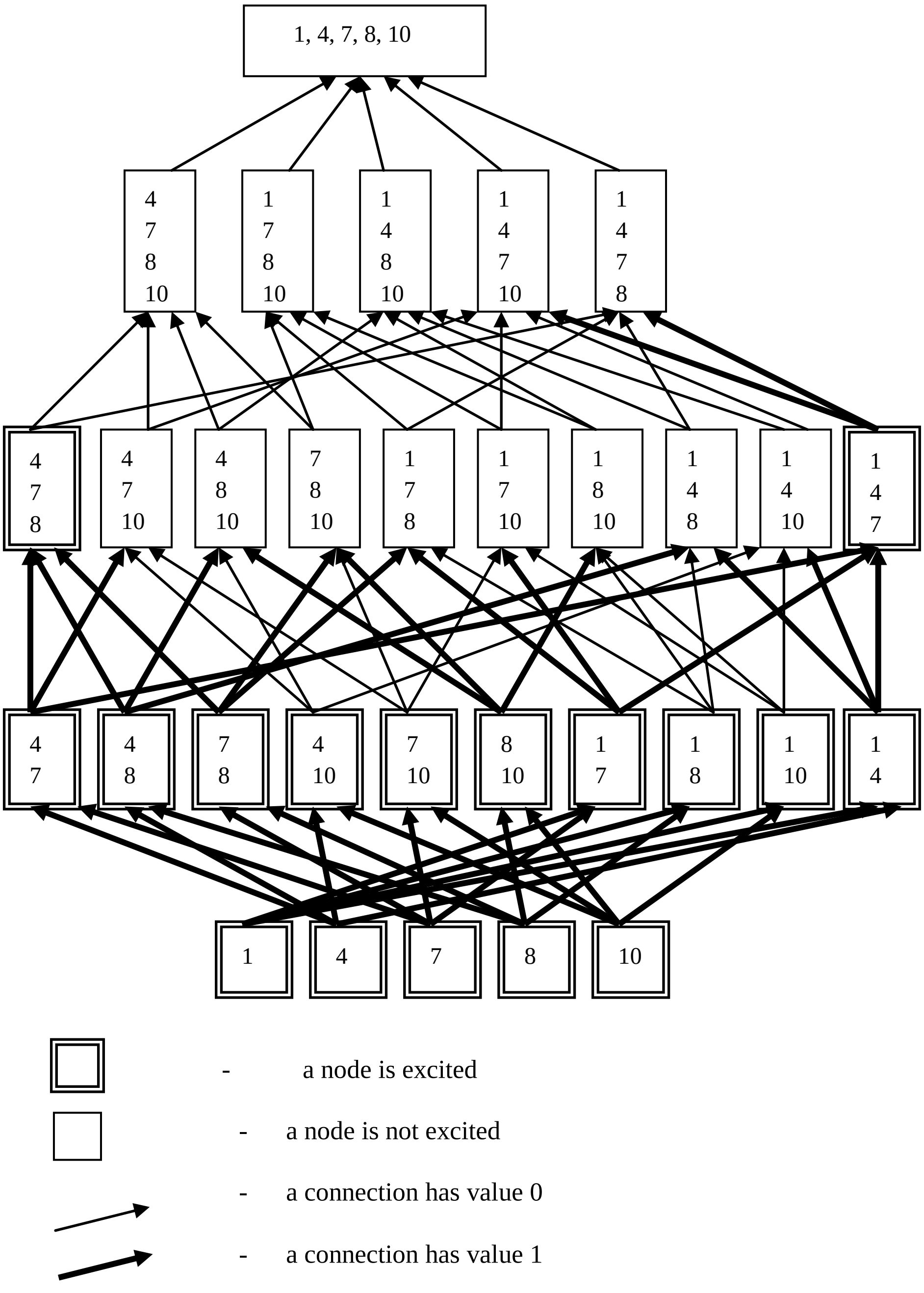

Fig. 1. An example of special combinatorial network

References

1. Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases. In: P. Buneman, S. Jajodia (eds.), Proceedings of the ACM SIGMOD Conference on Management of Data, pp. 207-216. ACM Press (1993).
2. Houtsma, M., Swami, A.: Set-oriented mining of association rules. Research Report RJ 9567 IBMAlmaden Research Center, San Jose, California (1993).
3. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: J. B. Bocca, M. Jarco, C. Zaniolo (eds.), Proceeding of the 20th VLDB Conference, pp. 487-489. Morgan Kaufman (1994).
4. Agraval, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A. I.: Fast discovery of association rules. In: Fayyard, U. M., Piatetsky-Shapiro, G., Smyth, P., Utchurusamy, R. (eds.), Advances in Knowledge Discovery and Data Mining, pp. 307-328. Menlo Park, CA: AAAI Press (1996).
5. Stumme G.: Efficient Data Mining Based on Formal Concept Analysis. In: R. Cicchetti at al. (eds.), DEXA 2002, LNCS 2453, pp. 534-546. Springer-Verlag, Berlin, Heidelberg (2002).
6. Huhtala, Y., Karkkainen, J., Porkka P., and Toivonen, H.: Tane: an efficient algorithm for discovering functional and approximate dependencies. The computer Journal, 42(2), pp. 100-111 (1999).
7. Naidenova, X.A.: Machine learning as a diagnostic task. In: Arefiev, I. (ed.), “Knowledge-Dialog-Solution”, Materials of the Short-Term Scientific Seminar, pp. 26-36. Saint-Petersburg, Russia: State North-West Technical University (1992).
8. Megretskaya, I. A.: Construction of natural classification tests for knowledge base generation. In: Y. Pecherskij (ed.), The Problem of the expert system application in the national economy: Reports of the Republican Workshop (pp. 89-93). Kishinev, Moldava: Mathematical Institute with Computer Centre of Moldova Academy of Sciences (1988).
9. Naidenova, X.A.: DIAGARA: An incremental algorithm for inferring implicative rules from examples. International Journal “Information Theories & Applications, 12(2), 171-186 (2005).
10. Bhatnagar, D. Adlakha, N., Swaroopsaxena, A. A.: Distributed approach for mining frequent itemsets using optical neural network model. International Journal of Engineering Science and Technology, 3(5), pp. 3979-3981 (2011).