Multi-stage optimization of decision and inhibitory trees for decision tables with many-valued decisions

Mohammad Azad, Mikhail Moshkov

PII: S0377-2217(17)30565-9
DOI: 10.1016/j.ejor.2017.06.026
Reference: EOR 14508

To appear in: *European Journal of Operational Research*

Received date: 28 June 2016
Revised date: 6 June 2017
Accepted date: 7 June 2017

Please cite this article as: Mohammad Azad, Mikhail Moshkov, Multi-stage optimization of decision and inhibitory trees for decision tables with many-valued decisions, *European Journal of Operational Research* (2017), doi: 10.1016/j.ejor.2017.06.026

Highlights

- The notion of strictly optimal decision tree is introduced.
- Results of decision tree optimization are clarified based on this notion.
- The optimization technique is extended to the case of inhibitory trees.
- Tools for the study of totally optimal trees are created.
- The existence of totally optimal trees is proven for many decision tables.

Multi-stage optimization of decision and inhibitory trees for decision tables with many-valued decisions

Mohammad Azad**, Mikhail Moshkov*

Computer, Electrical and Mathematical Science and Engineering Division, King Abdullah University of Science and Technology, Thuwal 23955-6900, Saudi Arabia

Abstract

We study problems of optimization of decision and inhibitory trees for decision tables with many-valued decisions. As cost functions, we consider depth, average depth, number of nodes, and number of terminal/nonterminal nodes in trees. Decision tables with many-valued decisions (multi-label decision tables) are often more accurate models for real-life data sets than usual decision tables with single-valued decisions. Inhibitory trees can sometimes capture more information from decision tables than decision trees. In this paper, we create dynamic programming algorithms for multi-stage optimization of trees relative to a sequence of cost functions. We apply these algorithms to prove the existence of totally optimal (simultaneously optimal relative to a number of cost functions) decision and inhibitory trees for some modified decision tables from the UCI Machine Learning Repository.

Keywords: Multiple criteria analysis, Dynamic programming, Decision trees, Inhibitory trees, Totally optimal trees.

1. Introduction

This paper is devoted to the optimization of decision and inhibitory trees for decision tables with many-valued decisions. In usual decision tables, a single decision is associated with each observation (row). However, situations

*Corresponding author

**Principal corresponding author. Tel.: +966 54 058 9797; fax: +966 12 802 1291.
Email address: mohammad.azad@kaust.edu.sa (Mohammad Azad)

in which a set of decisions is associated with each row are possible. Such tables (multi-label decision tables) have been used in semantic annotation of images (Boutell et al., 2004), categorization of emotions from music (Wieczorkowska et al., 2005), functional genomics (Blockeel et al., 2006), and text categorization (Zhou et al., 2005). These tables are appropriate models in combinatorial optimization problems, such as finding a Hamiltonian circuit with the minimum length in the traveling salesman problem or finding the nearest post office in the post office problem, in the diagnostics of faults in circuits, etc. (Moshkov & Zielosko, 2011). In these examples, objects can have more than one decision: text can belong to different categories, new customer can have more than one nearest post office, etc.

Similar tables (inconsistent decision tables containing equal rows labeled with different decisions) are studied in rough set theory (Pawlak, 1997, 1992; Slowinski, 1992; Pawlak & Slowinski, 1994). In this case, for each object, we know a set of decisions to which the decision belongs, but we do not have enough information to recognize the decision.

We use the terminology of decision tables with many-valued decisions to handle both multi-label decision tables and inconsistent decision tables. For a decision table with many-valued decisions, we consider the following interpretation: for a given row, we should find a decision from the set of decisions attached to the row. This means that during the construction of a decision tree, we have to stop partitioning of a subtable when there exists a decision that belongs to all sets of decisions attached to rows of this subtable.

The considered interpretation is different from the generalized decision (Pawlak, 1992; Slowinski, 1992; Dembczynski et al., 2006, 2007) studied in rough set theory, where we should recognize the whole set of decisions attached to the row. However, our interpretation makes sense for both multi-label and inconsistent decision tables. For example, if we consider a decision table corresponding to an optimization problem, the choice of an arbitrary decision from the set attached to the row means that we would like to return not all optimal decisions but only one. In the case of inconsistent decision tables, for a given object, we return the decision for this object or the decision for an object indistinguishable from the given one. Our previous experiments (Azad & Moshkov, 2015, 2016, 2014d; Azad et al., 2013, 2015b) showed that decision trees constructed in this framework often have less complexity than decision trees built in the framework of generalized decision.

Decision trees are used as algorithms to solve problems given by decision tables and to represent knowledge from decision tables (in this paper, we

do not consider decision trees as classifiers). Depending on the goal, we should either minimize the time complexity of decision trees (depth or average depth), the space complexity of decision trees (number of nodes or number of terminal/nonterminal nodes), or both. It is well known that the considered problems are NP-hard (see, for example, (Hyafil & Rivest, 1976; Moshkov & Zielosko, 2011; Chikalov et al., 2016)). However, it is possible to use dynamic programming for optimization of decision trees for medium-sized decision tables (Garey, 1972; Azad & Moshkov, 2014a,b,c). The aim of these just cited papers was to find an optimal decision tree.

The optimization technique considered here allows us not only to find an optimal tree but also to describe the set of all optimal trees or its subset by a directed acyclic graph. As a result, we can continue the process of optimization relative to another cost function. If we have a number of criteria (cost functions) ordered from the most important to the least important, we can make multi-stage optimization relative to the considered sequence of cost functions. For example, if the average depth is the most important for us and the number of nodes is not quit as important, then we can first describe the set of all decision trees with the minimum average depth and, among these trees, we can describe all that have minimum number of nodes. To the best of our knowledge, no similar techniques have been described earlier. The standard way in such a situation is to optimize trees relative to a new cost function depending on the considered cost functions, for example, their linear combination. However, this way cannot guarantee that the constructed tree will have a minimum average depth.

The multi-stage optimization approach was created for usual decision tables by (Moshkov & Chikalov, 2004; Alkhalid et al., 2013). The first results in this direction for decision tables with many-valued decisions were obtained by (Chikalov et al., 2005).

In (Chikalov et al., 2005), it was proven that the result of the optimization of decision trees relative to a strictly increasing cost function (for example, the number of nodes or average depth) is the set of optimal trees and that the result of the optimization of decision trees relative to an increasing cost function (for example, depth) is a subset of the set of optimal trees. Here, we introduce the notion of a strictly optimal decision tree. For each node of such a tree, the subtree with its root in this node is optimal for the subtable corresponding to this node. We prove that the result of optimization of decision trees relative to an increasing cost function is the set of strictly optimal decision trees and, for a strictly increasing cost function, the set of

optimal decision trees coincides with the set of strictly optimal decision trees. Note that each strictly increasing cost function is an increasing cost function. We now essentially have better understanding of dynamic programming algorithms for the optimization of decision trees.

In this paper, we study the more realistic notion of a decision table with many-valued decisions over an information system. As a result, we can prove that the time complexity of the proposed algorithms for decision tables with many-valued decisions over a restricted information system is polynomial depending on the number of conditional attributes (not on the size of tables as was proved by (Chikalov et al., 2005)).

We consider not only decision trees which are intensively studied in operations research (Abellán & Masegosa, 2010; Frini et al., 2012; Muller & Wiederhold, 2002) but also inhibitory trees, which are a new subject of investigation. In inhibitory trees, terminal nodes are labeled with expressions of the "$\neq$ decision" kind. In this paper, we discuss some results that demonstrated the advantages of inhibitory trees and rules (inhibitory rules were studied earlier (Moshkov et al., 2008; Delimata et al., 2009)) over decision ones. Inhibitory trees and rules sometimes can capture more information from decision tables than decision trees and rules can. We prove that instead of inhibitory trees for a decision table, we can study decision trees for a decision table complementary to the initial one. As a result, we can generalize the most part of the considered optimization techniques to the case of inhibitory trees.

We can use multi-stage optimization as a tool for the study of totally optimal decision trees (decision trees that are optimal relative to two or more given cost functions simultaneously). Previously, the existence of such trees was proven in rare cases using nontrivial techniques (for example, the existence of decision trees for sorting six elements that are simultaneously optimal relative to depth, average depth, and number of nodes (Knuth, 1998)). Using multi-stage optimization, we show that totally optimal decision and inhibitory trees exist for many decision tables with many-valued decisions obtained from data sets from the UCI Machine Learning Repository (Bache & Lichman, 2013) by removal of some conditional attributes and for different combinations of cost functions. These results are unpredictable. Totally optimal trees can be useful in real-life applications.

Multi-stage optimization is the only experimental tool that can show the existence of totally optimal trees for arbitrary number of cost functions. This tool is efficient enough for medium-sized decision trees. In particular, for each

of the decision tables used in our experiments, the time required for multi-stage optimization of decision or inhibitory trees is at most 10 minutes. For two cost functions, we can use our other tool - construction of the set of Pareto optimal points. The totally optimal tree exists if and only if there is only one Pareto optimal point. However, this tool is more complex.

The remainder of this paper is organized as follows. In section 2, we consider the algorithms for decision tree optimization and describe classes of decision tables for which the considered algorithms have polynomial time complexity depending on the number of conditional attributes in the input decision table. In section 3, we discuss the problems of inhibitory tree optimization and the difference between decision and inhibitory trees. In section 4, we consider experimental results related to multi-stage optimization of decision and inhibitory trees including the existence of totally optimal trees. Section 5 contains short conclusions.

2. Optimization of decision trees

In this section, we consider the notions of a decision table with many-valued decisions and a decision tree for such a table. We also describe algorithm for constructing a directed acyclic graph representing such decision trees and the algorithm for decision tree optimization. We explain how these algorithms can be used for multi-stage optimization of decision trees relative to different cost functions and for the study of totally optimal decision trees. We also describe classes of decision tables for which the considered algorithms have polynomial time complexity depending on the number of conditional attributes in the input decision table.

2.1. Decision tables with many-valued decisions

First, we consider the notion of a decision table with many-valued decisions. A *decision table with many-valued decisions* is a rectangular table T with $n \geq 1$ columns filled with numbers from the set $\omega = \{0, 1, 2, \ldots\}$ of nonnegative integers. Columns of the table are labeled with *conditional* attributes $f_1, \ldots, f_n$. Rows of the table are pairwise different, and each row r is labeled with a finite nonempty subset $D(r)$ of ω, which is interpreted as a *set of decisions*. Rows of the table are interpreted as tuples of values of conditional attributes. We denote by $Row(T)$ the set of rows of T. Let $D(T) = \bigcup_{r \in Row(T)} D(r)$. An example of a decision table with many-valued decisions T_0 can be found in Fig. 1.

$T_0 =$

	f_1	f_2	f_3	
r_1	0	0	0	{1}
r_2	0	1	1	{1,2}
r_3	1	0	1	{1,3}
r_4	1	1	0	{2,3}
r_5	0	0	1	{2}

Figure 1: A decision table with many-valued decisions T_0

We now consider the notion of the size of a decision table which is the length of its description. A decision table can be represented by a word over the alphabet $\{0, 1, ;, :, |\}$ in which numbers from ω are in binary representation (are represented by words over the alphabet $\{0, 1\}$), the symbol ";" is used to separate two numbers from ω, and the symbol "|" is used to separate two rows (we add numbers from $D(r)$ at the end of each row r and separate these numbers from r by the symbol ":"). The length of this word will be called the *size* of the decision table.

We now describe the notion of a degenerate decision table and some parameters of decision tables. We denote by $\mathcal{T}$ the set of all decision tables with many-valued decisions. Let $T \in \mathcal{T}$. The decision table T is called *empty* if it has no rows. The table T is called *degenerate* if it is empty or has a *common decision* – a decision $d \in D(T)$ such that $d \in D(r)$ for any row r of T. We denote by $\dim(T)$ the number of columns (conditional attributes) in T. We denote by $N(T)$ the number of rows in the table T and, for any $d \in \omega$, we denote by $N_d(T)$ the number of rows r of T such that $d \in D(r)$. By $mcd(T)$, we denote the *most common decision* for T, which is the minimum decision d_0 from $D(T)$ such that $N_{d_0}(T) = \max\{N_d(T) : d \in D(T)\}$. If T is empty, then $mcd(T) = 0$.

For any conditional attribute $f_i \in \{f_1, \ldots, f_n\}$, we denote by $E(T, f_i)$ the set of values of the attribute f_i in the table T. We denote by $E(T)$ the set of conditional attributes of T for which $|E(T, f_i)| \geq 2$. In other words, $E(T)$ is the set of conditional attributes that are not constant on T. Let $range(T) = \max\{|E(T, f_i)| : i = 1, \ldots, n\}$.

Let T be a nonempty decision table. A *subtable* of T is a table obtained from T by removal of some rows. Let $f_{i_1}, \ldots, f_{i_m} \in \{f_1, \ldots, f_n\}$ and $a_1, \ldots, a_m \in \omega$. We denote by $T(f_{i_1}, a_1) \ldots (f_{i_m}, a_m)$ the subtable of the table

T containing the rows from T, which at the intersection with the columns $f_{i_1}, \ldots, f_{i_m}$ have numbers $a_1, \ldots, a_m$, respectively. Such nonempty subtables, including the table T, are called *separable* subtables of T. We denote by $SEP(T)$ the set of separable subtables of the table T. Separable subtables of T can be considered as subproblems of an optimization problem given by the table T. Separable subtables are nodes of the directed acyclic graph representing decision trees. Figure 2 gives us an example of a separable subtable $T_0(f_1, 0)(f_3, 1)$ of the table T_0, which is degenerate.

$T_0(f_1, 0)(f_3, 1) =$

	f_1	f_2	f_3	
r_2	0	1	1	{1,2}
r_5	0	0	1	{2}

Figure 2: Example of a separable subtable of table T_0

2.2. Directed acyclic graph $\Delta(T)$

Let T be a nonempty decision table with n conditional attributes $f_1, \ldots, f_n$. We now consider an algorithm $\mathcal{A}_1$ for the construction of a directed acyclic graph $\Delta(T)$ which will be used for the description and optimization of decision trees. Nodes of this graph are some separable subtables of the table T. During each iteration, we process one node. We start with the graph that consists of one node T which is not processed and finish when all nodes of the graph are processed. During each iteration, we check the unprocessed node (subtable) Θ: if it is degenerate as described in 3.a, then we stop partitioning the node; otherwise, as described in 3.b, we partition the node and continue the process.

Algorithm $\mathcal{A}_1$

Input: A nonempty decision table T with conditional attributes $f_1, \ldots, f_n$.
Output: Directed acyclic graph $\Delta(T)$.

1. Construct the graph that consists of one node T which is not marked as processed.

2. If all nodes of the graph are processed, then the work of algorithm is finished. Return the resulting graph as $\Delta(T)$. Otherwise, choose a node (table) Θ that has not been processed yet.
3. a. If Θ is degenerate, mark the node Θ as processed and proceed to step 2.

   b. If Θ is not degenerate, then for each $f_i \in E(\Theta)$, draw a bundle of edges from the node Θ (this bundle of edges will be called the *f_i-bundle*). Let $E(\Theta, f_i) = \{a_1, \ldots, a_k\}$. Then, draw k edges from Θ and label these edges with the pairs $(f_i, a_1), \ldots, (f_i, a_k)$. These edges enter nodes $\Theta(f_i, a_1), \ldots, \Theta(f_i, a_k)$, respectively. If some of the nodes $\Theta(f_i, a_1), \ldots, \Theta(f_i, a_k)$ are not present in the graph, then add these nodes to the graph. Mark the node Θ as processed and return to step 2.

Let $L(\Delta(T))$ be the number of nodes in the graph $\Delta(T)$. We now evaluate the time complexity of algorithm $\mathcal{A}_1$.

Proposition 1. *The time complexity of algorithm $\mathcal{A}_1$ is bounded from above by a polynomial on the size of the input table T and the number $|SEP(T)|$ of different separable subtables of T.*

Proof. It is easy to see that each step of algorithm $\mathcal{A}_1$ has polynomial time complexity depending on the size of the table T and the number $L(\Delta(T))$. The number of steps is $O(L(\Delta(T)))$. Therefore the time complexity of algorithm $\mathcal{A}_1$ is bounded from above by a polynomial on the size of the input table T and the number $L(\Delta(T))$. The number $L(\Delta(T))$ is bounded from above by the number $|SEP(T)|$. □

A *bundle-intact subgraph* of the graph $\Delta(T)$ is a graph G obtained from $\Delta(T)$ by removal of some bundles of edges such that each nonterminal node of $\Delta(T)$ keeps at least one bundle of edges starting in this node. By definition, $\Delta(T)$ is a bundle-intact subgraph of $\Delta(T)$. A node Θ of the graph G is called *terminal* if there are no edges starting in this node. The node Θ is terminal if and only if Θ is degenerate. We denote by $L(G)$ the number of nodes in the graph G. We will obtain bundle-intact subgraphs of the graph $\Delta(T)$ as a result of the optimization of decision trees for T relative to different cost functions.

2.3. Decision trees

We now discuss the main notions connected with decision trees. Let T be a decision table with n conditional attributes $f_1, \ldots, f_n$.

A decision tree over T is a finite directed tree with root in which nonterminal nodes are labeled with attributes from the set $\{f_1, \ldots, f_n\}$, terminal nodes are labeled with numbers from ω, and, for each nonterminal node, edges starting from this node are labeled with pairwise different numbers from ω.

We now define the notion of a decision tree for T, which for a given row r of T, should return a decision from the set $D(r)$ attached to row r.

Let Γ be a decision tree over T and v be a node of Γ. We denote by $\Gamma(v)$ the subtree of Γ for which v is the root. Let Θ be a subtable of T, We define a subtable $\Theta(v) = \Theta_\Gamma(v)$ of the table Θ in the following way: if v is the root of Γ, then $\Theta(v) = \Theta$. If v is not the root of Γ, then $\Theta(v) = \Theta(f_{i_1}, a_1) \ldots (f_{i_m}, a_m)$, where $v_1, e_1, \ldots, v_m, e_m, v_{m+1} = v$ is the directed path from the root of Γ to v in which nodes $v_1, \ldots, v_m$ are labeled with attributes $f_{i_1}, \ldots, f_{i_m}$ and edges $e_1, \ldots, e_m$ are labeled with numbers $a_1, \ldots, a_m$, respectively.

A decision tree Γ over T is called a *decision tree for* T if, for any node v of Γ,

- If $T(v)$ is a degenerate table, then v is a terminal node labeled with $mcd(T(v))$.
- If $T(v)$ is not degenerate, then v is a nonterminal node, which is labeled with an attribute $f_i \in E(T(v))$ and, if $E(T(v), f_i) = \{a_1, \ldots, a_t\}$, then t edges start from the node v that are labeled with $a_1, \ldots, a_t$, respectively.

We denote by $DT(T)$ the set of decision trees for T.

For $b \in \omega$, we denote by $tree(b)$ the decision tree that contains only one (terminal) node and the node is labeled with b.

Let $f_i \in \{f_1, \ldots, f_n\}$, $a_1, \ldots, a_t$ be pairwise different numbers from ω, and $\Gamma_1, \ldots, \Gamma_t$ be decision trees over T. By $tree(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t)$, we denote the following decision tree over T: the root of the tree is labeled with f_i, and t edges start from the root which are labeled with $a_1, \ldots, a_t$. These edges enter the roots of decision trees $\Gamma_1, \ldots, \Gamma_t$, respectively.

Let $f_i \in E(T)$ and $E(T, f_i) = \{a_1, \ldots, a_t\}$. We denote $DT(T, f_i) = \{tree(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t) : \Gamma_j \in DT(T(f_i, a_j)), j = 1, \ldots, t\}$. The fol-

lowing proposition clarifies the structure of the set $DT(T)$ of decision trees for T.

Proposition 2. *Let T be a decision table. Then, $DT(T) = \{tree(mcd(T))\}$ if T is degenerate, and $DT(T) = \bigcup_{f_i \in E(T)} DT(T, f_i)$ if T is nondegenerate.*

Proof. Let T be degenerate. Then, $tree(mcd(T))$ is the only decision tree for T. Let T be nondegenerate and $\Gamma \in DT(T)$. Then, by definition, $\Gamma = tree(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t)$, where $f_i \in E(T)$ and $\{a_1, \ldots, a_t\} = E(T, f_i)$. Since $\Gamma \in DT(T)$, we have $\Gamma_j \in DT(T(f_i, a_j))$ for $j = 1, \ldots, t$. From here it follows that $\Gamma \in DT(T, f_i)$ for $f_i \in E(T)$. Therefore $DT(T) \subseteq \bigcup_{f_i \in E(T)} DT(T, f_i)$.

Let, for some $f_i \in E(T)$, $\Gamma \in DT(T, f_i)$. Then,

$$\Gamma = tree(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t),$$

where $\{a_1, \ldots, a_t\} = E(T, f_i)$, and $\Gamma_j \in DT(T(f_i, a_j))$ for $j = 1, \ldots, t$. Using these facts, we can show that $\Gamma \in DT(T)$. Therefore $\bigcup_{f_i \in E(T)} DT(T, f_i) \subseteq DT(T)$. □

Let G be a bundle-intact subgraph of the graph $\Delta(T)$. For each nonterminal node Θ of the graph G, we denote by $E_G(\Theta)$ the set of attributes f_i from $E(\Theta)$ such that f_i-bundle of edges starts from Θ in G. For each terminal node Θ, $E_G(\Theta) = \emptyset$. For each node Θ of the graph G, we define the set $Tree(G, \Theta)$ of decision trees in the following way. If Θ is a terminal node of G (in this case Θ is degenerate), then $Tree(G, \Theta) = \{tree(mcd(\Theta))\}$. Let Θ be a nonterminal node of G (in this case, Θ is nondegenerate), $f_i \in E_G(\Theta)$, and $E(\Theta, f_i) = \{a_1, \ldots, a_t\}$. We denote $Tree(G, \Theta, f_i) = \{tree(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t) : \Gamma_j \in Tree(G, \Theta(f_i, a_j)), j = 1, \ldots, t\}$. Then,

$$Tree(G, \Theta) = \bigcup_{f_i \in E_G(T)} Tree(G, \Theta, f_i).$$

The next statement shows that if $G = \Delta(T)$, then for each node Θ of the graph G, the set $Tree(G, \Theta)$ is equal to the set $DT(\Theta)$ of all decision trees for Θ. This means that (in the node T) the graph $\Delta(T)$ describes the set of all decision trees for T.

Proposition 3. *Let T be a decision table. Then, for any node Θ of the graph $\Delta(T)$, the following equality holds: $Tree(\Delta(T), \Theta) = DT(\Theta)$.*

PROOF. We prove this statement by induction on nodes of $\Delta(T)$. Let Θ be a terminal node of $\Delta(T)$. Then, $Tree(\Delta(T), \Theta) = \{tree(mcd(\Theta))\} = DT(\Theta)$. Now let Θ be a nonterminal node of $\Delta(T)$, and let us assume that, for any $f_i \in E(\Theta)$ and $\alpha_j \in E(\Theta, f_i)$, $Tree(\Delta(T), \Theta(f_i, a_j)) = DT(\Theta(f_i, a_j))$. Then, for any $f_i \in E(\Theta)$, we have $Tree(\Delta(T), \Theta, f_i) = DT(\Theta, f_i)$. Using Proposition 2, we obtain $Tree(\Delta(T), \Theta) = DT(\Theta)$. □

We now discuss the notion of a cost function for decision trees. Let $\mathbb{R}$ be the set of real numbers, $\mathbb{R}_+$ be the set of nonnegative real numbers, and n be a natural number. We consider a partial order $\leq$ on the set $\mathbb{R}^n$: $(x_1, \ldots, x_n) \leq (y_1, \ldots, y_n)$ if $x_1 \leq y_1, \ldots, x_n \leq y_n$. A function $g : \mathbb{R}^n_+ \to \mathbb{R}_+$ is called *increasing* if $g(x) \leq g(y)$ for any $x, y \in \mathbb{R}^n_+$ such that $x \leq y$. A function $g : \mathbb{R}^n_+ \to \mathbb{R}_+$ is called *strictly increasing* if $g(x) < g(y)$ for any $x, y \in \mathbb{R}^n_+$ such that $x \leq y$ and $x \neq y$. If g is strictly increasing, then evidently, g is increasing. For example, $\max(x_1, x_2)$ is increasing and $x_1 + x_2$ is strictly increasing.

Let $f(x_1, x_2)$ be a function from $\mathbb{R}^2_+$ to $\mathbb{R}_+$. We can extend $f(x_1, x_2)$ to a function with an arbitrary number of variables in the following way: $f(x_1) = x_1$ and, if $n > 2$, then $f(x_1, \ldots, x_n) = f(f(x_1, \ldots, x_{n-1}), x_n)$. If $f(x_1, x_2)$ is increasing, then the function $f(x_1, \ldots, x_n)$ is increasing. If $f(x_1, x_2)$ is strictly increasing, then the function $f(x_1, \ldots, x_n)$ is strictly increasing.

A cost function for decision trees is the function $\psi(T, \Gamma)$ which is defined on pairs decision table T and a decision tree Γ for T, and has values from $\mathbb{R}_+$. The function ψ is given by three functions, $\psi^0 : \mathcal{T} \to \mathbb{R}_+$, $F : \mathbb{R}^2_+ \to \mathbb{R}_+$ and $w : \mathcal{T} \to \mathbb{R}_+$.

The value of $\psi(T, \Gamma)$ is defined by induction:

- If $\Gamma = tree(mcd(T))$, then $\psi(T, \Gamma) = \psi^0(T)$.
- If $\Gamma = tree(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t)$, then

$$\psi(T, \Gamma) = F(\psi(T(f_i, a_1), \Gamma_1), \ldots, \psi(T(f_i, a_t), \Gamma_t)) + w(T).$$

The cost function ψ is called *increasing* if F is an increasing function. The cost function ψ is called *strictly increasing* if F is a strictly increasing function.

We now consider examples of cost functions for decision trees. For each of these cost functions, we give both "natural" and "inductive" definitions. The latter will be used in the optimization algorithms.

- *Depth* $h(T, \Gamma) = h(\Gamma)$ of a decision tree Γ for a decision table T is the maximum length of a path in Γ from the root to a terminal node. For this cost function, $\psi^0(T) = 0$, $F(x, y) = \max(x, y)$, and $w(T) = 1$. This is an increasing cost function.

- *Total path length* $tpl(T, \Gamma)$ of a decision tree Γ for a decision table T is equal to $\sum_{r \in Row(T)} l_\Gamma(r)$ where $l_\Gamma(r)$ is the length of a path in Γ from the root to a terminal node v, such that the row r belongs to $T_\Gamma(v)$. For this cost function, $\psi^0(T) = 0$, $F(x, y) = x + y$, and $w(T) = N(T)$. This is a strictly increasing cost function. Let T be a nonempty decision table. The value $tpl(T, \Gamma)/N(T)$ is the *average depth* $h_{avg}(T, \Gamma)$ of a decision tree Γ for a decision table T.

- *Number of nodes* $L(T, \Gamma) = L(\Gamma)$ of a decision tree Γ for a decision table T. For this cost function, $\psi^0(T) = 1$, $F(x, y) = x + y$, and $w(T) = 1$. This is a strictly increasing cost function.

- *Number of nonterminal nodes* $L_n(T, \Gamma) = L_n(\Gamma)$ of a decision tree Γ for a decision table T. For this cost function, $\psi^0(T) = 0$, $F(x, y) = x + y$, and $w(T) = 1$. This is a strictly increasing cost function.

- *Number of terminal nodes* $L_t(T, \Gamma) = L_t(\Gamma)$ of a decision tree Γ for a decision table T. For this cost function, $\psi^0(T) = 1$, $F(x, y) = x + y$, and $w(T) = 0$. This is a strictly increasing cost function.

For each of the considered cost functions, corresponding functions $\psi^0(T)$ and $w(T)$ have polynomial time complexity depending on the size of the decision tables.

An example of a decision tree for the table T_0 can be found in Fig. 3. If v is the node labeled with the attribute f_3, then subtable $T_0(v)$ corresponding to the node v is equal to $T_0(f_1, 0)$. The depth of the tree is 2, the average depth is $8/5 = 1.6$, the number of nodes is 5, the number of nonterminal nodes is 2, and the number of terminal nodes is 3.

2.4. Procedure of decision tree optimization

We now discuss how to optimize decision trees represented by a bundle-intact subgraph of the graph $\Delta(T)$ relative to a cost function for decision trees. This subgraph can be obtained, for example, during the optimization of decision trees relative to other cost functions.

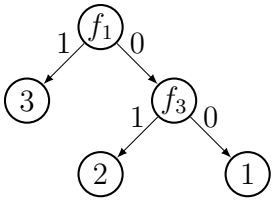

Figure 3: A decision tree for table T_0

Let ψ be an increasing cost function for decision trees given by the triple of functions ψ^0, F and w, T be a decision table with n conditional attributes $f_1, \ldots, f_n$, and G be a bundle-intact subgraph of the graph $\Delta(T)$.

Let Θ be a node of G, $\Gamma \in Tree(G, \Theta)$, and v be a node of Γ. One can show that the decision tree $\Gamma(v)$ belongs to the set $Tree(G, \Theta_\Gamma(v))$.

We consider not only optimal but also strictly optimal decision trees for which each subtree is optimal for the corresponding subtable.

A decision tree Γ from $Tree(G, \Theta)$ is called an *optimal decision tree for* Θ *relative to* ψ *and* G if $\psi(\Theta, \Gamma) = \min\{\psi(\Theta, \Gamma') : \Gamma' \in Tree(G, \Theta)\}$. A decision tree Γ from $Tree(G, \Theta)$ is called a *strictly optimal decision tree for* Θ *relative to* ψ *and* G if, for any node v of Γ, the decision tree $\Gamma(v)$ is an optimal decision tree for $\Theta_\Gamma(v)$ relative to ψ and G.

We denote by $Tree_\psi^{opt}(G, \Theta)$ the set of optimal decision trees for Θ relative to ψ and G. We denote by $Tree_\psi^{s-opt}(G, \Theta)$ the set of strictly optimal decision trees for Θ relative to ψ and G. Let $\Gamma \in Tree_\psi^{opt}(G, \Theta)$ and $\Gamma = tree(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t)$. Then, $\Gamma \in Tree_\psi^{s-opt}(G, \Theta)$ if and only if $\Gamma_j \in Tree_\psi^{s-opt}(G, \Theta(f_i, a_j))$ for $j = 1, \ldots, t$.

The following statement shows that, for a bundle-intact subgraph of the initial directed acyclic graph and a strictly increasing cost function, the set of optimal decision trees is equal to the set of strictly optimal decision trees.

Proposition 4. *Let ψ be a strictly increasing cost function for decision trees, T be a decision table, and G be a bundle-intact subgraph of the graph $\Delta(T)$. Then, for any node Θ of the graph G, $Tree_\psi^{opt}(G, \Theta) = Tree_\psi^{s-opt}(G, \Theta)$.*

Proof. It is clear that

$$Tree_\psi^{s-opt}(G, \Theta) \subseteq Tree_\psi^{opt}(G, \Theta).$$

Let $\Gamma \in Tree_\psi^{opt}(G, \Theta)$ and let us assume that $\Gamma \notin Tree_\psi^{s-opt}(G, \Theta)$. Then, there is a node v of Γ such that $\Gamma(v) \notin Tree_\psi^{opt}(G, \Theta_\Gamma(v))$. Let $\Gamma_0 \in$

$Tree_\psi^{opt}(G, \Theta_\Gamma(v))$ and Γ' be the decision tree obtained from Γ by replacing $\Gamma(v)$ with Γ_0. One can show that $\Gamma' \in Tree(G, \Theta)$. Since ψ is strictly increasing and $\psi(\Theta_\Gamma(v), \Gamma_0) < \psi(\Theta_\Gamma(v), \Gamma(v))$, we have $\psi(\Theta, \Gamma') < \psi(\Theta, \Gamma)$. Therefore, $\Gamma \notin Tree_\psi^{opt}(G, \Theta)$ which is impossible. Thus, $Tree_\psi^{opt}(G, \Theta) \subseteq Tree_\psi^{s-opt}(G, \Theta)$. □

We describe now an algorithm $\mathcal{A}_2$ (a *procedure of optimization relative to the cost function* ψ). The algorithm $\mathcal{A}_2$ attaches to each node Θ of G the number $c(\Theta) = \min\{\psi(\Theta, \Gamma) : \Gamma \in Tree(G, \Theta)\}$ and, probably, removes some f_i-bundles of edges starting from nonterminal nodes of G. As a result, we obtain a bundle-intact subgraph G^ψ of the graph G. It is clear that G^ψ is also a bundle-intact subgraph of the graph $\Delta(T)$. Later we will show that the graph G^ψ describes the set of strictly optimal decision trees for T relative to ψ and G.

Algorithm $\mathcal{A}_2$

Input: A bundle-intact subgraph G of the graph $\Delta(T)$ for some decision table T, and an increasing cost function ψ for decision trees given by the triple of functions ψ^0, F and w.
Output: The bundle-intact subgraph G^ψ of the graph G.

1. If all nodes of the graph G are processed, then return the obtained graph as G^ψ and finish the work of algorithm. Otherwise, choose a node Θ of the graph G that is not yet processed and which is either a terminal node of G or a nonterminal node of G for which all children are processed.
2. If Θ is a terminal node, then set $c(\Theta) = \psi^0(\Theta)$, mark node Θ as processed and proceed to step 1.
3. If Θ is a nonterminal node, then for each $f_i \in E_G(\Theta)$, compute the value $c(\Theta, f_i) = F(c(\Theta(f_i, a_1)), \ldots, c(\Theta(f_i, a_t))) + w(\Theta)$, where
$$\{a_1, \ldots, a_t\} = E(\Theta, f_i)$$
and set $c(\Theta) = \min\{c(\Theta, f_i) : f_i \in E_G(\Theta)\}$. Remove all f_i-bundles of edges starting from Θ for which $c(\Theta) < c(\Theta, f_i)$. Mark the node Θ as processed and proceed to step 1.

The following statement characterizes the time complexity of algorithm $\mathcal{A}_2$.

Proposition 5. *Let G be a bundle-intact subgraph of the graph $\Delta(T)$ for some decision table T with n conditional attributes and ψ be an increasing cost function for decision trees given by the triple of functions ψ^0, F and w. Then, to construct the graph G^ψ, algorithm $\mathcal{A}_2$ makes $O(nL(G)range(T))$ elementary operations (computations of F, w, ψ^0, comparisons, and additions).*

Proof. In each terminal node of the graph G, algorithm $\mathcal{A}_2$ computes the value of ψ^0. In each nonterminal node of G, the algorithm $\mathcal{A}_2$ computes the value of F (as the function with two variables) at most $range(T)n$ times, where $range(T) = \max\{|E(T, f_i)| : i = 1, \ldots, n\}$, and the value of w at most n times, makes at most n additions and at most $2n$ comparisons. Therefore algorithm $\mathcal{A}_2$ makes $O(nL(G)range(T))$ elementary operations. □

For any node Θ of the graph G and for any $f_i \in E_G(\Theta)$, we denote $\psi_G(\Theta) = \min\{\psi(\Theta, \Gamma) : \Gamma \in Tree(G, \Theta)\}$ and

$$\psi_G(\Theta, f_i) = \min\{\psi(\Theta, \Gamma) : \Gamma \in Tree(G, \Theta, f_i)\}.$$

The next statement will be used in the proof of algorithm $\mathcal{A}_2$'s correctness.

Lemma 6. *Let G be a bundle-intact subgraph of the graph $\Delta(T)$ for some decision table T with n conditional attributes and ψ be an increasing cost function for decision trees given by the triple of functions ψ^0, F and w. Then, for any node Θ of the graph G and for any attribute $f_i \in E_G(\Theta)$, algorithm $\mathcal{A}_2$ computes values $c(\Theta) = \psi_G(\Theta)$ and $c(\Theta, f_i) = \psi_G(\Theta, f_i)$.*

Proof. We prove this lemma by induction on the nodes of G. Let Θ be a terminal node of G. Then, $Tree(G, \Theta) = \{tree(mcd(\Theta))\}$ and $\psi_G(\Theta) = \psi^0(\Theta)$. Therefore, $c(\Theta) = \psi_G(\Theta)$. Since $E_G(\Theta) = \emptyset$, the considered statement holds for Θ.

Let now Θ be a nonterminal node of G such that the considered statement holds for each node $\Theta(f_i, a_j)$ with $f_i \in E_G(\Theta)$ and $a_j \in E(\Theta, f_i)$. By definition, $Tree(G, \Theta) = \bigcup_{f_i \in E_G(\Theta)} Tree(G, \Theta, f_i)$ and, for each $f_i \in E_G(\Theta)$,

$$Tree(G, \Theta, f_i) = \{tree(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t) : \Gamma_j \in Tree(G, \Theta(f_i, a_j)), j = 1, \ldots, t\},$$

where $\{a_1, \ldots, a_t\} = E(\Theta, f_i)$. Since ψ is an increasing cost function,

$$\psi_G(\Theta, f_i) = F(\psi_G(\Theta(f_i, a_1)), \ldots, \psi_G(\Theta(f_i, a_t))) + w(\Theta),$$

where $\{a_1, \ldots, a_t\} = E(\Theta, f_i)$. It is clear that $\psi_G(\Theta) = \min\{\psi_G(\Theta, f_i) : f_i \in E_G(\Theta)\}$. By the induction hypothesis, $\psi_G(\Theta(f_i, a_j)) = c(\Theta(f_i, a_j))$ for each $f_i \in E_G(\Theta)$ and $a_j \in E(\Theta, f_i)$. Therefore, $c(\Theta, f_i) = \psi_G(\Theta, f_i)$ for each $f_i \in E_G(\Theta)$, and $c(\Theta) = \psi_G(\Theta)$. □

Now we prove the correctness of algorithm $\mathcal{A}_2$.

Theorem 7. *Let ψ be an increasing cost function for decision trees, T be a decision table, and G be a bundle-intact subgraph of the graph $\Delta(T)$. Then, for any node Θ of the graph G^ψ, the following equality holds: $Tree(G^\psi, \Theta) = Tree_\psi^{s-opt}(G, \Theta)$.*

Proof. We prove this theorem by induction on nodes of G^ψ. We use Lemma 6, which shows that, for any node Θ of the graph G and for any $f_i \in E_G(\Theta)$, $c(\Theta) = \psi_G(\Theta)$ and $c(\Theta, f_i) = \psi_G(\Theta, f_i)$.

Let Θ be a terminal node of G^ψ. Then, $Tree(G^\psi, \Theta) = \{tree(mcd(\Theta))\}$. It is clear that $Tree(G^\psi, \Theta) = Tree_\psi^{s-opt}(G, \Theta)$. Therefore, the considered statement holds for Θ.

Let Θ be a nonterminal node of G^ψ such that the considered statement holds for each node $\Theta(f_i, a_j)$ with $f_i \in E_G(\Theta)$ and $a_j \in E(\Theta, f_i)$. By definition,

$$Tree(G^\psi, \Theta) = \bigcup_{f_i \in E_{G^\psi}(\Theta)} Tree(G^\psi, \Theta, f_i)$$

and, for each $f_i \in E_{G^\psi}(\Theta)$, $Tree(G^\psi, \Theta, f_i) = \{tree(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t) : \Gamma_j \in Tree(G^\psi, \Theta(f_i, a_j)), j = 1, \ldots, t\}$, where $\{a_1, \ldots, a_t\} = E(\Theta, f_i)$.

We know that $E_{G^\psi}(\Theta) = \{f_i : f_i \in E_G(\Theta), \psi_G(\Theta, f_i) = \psi_G(\Theta)\}$. Let $f_i \in E_{G^\psi}(\Theta)$ and $\Gamma \in Tree(G^\psi, \Theta, f_i)$. Then, $\Gamma = tree(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t)$, where $\{a_1, \ldots, a_t\} = E(\Theta, f_i)$ and $\Gamma_j \in Tree(G^\psi, \Theta(f_i, a_j))$ for $j = 1, \ldots, t$. According to the induction hypothesis,

$$Tree(G^\psi, \Theta(f_i, a_j)) = Tree_\psi^{s-opt}(G, \Theta(f_i, a_j))$$

and $\Gamma_j \in Tree_\psi^{s-opt}(G^\psi, \Theta(f_i, a_j))$ for $j = 1, \ldots, t$. In particular,

$$\psi(\Theta(f_i, a_j), \Gamma_j) = \psi_G(\Theta(f_i, a_j))$$

for $j = 1, \ldots, t$. Since $\psi_G(\Theta, f_i) = \psi_G(\Theta)$, we have

$$F(\psi_G(\Theta(f_i, a_1)), \ldots, \psi_G(\Theta(f_i, a_t))) + w(\Theta) = \psi_G(\Theta)$$

and $\psi(\Theta, \Gamma) = \psi_G(\Theta)$. Therefore, $\Gamma \in Tree_\psi^{opt}(G, \Theta)$, $\Gamma \in Tree_\psi^{s-opt}(G, \Theta)$ and $Tree(G^\psi, \Theta) \subseteq Tree_\psi^{s-opt}(G, \Theta)$.

Let $\Gamma \in Tree_\psi^{s-opt}(G, \Theta)$. Since Θ is a nonterminal node, Γ can be represented in the form $\Gamma = tree(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t)$, where $f_i \in E_G(\Theta)$, $\{a_1, \ldots, a_t\} = E(\Theta, f_i)$, and $\Gamma_j \in Tree_\psi^{s-opt}(G, \Theta(f_i, a_j))$ for $j = 1, \ldots, t$. Since $\Gamma \in Tree_\psi^{s-opt}(G, \Theta)$, $\psi_G(\Theta, f_i) = \psi_G(\Theta)$ and $f_i \in E_{G^\psi}(T)$. According to the induction hypothesis, $Tree(G^\psi, \Theta(f_i, a_j)) = Tree_\psi^{s-opt}(G, \Theta(f_i, a_j))$ for $j = 1, \ldots, t$. Therefore, $\Gamma \in Tree(G^\psi, \Theta, f_i) \subseteq Tree(G^\psi, \Theta)$. As a result, we have $Tree_\psi^{s-opt}(G, \Theta) \subseteq Tree(G^\psi, \Theta)$. □

If ψ is a strictly increasing cost function, then the graph G^ψ describes the set of optimal decision trees for T relative to ψ and G.

Corollary 8. *Let ψ be a strictly increasing cost function, T be a decision table, and G be a bundle-intact subgraph of the graph $\Delta(T)$. Then, for any node Θ of the graph G^ψ, $Tree(G^\psi, \Theta) = Tree_\psi^{opt}(G, \Theta)$.*

This corollary follows immediately from Proposition 4 and Theorem 7.

2.5. Multi-stage optimization of decision trees

We now explain the possibilities of multi-stage optimization of decision trees relative to a sequence of cost functions.

Let $\psi_1, \ldots, \psi_t$ be cost functions and T be a decision table. For $i = 1, \ldots, t$, we define the set $DT^{\psi_1, \ldots, \psi_i}(T)$ of decision trees for T in the following way: $DT^{\psi_1}(T)$ is the set of all decision trees from $DT(T)$ that have minimum cost relative to ψ_1 among all trees from $DT(T)$. For $i = 1, \ldots, t-1$, $DT^{\psi_1, \ldots, \psi_{i+1}}(T)$ is the set of all decision trees from $DT^{\psi_1, \ldots, \psi_i}(T)$ that have minimum cost relative to ψ_{i+1} among all trees from $DT^{\psi_1, \ldots, \psi_i}(T)$.

We describe now a procedure of multi-stage optimization of decision trees for T relative to the sequence of cost functions $\psi_1, \ldots, \psi_t$. We begin from the graph $G = \Delta(T)$ and apply to it the procedure of optimization relative to the cost function ψ_1 (algorithm $\mathcal{A}_2$). As a result, we obtain a bundle-intact subgraph G^{ψ_1} of the graph G. We apply to graph G^{ψ_1} the procedure of optimization relative to the cost function ψ_2 and obtain a bundle-intact

subgraph G^{ψ_1,ψ_2} of the graph G, etc. At the last step, we apply to graph $G^{\psi_1,\dots,\psi_{t-1}}$ the procedure of optimization relative to the cost function ψ_t and obtain a bundle-intact subgraph $G^{\psi_1,\dots,\psi_t}$ of the graph G.

The following statement describes the results of multi-stage optimization of decision trees.

Proposition 9. *Let T be a decision table, $G = \Delta(T)$, $t \geq 2$, and let $\psi_1, \dots, \psi_t$ be cost functions.*

1. *If $\psi_1, \dots, \psi_t$ are strictly increasing then, for $i = 1, \dots, t$,*

$$Tree(G^{\psi_1,\dots,\psi_i}, T) = DT^{\psi_1,\dots,\psi_i}(T).$$

2. *If $\psi_1, \dots, \psi_{t-1}$ are strictly increasing and ψ_t is increasing, then*

$$Tree(G^{\psi_1,\dots,\psi_i}, T) = DT^{\psi_1,\dots,\psi_i}(T)$$

for $i = 1, \dots, t-1$ and $Tree(G^{\psi_1,\dots,\psi_t}, T) \subseteq DT^{\psi_1,\dots,\psi_t}(T)$.

Proof. Let the cost functions $\psi_1, \dots, \psi_t$ be strictly increasing. We prove by induction on i that $Tree(G^{\psi_1,\dots,\psi_i}, T) = DT^{\psi_1,\dots,\psi_i}(T)$ for $i = 1, \dots, t$. By Proposition 3, the set $Tree(G, T)$ is equal to the set $DT(T)$ of all decision trees for T. Using Corollary 8, we obtain that the set $Tree(G^{\psi_1}, T)$ coincides with the set $DT^{\psi_1}(T)$. Let the considered statement be true for some $i \in \{1, \dots, t-1\}$, i.e., $Tree(G^{\psi_1,\dots,\psi_i}, T) = DT^{\psi_1,\dots,\psi_i}(T)$. By Corollary 8, the set $Tree(G^{\psi_1,\dots,\psi_{i+1}}, T)$ coincides with the set of all decision trees from $Tree(G^{\psi_1,\dots,\psi_i}, T)$ that have minimum cost relative to ψ_{i+1} among all trees from $Tree(G^{\psi_1,\dots,\psi_i}, T)$. Using the inductive hypothesis, we obtain

$$Tree(G^{\psi_1,\dots,\psi_{i+1}}, T) = DT^{\psi_1,\dots,\psi_{i+1}}(T).$$

Therefore, the first statement of the considered proposition holds.

Let the cost functions $\psi_1, \dots, \psi_{t-1}$ be strictly increasing and the cost function ψ_t be increasing. We already proved that $Tree(G^{\psi_1,\dots,\psi_i}, T) = DT^{\psi_1,\dots,\psi_i}(T)$ for $i = 1, \dots, t-1$. By Theorem 7, the set $Tree(G^{\psi_1,\dots,\psi_t}, T)$ coincides with the set $Tree^{s-opt}_{\psi t}(G^{\psi_1,\dots,\psi_{t-1}}, T)$, which is a subset of the set of all decision trees from $Tree(G^{\psi_1,\dots,\psi_{t-1}}, T)$ that have minimum cost relative to ψ_t among all trees from $Tree(G^{\psi_1,\dots,\psi_{t-1}}, T) = DT^{\psi_1,\dots,\psi_{t-1}}(T)$. Therefore, $Tree(G^{\psi_1,\dots,\psi_t}, T) \subseteq DT^{\psi_1,\dots,\psi_t}(T)$. □

2.6. Totally optimal decision trees

We now discuss the notion of a totally optimal decision tree relative to a number of cost functions. This is a decision tree that is simultaneously optimal for each of the considered cost functions.

For a cost function ψ, we denote $\psi(T) = \min\{\psi(T, \Gamma) : \Gamma \in DT(T)\}$, i.e., $\psi(T)$ is the minimum cost of a decision tree for T relative to the cost function ψ. Let $\psi_1, \ldots, \psi_t$ be cost functions and $t \geq 2$. A decision tree Γ for T is called a *totally optimal decision tree for T relative to the cost functions $\psi_1, \ldots, \psi_t$* if $\psi_1(T, \Gamma) = \psi_1(T), \ldots, \psi_t(T, \Gamma) = \psi_t(T)$, i.e., if Γ is simultaneously optimal relative to $\psi_1, \ldots, \psi_t$.

Assume that $\psi_1, \ldots, \psi_{t-1}$ are strictly increasing cost functions and that ψ_t is increasing or strictly increasing. We now describe how to recognize the existence of a decision tree for T that is a totally optimal decision tree for T relative to the cost functions $\psi_1, \ldots, \psi_t$.

First, we construct the graph $G = \Delta(T)$ using algorithm $\mathcal{A}_1$. For $i = 1, \ldots, t$, we apply to G the procedure of optimization relative to ψ_i (algorithm $\mathcal{A}_2$). For $i = 1, \ldots, t$, we obtain the graph G^{ψ_i} and the number $c(T)$ attached to the node T of G^{ψ_i} (see step 3 of algorithm $\mathcal{A}_2$). We denote this number as $c_i(T)$. Next, we apply to G the procedure of multi-stage optimization of decision trees for T relative to the sequence of cost functions $\psi_1, \ldots, \psi_t$. As a result, we obtain graphs $G^{\psi_1}, G^{\psi_1,\psi_2}, \ldots, G^{\psi_1,\ldots,\psi_t}$ and numbers attached to the node T of these graphs. For $i = 1, \ldots, t$, we denote by $\varphi_i(T)$ the number $c(T)$ attached to the node T of the graph $G^{\psi_1,\ldots,\psi_i}$ (see step 3 of algorithm $\mathcal{A}_2$).

Proposition 10. *Let T be a decision table, $t \geq 2$, $\psi_1, \ldots, \psi_{t-1}$ be strictly increasing cost functions, and let ψ_t be an increasing or strictly increasing cost function. Then, a totally optimal decision tree for T relative to the cost functions $\psi_1, \ldots, \psi_t$ exists if and only if $\varphi_i(T) = c_i(T)$ for $i = 1, \ldots, t$.*

Proof. By Proposition 3, the set $Tree(G, T)$ is equal to the set $DT(T)$ of all decision trees for T. From Lemma 6, it follows that, for $i = 1, \ldots, t$, $c_i(T) = \min\{\psi_i(T, \Gamma) : \Gamma \in Tree(G, T)\} = \min\{\psi_i(T, \Gamma) : \Gamma \in DT(T)\} = \psi_i(T)$.

It is clear that $\varphi_1(T) = c_1(T)$. From Lemma 6 and from Proposition 9, it follows that $\varphi_i(T) = \min\{\psi_i(T, \Gamma) : \Gamma \in Tree(G^{\psi_1,\ldots,\psi_{i-1}}, T)\} = \min\{\psi_i(T, \Gamma) : \Gamma \in DT^{\psi_1,\ldots,\psi_{i-1}}(T)\}$ for $i = 2, \ldots, t$. From these equalities

and Proposition 9, it follows that, for any decision tree $\Gamma \in Tree(G^{\psi_1,\dots,\psi_t}, T)$, $\psi_i(T,\Gamma) = \varphi_i(T)$ for $i = 1,\dots,t$.

If $\varphi_i(T) = c_i(T)$ for $i = 1,\dots,t$, then for any $\Gamma \in Tree(G^{\psi_1,\dots,\psi_t}, T)$, $\psi_i(T,\Gamma) = \psi_i(T)$ for $i = 1,\dots,t$, and Γ is a totally optimal decision tree for T relative to the cost functions $\psi_1,\dots,\psi_t$.

Let there exist a totally optimal decision tree Γ for T relative to the cost functions $\psi_1,\dots,\psi_t$. Then, $\psi_i(T,\Gamma) = \psi_i(T) = c_i(T)$ for $i = 1,\dots,t$. We know that $\varphi_1(T) = c_1(T)$. One can show that $\Gamma \in DT^{\psi_1,\dots,\psi_i}(T)$ for $i = 1,\dots,t$. Using Proposition 9, we obtain that $\varphi_i(T) = \min\{\psi_i(T,\Gamma) : \Gamma \in Tree(G^{\psi_1,\dots,\psi_{i-1}}, T)\} = \min\{\psi_i(T,\Gamma) : \Gamma \in DT^{\psi_1,\dots,\psi_{i-1}}(T)\} = c_i(T)$ for $i = 2,\dots,t$. □

2.7. Restricted information systems

We now describe classes of decision tables for which algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$ have polynomial time complexity depending on the number of conditional attributes in the input table.

Let A be a nonempty set and let F be a nonempty set of non-constant functions from A to $E_k = \{0,\dots,k-1\}$, $k \geq 2$. Functions from F are called *attributes*, and the pair $\mathcal{U} = (A, F)$ is called a *k-valued information system*.

For arbitrary attributes $f_1,\dots,f_n \in F$ and a mapping ν which corresponds to each tuple $(\delta_1,\dots,\delta_n) \in E_k^n$ a nonempty subset $\nu(\delta_1,\dots,\delta_n)$ of the set $\{0,\dots,k^n-1\}$ for which the cardinality is at most n^k, we denote by $T_\nu(f_1,\dots,f_n)$ the decision table with n conditional attributes $f_1,\dots,f_n$. This table contains the row $(\delta_1,\dots,\delta_n) \in E_k^n$ if and only if the system of equations

$$\{f_1(x) = \delta_1,\dots,f_n(x) = \delta_n\} \tag{1}$$

is *compatible* (has a solution from the set A). This row is labeled with the set of decisions $\nu(\delta_1,\dots,\delta_n)$. The table $T_\nu(f_1,\dots,f_n)$ is called a *decision table over the information system* $\mathcal{U}$. We denote by $\mathcal{T}(\mathcal{U})$ the set of decision tables over $\mathcal{U}$.

Let us consider the function

$$SEP_{\mathcal{U}}(n) = \max\{|SEP(T)| : T \in \mathcal{T}(\mathcal{U}), \dim(T) \leq n\},$$

where $\dim(T)$ is the number of conditional attributes in T, which characterizes the maximum number of separable subtables depending on the number of conditional attributes in decision tables over $\mathcal{U}$.

A system of equations of the kind (1) is called a *system of equations over* $\mathcal{U}$. Two systems of equations are called *equivalent* if they have the same set of solutions from A. A compatible system of equations will be called *irreducible* if each of its proper subsystems is not equivalent to the system. Let r be a natural number. An information system $\mathcal{U}$ will be called *r-restricted* if each irreducible system of equations over $\mathcal{U}$ consists of at most r equations. An information system $\mathcal{U}$ will be called *restricted* if it is r-restricted for some natural r.

Theorem 11. *(Moshkov & Chikalov, 2000) Let $\mathcal{U} = (A, F)$ be a k-valued information system. Then, the following statements hold:*

1. *If $\mathcal{U}$ is an r-restricted information system, then $SEP_{\mathcal{U}}(n) \leq (nk)^r + 1$ for any natural n.*
2. *If $\mathcal{U}$ is not a restricted information system, then $SEP_{\mathcal{U}}(n) \geq 2^n$ for any natural n.*

Let us consider a family of restricted information systems. Let d and t be natural numbers, $f_1, \ldots, f_t$ be functions from $\mathbb{R}^d$ to $\mathbb{R}$, and s be a function from $\mathbb{R}$ to $\{0, 1\}$ such that $s(x) = 0$ if $x < 0$ and $s(x) = 1$ if $x \geq 0$. Then, the 2-valued information system $U = (\mathbb{R}^d, F)$ where $F = \{s(f_i + c) : i = 1, \ldots, t, c \in \mathbb{R}\}$ is a $2t$-restricted information system.

If $f_1, \ldots, f_t$ are linear functions, then we deal with attributes corresponding to t families of parallel hyperplanes in $\mathbb{R}^d$ which is usual for decision trees for datasets with only t numerical attributes (Breiman et al., 1984).

We now consider a class of so-called linear information systems for which all restricted systems are known. Let P be the set of all points in the plane and let l be a straight line (line in short) in the plane. This line l divides the plane into three parts: two open half-planes H_1 and H_2 and the line l. Two attributes correspond to the line l. The first attribute takes the value 0 on points from H_1, and value 1 on points from H_2 and l. The second one takes the value 0 on points from H_2, and the value 1 on points from H_1 and l. We denote by $\mathcal{L}$ the set of all attributes corresponding to lines in the plane. We call information systems of the kind (P, F), where $F \subseteq \mathcal{L}$, as *linear* information systems. We describe all restricted linear information systems.

Let l be a line in the plane. Let us denote by $\mathcal{L}(l)$ the set of all attributes corresponding to lines that are parallel to l. Let p be a point in the plane.

We denote by $\mathcal{L}(p)$ the set of all attributes corresponding to lines that pass through p. A set C of attributes from $\mathcal{L}$ is called a *clone* if $C \subseteq \mathcal{L}(l)$ for some line l or $C \subseteq \mathcal{L}(p)$ for some point p.

Theorem 12. *(Moshkov, 2007) A linear information system (P, F) is restricted if and only if F is the union of a finite number of clones.*

We now evaluate the time complexity of algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$ for decision tables over a restricted information system $\mathcal{U}$. We begin from an auxiliary statement.

Lemma 13. *Let $\mathcal{U}$ be a restricted information system. Then, for decision tables from $\mathcal{T}(\mathcal{U})$, both the size and the number of separable subtables are bounded from above by polynomials on the number of conditional attributes.*

Proof. Let $\mathcal{U}$ be a k-valued information system that is r-restricted. For any decision table $T \in \mathcal{T}(\mathcal{U})$, each value of each conditional attribute is at most k, the value of each decision is at most $k^{\dim(T)}$, the cardinality of each set of decisions attached to rows of T is at most $\dim(T)^k$ and, by Theorem 11, $N(T) \leq |SEP(T)| \leq (\dim(T)k)^r + 1$. From here, it follows that the size of the decision tables from $\mathcal{T}(\mathcal{U})$ is bounded from above by a polynomial on the number of conditional attributes in the decision tables. By Theorem 11, the number of separable subtables for decision tables from $\mathcal{T}(\mathcal{U})$ is bounded from above by a polynomial on the number of conditional attributes in the decision tables. □

We now evaluate the time complexity of algorithm $\mathcal{A}_1$ for decision tables over a restricted information system.

Proposition 14. *Let $\mathcal{U}$ be a restricted information system. Then, algorithm $\mathcal{A}_1$ has polynomial time complexity for decision tables from $\mathcal{T}(\mathcal{U})$ depending on the number of conditional attributes.*

Proof. By Proposition 1, time complexity of algorithm $\mathcal{A}_1$ is bounded from above by a polynomial on the size of the input table T and the number $|SEP(T)|$ of different separable subtables of T. From Lemma 13, it follows that, for decision tables from $\mathcal{T}(\mathcal{U})$, both the size and the number of separable subtables are bounded from above by polynomials on the number of conditional attributes. □

We assume that numerical operations $\max(x, y)$ and $x+y$ have time complexity $O(1)$. This assumption is reasonable for computations with floating-point numbers. Based on this assumption, we evaluate the time complexity of algorithm $\mathcal{A}_2$ for the decision tables over a restricted information system.

Proposition 15. *Let $\psi \in \{h, tpl, L, L_n, L_t\}$ and $\mathcal{U}$ be a restricted information system. Then, algorithm $\mathcal{A}_2$ has polynomial time complexity for the decision tables from $\mathcal{T}(\mathcal{U})$ depending on the number of conditional attributes in these tables.*

Proof. Since $\psi \in \{h, tpl, L, L_n, L_t\}$, ψ^0 is a constant, F is either $\max(x, y)$ or $x + y$, and w is either a constant or $N(T)$. Therefore the elementary operations used by algorithm $\mathcal{A}_2$ are either constants, or numerical operations $\max(x, y)$ and $x + y$ that have time complexity $O(1)$, or computations of the parameter $N(T)$ of the decision tables that have polynomial time complexity depending on the size of the decision tables. From Proposition 5, it follows that the number of elementary operations is bounded from above by a polynomial depending on the size of input table T and on the number of separable subtables of T. Using Lemma 13, we obtain that algorithm $\mathcal{A}_2$ has polynomial time complexity for the decision tables from $\mathcal{T}(\mathcal{U})$ depending on the number of conditional attributes in these tables. □

3. Optimization of inhibitory trees

We now consider the notion of inhibitory tree for a decision table with many-valued decisions. We show that the study of inhibitory trees for a decision table with many-valued decisions T can be reduced to the study of decision trees for a decision table T^C complementary to T. We consider possibilities of multi-stage optimization of inhibitory trees including study of totally optimal inhibitory trees. We also discuss some results showing advantages of inhibitory trees and rules over decision ones.

3.1. Inhibitory trees

First, we consider notions connected with inhibitory trees, including the notion of complementary decision table and some parameters of decision tables. Let T be a nondegenerate decision table with n conditional attributes $f_1, \ldots, f_n$. We denote by T^C *complementary to* T decision table obtained from the table T by changing, for each row $r \in Row(T)$, the set $D(r)$ with

the set $D(T) \setminus D(r)$. The complementary table T_0^C to T_0 is depicted in Fig. 4. When we consider complementary to T table T^C or when we study inhibitory trees for T, we assume that, for any row r of T, $D(r) \neq D(T)$.

$T_0^C =$

	f_1	f_2	f_3	
r_1	0	0	0	{2,3}
r_2	0	1	1	{3}
r_3	1	0	1	{2}
r_4	1	1	0	{1}
r_5	0	0	1	{1,3}

Figure 4: A decision table with many-valued decisions T_0^C complementary to T_0

Let Θ be a subtable of T. The subtable Θ is called *incomplete relative to* T if $D(\Theta) \subset D(T)$. By $lcd(T, \Theta)$ we denote the *least common decision for* Θ *relative to* T, which is the minimum decision d_0 from $D(T)$ such that $N_{d_0}(\Theta) = \min\{N_d(\Theta) : d \in D(T)\}$.

We now consider the notions of an inhibitory tree over a decision table and for a decision table.

An inhibitory tree over T is a finite directed tree with root in which nonterminal nodes are labeled with attributes from the set $\{f_1, \ldots, f_n\}$, terminal nodes are labeled with expressions of the kind $\neq t$, $t \in \omega$, and, for each nonterminal node, edges starting from this node are labeled with pairwise different numbers from ω.

Let Γ be an inhibitory tree over T and let v be a node of Γ. We denote by $\Gamma(v)$ the subtree of Γ for which v is the root. We now define a subtable $T(v) = T_\Gamma(v)$ of the table T. If v is the root of Γ, then $T(v) = T$. Let v not be the root of Γ and let $v_1, e_1, \ldots, v_m, e_m, v_{m+1} = v$ be the directed path from the root of Γ to v in which nodes $v_1, \ldots, v_m$ are labeled with attributes $f_{i_1}, \ldots, f_{i_m}$ and edges $e_1, \ldots, e_m$ are labeled with numbers $a_1, \ldots, a_m$, respectively. Then, $T(v) = T(f_{i_1}, a_1) \ldots (f_{i_m}, a_m)$.

An inhibitory tree Γ over T is called an *inhibitory tree for* T if, for any node v of Γ,

- If $T(v)$ is an incomplete subtable relative to T, then v is a terminal node labeled with $\neq lcd(T, T(v))$.

- If $T(v)$ is not an incomplete subtable relative to T, then v is a nonterminal node that is labeled with an attribute $f_i \in E(T(v))$ and, if $E(T(v), f_i) = \{a_1, \ldots, a_t\}$, then t edges start from the node v that are labeled with $a_1, \ldots, a_t$, respectively.

We denote by $IT(T)$ the set of inhibitory trees for T.

A cost function for inhibitory trees is a function $\psi(T, \Gamma)$ that is defined on pairs decision table T and an inhibitory tree Γ for T, and has values from $\mathbb{R}_+$.

We now consider examples of cost functions for inhibitory trees:

- *Depth* $h(T, \Gamma) = h(\Gamma)$ of an inhibitory tree Γ for a decision table T is the maximum length of a path in Γ from the root to a terminal node.
- *Total path length* $tpl(T, \Gamma)$ of an inhibitory tree Γ for a decision table T is equal to $\sum_{r \in Row(T)} l_\Gamma(r)$ where $l_\Gamma(r)$ is the length of a path in Γ from the root to a terminal node v such that the row r belongs to $T_\Gamma(v)$. Let T be a nonempty decision table. The value $tpl(T, \Gamma)/N(T)$ is the *average depth* $h_{avg}(T, \Gamma)$ of an inhibitory tree Γ for a decision table T.
- *Number of nodes* $L(T, \Gamma) = L(\Gamma)$ of an inhibitory tree Γ for a decision table T.
- *Number of nonterminal nodes* $L_n(T, \Gamma) = L_n(\Gamma)$ of an inhibitory tree Γ for a decision table T.
- *Number of terminal nodes* $L_t(T, \Gamma) = L_t(\Gamma)$ of an inhibitory tree Γ for a decision table T.

An example of an inhibitory tree for the table T_0 can be found in Fig. 5. The depth of the tree is 2, the average depth is $7/5 = 1.4$, the number of nodes is 5, the number of nonterminal nodes is 2, and the number of terminal nodes is 3.

We now discuss relationships between inhibitory trees for a decision table and decision trees for the complementary decision table.

Let Γ be a decision tree over T^C. We denote by Γ^- an inhibitory tree over T obtained from Γ by changing expressions attached to terminal nodes: if a terminal node in Γ is labeled with t, then the corresponding node in Γ^- is labeled with $\neq t$. Let B be a set of decision trees over T^C. We denote $B^- = \{\Gamma^- : \Gamma \in B\}$.

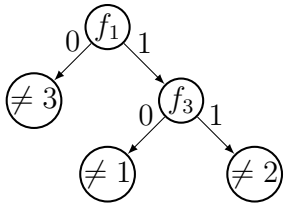

Figure 5: An inhibitory tree for table T_0

Proposition 16. *Let T be a nondegenerate decision table with attributes $f_1, \ldots, f_n$ and Γ be a decision tree over T^C. Then,*

1. *$\Gamma \in DT(T^C)$ if and only if $\Gamma^- \in IT(T)$;*
2. *If $\Gamma \in DT(T^C)$, then $\psi(T, \Gamma^-) = \psi(T^C, \Gamma)$ for any $\psi \in \{h, tpl, L, L_n, L_t\}$.*

Proof. Let v be a node of Γ (we keep the notation v for corresponding to v node in Γ^-). One can show that $mcd(T^C_\Gamma(v)) = lcd(T, T_{\Gamma^-}(v))$ and $T^C_\Gamma(v)$ is degenerate if and only if $T_{\Gamma^-}(v)$ is an incomplete subtable relative to T. It is clear that $E(T^C_\Gamma(v), f_i) = E(T_{\Gamma^-}(v), f_i)$ for any $f_i \in \{f_1, \ldots, f_n\}$, and $E(T^C_\Gamma(v)) = E(T_{\Gamma^-}(v))$. Using these facts, we obtain that $\Gamma \in DT(T^C)$ if and only if $\Gamma^- \in IT(T)$. If $\Gamma \in DT(T^C)$ and $\psi \in \{h, tpl, L, L_n, L_t\}$, then it is easy to show that $\psi(T, \Gamma^-) = \psi(T^C, \Gamma)$. □

Corollary 17. *Let T be a nondegenerate decision table. Then, $IT(T) = DT(T^C)^-$.*

3.2. Multi-stage optimization of inhibitory trees

We now consider possibilities of optimization of inhibitory trees including multi-stage optimization relative to a sequence of cost functions.

Let T be a nondegenerate decision table with n conditional attributes $f_1, \ldots, f_n$ and let T^C be the decision table complementary to T.

Let G be a bundle-intact subgraph of the graph $\Delta(T^C)$. We correspond to the node T of G a set $Tree(G, T^C)$ of decision trees for T^C. If $G = \Delta(T^C)$, then by Proposition 3, the set $Tree(G, T^C)$ is equal to the set $DT(T^C)$ of all decision trees for T^C. In the general case, $Tree(G, T^C) \subseteq DT(T^C)$.

Let us consider the set $Tree(G, T^C)^-$. From Corollary 17, it follows that $IT(T) = DT(T^C)^-$. Therefore, if $G = \Delta(T^C)$, then $Tree(G, T^C)^- = IT(T)$. In the general case, $Tree(G, T^C)^- \subseteq DT(T^C)^- = IT(T)$.

In Section 2.4, algorithm $\mathcal{A}_2$ is considered which, for the graph G and increasing cost function ψ for decision trees, constructs bundle-intact subgraph G^{ψ} of the graph G.

We begin from the study of optimization of inhibitory trees relative to a cost function from the set $\{tpl, L, L_n, L_t\}$.

Proposition 18. *Let T be a nondegenerate decision table, G be a bundle-intact subgraph of the graph $\Delta(T^C)$, and $\psi \in \{tpl, L, L_n, L_t\}$. Then the set $Tree(G^{\psi}, T^C)^-$ is equal to the set of all inhibitory trees from $Tree(G, T^C)^-$ that have minimum cost relative to ψ among all inhibitory trees from the set $Tree(G, T^C)^-$.*

Proof. Since $\psi \in \{tpl, L, L_n, L_t\}$, ψ is strictly increasing and, by Corollary 8, the set $Tree(G^{\psi}, T^C)$ is equal to the set of all decision trees from $Tree(G, T^C)$ that have minimum cost relative to ψ among all decision trees from the set $Tree(G, T^C)$. From Proposition 16 it follows that, for any $\Gamma \in Tree(G, T^C)$, $\psi(T^C, \Gamma) = \psi(T, \Gamma^-)$. Therefore, the set $Tree(G^{\psi}, T^C)^-$ is equal to the set of all inhibitory trees from $Tree(G, T^C)^-$ that have minimum cost relative to ψ among all inhibitory trees from $Tree(G, T^C)^-$. □

We now consider possibilities of optimization of inhibitory trees relative to the depth h.

Proposition 19. *Let T be a nondegenerate decision table, G be a bundle-intact subgraph of the graph $\Delta(T^C)$, and $\psi = h$. Then, the set $Tree(G^{\psi}, T^C)^-$ is a subset of the set of all inhibitory trees from $Tree(G, T^C)^-$ that have minimum cost relative to ψ among all inhibitory trees from $Tree(G, T^C)^-$.*

Proof. By Theorem 7, the set $Tree(G^{\psi}, T^C)$ is a subset of the set of all decision trees from $Tree(G, T^C)$ that have minimum cost relative to ψ among all decision trees from the set $Tree(G, T^C)$. From Proposition 16, it follows that, for any $\Gamma \in Tree(G, T^C)$, $\psi(T^C, \Gamma) = \psi(T, \Gamma^-)$. Therefore, the set $Tree(G^{\psi}, T^C)^-$ is a subset of the set of all inhibitory trees from $Tree(G, T^C)^-$ that have minimum cost relative to ψ among all inhibitory trees from $Tree(G, T^C)^-$. □

More accurate analysis of optimization relative to h can be done if we consider the notion of a strictly optimal inhibitory tree.

We can make multi-stage optimization of inhibitory trees relative to a sequence of cost functions $\psi_1, \psi_2, \ldots$ from $\{h, tpl, L, L_n, L_t\}$. We begin from the graph $G = \Delta(T^C)$ and apply to it the optimization procedure relative to the cost function ψ_1 (Algorithm $\mathcal{A}_2$). As a result, we obtain a bundle-intact subgraph G^{ψ_1} of the graph G. We know that the set $Tree(G, T^C)^-$ is equal to the set $IT(T)$ of all inhibitory trees for T. If $\psi_1 \neq h$, then by Proposition 18, the set $Tree(G^{\psi_1}, T^C)^-$ coincides with the set of all inhibitory trees from $Tree(G, T^C)^-$ that have minimum cost relative to ψ_1 among all trees from the set $Tree(G, T^C)^-$. Next, we apply to G^{ψ_1} the procedure of optimization relative to the cost function ψ_2. As a result, we obtain a bundle-intact subgraph G^{ψ_1,ψ_2} of the graph G. If $\psi_2 \neq h$, then by Proposition 18, the set $Tree(G^{\psi_1,\psi_2}, T^C)^-$ coincides with the set of all inhibitory trees from $Tree(G^{\psi_1}, T^C)^-$ that have minimum cost relative to ψ_2 among all trees from $Tree(G^{\psi_1}, T^C)^-$, etc.

If one of the cost functions ψ_i is equal to h, then by Proposition 19, the set $Tree(G^{\psi_1,\ldots,\psi_i}, T^C)^-$ is a subset of the set of all inhibitory trees from $Tree(G^{\psi_1,\ldots,\psi_{i-1}}, T^C)^-$ that have minimum cost relative to h among all trees from $Tree(G^{\psi_1,\ldots,\psi_{i-1}}, T^C)^-$.

3.3. Totally optimal inhibitory trees

We now discuss the notion of a totally optimal inhibitory tree relative to a number of cost functions.

For a cost function ψ, we denote $\psi^-(T) = \min\{\psi(T, \Gamma) : \Gamma \in IT(T)\}$, i.e., $\psi^-(T)$ is the minimum cost of an inhibitory tree for T relative to the cost function ψ. Let $\psi_1, \ldots, \psi_m$ be the cost functions with $m \geq 2$. An inhibitory tree Γ for T is called a *totally optimal inhibitory tree for T relative to the cost functions $\psi_1, \ldots, \psi_m$* if $\psi_1(T, \Gamma) = \psi_1^-(T), \ldots, \psi_m(T, \Gamma) = \psi_m^-(T)$, i.e., Γ is simultaneously optimal relative to $\psi_1, \ldots, \psi_m$.

Proposition 20. *Let T be a nondegenerate decision table, $\psi_1, \ldots, \psi_{m-1} \in \{tpl, L, L_n, L_t\}$, and $\psi_m \in \{h, tpl, L, L_n, L_t\}$. Then, a totally optimal inhibitory tree for T relative to the cost functions $\psi_1, \ldots, \psi_m$ exists if and only if a totally optimal decision tree for T^C relative to the cost functions $\psi_1, \ldots, \psi_m$ exists.*

Proof. The considered statement follows immediately from Proposition 16. □

We now describe how to recognize the existence of an inhibitory tree for T that is a totally optimal inhibitory tree for T relative to the cost functions $\psi_1, \ldots, \psi_m$.

First, we construct the graph $G = \Delta(T^C)$ using the Algorithm $\mathcal{A}_1$. For $i = 1, \ldots, m$, we apply to G the procedure of optimization relative to ψ_i (the Algorithm $\mathcal{A}_2$). As a result, we obtain, for $i = 1, \ldots, m$, the graph G^{ψ_i} and the number $c_i(T^C)$ attached to the node T^C of G^{ψ_i}. Next, we sequentially apply to G the optimization procedures relative to the cost functions $\psi_1, \ldots, \psi_m$. As a result, we obtain graphs $G^{\psi_1}, G^{\psi_1,\psi_2}, \ldots, G^{\psi_1,\ldots,\psi_m}$ and numbers $\varphi_1(T^C), \varphi_2(T^C), \ldots, \varphi_m(T^C)$ attached to the node T^C of these graphs. From Propositions 10 and 20, it follows that a totally optimal inhibitory tree for T relative to the cost functions $\psi_1, \ldots, \psi_m$ exists if and only if $\varphi_i(T^C) = c_i(T^C)$ for $i = 1, \ldots, m$.

3.4. Decision trees vs. inhibitory trees

In this and the next subsections, we consider two reasons why inhibitory trees and rules can be more powerful tools than decision trees and rules in data analysis.

Let us consider the decision table T_1 depicted in Fig. 6 and the problem of prediction of a decision value for a new object given by values of conditional attributes $f_1 = 0$ and $f_2 = 0$. To solve this problem, we use decision and inhibitory trees for T_1.

It is clear that there is no decision tree for T_1 the work of which, for a new object given by values of conditional attributes $f_1 = 0$ and $f_2 = 0$, finishes in a terminal node. This means that the decision trees will not give us any information about decisions corresponding to this new object. However, the inhibitory trees Γ_1 and Γ_2 for the decision table T_1 (see Figs. 7 and 8) will restrict the set of possible decisions to $\{2\}$. It is easy to see that there is no inhibitory tree for T_1 that returns $\neq 2$ for the new object.

$T_1 =$

f_1	f_2	
0	1	{1}
0	2	{2}
1	0	{2}
2	0	{3}

Figure 6: Decision table T_1

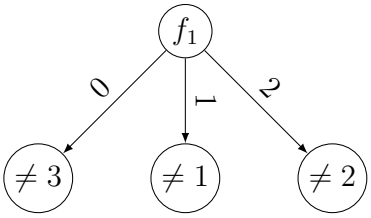

Figure 7: Inhibitory tree Γ_1 for decision table T_1

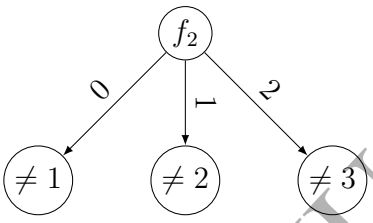

Figure 8: Inhibitory tree Γ_2 for decision table T_1

This example shows that the inhibitory trees can give us more information that the decision trees about a decision table.

3.5. Deterministic association rules vs. inhibitory association rules

We now compare expressive possibilities of deterministic association rules (usual association rules) and inhibitory association rules for information systems.

Let $S = (U, F)$ be an information system where U is a finite set of objects and F is a finite set of attributes (functions defined on U). We consider both deterministic and inhibitory association rules of the following form:

$$(f_1 = a_1) \wedge \ldots \wedge (f_t = a_t) \rightarrow f_{t+1} = a_{t+1},$$
$$(f_1 = a_1) \wedge \ldots \wedge (f_t = a_t) \rightarrow f_{t+1} \neq a_{t+1},$$

where $f_1, \ldots, f_{t+1}$ are attributes from F and $a_1, \ldots, a_{t+1}$ are values of these attributes. We consider only true and realizable rules. True means that the rule is true for any object from U. Realizable means that the left-hand side of the rule is true for at least one object from U.

We assume that different objects from U have different tuples of values of attributes from F, and we identify objects from U and tuples of values of attributes from F on these objects. Let V be the set of all tuples of known

values of attributes from F, i.e., V is the Cartesian product of ranges of attributes from F. It is clear that $U \subseteq V$. We say that the set U can be described by deterministic (inhibitory) rules if there exists a set P of true and realizable deterministic (inhibitory) rules such that the set of objects from V, for which all rules from P are true, coincides with U. In (Skowron & Suraj, 1993; Suraj, 2001) it was shown that there exist information systems $S = (U, F)$ such that the set U cannot be described by deterministic rules. In (Moshkov et al., 2008; Delimata et al., 2009) it was shown that, for any information system $S = (U, F)$, the set U can be described by inhibitory rules.

The mentioned results show that the inhibitory association rules can give us more information than the deterministic association rules about information systems. These results can be extended to so-called deterministic association trees (the analog of decision trees) and inhibitory association trees (the analog of inhibitory trees) but this question should be considered separately.

4. Experimental results

We conducted experiments to study the existence of totally optimal decision and inhibitory trees relative to the depth, average depth, and number of nodes. Instead of the investigation of inhibitory trees for a decision table with many-valued decisions T, we studied decision trees for the table T^C complementary to T.

4.1. Decision tables used in experiments

We took data sets (decision tables) from UCI ML Repository (Bache & Lichman, 2013) and removed one or more conditional attributes from them; as a result, for some tables, there were multiple rows that had equal values of conditional attributes but different decisions that were then merged into a single row labeled with the set of decisions from the group of equal rows. Before the experiment, we performed some preprocessing procedures. We removed an attribute if it had a unique value for each row. We filled the missing value for an attribute with the most common value for that attribute.

In Table 1, the first column 'Decision table T' refers to the name of the new decision table T (which we get after removing attributes from the data set from UCI ML Repository); the second column 'Original data set - Columns' refers to the name of the original data set along with the indexes

Table 1: Decision tables with many-valued decisions used in experiments

Decision table T	**Original data set - Columns**	**Rows**	**Attr**	$\lvert D(T)\rvert$	**Spectrum** #1, #2, #3, ...
CARS-1	CARS -1	432	5	4	258, 161, 13
FLAGS-4	FLAGS -1,2,3,19	176	22	6	168, 8
FLAGS-5	FLAGS -1,2,3,5,15	177	21	6	166, 10, 1
FLAGS-3	FLAGS -1,2,3	184	23	6	178, 6
FLAGS-1	FLAGS -1	190	25	6	188, 2
LYMPH-5	LYMPHOGRAPHY -1,13,14,15,18	122	13	4	113, 9
LYMPH-4	LYMPHOGRAPHY -13,14,15,18	136	14	4	132, 4
NURSERY-4	NURSERY -1,5,6,7	240	4	5	97, 96, 47
NURSERY-1	NURSERY -1	4320	7	5	2858, 1460, 2
POKER-5A	POKER-HAND -1,2,4,6,8	3324	5	10	128, 1877, 1115 198, 5, 1
POKER-5B	POKER-HAND -2,3,4,6,8	3323	5	10	130, 1850, 1137 199, 6, 1
POKER-5C	POKER-HAND -2,4,6,8,10	1024	5	10	0, 246, 444 286, 44, 4
ZOO-5	ZOO-DATA -2,6,8,9,13	43	11	7	40, 1, 2
ZOO-4	ZOO-DATA -2,9,13,14	44	12	7	40, 4
ZOO-2	ZOO-DATA -6,13	46	14	7	44, 2

Table 2: Existence of totally optimal decision and inhibitory trees for two cost functions

Decision table	Has totally optimal decision trees?			Has totally optimal inhibitory trees?		
	(h_{avg}, h)	(L, h)	(L, h_{avg})	(h_{avg}, h)	(L, h)	(L, h_{avg})
CARS-1	No	Yes	No	Yes	Yes	Yes
FLAGS-4	Yes	No	No	Yes	Yes	No
FLAGS-5	No	No	No	Yes	Yes	No
FLAGS-3	No	No	No	Yes	Yes	No
FLAGS-1	Yes	No	No	Yes	Yes	Yes
LYMPH-5	No	No	No	Yes	Yes	Yes
LYMPH-4	No	No	No	Yes	Yes	Yes
NURSERY-4	Yes	Yes	Yes	Yes	Yes	Yes
NURSERY-1	Yes	Yes	No	Yes	Yes	Yes
POKER-5A	Yes	Yes	No	Yes	No	No
POKER-5B	Yes	Yes	No	Yes	No	No
POKER-5C	Yes	Yes	Yes	Yes	Yes	Yes
ZOO-5	No	No	Yes	Yes	Yes	Yes
ZOO-4	No	Yes	No	Yes	Yes	Yes
ZOO-2	No	No	Yes	Yes	Yes	Yes

of attributes removed from the original data set; the column 'Rows' refers to the number of rows; the column 'Attr' refers to the number of attributes; the column '$|D(T)|$' refers to the total number of decisions in T; and the column 'Spectrum' refers to a sequence #1, #2, #3, ..., where $\#i$ means the number of rows in T that are labeled with sets of decisions containing i decisions.

4.2. Totally optimal trees relative to two cost functions

In Table 2, we show the results for the existence of totally optimal decision and inhibitory trees relative to two cost functions. The results are grouped according to the three pairs of cost functions: (average depth, depth), (number of nodes, depth), and (number of nodes, average depth).

For the decision trees, more than half of the considered decision tables do not have totally optimal decision trees for each pair of cost functions. On the other hand, for the inhibitory trees, we have totally optimal inhibitory trees for all the decision tables for the pair (average depth, depth), for almost all the decision tables except two cases for the pair (number of nodes, depth), and for ten cases for the pair (number of nodes, average depth).

Table 3: Existence of totally optimal decision trees for three cost functions

Decision table	Non-sequential			Sequential			Has totally optimal decision trees?
	L	h_{avg}	h	L	h_{avg}	h	
CARS-1	28	1.96	4	28	2.06	4	No
FLAGS-5	152	3.6	5	152	5.93	10	No
FLAGS-4	149	3.73	6	149	6.7	11	No
FLAGS-3	151	3.64	5	151	6.6	11	No
FLAGS-1	112	2.77	5	112	3.9	7	No
LYMPH-5	56	3.65	5	56	4.72	9	No
LYMPH-4	67	3.73	5	67	4.79	7	No
NURSERY-4	9	1.33	2	9	1.33	2	**Yes**
NURSERY-1	117	2.127	7	117	2.134	7	No
POKER-5A	230	2.62	4	230	2.63	4	No
POKER-5B	226	2.529	5	226	2.531	5	No
POKER-5C	69	2.33	5	69	2.33	5	**Yes**
ZOO-5	17	2.77	4	17	2.77	6	No
ZOO-4	23	3.34	5	23	3.41	7	No
ZOO-2	13	2.72	4	13	2.72	5	No

4.3. Totally optimal trees relative to three cost functions

In Tables 3 and 4, we show the results of the experiments with multi-stage optimization for the three cost functions L, h_{avg}, and h. We list first the values of the cost functions for non-sequential, i.e., individual cost optimization. Then, we list the values of cost functions for sequential, i.e., multi-stage optimization. Then, we list whether or not the decision tables have totally optimal trees. A totally optimal tree exists if and only if the values after non-sequential optimization are equal to the values after sequential optimization.

We can see that, for the case of decision trees, there are only two decision tables that have totally optimal trees. On the other hand, for the case of inhibitory trees, there are ten decision tables that have totally optimal trees. Moreover, the optimal inhibitory trees usually have smaller depth, average depth and number of nodes compared to optimal decision trees.

5. Conclusions

We studied the optimization of decision and inhibitory trees for decision tables with many-valued decisions. We created procedures to sequentially

Table 4: Existence of totally optimal inhibitory trees for three cost functions

Decision table	Non-sequential			Sequential			Has totally optimal inhibitory trees?
	L	h_{avg}	h	L	h_{avg}	h	
CARS-1	14	1.47	4	14	1.47	4	**Yes**
FLAGS-5	17	1.9	3	17	2.43	3	No
FLAGS-4	17	1.81	3	17	2.43	3	No
FLAGS-3	17	1.83	3	17	2.45	3	No
FLAGS-1	9	1	1	9	1	1	**Yes**
LYMPH-5	3	1	1	3	1	1	**Yes**
LYMPH-4	3	1	1	3	1	1	**Yes**
NURSERY-4	4	1	1	4	1	1	**Yes**
NURSERY-1	4	1	1	4	1	1	**Yes**
POKER-5A	13	1	1	13	1.5	2	No
POKER-5B	13	1	1	13	1.5	2	No
POKER-5C	13	1.5	2	13	1.5	2	**Yes**
ZOO-5	3	1	1	3	1	1	**Yes**
ZOO-4	3	1	1	3	1	1	**Yes**
ZOO-2	3	1	1	4	1	1	**Yes**

optimize cost functions and to find the existence of totally optimal decision and inhibitory trees.

We found that the optimal inhibitory trees for the considered decision tables have smaller values of cost functions compared to the optimal decision trees for the same decision tables. Additionally, we found that totally optimal inhibitory trees exist in more cases than totally optimal decision trees.

In the future, we are planning to study bi-criteria optimization of decision trees for decision tables with many-valued decisions by constructing the set of Pareto optimal points. The first results in this direction can be found in (Azad et al., 2015a).

Acknowledgment

Research reported in this publication was supported by the King Abdullah University of Science and Technology (KAUST). We are greatly indebted to the anonymous reviewers for useful comments and suggestions.

References

References

Abellán, J., & Masegosa, A. R. (2010). An ensemble method using credal decision trees. *European Journal of Operational Research*, *205*, 218–226.

Alkhalid, A., Amin, T., Chikalov, I., Hussain, S., Moshkov, M., & Zielosko, B. (2013). Optimization and analysis of decision trees and rules: dynamic programming approach. *Int. J. General Systems*, *42*, 614–634.

Azad, M., Chikalov, I., Hussain, S., & Moshkov, M. (2015a). Multi-pruning of decision trees for knowledge representation and classification. In *3rd IAPR Asian Conference on Pattern Recognition, ACPR 2015* (pp. 604–608). IEEE.

Azad, M., Chikalov, I., & Moshkov, M. (2013). Three approaches to deal with inconsistent decision tables – comparison of decision tree complexity. In *International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing* (pp. 46–54). Springer.

Azad, M., Chikalov, I., Moshkov, M., & Zielosko, B. (2015b). Three approaches to deal with tests for inconsistent decision tables – comparative study. In *Transactions on Rough Sets XIX* (pp. 38–50). Springer.

Azad, M., & Moshkov, M. (2014a). Minimization of decision tree average depth for decision tables with many-valued decisions. In *18th International Conference in Knowledge Based and Intelligent Information and Engineering Systems, KES 2014* (pp. 368–377). Elsevier.

Azad, M., & Moshkov, M. (2014b). Minimization of decision tree depth for multi-label decision tables. In *2014 IEEE International Conference on Granular Computing, GrC 2014* (pp. 7–12). IEEE.

Azad, M., & Moshkov, M. (2014c). Minimizing size of decision trees for multi-label decision tables. In *2014 Federated Conference on Computer Science and Information Systems* (pp. 67–74). IEEE.

Azad, M., & Moshkov, M. (2014d). 'Misclassification error' greedy heuristic to construct decision trees for inconsistent decision tables. In *International Conference on Knowledge Discovery and Information Retrieval* (pp. 184–191). SCITEPRESS.

Azad, M., & Moshkov, M. (2015). Classification and optimization of decision trees for inconsistent decision tables represented as MVD tables. In *2015 Federated Conference on Computer Science and Information Systems* (pp. 31–38). IEEE.

Azad, M., & Moshkov, M. (2016). Classification for inconsistent decision tables. In *International Joint Conference on Rough Sets* (pp. 525–534). Springer.

Bache, K., & Lichman, M. (2013). UCI Machine Learning Repository.

Blockeel, H., Schietgat, L., Struyf, J., Dzeroski, S., & Clare, A. (2006). Decision trees for hierarchical multilabel classification: A case study in functional genomics. In *10th European Conference on Principles and Practice of Knowledge Discovery in Databases* (pp. 18–29). Springer.

Boutell, M. R., Luo, J., Shen, X., & Brown, C. M. (2004). Learning multi-label scene classification. *Pattern Recognition*, *37*, 1757–1771.

Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). *Classification and Regression Trees*. Monterey: Wadsworth and Brooks.

Chikalov, I., Hussain, S., & Moshkov, M. (2016). Totally optimal decision trees for Boolean functions. *Discrete Applied Mathematics*, *215*, 1–13.

Chikalov, I., Moshkov, M., & Zelentsova, M. (2005). On optimization of decision trees. In *Transactions on Rough Sets IV* (pp. 18–36). Springer.

Delimata, P., Moshkov, M., Skowron, A., & Suraj, Z. (2009). *Inhibitory Rules in Data Analysis: A Rough Set Approach,* volume 163 of *Studies in Computational Intelligence*. Heidelberg: Springer.

Dembczynski, K., Greco, S., Kotlowski, W., & Slowinski, R. (2006). Quality of rough approximation in multi-criteria classification problems. In *Rough Sets and Current Trends in Computing: 5th International Conference, RSCTC 2006* (pp. 318–327). Springer.

Dembczynski, K., Greco, S., Kotlowski, W., & Slowinski, R. (2007). Optimized generalized decision in dominance-based rough set approach. In *Rough Sets and Knowledge Technology: Second International Conference, RSKT 2007* (pp. 118–125). Springer.

Frini, A., Guitouni, A., & Martel, J.-M. (2012). A general decomposition approach for multi-criteria decision trees. *European Journal of Operational Research*, *220*, 452–460.

Garey, M. R. (1972). Optimal binary identification procedures. *SIAM Journal on Applied Mathematics*, *23*, 173–186.

Hyafil, L., & Rivest, R. L. (1976). Constructing optimal binary decision trees is NP-complete. *Inf. Process. Lett.*, *5*, 15–17.

Knuth, D. E. (1998). *The Art of Computer Programming: Sorting and Searching,* volume 3. (2nd ed.). Boston: Pearson Education.

Moshkov, M. (2007). On the class of restricted linear information systems. *Discrete Math.*, *307*, 2837–2844.

Moshkov, M., & Chikalov, I. (2000). On algorithm for constructing of decision trees with minimal depth. *Fundam. Inform.*, *41*, 295–299.

Moshkov, M., & Chikalov, I. (2004). Consecutive optimization of decision trees concerning various complexity measures. *Fundam. Inform.*, *61*, 87–96.

Moshkov, M., Skowron, A., & Suraj, Z. (2008). Maximal consistent extensions of information systems relative to their theories. *Inf. Sci.*, *178*, 2600–2620.

Moshkov, M., & Zielosko, B. (2011). *Combinatorial Machine Learning – A Rough Set Approach,* volume 360 of *Studies in Computational Intelligence*. Heidelberg: Springer.

Muller, W., & Wiederhold, E. (2002). Applying decision tree methodology for rules extraction under cognitive constraints. *European Journal of Operational Research*, *136*, 282–289.

Pawlak, Z. (1992). *Rough Sets: Theoretical Aspects of Reasoning About Data*. Norwell: Kluwer Academic Publishers.

Pawlak, Z. (1997). Rough set approach to knowledge-based decision support. *European Journal of Operational Research*, *99*, 48–57.

Pawlak, Z., & Slowinski, R. (1994). Rough set approach to multi-attribute decision analysis. *European Journal of Operational Research*, *72*, 443–459.

Skowron, A., & Suraj, Z. (1993). Rough sets and concurrency. *Bulletin of the Polish Academy of Sciences*, *41*, 237–254.

Slowinski, R. (1992). *Intelligent Decision Support: Handbook of Applications and Advances of the Rough Sets Theory*. Norwell: Kluwer Academic Publishers.

Suraj, Z. (2001). Some remarks on extensions and restrictions of information systems. In *Rough Sets and Current Trends in Computing, Second International Conference, RSCTC 2000, Revised Papers* (pp. 204–211). Springer.

Wieczorkowska, A., Synak, P., Lewis, R. A., & Ras, Z. W. (2005). Extracting emotions from music data. In *Foundations of Intelligent Systems, 15th International Symposium, ISMIS 2005* (pp. 456–465). Springer.

Zhou, Z.-H., Jiang, K., & Li, M. (2005). Multi-instance learning based web mining. *Appl. Intell.*, *22*, 135–147.