Bi-Criteria Optimization of Decision Trees with Applications to Data Analysis

Igor Chikalov, Shahid Hussain, Mikhail Moshkov

PII: S0377-2217(17)30934-7
DOI: 10.1016/j.ejor.2017.10.021
Reference: EOR 14745

To appear in: *European Journal of Operational Research*

Received date: 26 June 2016
Revised date: 9 October 2017
Accepted date: 10 October 2017

Please cite this article as: Igor Chikalov, Shahid Hussain, Mikhail Moshkov, Bi-Criteria Optimization of Decision Trees with Applications to Data Analysis, *European Journal of Operational Research* (2017), doi: 10.1016/j.ejor.2017.10.021

Highlights

- Algorithms for bi-criteria optimization of decision trees are created.
- We compare twenty greedy heuristics for bi-criteria optimization of decision trees.
- We discover very small number of Pareto Optimal Points for an optimization problem.

Bi-Criteria Optimization of Decision Trees with Applications to Data Analysis

Igor Chikalov[a], Shahid Hussain*[b], Mikhail Moshkov[a]

[a] *Computer, Electrical and Mathematical Sciences and Engineering Division, King Abdullah University of Science and Technology, Thuwal 23955-6900, Saudi Arabia*
[b] *School of Science and Engineering, Habib University, Karachi, Pakistan*

Abstract

This paper is devoted to the study of bi-criteria optimization problems for decision trees. We consider different cost functions such as depth, average depth, and number of nodes. We design algorithms that allow us to construct the set of Pareto optimal points (POPs) for a given decision table and the corresponding bi-criteria optimization problem. These algorithms are suitable for investigation of medium-sized decision tables. We discuss three examples of applications of the created tools: the study of relationships among depth, average depth and number of nodes for decision trees for corner point detection (such trees are used in computer vision for object tracking), study of systems of decision rules derived from decision trees, and comparison of different greedy algorithms for decision tree construction as single- and bi-criteria optimization algorithms.

Keywords: Multiple criteria analysis, bi criteria optimization, Dynamic programming, Decision trees, Pareto optimal points, Heuristics

1. Introduction

Decision trees are widely used as classifiers [1, 9, 15] where a decision tree is considered as a model of a data and is used to predict the value of the decision attribute for a new object given by values of conditional attributes. In this paper, we concentrate on the two other areas of applications where decision trees are considered as algorithms for problem solving [21] and as tools for data mining and knowledge representation [31].

When decision trees are considered as algorithms, we are interested in minimizing the depth of decision trees (worst-case time complexity), average depth of decision trees (average-case time complexity), and number of nodes in decision

*Corresponding author
Email addresses: igor.chikalov@gmail.com (Igor Chikalov), shahid.hussain@sse.habib.edu.pk (Shahid Hussain*), mikhail.moshkov@kaust.edu.sa (Mikhail Moshkov)

trees (space complexity). It is natural to study not only single- but also bi-criteria optimization of decision trees. Note that the question about space-time tradeoff in universal models of computation as well as non-universal models such as branching programs and decision trees was studied by many authors [6, 8].

When decision trees are considered as tools for knowledge representation the goal is to have a comprehensive tree. The standard way is to try to construct a decision tree with minimum number of nodes. However, the depth of decision trees is also important since we need to understand conjunctions of conditions corresponding to paths in decision trees from the root to terminal nodes.

Another example of decision tree usage, where bi-criteria optimization is required, is connected with induction of decision rules from the decision trees [25, 30]: the number of rules is equal to the number of terminal nodes in the tree and the maximum length of a rule is equal to the depth of the tree.

The mentioned single- and bi-criteria optimization problems are complicated. In particular, the following problems are NP-hard for exact decision trees: minimization of the average depth [17], minimization of the depth and number of nodes (see, for example [11]), minimization of the number of terminal/nonterminal nodes (follow directly from the results obtained in [11]). Based on results from [35] it is possible to prove that the problem of minimization of the depth of approximate decision trees is also NP-hard.

Using extensions of the dynamic programming, we can solve the considered bi-criteria optimization problems in a way which is better than the exhaustive search. We construct a directed acyclic graph (DAG) such that the nodes of graph are subtables of the initial decision table given by conditions "`attribute = value`". For each node (subtable) of the DAG, we construct the set of Pareto optimal points (POPs) for the considered bi-criteria optimization problem. We also show how to describe relationships between two criteria using the constructed set of POPs.

The main peculiarities of the considered bi-criteria optimization problems are the following: the number of decision trees under consideration can be huge but the number of POPs for the mentioned cost functions is comparable with the size of the input decision table. In particular, if one of the cost functions is the depth of decision trees, then the number of POPs is at most the number of conditional attributes in the input table plus one. For example, for the decision table BREAST-CANCER from UCI Machine Learning Repository [18] (see Table 2) with 266 rows and 10 conditional attributes, the number of exact decision trees is equal to 7.3×10^{88}, but the number of POPs for the bi-criteria optimization problem related to the depth and number of terminal nodes is equal to one.

The designed algorithms which were adapted to these peculiarities allow us to construct the set of POPs for medium-sized decision tables. For example, the largest decision table BOX-20 used in experiments described in this paper contains 16 conditional attributes and 81,900 rows (see Table 1). Experiments described in this paper (construction of the set of Pareto optimal points for a decision table) required usually few minutes and, rarely, few hours. In the worst case, these algorithms have exponential time complexity. However, in [23, 22] the classes of decision tables over so-called restricted information systems were

described for which the considered algorithms have polynomial time complexity depending on the size of decision tables.

Another limitation of the designed algorithms is that they work with decision tables containing only categorical attributes, and these attributes are used as they are: if a node in a decision tree is labeled with an attribute then this node has an outgoing edge for each possible value of the considered attribute.

We discuss three examples of applications of the constructed algorithms. First, we study the set of Pareto optimal points and corresponding tradeoffs for each pair of cost functions depth, average depth and number of nodes for the corner point detection problem. This problem is known in computer vision and is connected with tracking of objects [32, 33]. Earlier, we have shown that the average depth of decision trees for this problem constructed by dynamic programming algorithm is, on the average, 7% less in comparison with the best results obtained by greedy algorithms [3].

The second example of applications of the algorithms presented in this paper is devoted to the study of decision rule systems derived from decision trees [25, 30]: we study relationships between depth and number of terminal nodes in decision trees which correspond to relationships between maximum length and number of rules in derived systems.

The first two applications should be considered mainly as illustrative examples since we can work efficiently only with medium-sized decision tables. However, we found a number of interesting results. In particular, in many cases the set of POPs contains the only one point. It means that there exists a decision tree which is optimal relative to the both criteria simultaneously (totally optimal decision tree relative to these criteria).

The third example of applications is connected with the evaluation of 20 approximate greedy algorithms for decision tree construction as algorithms for single- and bi-criteria optimization. These algorithms are essentially faster than dynamic programming algorithms and have polynomial time complexity depending on the size of decision tables. If we consider single-criterion optimization, it is easy to compare heuristics by comparing the obtained results. For the case of two criteria, the results obtained by heuristics can be incomparable. However, if we know the set of POPs, we can find the minimum distance from a point corresponding to a heuristic to a POP and compare heuristics based on these distances. It is easy to derive minimum values of cost functions from the set of POPs. As a result, we can compare not only the costs of trees constructed by heuristics but also average relative differences of costs of trees constructed by heuristics and optimal trees. It is interesting that, in some cases, the best for bi-criteria optimization heuristic is not good for single-criterion optimization.

The usual dynamic programming approach for single-criterion optimization of decision trees was created earlier by Garey [14] (see also [19, 34]). In [10, 16], we considered relationships for different pairs of cost functions for decision trees separately with individual algorithm for each pair of cost functions. The papers of other authors devoted to bi-criteria optimization of decision trees consider mainly the classification accuracy and the size of decision trees (often training error and number of nodes, respectively) [7]. Most of the bi-criteria

optimization algorithms are evolutionary in nature [5, 26, 36] or are based on related techniques such as particle swarm optimization [12]. These and other approaches [13] do not guarantee the construction of the set of POPs, they even cannot guarantee that the constructed points are Pareto optimal.

The novelty of this paper lies in (i) the creation of algorithms for bi-criteria optimization of decision trees, (ii) the comparison of 20 greedy heuristics as algorithms for bi-criteria optimization of decision trees, and (iii) the discovery of abnormally small number of POPs for bi-criteria optimization problem for decision trees relative to the depth and number of terminal nodes.

The created tools can be used for the study of complexity of algorithms represented in the form of decision trees [2] and for the study of decision trees as a way for knowledge representation. In particular, these tools can be useful in the rough set theory and its applications [27, 28, 29] where decision trees are widely used.

This paper consists of nine sections and appendix. In Section 2, we discuss the notions of decision tables, uncertainty measures, decision trees, and cost functions for decision trees. Section 3 is devoted to the study of the algorithm for construction of DAG. In Section 4, we discuss two auxiliary algorithms for processing of POPs. Section 5 is devoted to the study of bi-criteria optimization problems for decision trees. We discuss time complexity of the two main algorithms in Section 6. In Sections 7 and 8, we consider three examples of applications and conclude the paper in Section 9. The appendix contains proofs of some statements presented in the paper.

2. Decision Tables and Decision Trees

In this section, we consider the notions of decision tables, uncertainty measures, decision trees, and cost functions for decision trees.

2.1. Decision Tables

In this section, we discuss the notions of decision table and separable subtable of decision table, and consider some parameters of decision tables including their size.

A *decision table* is a rectangular table T with $n \geq 1$ columns filled with numbers from the set $\omega = \mathsf{N} \cup \{0\} = \{0, 1, 2, \ldots\}$ 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 is labeled with a number from ω which is interpreted as a decision (a value of the *decision attribute* d). Rows of the table are interpreted as tuples of values of conditional attributes.

A decision table is called *empty* if it has no rows. We denote by $\mathcal{T}$ the set of all decision tables and by $\mathcal{T}^+$, the set of nonempty decision tables. Let $T \in \mathcal{T}$, the table T is called *degenerate* if it is empty or all rows of T are labeled with the same decision. Let $D(T)$ be the set of decisions attached to rows of T. We denote by $N(T)$, the number of rows in the table T, and for any $t \in \omega$, we denote by $N_t(T)$ the number of rows of T labeled with the decision t. By $mcd(T)$

we denote the *most common decision* for T which is the minimum decision t_0 from $D(T)$ such that $N_{t_0}(T) = \max\{N_t(T) : t \in D(T)\}$. 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 for which $|E(T, f_i)| \geq 2$.

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.

A decision table T 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 to each row corresponding decision as the last number in the row). The length of this word will be called the *size* of the decision table T.

2.2. Uncertainty Measures

In this section, we consider the notion of uncertainty measure and five examples of uncertainty measures. We use uncertainty measures to define the notion of approximate decision trees (see Section 2.3) and the notion of an impurity function which determines greedy algorithms for decision tree construction (see Section 8).

Let R be the set of real numbers and R_+ be the set of all nonnegative real numbers. An *uncertainty measure* is a function $U : \mathcal{T} \to \mathsf{R}$ such that $U(T) \geq 0$ for any $T \in \mathcal{T}$, and $U(T) = 0$ if and only if T is a degenerate table. One can show that the following functions (we assume that, for any empty table, the value of each of the considered functions is equal to 0) are uncertainty measures:

- *Misclassification error:* $me(T) = N(T) - N_{mcd(T)}(T)$.
- *Relative misclassification error:* $rme(T) = (N(T) - N_{mcd(T)}(T))/N(T)$.
- *Entropy:* $ent(T) = -\sum_{t \in D(T)} \frac{N_t(T)}{N(T)} \log_2 \frac{N_t(T)}{N(T)}$.
- *Gini index:* $gini(T) = 1 - \sum_{t \in D(T)} \left(\frac{N_t(T)}{N(T)}\right)^2$.
- Function rt where $rt(T)$ is the number of unordered pairs of rows of T labeled with different decisions (note that $rt(T) = N(T)^2 gini(T)/2$).

In this paper, we assume that each operation with numbers $\max(x, y)$, $x+y$, $x-y$, $x \times y$, $x \div y$, $\log_2 x$ has time complexity $O(1)$. This assumption is reasonable for computations with floating-point numbers. Under this assumption, each of the considered five uncertainty measures has polynomial time complexity depending on the size of decision tables.

2.3. Decision Trees

In this section, we consider the notions of exact and approximate decision trees for decision tables, and discuss the structure of the set of approximate 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 (see Figure 3).

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. We define now a subtable $T(v) = T_\Gamma(v)$ of the table T. If v is the root of Γ then $T(v) = T$. Let v be not the root of Γ and $v_1, e_1, \ldots, v_m, e_m, v_{m+1} = v$ 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 $T(v) = T(f_{i_1}, a_1) \ldots (f_{i_m}, a_m)$.

We now define the notion of a decision tree for a decision table which covers all decision trees under consideration including exact and approximate decision trees.

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 either v is a terminal node labeled with $mcd(T(v))$, or 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.

Let U be an uncertainty measure and $\alpha \in \mathsf{R}_+$. A decision tree Γ over T is called a (U, α)-*decision tree for* T if, for any node v of Γ,

- If $U(T(v)) \leq \alpha$ then v is a terminal node which is labeled with $mcd(T(v))$.
- If $U(T(v)) > \alpha$ then v is a nonterminal node 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 which are labeled with $a_1, \ldots, a_t$, respectively.

We denote by $DT_{U,\alpha}(T)$ the set of (U, α)-decision trees for T. It is easy to show that $DT_{U,\alpha}(T) \subseteq DT(T)$.

If $\alpha > 0$ then we have the notion of an approximate decision tree. For $\alpha = 0$, we obtain the notion of exact decision tree which does not depend on the considered uncertainty measure U.

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

by $tree(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t)$ 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 $a_1, \ldots, a_t$ and 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:

$$DT_{U,\alpha}(T, f_i) = \{tree(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t) : \Gamma_j \in DT_{U,\alpha}(T(f_i, a_j)), j = 1, \ldots, t\}.$$

The next statement allows us to understand the structure of the set $DT_{U,\alpha}(T)$. We will use this statement in Section 3 to study the set of decision trees represented by a directed acyclic graph $\Delta_{U,\alpha}(T)$. The proof of this statement can be found in the appendix.

Proposition 1. *Let U be an uncertainty measure, $\alpha \in \mathsf{R}_+$, and T be a decision table. Then*

$$DT_{U,\alpha}(T) = \begin{cases} \{tree(mcd(T))\}, & \text{if } U(T) \le \alpha \\ \bigcup_{f_i \in E(T)} DT_{U,\alpha}(T, f_i), & \text{if } U(T) > \alpha. \end{cases}$$

2.4. Cost Functions for Decision Trees

In this section, we discuss the notion of a cost function for decision trees, and consider five examples of cost functions related to the time complexity of decision trees in the worst and average cases and to its space complexity. Each cost function will be defined both in "natural" and in "inductive" way which is important for the considered later algorithms for bi-criteria optimization.

Let n be a natural number. We consider a partial order $\le$ on the set R_+^n: $(x_1, \ldots, x_n) \le (y_1, \ldots, y_n)$ if $x_1 \le y_1, \ldots, x_n \le y_n$. A function $g : \mathsf{R}_+^n \to \mathsf{R}_+$ is called *increasing* if $g(x) \le g(y)$ for any $x, y \in \mathsf{R}_+^n$ such that $x \le y$. For example, $\max(x_1, x_2)$ and $x_1 + x_2$ are increasing functions.

Let f be a function from R_+^2 to R_+. We can extend f to a function with arbitrary number of variables in the following way: $f(x_1) = x_1$ and if $n > 2$ then $f(x_1, \ldots, x_n) = f(f(x_1, \ldots, x_{n-1}), x_n)$. It is not difficult to prove the following statement.

Lemma 2. *Let f be an increasing function from R_+^2 to R_+. Then, for any natural n, the function $f(x_1, \ldots, x_n)$ is increasing.*

A cost function for decision trees is a function $\psi(T, \Gamma)$ which is defined on pairs of decision table T and a decision tree Γ for T, and has values from R_+. The function ψ is given by three functions $\psi^0 : \mathcal{T} \to \mathsf{R}_+$, $F : \mathsf{R}_+^2 \to \mathsf{R}_+$ and $w : \mathcal{T} \to \mathsf{R}_+$. The value $\psi(T, \Gamma)$ 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. We now consider examples of some increasing cost functions for decision trees:

- *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 $w(T) = 1$.
- *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 $Row(T)$ is the set of rows of T, and $l_\Gamma(r)$ 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 $w(T) = N(T)$. Let T be a nonempty decision table. The value $tpl(T,\Gamma)/N(T)$ is the *average depth* $h_{\mathrm{avg}}(T,\Gamma)$ 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 $w(T) = 1$.
- *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 $w(T) = 1$.
- *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 $w(T) = 0$.

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

3. Directed Acyclic Graph $\Delta_{U,\alpha}(T)$

In this section, we study a directed acyclic graph $\Delta_{U,\alpha}(T)$ which allows us to describe and investigate the set of (U,α)-decision trees for a decision table T.

Let T be a nonempty decision table with n conditional attributes $f_1,\dots,f_n$, U be an uncertainty measure, and $\alpha \in \mathsf{R}_+$. We now consider algorithm $\mathcal{A}_1$ for the construction of a directed acyclic graph $\Delta_{U,\alpha}(T)$ which will be used for the description and study 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. The algorithm $\mathcal{A}_1$ can be considered also as a definition of the graph $\Delta_{U,\alpha}(T)$.

Algorithm $\mathcal{A}_1$

Input:	A nonempty decision table T with n conditional attributes $f_1,\dots,f_n$, an uncertainty measure U, and a number $\alpha \in \mathsf{R}_+$.
Output:	Directed acyclic graph $\Delta_{U,\alpha}(T)$.

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 the algorithm is finished. Return the resulting graph as $\Delta_{U,\alpha}(T)$. Otherwise, choose a node (table) Θ that has not been processed yet.
3. If $U(\Theta) \leq \alpha$ mark the node Θ as processed and proceed to the step 2.
4. If $U(\Theta) > \alpha$ then, for each $f_i \in E(\Theta)$, draw a bundle of edges from the node Θ. Let $E(\Theta, f_i) = \{a_1, \ldots, a_k\}$. Then draw k edges from Θ and label these edges with 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 proceed to the step 2.

A node Θ of the graph $\Delta_{U,\alpha}(T)$ is called *terminal* if there are no outgoing edges from this node. It is easy to see that the node Θ is terminal if and only if $U(\Theta) \leq \alpha$.

For each node Θ of the graph $\Delta_{U,\alpha}(T)$, we define the set $Tree(\Delta_{U,\alpha}(T), \Theta)$ of decision trees corresponding to this node in the following way. If Θ is a terminal node of $\Delta_{U,\alpha}(T)$, then $Tree(\Delta_{U,\alpha}(T), \Theta) = \{tree(mcd(\Theta))\}$. Let Θ be a nonterminal node of $\Delta_{U,\alpha}(T)$, $f_i \in E(\Theta)$, and $E(\Theta, f_i) = \{a_1, \ldots, a_t\}$. We denote:

$$\begin{aligned} Tree(\Delta_{U,\alpha}(T), \Theta, f_i) \quad = \quad & \{tree(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t) : \\ & \Gamma_j \in Tree(\Delta_{U,\alpha}(T), \Theta(f_i, a_j)), j = 1, \ldots, t\}. \end{aligned}$$

Then,

$$Tree(\Delta_{U,\alpha}(T), \Theta) = \bigcup_{f_i \in E(T)} Tree(G, \Theta, f_i).$$

The next statement shows that, for each node Θ of the the graph $\Delta_{U,\alpha}(T)$, the set $Tree(\Delta_{U,\alpha}(T), \Theta)$ of decision trees corresponding to the node Θ is equal to the set $DT_{U,\alpha}(\Theta)$ of (U, α)-decision trees for the table Θ. It means that the graph $\Delta_{U,\alpha}(T)$ describes the set of (U, α)-decision trees for the table T. The proof of this statement can be found in the appendix.

Proposition 3. *Let U be an uncertainty measure, $\alpha \in \mathsf{R}_+$, and T be a decision table. Then, for any node Θ of the graph $\Delta_{U,\alpha}(T)$, the following equality holds:*

$$Tree(\Delta_{U,\alpha}(T), \Theta) = DT_{U,\alpha}(\Theta).$$

4. Two Algorithms for Processing of Pareto Optimal Points

In this section, we consider two auxiliary algorithms $\mathcal{A}_2$ and $\mathcal{A}_3$ which are used for the construction of Pareto optimal points for bi-criteria optimization problems related to decision trees (see algorithm $\mathcal{A}_4$).

Let R^2 be the set of pairs of real numbers (*points*). We consider a partial order $\leq$ on the set R^2: $(c, d) \leq (a, b)$ if $c \leq a$ and $d \leq b$. Two points α and β are *comparable* if $\alpha \leq \beta$ or $\beta \leq \alpha$. A subset of R^2 in which no two different points are comparable is called an *antichain*. We will write $\alpha < \beta$ if $\alpha \leq \beta$ and $\alpha \neq \beta$. If α and β are comparable then $\min(\alpha, \beta) = \alpha$ if $\alpha \leq \beta$ and $\min(\alpha, \beta) = \beta$ if $\alpha > \beta$.

Let A be a nonempty finite subset of R^2. A point $\alpha \in A$ is called a *Pareto optimal point for* A if there is no point $\beta \in A$ such that $\beta < \alpha$. We denote by $Par(A)$ the set of Pareto optimal points for A. It is clear that $Par(A)$ is an antichain.

Points from $Par(A)$ can be ordered in the following way: $(a_1, b_1), \ldots, (a_t, b_t)$ where $a_1 < \cdots < a_t$. Since points from $Par(A)$ are incomparable therefore, $b_1 > \cdots > b_t$. We will say the sequence $(a_1, b_1), \ldots, (a_t, b_t)$ is *normal representation of* $Par(A)$.

We now describe an algorithm which, for a given nonempty finite subset A of the set R^2, constructs the normal representation of the set $Par(A)$. We assume that A is a multiset containing, possibly, repeating elements.

Algorithm $\mathcal{A}_2$

Input: A nonempty finite subset A of the set R^2 containing, possibly, repeating elements (multiset).

Output: Normal representation P of the set $Par(A)$ of Pareto optimal points for A.

1. Construct a sequence B of all points from A ordered according to the first coordinate in the ascending order. Set P equal to the empty sequence.
2. If there is only one point in the sequence B, add this point to the end of the sequence P, return P, and finish the work of the algorithm. Otherwise, choose the first $\alpha = (\alpha_1, \alpha_2)$ and the second $\beta = (\beta_1, \beta_2)$ points from B.
3. If α and β are comparable then remove α and β from B, add the point $\min(\alpha, \beta)$ to the beginning of B, and proceed to the step 2.
4. If α and β are not comparable (in this case $\alpha_1 < \beta_1$ and $\alpha_2 > \beta_2$) then remove α from B, add the point α to the end of P, and proceed to the step 2.

The proof of the following statement can be found in the appendix.

Proposition 4. *Let A be a nonempty finite subset of the set R^2 containing, possibly, repeating elements (multiset). Then the algorithm $\mathcal{A}_2$ returns the normal representation of the set $Par(A)$ of Pareto optimal points for A.*

The algorithm $\mathcal{A}_2$ can construct the set $Par(A)$ if the cardinality of the set A is reasonable. The number of decision trees can be huge. However, the number of Pareto optimal points is bounded from above by a polynomial in the size of

the input decision table for the most interesting cost functions. In particular, if one of the considered cost functions is the depth of decision tree then the number of Pareto optimal points is at most the number of conditional attributes plus one. In such situation, instead of the algorithm $\mathcal{A}_2$ we use a special technique based on the construction of the set of Pareto optimal points for the set of (U, α)-decision trees for each node (subtable) Θ of the graph $\Delta_{U,\alpha}(T)$.

It is easy to construct the only Pareto optimal point for a terminal node Θ of the graph $\Delta_{U,\alpha}(T)$. Let Θ be a nonterminal node of $\Delta_{U,\alpha}(T)$. First, for each attribute f_i which is not constant on Θ, we construct the set of Pareto optimal points for the set of (U, α)-decision trees for Θ in which the root is labeled with the attribute f_i. After that, we construct the union of these sets for all non-constant attributes f_i and apply the algorithm $\mathcal{A}_2$ to this union.

To construct the set of Pareto optimal points for the set of (U, α)-decision trees for Θ in which the root is labeled with the attribute f_i, we apply the algorithm $\mathcal{A}_3$. This algorithm "merges" step by step the sets of Pareto optimal points corresponding to the nodes of the kind $\Theta(f_i, a)$ where a is a value of f_i in Θ. We apply the algorithm $\mathcal{A}_2$ after each step to keep the reasonable size of the constructed set of points. The increasing functions F, H mentioned in the description of the algorithm $\mathcal{A}_3$ correspond to the considered cost functions, and the sets $Par(P_1), \ldots, Par(P_t)$ correspond to the sets of Pareto optimal points for the nodes of the kind $\Theta(f_i, a)$ where a is a value of f_i in Θ.

Let F, H be increasing functions from R^2 to R, and A, B be nonempty finite subsets of the set R^2. We denote by $A \langle FH \rangle B$ the set $\{(F(a,c), H(b,d)) : (a,b) \in A, (c,d) \in B\}$.

Let $P_1, \ldots, P_t$ be nonempty finite subsets of R^2, $Q_1 = P_1$, and, for $i = 2, \ldots, t$, $Q_i = Q_{i-1} \langle FH \rangle P_i$. We assume that, the sets $Par(P_1), \ldots, Par(P_t)$ are already constructed for $i = 1, \ldots, t$. We now describe an algorithm that constructs the sets $Par(Q_1), \ldots, Par(Q_t)$ and returns $Par(Q_t)$.

Algorithm $\mathcal{A}_3$

Input: Increasing functions F, H from R^2 to R and the sets $Par(P_1), \ldots, Par(P_t)$ for nonempty finite subsets $P_1, \ldots, P_t$ of R^2.

Output: The set $Par(Q_t)$ where $Q_1 = P_1$, and, for $i = 2, \ldots, t$, $Q_i = Q_{i-1} \langle FH \rangle P_i$.

1. Set $B_1 = Par(P_1)$ and $i = 2$.
2. Construct the multiset

$$\begin{aligned} A_i &= B_{i-1} \langle FH \rangle Par(P_i) \\ &= \{(F(a,c), H(b,d)) : (a,b) \in B_{i-1}, (c,d) \in Par(P_i)\} \end{aligned}$$

   (we will not remove equal pairs from the constructed set).
3. Using algorithm $\mathcal{A}_2$ construct the set $B_i = Par(A_i)$. If $i = t$ then return B_i and finish the work of the algorithm. Otherwise, set $i = i + 1$ and proceed to the step 2.

The proof of the next statement can be found in the appendix.

Proposition 5. *Let F, H be increasing functions from R^2 to R, $P_1, \ldots, P_t$ be nonempty finite subsets of R^2, $Q_1 = P_1$, and, for $i = 2, \ldots, t$,*

$$Q_i = Q_{i-1} \langle FH \rangle P_i.$$

Then the algorithm $\mathcal{A}_3$ returns the set $Par(Q_t)$.

5. Study of Bi-Criteria Optimization Problems

In this section, we consider an algorithm $\mathcal{A}_4$ which constructs the sets of Pareto optimal points for bi-criteria optimization problems for decision trees. This algorithms is based on the algorithms $\mathcal{A}_2$ and $\mathcal{A}_3$. We also show how the constructed set of Pareto optimal points can be transformed into the graph of a function which describes the relationship between the considered cost functions.

5.1. Construction of the Set of Pareto Optimal Points

Let ψ and φ be increasing cost functions for decision trees given by triples of functions ψ^0, F, w and φ^0, H, u, respectively, U be an uncertainty measure, $\alpha \in \mathsf{R}_+$, T be a decision table with n conditional attributes $f_1, \ldots, f_n$, and $G = \Delta_{U,\alpha}(T)$.

For each node Θ of the graph G, we denote

$$t_{\psi,\varphi}(G, \Theta) = \{(\psi(\Theta, \Gamma), \varphi(\Theta, \Gamma)) : \Gamma \in \mathit{Tree}(G, \Theta)\}.$$

Note that, by Proposition 3, the set $\mathit{Tree}(G, \Theta)$ is equal to the set of (U, α)-decision trees for Θ. In other words, $t_{\psi,\varphi}(G, \Theta)$ is the set of points obtained in the following way: for each (U, α)-decision tree Γ for the table Θ, we construct the point which coordinates are values of the cost functions ψ and φ for the decision tree Γ and the table Θ. We denote by $Par(t_{\psi,\varphi}(G, \Theta))$ the set of Pareto optimal points for $t_{\psi,\varphi}(G, \Theta)$.

We now describe an algorithm $\mathcal{A}_4$ which constructs the set $Par(t_{\psi,\varphi}(G, T))$. In fact, this algorithm constructs, for each node Θ of the graph G, the set $B(\Theta) = Par(t_{\psi,\varphi}(G, \Theta))$.

Algorithm $\mathcal{A}_4$

Input:	Increasing cost functions for decision trees ψ and φ given by triples of functions ψ^0, F, w and φ^0, H, u, respectively, a decision table T with n conditional attributes $f_1, \ldots, f_n$, and the graph $G = \Delta_{U,\alpha}(T)$ where U is an uncertainty measure and $\alpha \in \mathsf{R}_+$.
Output:	The set $Par(t_{\psi,\varphi}(G, T))$ of Pareto optimal points for the set of pairs $t_{\psi,\varphi}(G, T) = \{(\psi(T, \Gamma), \varphi(T, \Gamma)) : \Gamma \in \mathit{Tree}(G, T)\}$.

1. If all nodes in G are processed, then return the set $B(T)$. Otherwise, choose a node Θ in the graph G which is not processed yet and which is either a terminal node of G or a nonterminal node of G such that, for any $f_i \in E(\Theta)$ and any $a_j \in E(\Theta, f_i)$, the node $\Theta(f_i, a_j)$ is already processed, i.e., the set $B(\Theta(f_i, a_j))$ is already constructed.
2. If Θ is a terminal node, then set $B(\Theta) = \{(\psi^0(\Theta), \varphi^0(\Theta))\}$. Mark the node Θ as processed and proceed to the step 1.
3. If Θ is a nonterminal node then, for each $f_i \in E(\Theta)$, apply the algorithm $\mathcal{A}_3$ to the functions F, H and the sets $B(\Theta(f_i, a_1)), \ldots, B(\Theta(f_i, a_t))$, where $\{a_1, \ldots, a_t\} = E(\Theta, f_i)$. Set $C(\Theta, f_i)$ the output of the algorithm $\mathcal{A}_3$ and
$$\begin{aligned} B(\Theta, f_i) &= C(\Theta, f_i) \langle ++ \rangle \{(w(\Theta), u(\Theta))\} \\ &= \{(a + w(\Theta), b + u(\Theta)) : (a, b) \in C(\Theta, f_i)\}. \end{aligned}$$
4. Construct the multiset $A(\Theta) = \bigcup_{f_i \in E(\Theta)} B(\Theta, f_i)$ by simple transcription of elements from the sets $B(\Theta, f_i)$, $f_i \in E(\Theta)$. Apply to the obtained multiset $A(\Theta)$ the algorithm $\mathcal{A}_2$ which constructs the set $Par(A(\Theta))$. Set $B(\Theta) = Par(A(\Theta))$. Mark the node Θ as processed and proceed to the step 1.

The proof of the following theorem can be found in the appendix.

Theorem 6. *Let ψ and φ be increasing cost functions for decision trees given by triples of functions ψ^0, F, w and φ^0, H, u, respectively, U be an uncertainty measure, $\alpha \in \mathsf{R}_+$, T be a decision table with n conditional attributes $f_1, \ldots, f_n$, and $G = \Delta_{U,\alpha}(T)$. Then, for each node Θ of the graph G, the algorithm $\mathcal{A}_4$ constructs the set $B(\Theta) = Par(t_{\psi,\varphi}(G, \Theta))$.*

5.2. Relationship between Two Cost Functions

Let ψ and φ be increasing cost functions for decision trees, U be an uncertainty measure, $\alpha \in \mathsf{R}_+$, T be a decision table, and $G = \Delta_{U,\alpha}(T)$.

To study the relationship between cost functions ψ and φ on the set of decision trees $Tree(G, T)$ we consider partial function $\mathcal{F}^{\psi,\varphi}_{G,T} : \mathsf{R} \to \mathsf{R}$ defined as follows:
$$\mathcal{F}^{\psi,\varphi}_{G,T}(x) = \min\{\varphi(T, \Gamma) : \Gamma \in Tree(G, T), \psi(T, \Gamma) \leq x\}.$$

The proof of the next statement is in the appendix.

Proposition 7. *Let $(a_1, b_1), \ldots, (a_k, b_k)$ be the normal representation of the set $Par(t_{\psi,\varphi}(G, T))$, where $a_1 < \cdots < a_k$ and $b_1 > \cdots > b_k$. Then, for any $x \in \mathsf{R}$, $\mathcal{F}^{\psi,\varphi}_{G,T}(x) = \mathcal{F}(x)$ where*
$$\mathcal{F}(x) = \begin{cases} \textit{undefined}, & x < a_1 \\ b_1, & a_1 \leq x < a_2 \\ \ldots & \ldots \\ b_{t-1}, & a_{t-1} \leq x < a_t \\ b_t, & a_t \leq x \end{cases}.$$

6. On Complexity of Algorithms $\mathcal{A}_1$ and $\mathcal{A}_4$

In this section, we consider uncertainty measures from the set $\{me, rme, ent, gini, rt\}$. Each of these uncertainty measures has polynomial time complexity depending on the size of decision tables. We consider here cost functions from the set $\{h, tpl, L, L_n, L_t\}$. Each of these cost functions is given by three functions ψ^0, F and w such that $F \in \{x + y, \max(x, y)\}$ and functions ψ^0 and w have polynomial time complexity depending on the size of decision tables.

Let T be a decision table, $G = \Delta_{U,\alpha}(T)$ where $U \in \{me, rme, ent, gini, rt\}$ and $\alpha \in \mathsf{R}_+$, and $\psi, \varphi \in \{h, tpl, L, L_n, L_t\}$.

To construct the set $Par(t_{\psi,\varphi}(G, T))$ of Pareto optimal points for (U, α)-decision trees for T relative to ψ and φ, we apply algorithm $\mathcal{A}_1$ to T, U and α and construct the graph $G = \Delta_{U,\alpha}(T)$. After that, we apply the algorithm $\mathcal{A}_4$ to T, G, ψ and φ and construct the set $Par(t_{\psi,\varphi}(G, T))$.

One can show that the time complexity of each of algorithms $\mathcal{A}_1$ and $\mathcal{A}_4$ 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.

In [23, 22] infinite sets of attributes (so-called restricted information systems) were studied for each of which the number of separable subtables in decision tables over the considered set of attributes is bounded from above by a polynomial on the number of attributes in the table. For decision tables over such set of attributes, the time complexity of each of algorithms $\mathcal{A}_1$ and $\mathcal{A}_4$ is bounded from above by a polynomial on the size of the input table.

Let us consider a family of restricted information systems. Let d and t be natural numbers, $f_1, \ldots, f_t$ be functions from R^d to R, and s be a function from R to $\{0, 1\}$ such that $s(x) = 0$ if $x < 0$ and $s(x) = 1$ if $x \geq 0$. Then the information system with the universe R^d and the set of attributes $\{s(f_i + c) : i = 1, \ldots, t, c \in \mathsf{R}\}$ is a 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 R^d which is usual for decision trees for datasets with numerical attributes only [9].

In general case, algorithms $\mathcal{A}_1$ and $\mathcal{A}_4$ have exponential time complexity depending on the size of the input tables. However, the considered algorithms are applicable to medium-sized decision tables which allows us to use them for data analysis (see next two sections).

7. Illustrative Examples

In this section, we consider two illustrative examples of applications of the designed algorithms: we study relationships among different parameters of decision trees for corner point detection, and investigate systems of decision rules derived from decision trees. It is impossible to consider these examples as real applications since our bi-criteria optimization algorithms can work only with medium-sized decision tables.

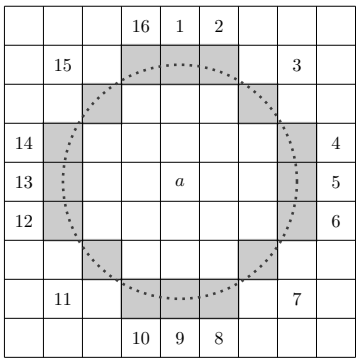


Figure 1: Bresenham circle of radius 3 surrounding pixel a

7.1. Decision Trees for Corner Point Detection

Object tracking (locating an object and determining its position) in a video stream is an important problem in computer vision. One of the approaches to solve this problem is based on the use of distinctive points located on the image. FAST algorithm devised by Rosten and Drummond [32, 33] is an example of such a solution. This algorithm iterates through all pixels and detects *corner points* by comparing the intensity of the current pixel and surrounding pixels.

In order to determine if an image pixel a is a corner point, a circle of 16 pixels (a Bresenham circle of radius 3) surrounding a is examined (see Figure 1). The intensity of each pixel of the circle is compared with the intensity of a. The pixel a is assumed to be a corner point if at least 12 contiguous pixels on the circle are all either brighter or darker than a by a given threshold $\gamma > 0$.

The trivial algorithm for corner point detection requires the consideration of all 16 pixels from the Bresenham circle of radius 3 surrounding a given pixel. The decreasing of the number of considered pixels will lead to the decreasing of time complexity of the FAST algorithm. Rosten and Drummond [32, 33] proposed to use decision trees constructed by greedy algorithms as a way for corner point detection. Our earlier results [3] show that the average depth of decision trees for this problem constructed by dynamic programming algorithm is, on the average, 7% less in comparison with the best results obtained by greedy algorithms. For example, for the decision table BOX-20 corresponding to the set of images BOX and threshold $\gamma = 20$, the dynamic programming algorithm constructs a decision tree with minimum average depth equals to 3.378. In this section, we study the set of Pareto optimal points and corresponding tradeoffs for each pair of cost functions depth, average depth and number of nodes for decision trees for corner point detection.

For an arbitrary pixel a and for $i = 1, \ldots, 16$, $\phi_i(a)$ denotes the intensity of the i-th pixel in the circle surrounding a (ordering as shown in Figure 1) and $\phi(a)$ denotes the intensity of the pixel a. The pixel a can be represented as an

Decision table	# Rows	# POPs		
		(L, tpl)	(h, L)	(h, tpl)
BOX-20	81,900	280	1	2
BOX-50	10,972	37	3	2
BOX-100	680	1	1	2
JUNK-50	3,509	12	2	3
JUNK-70	980	1	2	2
MAZE-70	5,303	15	4	4
MAZE-100	1,343	1	2	3

Table 1: Corner point detection experiments: number of Pareto optimal points

object that is characterized by the attributes $f_1, \ldots, f_{16}$ where, for $i = 1, \ldots, 16$,

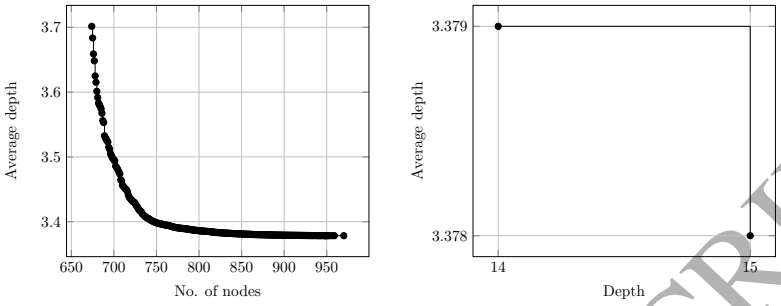
$$f_i(a) = \begin{cases} 0, & \text{if} \quad \phi(a) - \phi_i(a) > \gamma, \\ 1, & \text{if} \quad |\phi_i(a) - \phi(a)| \leq \gamma, \\ 2, & \text{if} \quad \phi_i(a) - \phi(a) > \gamma. \end{cases}$$

Then the problem of corner point detection for a given set of images can be represented as a decision table T with conditional attributes $f_1, \ldots, f_{16}$. The table T contains rows (tuples of attribute values) for all pixels from the considered images with the exception of outer boundaries. A row of T is labeled with the decision 1 if this tuple of attribute values describes a corner point, and the decision 0 otherwise.

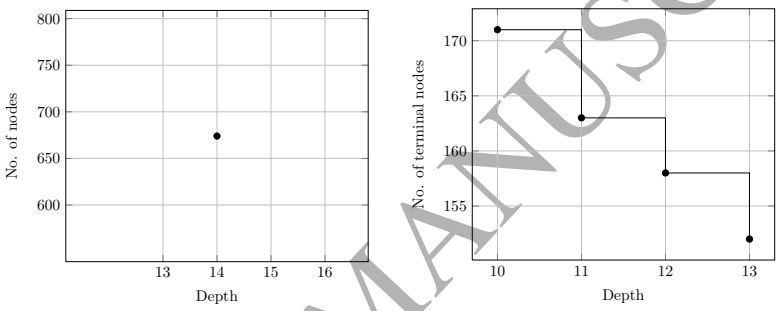
Study of relationships among different cost functions for decision trees for the table T allows us to find a decision tree for corner point detection problem which is adapted to the considered set of images and has appropriate values of cost functions.

In our experiments, we use decision tables corresponding to the sets of images BOX, MAZE, and JUNK considered in [33] (see also [3]) and different thresholds γ. For each such table T, we construct the set of Pareto optimal points $Par(t_{\psi,\varphi}(\Delta_{rme,0}(T), T))$ for the following pairs of cost functions (ψ, φ): (L, tpl), (h, tpl), and (h, L). Some results of experiments including the number of rows in the considered decision tables and the number of Pareto optimal points (POPs) for the three pairs of cost functions can be found in Table 1. The name of each table consists of the name of a set of images and a threshold γ.

Figure 2 shows in detail relationships (see Section 5.2) for the decision table BOX-20 and pairs of cost functions (L, h_{avg}), (h, h_{avg}), and (h, L) ($h_{\text{avg}}(T, \Gamma) = tpl(T, \Gamma)/N(T)$ for a decision tree Γ for the decision table T), and for the decision table BOX-40 and pair of cost functions (h, L_t), where the small filled circles are Pareto optimal points. The relationship for (h, h_{avg}) shows that we can decrease the depth of decision trees from 15 to 14 by increasing the average depth from 3.378 to 3.379 only. In the case (h, L), there is only one Pareto optimal point. It means that there exists a decision tree for corner point detection for BOX-20, which has simultaneously the minimum depth and the minimum number

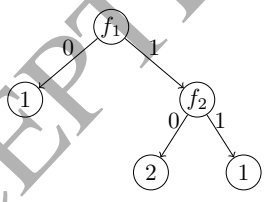


(a) BOX-20: (L, h_{avg}) (b) BOX-20: (h, h_{avg})



(c) BOX-20: (h, L) (d) BOX-40: – pair (h, L_t)

Figure 2: Relationships for decision trees for corner point detection



(a) Decision tree Γ

$$f_1 = 0 \rightarrow 1$$
$$f_1 = 1 \wedge f_2 = 0 \rightarrow 2$$
$$f_1 = 1 \wedge f_2 = 1 \rightarrow 1.$$

(b) System of decision rules for Γ

Figure 3: System of decision rules for given decision tree

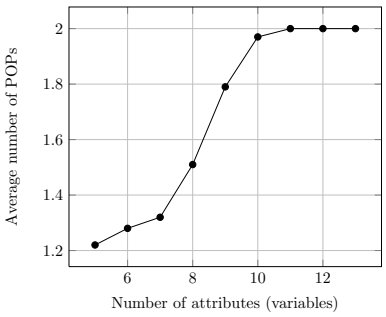


Figure 4: Average number of Pareto optimal points for decision tables corresponding to partial Boolean functions

Table 2: Number of Pareto optimal points for optimization relative to (h, L_t)

			Alpha										
Decision table	Rows	Attrs.	0	0.01	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
ADULT-STRETCH	16	5	1	1	1	1	1	1	1	1	1	1	1
BALANCE-SCALE	8,124	22	1	1	1	1	1	1	1	1	1	1	1
BREAST-CANCER	266	10	1	1	1	1	1	2	4	3	3	2	1
CARS	1,728	6	1	1	1	1	1	1	1	1	1	1	1
FLAGS	194	26	4	4	4	3	2	3	3	2	3	3	1
HAYES-ROTH-DATA	69	4	1	1	1	1	1	1	1	1	1	1	1
HOUSE-VOTES-84	279	16	1	1	3	1	2	1	1	1	1	1	1
LENSES	11	4	1	1	1	1	1	1	1	1	1	1	1
LYMPHOGRAPHY	148	18	3	3	3	3	2	2	3	1	1	1	1
MONKS-1-TEST	432	6	1	1	1	1	1	1	1	1	1	1	1
MONKS-1-TRAIN	124	6	1	1	1	1	1	1	2	2	1	1	1
MONKS-2-TEST	432	6	1	1	1	1	1	1	1	1	1	1	1
MONKS-2-TRAIN	169	6	2	2	2	2	2	2	2	2	2	2	1
MONKS-3-TEST	432	6	1	1	1	1	1	1	1	1	1	1	1
MONKS-3-TRAIN	122	6	2	2	2	2	2	2	2	1	1	1	1
MUSHROOM	8,124	22	2	1	1	1	2	3	2	2	2	1	1
NURSERY	12,960	8	1	1	1	1	1	2	2	1	1	1	1
SHUTTLE-LAND.CONT.	15	6	1	1	1	1	1	1	1	1	1	1	1
SOYBEAN-SMALL	47	35	1	1	1	1	1	1	1	1	2	2	1
SPECT-TEST	169	22	1	1	1	1	1	1	1	2	2	2	1
TEETH	23	8	1	1	1	1	1	1	1	1	2	2	1
TIC-TAC-TOE	958	9	2	2	2	2	2	2	2	1	2	2	1
ZOO-DATA	59	16	1	1	1	1	1	2	1	1	1	1	1

of nodes (a totally optimal decision tree relative to h and L). We find totally optimal decision trees relative to different pairs of cost functions in four more experiments (see Table 1). The relationship for (L, h_{avg}) shows that we can decrease the number of nodes in the decision trees from more than 950 to 750 by small increment in average depth (from 3.378 to 3.4). The study of the last pair (h, L_t) allows us to understand the relationship between the maximum length and the number of decision rules derived from decision trees for BOX-40 (see Section 7.2).

The obtained results show that the bi-criteria optimization algorithms can give us useful information about decision trees for corner point detection if the decision table corresponding to a collection of images has about hundred thousand of rows.

7.2. Systems of Decision Rules Derived from Decision Trees

Decision rules are widely used in machine learning and data mining. One of well known approaches to decision rule construction is to build a decision tree Γ and derive a decision rule from each path from the root to a terminal node of Γ [30, 20, 24] (see Figure 3).

It is clear that the number of constructed rules is equal to the number $L_t(\Gamma)$ of terminal nodes in Γ, and the maximum length of the constructed rule is equal to the depth $h(\Gamma)$ of Γ. In this section, we study bi-criteria optimization of decision trees relative to h and L_t.

The number of Pareto optimal points for bi-criteria optimization relative to h and L_t is small: see the bottom right subfigure of Figure 2 and Table 2 which contains the number of Pareto optimal points from the set

$$Par(t_{h,L_t}(\Delta_{rme,\alpha}(T), T)),$$

where T is one of 23 decision tables from UCI ML Repository described in Table 2 and $\alpha \in \{0, 0.01, 0.05, 0.1, \ldots, 0.9\}$. The maximum number of Pareto optimal points in the considered examples is equal to four. Totally optimal decision trees relative to h and L_t exist in many cases (in all cases where the number of POPs is equal to one).

For $n = 5, \ldots, 13$, we randomly generate 100 partial Boolean functions with n variables and 0.7×2^n n-tuples of variable values for which functions are defined, represent these functions as decision tables and count for each n the average number of Pareto optimal points. Figure 4 shows that the average number of Pareto optimal points is at most two.

The most interesting result of our experiments is that the sets of POPs for (h, L_t) are not diverse. However, the construction of these sets can help in choosing the appropriate systems of rules (appropriate decision tree). See for example, the bottom right subfigure in Figure 2 where the values for h are changing from 10 to 13 and the values for L_t are changing from 152 to 171.

The considered bi-criteria optimization algorithms will work efficiently only for medium-sized decision tables. For big decision tables, we can use only approximate algorithms for decision tree optimization. At the end of the next section, we consider greedy algorithms that are good enough from the point of view of single-criterion optimization relative to h or L_t and from the point of view of bi-criteria optimization relative to h and L_t.

8. Comparison of Greedy Algorithms for Decision Tree Construction

In this section, we compare 20 greedy algorithms for decision tree construction as single- and bi-criteria optimization algorithms. This can be considered as a real-life application of the bi-criteria optimization since to understand the accuracy of an algorithm it is not necessary to use big decision tables. To compare the greedy algorithms as single-criterion optimization algorithms, we use average relative difference. The relative difference for a given cost function, greedy algorithm and decision table is equal to $(greedy - opt)/opt$ where $greedy$ is the cost of the decision tree constructed by the greedy algorithm for the given table and opt is the minimum cost of a decision tree for this table. Values of the average relative difference for four cost functions, seven greedy algorithms and different thresholds α can be found in Figures 5-8. To compare the greedy algorithms as bi-criteria optimization algorithms, we use the normalized distance from the point corresponding to the decision tree constructed by the greedy algorithm (coordinates of this point are the values of the considered cost functions for the constructed tree) to the set of Pareto optimal points. Values of

the average minimum distances for four pairs of cost functions, seven greedy algorithms and different thresholds α can be found in Figures 9-12.

We consider five uncertainty measures for decision tables: entropy *ent*, gini index *gini*, function *rt*, misclassification error *me*, and relative misclassification error *rme* (see Section 2.2). Let T be a decision table, $f_i \in E(T)$ and $E(T, f_i) = \{a_1, \ldots, a_t\}$. The attribute f_i divides the table T into subtables $T_1 = T(f_i, a_1), \ldots, T_t = T(f_i, a_t)$. For a fixed uncertainty measure U, we can define *impurity functions* $I(T, f_i)$ of four types which give "impurity"of this partition: $\sum_{j=1}^{t} U(T_j)$ (*sum* type denoted by s), $\max_{1 \le j \le t} U(T_j)$ (*max* type denoted by m), $\sum_{j=1}^{t} U(T_j)N(T_j)$ (*weighted sum* type denoted by *ws*), and $\max_{1 \le j \le t} U(T_j)N(T_j)$ (*weighted max* type denoted by *wm*).

As a result, we have 20 impurity functions defined by pairs *type of impurity function-uncertainty measure*, for example, *wm-gini*. For each impurity function I, we describe a greedy algorithm $\mathcal{A}_I$ which, for a given decision table T and real number α, $0 \le \alpha < 1$, constructs a (rme, α)-decision tree for the table T.

Algorithm $\mathcal{A}_I$

Input: Decision table T with n conditional attributes $f_1, \ldots, f_n$, and real number α, $0 \le \alpha < 1$.

Output: An (rme, α)-decision tree for the table T.

1. Construct a tree G consisting of a single node labeled with T.
2. If no node of the tree G is labeled with a table then the algorithm $\mathcal{A}_I$ ends and returns the tree G.
3. Choose a node v in G which is labeled with a subtable Θ of the table T.
4. If $rme(\Theta) \le \alpha$ then we label the node v by $mcd(\Theta)$ and proceed to the step 2.
5. If $rme(\Theta) > \alpha$ then, for each $f_i \in E(\Theta)$, we compute the value $I(T, f_i)$ and label the node v with the attribute $f_{i_0} \in E(\Theta)$ with minimum index i_0 such that $I(T, f_{i_0}) = \min\{I(T, f_i) : f_i \in E(\Theta)\}$. For each $\delta \in E(\Theta, f_{i_0})$, we add to the tree G a node $v(\delta)$ and an edge e_δ connecting v and $v(\delta)$. We label the node $v(\delta)$ with the subtable $\Theta(f_{i_0}, \delta)$, and label the edge e_δ with the number δ. We proceed to the step 2.

We compare experimentally the obtained 20 greedy algorithms. The methodology of experiments is the following. Let $V = \{0, 0.1, \ldots, 0.9\}$. By S we denote the set of 23 decision tables from UCI ML Repository [18] (information about these decision tables can be found in Table 2). We study the following pairs of increasing cost functions (ψ, φ): (h, L), (h, h_{avg}), (h_{avg}, L), and (h, L_t) where $h_{\mathrm{avg}}(T, \Gamma) = tpl(T, \Gamma)/N(T)$.

For each such pair (ψ, φ), each $T \in S$ and each $\alpha \in V$, we construct the set $Par_{\psi,\varphi,\alpha}(T) = Par(t_{\psi,\varphi}(\Delta_{rme,\alpha}(T), T))$ of Pareto optimal points for the

problem of optimization of (rme, α)-decision trees for T relative to ψ and φ. Next we find the values $\psi_\alpha(T) = \min\{\psi(T, \Gamma) : \Gamma \in DT_{rme,\alpha}(T)\}$ and $\varphi_\alpha(T) = \min\{\varphi(T, \Gamma) : \Gamma \in DT_{rme,\alpha}(T)\}$ in the following way: $\psi_\alpha(T) = \min\{a : (a, b) \in Par_{\psi,\varphi,\alpha}(T)\}$ and $\varphi_\alpha(T) = \min\{b : (a, b) \in Par_{\psi,\varphi,\alpha}(T)\}$. For each $T \in S$, each $\alpha \in V$, and each of the considered 20 impurity functions I, we apply the algorithm A_I to the table T and the number α, and construct an (rme, α)-decision tree $\Gamma_{I,\alpha}(T)$ for T.

We compute:

$$\begin{aligned} \psi_{I,\alpha}(S) &= \frac{1}{23}\sum_{T\in S}(\psi_\alpha(T, \Gamma_{I,\alpha}(T)) - \psi_\alpha(T))/\psi_\alpha(T), \text{and} \\ \varphi_{I,\alpha}(S) &= \frac{1}{23}\sum_{T\in S}(\varphi_\alpha(T, \Gamma_{I,\alpha}(T)) - \varphi_\alpha(T))/\varphi_\alpha(T). \end{aligned}$$

The values $\psi_{I,\alpha}(S)$ and $\varphi_{I,\alpha}(S)$ characterize the accuracy of the algorithm $\mathcal{A}_I$ as an algorithm for optimization of (rme, α)-decision trees for tables from the set S relative to ψ and φ, respectively. We also compute the values:

$$\begin{aligned} \psi_I(S) &= \frac{1}{10}\sum_{\alpha\in V}\psi_{I,\alpha}(S), \text{and} \\ \varphi_I(S) &= \frac{1}{10}\sum_{\alpha\in V}\varphi_{I,\alpha}(S), \end{aligned}$$

which characterize the accuracy of the algorithm $\mathcal{A}_I$ as an algorithm for optimization of decision trees for tables from the set S relative to ψ and φ, respectively.

Let $A_\alpha = \max\{a : (a, b) \in Par_{\psi,\varphi,\alpha}(T)\}$, $B_\alpha = \max\{b : (a, b) \in Par_{\psi,\varphi,\alpha}(T)\}$, and $Par^*_{\psi,\varphi,\alpha}(T) = \{(a/A_\alpha, b/B_\alpha) : (a, b) \in Par_{\psi,\varphi,\alpha}(T)\}$. We denote by $d^{\psi,\varphi}_{I,\alpha}(T)$ the Euclidean distance between the point

$$(\psi(\Gamma_{I,\alpha}(T))/A_\alpha, \varphi(\Gamma_{I,\alpha}(T))/B_\alpha)$$

and the set $Par^*_{\psi,\varphi,\alpha}(T)$ (the distance between a point p and a finite set P of points is the minimum distance between the point p and a point from the set P). We compute the value

$$d^{\psi,\varphi}_{I,\alpha}(S) = \frac{1}{23}\sum_{T\in S} d^{\psi,\varphi}_{I,\alpha}(T),$$

which characterizes the accuracy of the algorithm $\mathcal{A}_I$ as an algorithm for bi-criteria optimization of (rme, α)-decision trees for tables from the set S relative to ψ and φ. We also compute the value

$$d^{\psi,\varphi}_{I}(S) = \frac{1}{10}\sum_{\alpha\in V} d^{\psi,\varphi}_{I,\alpha}(S),$$

which characterizes the accuracy of the algorithm $\mathcal{A}_I$ as an algorithm for bi-criteria optimization of decision trees for tables from the set S relative to ψ and φ.

For each cost function $\psi \in \{h, h_{\text{avg}}, L, L_t\}$, we order 20 greedy algorithms A_I according to the value $\psi_I(S)$ and choose three impurity functions corresponding to the three best algorithms (see Table 3). For each pair of cost functions $(\psi, \varphi) \in \{(h, L), (h, h_{\text{avg}}), (h_{\text{avg}}, L), (h, L_t)\}$, we order 20 greedy algorithms A_I according to the value $d_I^{\psi,\varphi}(S)$ and choose three impurity functions corresponding to the three best algorithms (see Table 3).

	Position		
Cost function(s)	1-st	2-nd	3-rd
h	*s-rt*	*ws-gini*	*ws-ent*
h_{avg}	*ws-gini*	*ws-ent*	*s-rt*
L	*ws-gini*	*ws-ent*	*s-me*
L_t	*ws-gini*	*ws-ent*	*s-me*
(h, L)	*ws-gini*	*ws-ent*	*s-rt*
(h, h_{avg})	*ws-rt*	*wm-rt*	*s-rt*
(h_{avg}, L)	*ws-gini*	*ws-ent*	*s-rt*
(h, L_t)	*s-rt*	*ws-rt*	*ws-me*

Table 3: Best impurity functions for single- and bi-criteria optimization of decision trees

We consider now the following question: are the best heuristics for a bi-criteria optimization problem the best ones also for corresponding single-criterion optimization problems?

Results for pairs (h, L) and (h_{avg}, L) are predictable. For example, the best impurity function *ws-gini* for bi-criteria optimization relative to (h, L) is the first for single-criterion optimization relative to L. Similar situation is for all three best impurity functions for bi-criteria optimization relative to (h, L) and relative to (h_{avg}, L).

Results for pairs (h, h_{avg}) and (h, L_t) are more interesting. For example, the best impurity function *ws-rt* for bi-criteria optimization relative to (h, h_{avg}) is the 12th for single-criterion optimization relative to h and the 5th for single-criterion optimization relative to h_{avg}. The second impurity function *wm-rt* for bi-criteria optimization relative to (h, h_{avg}) is the 14th for single-criterion optimization relative to h and the 14th for single-criterion optimization relative to h_{avg}.

The second impurity function *ws-rt* for bi-criteria optimization relative to (h, L_t) is the 12th for single-criterion optimization relative to h and the 16th for single-criterion optimization relative to L_t. The third impurity function *ws-me* for bi-criteria optimization relative to (h, L_t) is the 4th for single-criterion optimization relative to h and the 7th for single-criterion optimization relative to L_t.

Table 3 contains seven different impurity functions: *s-rt*, *s-me*, *wm-rt*, *ws-*

gini, *ws-rt*, *ws-ent*, and *ws-me*. For these impurity functions I, the graphs of the functions $\psi_{I,\alpha}(S)$ (values of these functions are called *average relative difference*) depending on α for each $\psi \in \{h, h_{\mathrm{avg}}, L, L_t\}$ can be found on Figures 5, 6, 7, and 8. The graphs of the functions $d^{\psi,\varphi}_{I,\alpha}(S)$ (values of these functions are called *average minimum distance*) depending on α for each $(\psi, \varphi) \in \{(h, L), (h, L_t), (h, h_{\mathrm{avg}}), (h_{\mathrm{avg}}, L)\}$ can be found on Figures 9, 10, 11, and 12.

The obtained results allow us to choose appropriate greedy algorithms if we are interested in single- or bi-criteria optimization of decision trees. In particular, we know that the bi-criteria optimization of decision trees relative to h and L_t is important when we study systems of decision rules derived from decision trees (see Section 7.2). According to Table 3, the best three impurity functions for h are *s-rt*, *ws-gini*, *ws-ent*; for L_t are *ws-gini*, *ws-ent*, *s-me*; and for (h, L_t) are *s-rt*, *ws-rt*, *ws-me*.

Figures 5 and 8 show the average relative difference for h and L_t for the considered algorithms and different values of the threshold α. Figure 10 shows that the greedy algorithm based on *s-rt* should be used for bi-criteria optimization relative to h and L_t if $\alpha \leq 0.55$. If $\alpha > 0.55$, then we should use the greedy algorithm based on *ws-rt*.

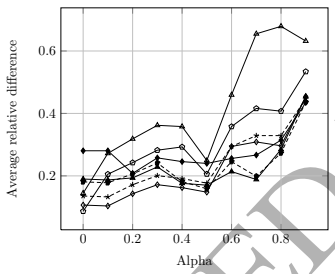


Figure 5: Comparison of greedy algorithms for h

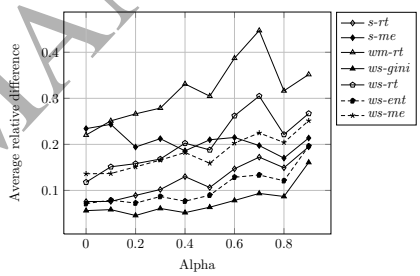


Figure 6: Comparison of greedy algorithms for h_{avg}

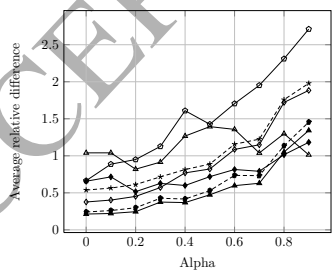


Figure 7: Comparison of greedy algorithms for L

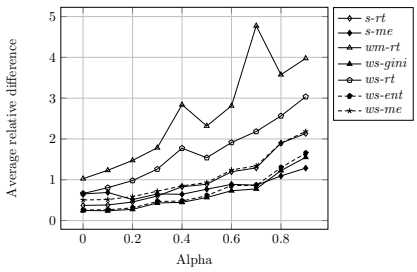


Figure 8: Comparison of greedy algorithms for L_t

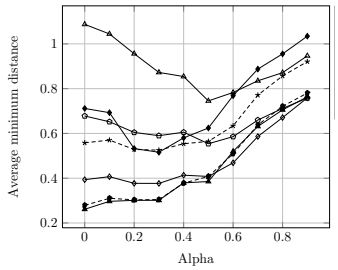


Figure 9: Comparison of greedy algorithms for (h, L)

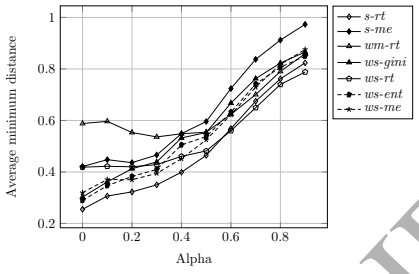


Figure 10: Comparison of greedy algorithms for (h, L_t)

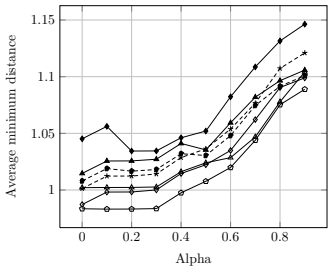


Figure 11: Comparison of greedy algorithms for (h, h_{avg})

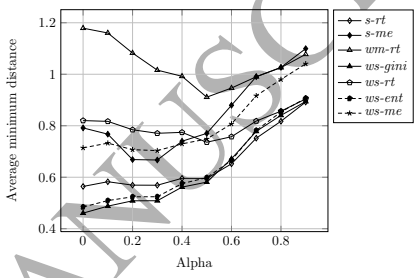


Figure 12: Comparison of greedy algorithms for (h_{avg}, L)

9. Conclusions

We described algorithms which allow us to construct the set of Pareto optimal points for the problem of bi-criteria optimization of decision trees. These algorithms can work with depth, average depth, number of nodes, number of terminal nodes of decision trees, etc. The main peculiarities of the considered bi-criteria optimization problems are that the number of decision trees under consideration can be huge but the number of Pareto optimal points is comparable with the size of the input decision table. We created a special technique which is adapted for these peculiarities and allows us to work with medium-sized decision tables. It means that the constructed algorithms can be used mainly as research tools.

We discussed three applications: studied decision trees for corner point detection, investigated systems of decision rules derived from decision trees, and compared 20 greedy algorithms for decision tree construction as single- and bi-criteria optimization algorithms.

The first two applications should be considered as illustrative examples rather than real applications since we can work only with medium-sized decision table. However, we found interesting tradeoffs for number of nodes vs. average depth of decision trees for know collection of images in the problem of corner point detection. We also discovered an abnormally small number of

Pareto optimal points for depth vs. number of terminal nodes bi-criteria optimization problem related to the derivation of decision rules from decision trees.

The comparison of greedy algorithms can be considered as a real application of the created tools for bi-criteria optimization: in this case it is enough to study medium-sized decision tables. The use of bi-criteria optimization here is crucial: we measure the quality of a heuristic by a distance from the point corresponding to heuristic to the set of Pareto optimal points. It is interesting that the best heuristic for a bi-criteria optimization problem can be far from the best ones for the corresponding single-criterion optimization problems.

The considered approach can be extended to the study of decision trees already optimized relative to some criteria and to the study of relationships between cost and accuracy of decision trees (the latter can be useful for classifier construction [4]). We are also planning to work with partial DAGs. This approach will reduce accuracy as well as time complexity of the considered algorithms (some results in this direction for decision rules can be found in [37]).

Acknowledgements

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.

Appendix A. Proofs

The appendix contains proofs of some statements considered in the paper.

Proof (of Proposition 1). Let $U(T) \leq \alpha$. Then $tree(mcd(T))$ is the only (U, α)-decision tree for T. Let $U(T) > \alpha$ and $\Gamma \in DT_{U,\alpha}(T)$. Then, by definition, $\Gamma = tree(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t)$ where $f_i \in E(T)$ and $\{a_1, \ldots, a_t\} = E(T, f_i)$. Using the fact that $\Gamma \in DT_{U,\alpha}(T)$ it is not difficult to show that $\Gamma_j \in DT_{U,\alpha}(T(f_i, a_j))$ for $j = 1, \ldots, t$. From here it follows that $\Gamma \in DT_{U,\alpha}(T, f_i)$ for $f_i \in E(T)$. Therefore $DT_{U,\alpha}(T) \subseteq \bigcup_{f_i \in E(T)} DT_{U,\alpha}(T, f_i)$.

Let, for some $f_i \in E(T)$, $\Gamma \in DT_{U,\alpha}(T, f_i)$. 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 $\Gamma_j \in DT_{U,\alpha}(T(f_i, a_j))$ for $j = 1, \ldots, t$. Using these facts it is not difficult to show that $\Gamma \in DT_{U,\alpha}(T)$. Therefore,

$$\bigcup_{f_i \in E(T)} DT_{U,\alpha}(T, f_i) \subseteq DT_{U,\alpha}(T).$$

□

Proof (of Proposition 3). We prove this statement by induction on nodes of $\Delta_{U,\alpha}(T)$. Let Θ be a terminal node of $\Delta_{U,\alpha}(T)$. Then $Tree(\Delta_{U,\alpha}(T), \Theta) =$

$\{tree(mcd(\Theta))\} = DT_{U,\alpha}(\Theta)$. Now let Θ be a nonterminal node of $\Delta_{U,\alpha}(T)$, and let us assume that $Tree(\Delta_{U,\alpha}(T), \Theta(f_i, a_j)) = DT_{U,\alpha}(\Theta(f_i, a_j))$ for any $f_i \in E(\Theta)$ and $\alpha_j \in E(\Theta, f_i)$. Then, for any $f_i \in E(\Theta)$, we have

$$Tree(\Delta_{U,\alpha}(T), \Theta, f_i) = DT_{U,\alpha}(\Theta, f_i).$$

Using Proposition 1, we obtain $Tree(\Delta_{U,\alpha}(T), \Theta) = DT_{U,\alpha}(\Theta)$. □

Proof (of Proposition 4). Let $(a_1, b_1), \ldots, (a_t, b_t)$ be the output sequence P and $Q = \{(a_1, b_1), \cdots, (a_t, b_t)\}$. It is clear that $a_1 < \cdots < a_t$, $b_1 > \cdots > b_t$ and, for any $\alpha \in A$, $\alpha \notin Q$, there exists $\beta \in Q$ such that $\beta < \alpha$. From here it follows that $Par(A) \subseteq Q$ and Q is an antichain. Let us assume that there exists $\gamma \in Q$ which does not belong to $Par(A)$. Then there exists $\alpha \in A$ such that $\alpha < \gamma$. Since Q is an antichain, $\alpha \notin Q$. We know that there exists $\beta \in Q$ such that $\beta \leq \alpha$. So two different points β and γ from Q are comparable which is impossible. Therefore $Q = Par(A)$ and P is the normal representation of the set $Par(A)$. □

Proof (of Proposition 5). We will prove by induction on i that, for $i = 1, \ldots, t$, the set B_i (see the description of the algorithm $\mathcal{A}_3$) is equal to the set $Par(Q_i)$. Since $B_1 = Par(P_1)$ and $Q_1 = P_1$, we have $B_1 = Par(Q_1)$. Let for some $i - 1$, $2 \leq i \leq t$, the considered statement hold, i.e., $B_{i-1} = Par(Q_{i-1})$. Then $B_i = Par(B_{i-1} \langle FH \rangle Par(P_i)) = Par(Par(Q_{i-1}) \langle FH \rangle Par(P_i))$.

We know that $Q_i = Q_{i-1} \langle FH \rangle P_i$. Let us show that

$$Par(Q_i) \subseteq Par(Q_{i-1}) \langle FH \rangle Par(P_i).$$

Let $\beta \in Par(Q_{i-1} \langle FH \rangle P_i)$ and $\beta = (F(a, c), H(b, d))$ where $(a, b) \in Q_{i-1}$ and $(c, d) \in P_i$. One can show that there exist $(a', b') \in Par(Q_{i-1})$ and $(c', d') \in Par(P_i)$ such that $(a', b') \leq (a, b)$ and $(c', d') \leq (c, d)$. It is clear that $\alpha = (F(a', c'), H(b', d')) \leq (F(a, c), H(b, d)) = \beta$ and $\alpha \in Par(Q_{i-1}) \langle FH \rangle Par(P_i)$. Since $\beta \in Par(Q_{i-1} \langle FH \rangle P_i)$, we have $\beta = \alpha$. Therefore $Par(Q_{i-1} \langle FH \rangle P_i) \subseteq Par(Q_{i-1}) \langle FH \rangle Par(P_i)$.

We now show that $Par(Q_i) = Par(Par(Q_{i-1}) \langle FH \rangle Par(P_i))$. Since

$$Par(Q_{i-1} \langle FH \rangle P_i) \subseteq Par(Q_{i-1}) \langle FH \rangle Par(P_i),$$

we have $Par(Q_{i-1} \langle FH \rangle P_i) \subseteq Par(Par(Q_{i-1}) \langle FH \rangle Par(P_i))$. Let us assume that, for some β, $\beta \in Par(Par(Q_{i-1}) \langle FH \rangle Par(P_i))$ and

$$\beta \notin Par(Q_{i-1} \langle FH \rangle P_i).$$

Then there exists $\alpha \in Q_{i-1} \langle FH \rangle P_i$ such that $\alpha < \beta$. One can show that there exists $\gamma \in Par(Q_{i-1} \langle FH \rangle P_i) \subseteq Par(Q_{i-1}) \langle FH \rangle Par(P_i)$ such that $\gamma \leq \alpha$. Therefore $\gamma < \beta$ and $\beta \notin Par(Par(Q_{i-1}) \langle FH \rangle Par(P_i))$. Hence $Par(Q_i) = Par(Q_{i-1} \langle FH \rangle P_i) = Par(Par(Q_{i-1}) \langle FH \rangle Par(P_i))$.

Therefore $B_i = Par(Q_i)$. So we have $B_t = Par(Q_t)$, and the algorithm $\mathcal{A}_3$ returns the set $Par(Q_t)$. □

PROOF (OF THEOREM 6). We prove the considered statement by induction on nodes of G. Let Θ be a terminal node of G. Then,

$$\begin{aligned} Tree(G,\Theta) &= \{tree(mcd(\Theta))\},\\ t_{\psi,\varphi}(G,\Theta) &= Par(t_{\psi,\varphi}(G,\Theta)) = \{(\psi^0(\Theta),\varphi^0(\Theta))\}, \text{and}\\ B(\Theta) &= Par(t_{\psi,\varphi}(G,\Theta)). \end{aligned}$$

Let Θ be a nonterminal node of G such that, for any $f_i \in E(\Theta)$ and any $a_j \in E(\Theta, f_i)$, the considered statement holds for the node $\Theta(f_i, a_j)$, i.e., $B(\Theta(f_i,a_j)) = Par(t_{\psi,\varphi}(G,\Theta(f_i,a_j)))$.

Let $f_i \in E(\Theta)$ and $E(\Theta,f_i) = \{a_1,\dots,a_t\}$. We denote

$$\begin{aligned} P(f_i) &= \{(F(b_1,\dots,b_t)+w(\Theta), H(c_1,\dots,c_t)+u(\Theta))\\ &\quad : (b_j,c_j) \in t_{\psi,\varphi}(G,\Theta(f_i,a_j)), j=1,\dots,t\}, \end{aligned}$$

and, for $j = 1,\dots,t$, we denote $P_j = t_{\psi,\varphi}(G,\Theta(f_i,a_j))$.

If we apply the algorithm $\mathcal{A}_3$ to the functions F, H and the sets $Par(P_1),\dots,Par(P_t)$, we obtain the set $Par(Q_t)$ where $Q_1 = P_1$, and, for $j = 2,\dots,t$, $Q_j = Q_{j-1}\langle FH\rangle P_j$. It is not difficult to show that $P(f_i) = Q_t \langle{+}{+}\rangle \{(w(\Theta),u(\Theta))\} = \{(a+w(\Theta), b+u(\Theta)) : (a,b) \in Q_t\}$ and

$$Par(P(f_i)) = Par(Q_t)\langle{+}{+}\rangle\{(w(\Theta),u(\Theta))\}.$$

According to the induction hypothesis, $B(\Theta(f_i,a_j)) = Par(P_j)$ for $j = 1,\dots,t$. Therefore $C(\Theta,f_i) = Par(Q_t)$ and $B(\Theta,f_i) = Par(P(f_i))$.

One can show that $t_{\psi,\varphi}(G,\Theta) = \bigcup_{f_i\in E(\Theta)} P(f_i)$. We prove now that

$$Par(t_{\psi,\varphi}(G,\Theta)) = Par\left(\bigcup_{f_i\in E(\Theta)} P(f_i)\right) \subseteq \bigcup_{f_i\in E(\Theta)} Par(P(f_i)).$$

Let $\alpha \in \bigcup_{f_i\in E(\Theta)} P(f_i) \setminus \bigcup_{f_i\in E(\Theta)} Par(P(f_i))$. Then there is $f_i \in E(\Theta)$ such that $\alpha \in P(f_i)$ but $\alpha \notin Par(P(f_i))$. Therefore there is $\beta \in P(f_i)$ such that $\beta \prec \alpha$. Hence $\alpha \notin \bigcup_{f_i\in E(\Theta)} Par(P(f_i))$, and $Par\left(\bigcup_{f_i\in E(\Theta)} P(f_i)\right) \subseteq \bigcup_{f_i\in E(\Theta)} Par(P(f_i))$.

Let us show that $Par(t_{\psi,\varphi}(G,\Theta)) = Par\left(\bigcup_{f_i\in E(\Theta)} Par(P(f_i))\right)$. Since $Par(t_{\psi,\varphi}(G,\Theta)) \subseteq \bigcup_{f_i\in E(\Theta)} Par(P(f_i))$, we have

$$Par(t_{\psi,\varphi}(G,\Theta)) \subseteq Par\left(\bigcup_{f_i\in E(\Theta)} Par(P(f_i))\right).$$

Let us assume that, for some β, $\beta \in Par\left(\bigcup_{f_i\in E(\Theta)} Par(P(f_i))\right)$ and $\beta \notin Par(t_{\psi,\varphi}(G,\Theta))$. Then there exists $\alpha \in t_{\psi,\varphi}(G,\Theta)$ such that $\alpha \prec \beta$. One can show that there exists $\gamma \in Par(t_{\psi,\varphi}(G,\Theta)) \subseteq \bigcup_{f_i\in E(\Theta)} Par(P(f_i))$ such

that $\gamma \leq \alpha$. Therefore $\gamma < \beta$ and $\beta \notin Par\left(\bigcup_{f_i \in E(\Theta)} Par(P(f_i))\right)$. Hence $Par(t_{\psi,\varphi}(G,\Theta)) = Par\left(\bigcup_{f_i \in E(\Theta)} Par(P(f_i))\right)$.

Since $B(\Theta, f_i) = Par(P(f_i))$ for any $f_i \in E(\Theta)$, we have $Par(t_{\psi,\varphi}(G,\Theta)) = Par\left(A(\Theta)\right) = B(\Theta)$. □

Proof (of Proposition 7). Let us consider two functions $\mathcal{F}_{t_{\psi,\varphi}(G,T)} : \mathsf{R} \to \mathsf{R}$ and $\mathcal{F}_{Par(t_{\psi,\varphi}(G,T))} : \mathsf{R} \to \mathsf{R}$ defined in the following way:

$$\begin{aligned} \mathcal{F}_{t_{\psi,\varphi}(G,T)}(x) &= \min\{b : (a,b) \in t_{\psi,\varphi}(G,T), a \leq x\}, \\ \mathcal{F}_{Par(t_{\psi,\varphi}(G,T))}(x) &= \min\{b : (a,b) \in Par(t_{\psi,\varphi}(G,T)), a \leq x\}. \end{aligned}$$

One can show that $a_1 = \min\{a : (a,b) \in t_{\psi,\varphi}(G,T)\}$. Therefore the value $\mathcal{F}_{t_{\psi,\varphi}(G,T)}(x)$ is undefined if $x < a_1$. Let $x \geq a_1$. Then both values $\mathcal{F}(x)$ and $\mathcal{F}_{t_{\psi,\varphi}(G,T)}(x)$ are defined. It is easy to check that $\mathcal{F}(x) = \mathcal{F}_{Par(t_{\psi,\varphi}(G,T))}(x)$. Since $Par(t_{\psi,\varphi}(G,T)) \subseteq t_{\psi,\varphi}(G,T)$, we have $\mathcal{F}_{t_{\psi,\varphi}(G,T)}(x) \leq \mathcal{F}(x)$. One can show that, for any point $(a,b) \in t_{\psi,\varphi}(G,T)$, there is a point

$$(a_i, b_i) \in Par(t_{\psi,\varphi}(G,T))$$

such that $(a_i, b_i) \leq (a,b)$. Therefore $\mathcal{F}(x) \leq \mathcal{F}_{t_{\psi,\varphi}(G,T)}(x)$ and $\mathcal{F}_{t_{\psi,\varphi}(G,T)}(x) = \mathcal{F}(x)$. It is easy to see that $\mathcal{F}^{\psi,\varphi}_{G,T}(x) = \mathcal{F}_{t_{\psi,\varphi}(G,T)}(x)$. Hence $\mathcal{F}^{\psi,\varphi}_{G,T}(x) = \mathcal{F}(x)$. □

References

[1] Abellán, J., Masegosa, A. R., 2010. An ensemble method using credal decision trees. European Journal of Operational Research 205 (1), 218–226.

[2] AbouEisha, H., Chikalov, I., Moshkov, M., 2016. Decision trees with minimum average depth for sorting eight elements. Discrete Applied Mathematics 204, 203–207.

[3] Alkhalid, A., Chikalov, I., Moshkov, M., 2011. Constructing an optimal decision tree for FAST corner point detection. In: Yao, J., Ramanna, S., Wang, G., Suraj, Z. (Eds.), Rough Sets and Knowledge Technology - 6th International Conference, RSKT 2011, Banff, Canada, October 9-12, 2011. Vol. 6954 of Lecture Notes in Computer Science. Springer, Heidelberg, pp. 187–194.

[4] Azad, M., Chikalov, I., Hussain, S., Moshkov, M., 2015. Multi-pruning of decision trees for knowledge representation and classification. In: 3rd IAPR Asian Conference on Pattern Recognition, ACPR 2015, Kuala Lumpur, Malaysia, November 3-6, 2015. IEEE, pp. 604–608.

[5] Barros, R. C., Basgalupp, M. P., de Carvalho, A. C. P. L. F., Freitas, A. A., 2012. A survey of evolutionary algorithms for decision-tree induction. IEEE Transactions on Systems, Man, and Cybernetics, Part C 42 (3), 291–312.

[6] Beame, P., Saks, M. E., Thathachar, J. S., 1998. Time-space tradeoffs for branching programs. In: 39th Annual Symposium on Foundations of Computer Science, FOCS '98, Palo Alto, California, USA, November 8-11, 1998. IEEE Computer Society, pp. 254–263.

[7] Bohanec, M., Bratko, I., 1994. Trading accuracy for simplicity in decision trees. Machine Learning 15 (3), 223–250.

[8] Borodin, A., Cook, S. A., 1982. A time-space tradeoff for sorting on a general sequential model of computation. SIAM J. Comput. 11 (2), 287–297.

[9] Breiman, L., Friedman, J. H., Olshen, R. A., Stone, C. J., 1984. Classification and Regression Trees. Wadsworth and Brooks, Monterey, CA.

[10] Chikalov, I., Hussain, S., Moshkov, M., 2014. Relationships between average depth and number of nodes for decision trees. In: Sun, F., Li, T., Li, H. (Eds.), Knowledge Engineering and Management. Vol. 214 of Advances in Intelligent Systems and Computing. Springer, Heidelberg, pp. 519–529.

[11] Chikalov, I., Hussain, S., Moshkov, M., 2016. Totally optimal decision trees for Boolean functions. Discrete Applied Mathematics 215, 1–13.

[12] Fieldsend, J. E., 2009. Optimizing decision trees using multi-objective particle swarm optimization. In: Coello, C. A., Dehuri, S., Ghosh, S. (Eds.), Swarm Intelligence for Multi-objective Problems in Data Mining. Vol. 242 of Studies in Computational Intelligence. Springer, Heidelberg, pp. 93–114.

[13] Frini, A., Guitouni, A., Martel, J.-M., 2012. A general decomposition approach for multi-criteria decision trees. European Journal of Operational Research 220 (2), 452–460.

[14] Garey, M. R., 1972. Optimal binary identification procedures. SIAM Journal on Applied Mathematics 23, 173–186.

[15] Hastie, T., Tibshirani, R., Friedman, J. H., 2001. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer-Verlag, New York.

[16] Hussain, S., 2014. Relationships among various parameters for decision tree optimization. In: Faucher, C., Jain, L. C. (Eds.), Innovations in Intelligent Machines-4 - Recent Advances in Knowledge Engineering. Vol. 514 of Studies in Computational Intelligence. Springer, Heidelberg, pp. 393–410.

[17] Hyafil, L., Rivest, R. L., 1976. Constructing optimal binary decision trees is NP-complete. Information Processing Letters 5 (1), 15–17.

[18] Lichman, M., 2013. UCI Machine Learning Repository, University of California, Irvine, School of Information and Computer Sciences.
URL `http://archive.ics.uci.edu/ml`

[19] Martelli, A., Montanari, U., 1978. Optimizing decision trees through heuristically guided search. Commun. ACM 21 (12), 1025–1039.

[20] Michalski, R. S., Pietrzykowski, J., 2007. iAQ: a program that discovers rules. In: 22nd AAAI Conference on Artificial Intelligence, AI Video Competition.

[21] Moshkov, M., 2005. Time complexity of decision trees. In: Peters, J. F., Skowron, A. (Eds.), T. Rough Sets III. Vol. 3400 of Lecture Notes in Computer Science. Springer, Heidelberg, pp. 244–459.

[22] Moshkov, M., 2007. On the class of restricted linear information systems. Discrete Mathematics 307 (22), 2837–2844.

[23] Moshkov, M., Chikalov, I., 2000. On algorithm for constructing of decision trees with minimal depth. Fundamenta Informaticae 41 (3), 295–299.

[24] Moshkov, M., Zielosko, B., 2011. Combinatorial Machine Learning - A Rough Set Approach. Vol. 360 of Studies in Computational Intelligence. Springer, Heidelberg.

[25] Muller, W., Wiederhold, E., 2002. Applying decision tree methodology for rules extraction under cognitive constraints. European Journal of Operational Research 136 (2), 282–289.

[26] Pangilinan, J. M., Janssens, G. K., 2011. Pareto-optimality of oblique decision trees from evolutionary algorithms. J. Global Optimization 51 (2), 301–311.

[27] Pawlak, Z., 1992. Rough Sets: Theoretical Aspects of Reasoning About Data. Kluwer Academic Publishers, Norwell, MA, USA.

[28] Pawlak, Z., Skowron, A., 2007. Rudiments of rough sets. Inf. Sci. 177 (1), 3–27.

[29] Pawlak, Z., Słowiński, R., 1994. Rough set approach to multi-attribute decision analysis. European Journal of Operational Research 72 (3), 443–459.

[30] Quinlan, J. R., 1993. C4.5: Programs for Machine Learning. Morgan Kaufmann.

[31] Rokach, L., Maimon, O., 2008. Data Mining with Decision Trees: Theory and Applications. World Scientific Publishing Co., Inc., River Edge, NJ, USA.

[32] Rosten, E., Drummond, T., 2005. Fusing points and lines for high performance tracking. In: 10th IEEE International Conference on Computer Vision (ICCV 2005), Beijing, China, October 17-20, 2005. IEEE Computer Society, pp. 1508–1515.

[33] Rosten, E., Drummond, T., 2006. Machine learning for high-speed corner detection. In: Computer Vision - ECCV 2006, 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006, Proceedings, Part I. Vol. 3951 of Lecture Notes in Computer Science. Springer, Heidelberg, pp. 430–443.

[34] Schumacher, H., Sevcik, K. C., 1976. The synthetic approach to decision table conversion. Commun. ACM 19 (6), 343–351.

[35] Slezak, D., 2002. Approximate entropy reducts. Fundam. Inform. 53 (3-4), 365–390.

[36] Zhao, H., 2007. A multi-objective genetic programming approach to developing pareto optimal decision trees. Decision Support Systems 43 (3), 809–826.

[37] Zielosko, B., 2014. Optimization of approximate decision rules relative to coverage. In: Kozielski, S., Mrozek, D., Kasprowski, P., Malysiak-Mrozek, B., Kostrzewa, D. (Eds.), Beyond Databases, Architectures, and Structures - 10th International Conference, BDAS 2014, Ustron, Poland, May 27-30, 2014. Vol. 424 of Communications in Computer and Information Science. Springer, Heidelberg, pp. 170–179.