CONSTRAINED OBJECT-CHARACTERIZATION TABLES AND ALGORITHMS[1]

Hasmik Sahakyan

Abstract: *Let C be a collection of objects, characterized by the set $A = \{a_1, \cdots, a_n\}$ of binary attributes. We consider problems of the following type: given an object-characterization table, it is to check if there exists a subset M in C of a given size, such that each attribute of A is satisfied by a given number of objects in M. Additional restriction may be applied such as - the number of matches of each object in M is limited. In this paper we investigate particular cases of the general problem, and consider approximation solutions by means of binary classification trees.*

Keywords: *Classification tree, covering, greedy algorithm.*

ACM Classification Keywords: *F.2.2 Nonnumerical Algorithms and Problems*

1. Introduction

Let C be a collection of objects (repetitions is allowed), where every object is given by the same set $A = \{a_1, \cdots, a_n\}$ of binary attributes. An object may satisfy the attribute a_i or not. Consider the following problem: is there an m-subset $M \subseteq C$, such that each attribute a_i is satisfied at least by one object of M for $i = 1, \cdots, n$? In practical level our interest is in representing the diversity of attribute values by the narrow subsets of objects. Machine learning techniques provide algorithmic means of the problem. Mathematically, the problem is related to transversals and sets of different representatives. As formal description related to our problem we will consider the MINIMUM COVER combinatorial problem which is one of the well known NP-complete algorithmic problems [Garey, Johnson, 1979].

MINIMUM COVER (MC)

> Given a finite set S, a collection C of subsets of S, and a positive integer $K \leq |C|$. Does there exist a cover $C' \subseteq C$ of S such that $|C'| \leq K$, i.e. does there exist a subset $C' \subseteq C$ such that $|C'| \leq K$ and every element of S is in at least one subset of C'.

Additional requirements that are common for application problems restrict the domain of possible solutions and create sub-problems, which can remain NP-complete or can have polynomial solutions. Restrictions can be applied on the number of occurrences of elements of S in subsets of C. For example, the number of occurrences of elements of S in subsets of C can be bounded from above by a constant t. However this particular sub-problem of MC is also NP-complete for $t > 1$ [Garey,Johnson, 1979]. The problem is not easier when the number of occurrence of i-th element of S is bounded by t_i, $i = 1, \cdots, |S|$.

We consider different sub-problems when restrictions are applied on the number of occurrence of the elements of S in C'. Assume that the number of occurrences of elements of S in subsets of C' is bounded from below: the i-th

[1] Partially supported by grants № SCS 13RF-088, and № 3-1B340 of State Committee of Science of Ministry of education and science of Republic of Armenia

element of S have to occur in at least s_i elements of C'. In this way the i-th element of S is covered at least s_i times.

MINIMUM $(s_1, \cdots, s_n)$-COVER

> Given a finite set S, a collection C of subsets of S, and a positive integer $K \leq |C|$. Does there exist an $(s_1, \cdots, s_n)$- cover $C' \subseteq C$ of S such that $|C'| \leq K$, i.e. does there exist a subset $C' \subseteq C$ such that $|C'| \leq K$ and the i-th element of S belongs to at least s_i subsets of collection C'?

MINIMUM $(s_1, \cdots, s_n)$-COVER is NP-complete, because of MINIMUM COVER with $s_i = 1$ is its sub-problem.

In this paper together with the decision versions of formulated problems we investigate also their *existence versions*.

$(s_1, \cdots, s_n)$-EXISTENCE

> Given a finite set S and a positive integer m. Does there exist a collection C of subsets of S such that $|C| \leq m$ and the i-th element of S belongs to at least s_i elements/subsets of C?

Further we apply more additional restrictions:

- On the number of repetitions in C;
- On the collection C of subsets of S.

The paper is organized as follows: in Section 2 below we present the problems in terms of binary matrices, and consider complexity issues. Section 3 describes hierarchical classification approach with constraints. We investigate counterparts of $(s_1, \cdots, s_n)$-existence in hierarchical classification area and describe approximation algorithms with help of binary classification trees.

2. Problems given in terms of binary matrices

Decision Problems

Let C be a collection of objects given by the value vectors of a set of binary attributes, $\{a_1, \cdots, a_n\}$. We identify each object of C with a binary sequence of size n in the following way: i-th component of the sequence is 1 if and only if the object satisfies the i-th attribute. In this manner we receive a binary matrix of size $|C| \times n$, where rows correspond to the objects of C, columns correspond to the attributes. If t_i is the number of objects satisfying attribute $a_i, i = 1, \cdots, n$, then the i-th column of matrix contains t_i ones. Now we present covering problems in terms of binary matrices and investigate complexity issues of these problems.

MINIMUM $(s_1, \cdots, s_n)$-COVER ($(s_1, \cdots, s_n)$-MC)

> Given a binary matrix M of size $m \times n$, a positive integer $m' \leq m$, and an integer sequence $(s_1, \cdots, s_n)$, where $0 \leq s_i \leq m'$ for $i = 1, \cdots, n$. Does M contain a submatrix M' of size $m' \times n$ such that the i-th column of M' contains at least s_i ones.

Whit an additional constraint applied on the number of repetitions of rows, the problem is formulated as follows:

MINIMUM $(s_1, \cdots, s_n)$-COVER WITH LIMITED REPETITIONS ($(s_1, \cdots, s_n)$-MC-LR)

> Given a binary matrix M of size $m \times n$, positive integers $m' \leq m$ and $r \geq 1$, and an integer sequence $(s_1, \cdots, s_n)$, where $0 \leq s_i \leq m'$ for $i = 1, \cdots, n$. Does M contain a submatrix M' of size $m' \times n$ with at most r repetitions of each row such that the i-th column of M' contains at least s_i ones.

MINIMUM $(s_1, \cdots, s_n)$-COVER WITH NO REPETITIONS ($(s_1, \cdots, s_n)$-MC-NR)

Given a binary matrix M of size $m \times n$, a positive integer $m' \leq m$, and an integer sequence $(s_1, \cdots, s_n)$, where $0 \leq s_i \leq m'$ for $i = 1, \cdots, n$. Does M contain a submatrix M' of size $m' \times n$ with different rows such that the i-th column of M' contains at least s_i ones.

In $(s_1, \cdots, s_n)$-MC-NR we can assume that M consists of different rows, and thus $m \leq 2^n$, and every column of M contains at most 2^{n-1} ones.

Theorem 1. $(s_1, \cdots, s_n)$-MC-NR is NP-complete.

Proof. We prove that $(s_1, \cdots, s_n)$-MC $\propto$ $(s_1, \cdots, s_n)$-MC-NR.

Consider an instance I_1 of $(s_1, \cdots, s_n)$-MC.

Instance I_1: matrix M_1 of size $m \times n$, a positive integer $m' \leq m$, and an integer sequence $(s_1, \cdots, s_n)$.

Now we transform I_1 into the instance I_2 of $(s_1, \cdots, s_n)$-MC-NR:

Instance I_2: matrix M_2 of size $m \times (n + m)$, where the first n columns of M_2 coincide with M_1, and the last m columns compose the unit $m \times m$ matrix; a positive integer $m' \leq m$ and an integer sequence $(s_1, \cdots, s_n, 1, \cdots, 1)$ of size $n + m$.

The Figure 1 below demonstrates the construction of M_2.

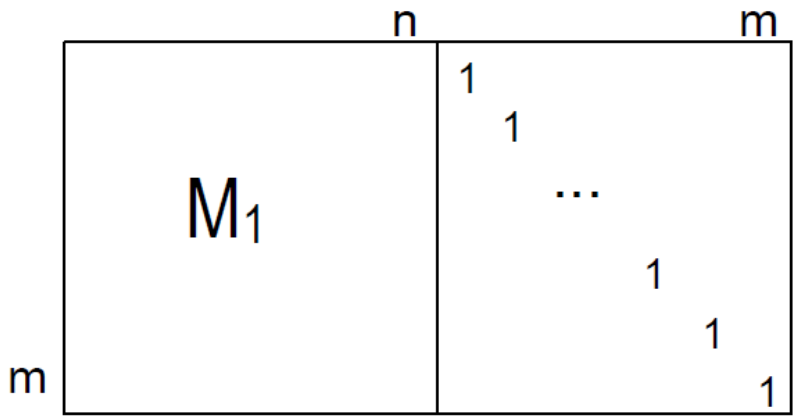

Figure1

It is easy to check that I_1 is a positive instance of $(s_1, \cdots, s_n)$-MC if and only if I_2 is a positive instance of $(s_1, \cdots, s_n)$-MC-NR. The transformations from one to the other instance are in polynomial time complexity.

Corollary. $(s_1, \cdots, s_n)$-MC-LR is NP-complete.

Existence versions

Below we consider the existence versions of the decision problems under consideration. While in covering problems above we require that the matrix contains at most given number of rows and each column of the matrix contains at least the given number of ones, - in existence versions we will require that that the matrix contains exactly m rows and the i-th column has exactly s_i ones. Theorem 2 and Theorem 3 below imply that in this way we do not make the problems easier.

$(s_1, \cdots, s_n)$- EXISTENCE

Given positive integers m, n, and an integer sequence $(s_1, \cdots, s_n)$, where $0 \leq s_i \leq m$ for $i = 1, \cdots, n$. Does there exist a binary matrix M of size $m \times n$ with the column sum vector $(s_1, \cdots, s_n)$.[1]

$(s_1, \cdots, s_n)$- EXISTENCE WITH LIMITED REPETITIONS ($(s_1, \cdots, s_n)$- EXISTENCE-LR).

Given positive integers m, n, r and an integer sequence $(s_1, \cdots, s_n)$, where $0 \leq s_i \leq m$ for $i = 1, \cdots, n$. Does there exist a binary matrix M of size $m \times n$ with the column sum vector $(s_1, \cdots, s_n)$, such that the number of repetitions of each row is limited by r.

$(s_1, \cdots, s_n)$- EXISTENCE WITH NO REPETITIONS ($(s_1, \cdots, s_n)$- EXISTENCE-NR)

Given positive integers m, n, and an integer sequence $(s_1, \cdots, s_n)$, where $0 \leq s_i \leq m$ for $i = 1, \cdots, n$. Does there exist a binary matrix M of size $m \times n$ with the column sum vector $(s_1, \cdots, s_n)$ and with different rows.

Notice that $(s_1, \cdots, s_n)$ - EXISTENCE-NR is a sub-problem of $((s_1, \cdots, s_n)$-MC-NR) with M being the $2^n \times n$ binary matrix whose rows are all binary sequences of size n. In this case, $m' \leq 2^n$, $0 \leq s_i \leq \min\{m', 2^{n-1}\}$.

Complexity

$(s_1, \cdots, s_n)$- EXISTENCE has obvious solution - the matrix exists if and only if $0 \leq s_i \leq m$.

$(s_1, \cdots, s_n)$- EXISTENCE-NR is equivalent to the known hypergraph degree sequence problem.

HYPERGRAPH DEGREE SEQUENCE PROBLEM

Given positive integers m, n, and an integer sequence $(s_1, \cdots, s_n)$. Does there exist a simple hypergraph with n vertices and m hyperedges, such that s_i is the degree of the i-th vertex (number of hyperedges containing the i-th vertex).

Hypergraph degree sequence problem is one of the known open problems in the hypergraph theory [Berge, 89], for which nor a polynomial algorithm is found neither the NP-completeness is proved.

On the other hand $(s_1, \cdots, s_n)$- EXISTENCE-NR is a sub-problem of $(s_1, \cdots, s_n)$- EXISTENCE-LR with $r = 1$, and thus:

$(s_1, \cdots, s_n)$- EXISTENCE-NR and $(s_1, \cdots, s_n)$- EXISTENCE-LR are open problems.

Theorem 2. [Sah, 2009]

Let M be a binary matrix of size $m \times n$ with different rows and with column sum vector $(s_1, \cdots, s_n)$, where $s_i > m/2$ for some index i. Then there exist M', a binary matrix of size $m \times n$ with different rows, with column sum vector $(s_1, \cdots, s_i - 1, \cdots, s_n)$.

Theorem 3. Let M be a binary matrix of size $m \times n$ with different rows and with column sum vector $(s_1, \cdots, s_n)$, where $s_i \geq m/2$ for $i = 1, \cdots, n$ and $s_j > m/2$ for some index j. Then there exists a binary matrix of size $(m + 1) \times n$ with different rows and with the column sum vector $(s_1, \cdots, s_n)$.

Proof. $s_j > m/2$ for some index j implies that in the j-th column of M the number of ones is greater than the number of zeros. Hence there is a binary sequence of size n with 0 in j-th position, which is not contained in M. Compose M' a binary matrix of size $(m + 1) \times n$ by appending this sequences into

[1] $(s_1, \cdots, s_n)$, is the column sum vector of M if the i-th column of M contains exactly s_i ones.

M. Then M' has column sum vector $(s'_1, \cdots, s'_n)$ greater than $(s_1, \cdots, s_n)$ in several components. According to Theorem 2 there exists a matrix with the column sum vector $(s_1, \cdots, s_n)$.

Further we will restrict attention with $(s_1, \cdots, s_n)$- EXISTENCE-NR and $(s_1, \cdots, s_n)$- EXISTENCE-LR.

3. Hierarchical classification with constraints

Hierarchical classification

In general, hierarchical classification addresses the problem of mapping the sets of classified objects into hierarchies of classes. Many important real-world classification problems are hierarchical in their nature, and it is useful when such classes are organized into a class hierarchy - typically in a tree diagram.

Top-down method in hierarchical classification starts from the root of the tree. The set of all objects is initially considered as a single class assigned to the root. Then a classifier/test applied to this set divides the set into smaller subsets/classes, each assigned to a child node of the root in the tree structure. This process is continued until each subset/class contains single object, assigned to nodes – and these are leaf nodes of the tree.

The resulting tree structure can be reduced at any level between the root and the leaf nodes – depending on how deep the classification in the hierarchy is performed. Depending on application problem, hierarchical classification algorithm can always reach a leaf node, or can stop at any level of the hierarchy, - in case if there is a constraint - limitation on the number of classes, sizes of classes, etc.

Constraints

Construction of classification trees, i.e. partitioning each set of objects assigned to a not leaf node, is performed by applying classifiers. The goal in classification problems is to create classifiers in such a way, that they divide the set of objects into classes such that objects in the same class are similar to one another and dissimilar to objects in other classes. For example the CART [Brei, 84] tree divides the sets by the impurity target. Coming from application problems, an additional functional for optimization can be given, and the problem is in constructing a classification tree with optimal value of the functional. In particular, optimality might be required for the height of the tree, the sub-tree weights on layers, the number of sub-trees, and others.

Solving $(s_1, \cdots, s_n)$ - EXISTENCE-NR by classification trees

C is a collection of objects given by the value vectors of a set $\{a_1, \cdots, a_n\}$ of binary attributes. Two objects of C are different if there is an attribute, which is satisfied for one object, and is not satisfied for the other. Otherwise two objects coincide. For each attribute a_i consider s_i - the number of objects satisfying a_i. In these conditions, the problem is in existence/construction of a set of m different vectors/objects.

We will seek approximate solutions of the problem with help of hierarchical classification trees and top down method.

Construction of the tree

The whole set of m objects, which is to be constructed is virtually assigned to the root vertex of the tree. Thus at the root vertex we have one class consisting of m objects. Then we apply a classifier to split the set of objects into two parts depending on whether or not objects satisfy the first attribute. The classifier C_0 on the root is obvious - it divides the whole set into an s_1-subset and an $(m - s_1)$-subset. On the first level of the tree we create two nodes (roots of the left and right subtrees), and assign them s_1 and $(m - s_1)$ objects. Further we will enumerate subtrees on each level and denote by $d_{i,j}$ the size of subset assigned to the j-th subtree on the i-th level. $d_{1,1} = s_1$ and $d_{1,2} = m - s_1$ on the first level. Then we build classifiers $C_{1,1}$, $C_{1,2}$ for partitioning subsets on the first level taking into account that s_2 objects must satisfy the second attribute. $C_{1,1}$ can generate any integer partition of $d_{1,1}$: $d_{1,1} = d_{2,1} + d_{2,2}$, and $C_{1,2}$ can generate any integer partition of $d_{1,2}$: $d_{1,2} = d_{2,3} +$

$d_{2,4}$, provided that $d_{2,1} + d_{2,3} = s_2$, and $d_{2,2} + d_{2,4} = m - s_2$. Let on the k-th level we have constructed p subtrees associated with $d_{k,1}, \cdots, d_{k,p}$-sets respectively. We apply classifiers $C_{k,1}, \cdots, C_{k,p}$ to divide these sets, given that the summary number of all objects associated with the left subtrees equals s_{k+1}. As far as all nodes on the n-th level of the tree must be assigned single objects, and on the other hand, the tree height is known beforehand, - then the classifiers must follow particular goals.

The global/per tree goal can be for example, to achieve required sizes of classes on closer to the root levels.

Local/per level/ goals /or optimization criteria/ must be chosen in such a way that they achieve the optimal value on the solution tree. For example, the criterion can be:

1. Maximization of the number of pairs of different objects;
2. Maximization of the number of different objects;
3. Maximization of the number of odd size-subsets.

Maximization of the number of pairs of different objects

This criterion was investigated in [Sah, 2010; Sah, Asl, 2011] and a greedy algorithm G was proposed, which on each level provides the optimal value of the number of pairs of different objects. According to G each subtree on the same level k, must be partitioned in such a manner that $s_k - (m - s_k)$ or $(m - s_k) - s_k$ difference is distributed equally.

Algorithm G

Without loss of generality we can assume that $s_k \geq (m - s_k)$ for $k = 1, \cdots, n$.

Step1. Construction of the first level of the tree

The first level contains the roots of two subtrees: the right subtree associated with the s_1-set; and the left subtree associated with the $(m - s_1)$-set. We denote the sizes of these sets by $d_{1,1}$ and $d_{1,2}$. Hereafter the first sub-index will indicate the number of level and the second – the number of subtrees at the level. Odd numbers correspond to right subtrees, and even numbers correspond to left subtrees. Thus,

$d_{1,1} + d_{1,2} = m$, $d_{1,1} = s_1$, and the number of pairs of different objects is equal to: $d_{1,1} \cdot d_{1,2}$.

Let we have constructed the first $k-1$ levels of the tree. In general, $(k-1)$-th level contains 2^{k-1} subtrees. Empty subtrees are possible, and let assume that $(k-1)$-th level contains p non-empty subtrees associated with $d_{k-1,1}, d_{k-1,2}, \cdots, d_{k-1,p}$-sets, respectively. Objects coincide within the same set and differ otherwise. When all subtrees on some level are associated with single objects, then at this level all objects are differentiated and the maximum number of pairs of different objects is already achieved. Further constructions are arbitrary.

Step k. Each subtree associated with $d_{k-1,i}$ -set is partitioned into left and right subtrees: $d_{k-1,i} = d_{k-1,i,l} + d_{k-1,i,r}$, taking into account that $\sum_{i=1}^{p} d_{k-1,i,l} = m - s_k$ and $\sum_{i=1}^{p} d_{k-1,i,r} = s_k$. The number of pairs of different objects will increase by $\sum_{i=1}^{p} d_{k-1,i,l} \cdot d_{k-1,i,r}$.

We will realize partitions of sets having a goal to minimize size differences.

The idea is in the following: if $s_k = m - s_k$ for $i = 1, \cdots, n$, then we would divide every set into 2 equal (± 1) parts and assign them to the left and right subtrees respectively. This will lead to 1-size sets in logarithmic number ([Knuth, 1973]) steps. Furthermore, among all integer partitions of $d_{k-1,i} = d_{k-1,i,l} + d_{k-1,i,r}$ the largest value of $d_{k-1,i,l} \cdot d_{k-1,i,r}$ is achieved when $d_{k-1,i,l} = d_{k-1,i,r}$. Thus following this strategy would lead to the goal, but in general at each step k we have $s_k - (m - s_k)$ difference. Trying to be closer to equal sizes of sets us:

1) Distribute $s_k - (m - s_k)$ difference among sets/subtrees keeping a "homogeneous" distribution;

2) Divide the remaining sets/subtrees into 2 equal parts.

Example

$m = 14$, $n = 6$,$(s_1, s_2, s_3, s_4, s_5, s_6) = (7,10,10,12,12,12)$. Figure 2 demonstrates the construction of a tree by the algorithm G. Circles correspond to the nodes, and the numbers in circles are sizes of associated subsets. On the last level of the tree there are 2 nodes associated with 2-sets (filled circles). Thus in this example two pairs of objects are not differentiated.

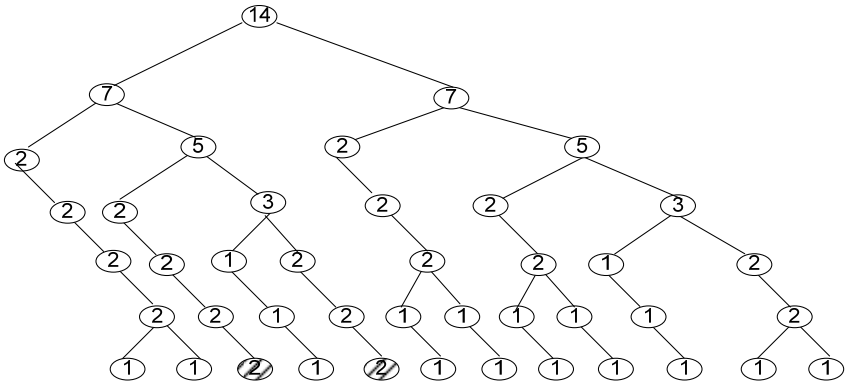

Figure 2

The second criterion, maximization of the number of different objects is not reasonable as creating 1-size classes at closer to the root levels will cause large sets afterwards, which may not be differentiated till the n-th level of the tree.

Maximization of the number of odd size sets

The main reasoning for using this criterion is as follows:

For each attribute a_k the number of objects is given satisfying to it. If $s_k = m - s_k$, then all classifiers on the k-th level would divide every set into 2 equal (± 1) parts. As much s_k is greater than $m - s_k$, so the partitions of the sets are unequally. The algorithm G distributes firstly the $s_k - (m - s_k)$ difference, and then divides the remaining sets into 2 equal parts. So, odd-size sets will reduce the difference $s_k - (m - s_k)$.

Below we construct algorithm G', which maximizes the number of odd-size sets in local step. G' can be considered as a modification of the algorithm G.

Algorithm G'

Assume that $s_1, \cdots, s_n$ are arranged in the increasing order. In each step we:

- Apply the algorithm G;
- Modify the partitioning.

Suppose that $d_{k-1,j}$ is the size of some set on the $(k-1)$-th level. Consider the partitioning of $d_{k-1,j}$ on the k-th level. We distinguish the following cases:

a) $d_{k-1,j}$ is odd.

Any partitioning of $d_{k-1,j}$ will produce one odd and one even size. Algorithm G' does not change the partitioning, except the cases mentioned in b) 2) below.

b) $d_{k-1,j}$ is even.

Suppose that: $d_{k-1,j} = d_{k-1,j,l} + d_{k-1,j,r}$ by G.

Distinguish the following cases:

1) $d_{k-1,j,l}$ and $d_{k-1,j,r}$ are both odd.
   Algorithm G' leaves this partitioning unchanged, except the cases mentioned in 2) below.
2) $d_{k-1,j,l}$ and $d_{k-1,j,r}$ are both even, or one of them is equal to 0.
   Algorithm G' partitions as:
   $d_{k-1,j} = (d_{k-1,j,l} + 1) + (d_{k-1,j,r} - 1)$ or $d_{k-1,j} = (d_{k-1,j,l} - 1) + (d_{k-1,j,r} + 1)$,
   if it is possible to shift the partitioning of some odd-size set; and s_k is enough to allow this modification.

It is clear that the algorithm G' is optimal locally, as any other partitioning cannot increase the number of odd sizes.

Two additional notes:

i. In case of the same number of odd sizes, it is preferable to have maximal number of non-empty sets.
ii. If $\log_2(d_{k-1,j}) > n - k$, then modify the partitioning to avoid this size.

Consider the example above and apply the algorithm G'. Figure 3 demonstrates the construction of the tree by G'.

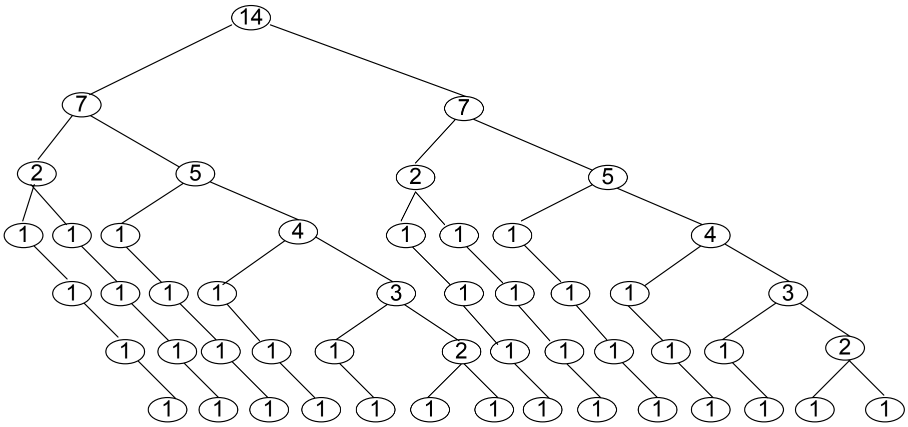

Figure 3

Now all pairs of objects are differentiated.

Solving $(s_1, \cdots, s_n)$- EXISTENCE-LR by classification trees

Construction of the tree is similar to the case of $(s_1, \cdots, s_n)$- EXISTENCE-NR. The only difference is that all nodes on the n-th level can be assigned at most r objects.

Local /per level/ goals /or optimization criteria/

1. Maximization of the number of pairs of different $\leq r$ size subsets;
2. Maximization of the number of $\leq r$ size-subsets;
3. Minimization of the largest size subset.

Maximization of the number of pairs of different $\leq r$ size subsets

It is obvious that the maximization of pairs of different objects provides the maximization of pairs of different $\leq r$ size subsets.Thus the algorithm G provides local optimal solution for $(s_1, \cdots, s_n)$ - EXISTENCE-LR.

By maximizing $\leq r$ -size classes or minimizing the largest size at closer to the root levels possibly will cause large sets of objects on low levels. Thus the two other criteria are seemed not reasonable.

Conclusion

Constrained object-characterization matrices appear in many areas raising questions about their existence or requiring algorithms that can construct them. Set theoretical and combinatorial relation is able to analyze the global picture but as we see here several hard fundamental problems appear that resist to be solved for many decades. The finite, combinatorial nature of the problem is complementarily related to step by step constructing technique such as greedy algorithms and hierarchical classifications. Combining the two parts – combinatorial and heuristic, gives a general picture of the current state in this area of subsets and set systems, element weights, classifications, hierarchies and algorithms of approximation.

Bibliography

[Asl, Kh, 2011] Aslanyan L., Khachatryan R., Association rule mining with n-dimensional unit cube chain split technique, Information Theories and Applications, Vol. 17, Number 2, pp. 108-123, 2010-11.

[Asl, Sah, 2009] L. Aslanyan H. Sahakyan, Chain Split and Computations in Practical Rule Mining, Book 08 Classification, Forecasting, Data mining, Institute of Information Theories and Applications FOI ITHEA 2009, pp.132-135.

[Berge, 89] C. Berge, Hypergraphs, North Holland, Amsterdam, 1989.

[Brei, 84] L. Breiman, J.H. Friedman, R.A. Olshen and C.J. Stone, Classification and Regression Trees, Belmont, CA: Wadsworh, Inc., 1984.

[Garey, Johnson, 1979] M.R. Garey, D.S.Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman & Co. New York, NY, USA ©1979, ISBN:0716710447.

[Knuth, 1973] D. Knuth, The Art of Computer Programming, vol.3. Sorting and Searching, Addison-Wesley Publishing Company, 1973.

[Kuhn, 05] F. Kuhn, P. von Rickenbach, R. Wattenhofer, E. Welzl, and A. Zollinger. Interference in cellular networks: The minimum membership set cover problem. In Proc. 11th COCOON, volume 3595 of LNCS, pages 188–198. Springer, 2005.

[Sah, 2009] H. Sahakyan, Numerical characterization of n-cube subset partitioning, Discrete Applied Mathematics 157 (2009) 2191-2197.

[Sah, 2010] H.Sahakyan, Approximation greedy algorithm for reconstructing of (0.1)-matrices with different rows, Information Theories and Applications, Vol. 17, Number 2, pp. 124-137, 2010-11.

[Sah, 2013] H. Sahakyan, (0,1)-Matrices with different rows, IEEE Explore, 7p. DOI:10.1109/CSITechnol.2013.6710342

[Sah, Asl, 2008], H. Sahakyan, L. Aslanyan, Lagrangean approximation for combinatorial inverse problems, Book 1 Algorithmic and Mathematical Foundations of the Artificial Intelligence, Institute of Information Theories and Applications FOI ITHEA, 2008, pp. 14-20.

[Sah, Asl, 2011] H. Sahakyan, L. Aslanyan, Evaluation of Greedy algorithm of constructing (0,1)-matrices with different rows, Information Technologies & Knowledge Vol.5, Number 1, pp. 55-66, 2011.

Authors' Information

Hasmik Sahakyan *– Scientific Secretary, Institute for Informatics and Automation Problems, NAS RA, P. Sevak St. 1, Yerevan 14, Armenia, e-mail: hasmik@ipia.sci.am*