electronics

MDPI

Article

Optimization of Decision Trees with Hypotheses for Knowledge Representation

Mohammad Azad [1], Igor Chikalov [2], Shahid Hussain [3] and Mikhail Moshkov [4,*]

1 Department of Computer Science, College of Computer and Information Sciences, Jouf University, Sakaka 72441, Saudi Arabia; mmazad@ju.edu.sa
2 Intel Corporation, 5000 W Chandler Blvd, Chandler, AZ 85226, USA; igor.chikalov@gmail.com
3 Computer Science Program, Dhanani School of Science and Engineering, Habib University, Karachi 75290, Pakistan; hussain.shahid@gmail.com
4 Computer, Electrical and Mathematical Sciences & Engineering Division, King Abdullah University of Science and Technology (KAUST), Thuwal 23955-6900, Saudi Arabia
* Correspondence: mikhail.moshkov@kaust.edu.sa

Abstract: In this paper, we consider decision trees that use two types of queries: queries based on one attribute each and queries based on hypotheses about values of all attributes. Such decision trees are similar to the ones studied in exact learning, where membership and equivalence queries are allowed. We present dynamic programming algorithms for minimization of the depth and number of nodes of above decision trees and discuss results of computer experiments on various data sets and randomly generated Boolean functions. Decision trees with hypotheses generally have less complexity, i.e., they are more understandable and more suitable as a means for knowledge representation.

Keywords: knowledge representation; decision tree; hypothesis; depth; number of nodes

Citation: Azad, M.; Chikalov, I.; Hussain, S.; Moshkov, M. Optimization of Decision Trees with Hypotheses for Knowledge Representation. *Electronics* **2021**, *10*, 1580. https://doi.org/10.3390/electronics10131580

Academic Editors: Dimitris Apostolou and Dionisios Sotiropoulos

Received: 26 May 2021
Accepted: 23 June 2021
Published: 30 June 2021

1. Introduction

Decision trees are used in many areas of computer science as a means for knowledge representation, as classifiers, and as algorithms to solve different problems of combinatorial optimization, computational geometry, etc. [1–3]. They are studied, in particular, in test theory initiated by Chegis and Yablonskii [4], rough set theory initiated by Pawlak [5–7], and exact learning initiated by Angluin [8,9]. These theories are closely related: attributes from rough set theory and test theory correspond to membership queries from exact learning. Exact learning studies additionally the so-called equivalence queries. The notion of "minimally adequate teacher" that allows both membership and equivalence queries was discussed by Angluin in Reference [10]. Relations between exact learning and PAC learning proposed by Valiant [11] are discussed in Reference [8].

In this paper, which is an extension of two conference papers [12,13], we add the notion of a hypothesis to the model that has been considered in rough set theory, as well as in test theory. This model allows us to use an analog of equivalence queries. Our goal is to check whether it is possible to reduce the time and space complexity of decision trees if we use additionally hypotheses. Decision trees with less complexity are more understandable and more suitable as a means for knowledge representation. Note that, to improve the understandability, we should not only try to minimize the number of nodes in a decision tree but also its depth that is the unimprovable upper bound on the number of conditions describing objects accepted by a path from the root to a terminal node of the tree. In this paper, we concentrate only on the consideration of complexity of decision trees and do not study many recent problems considered in machine learning [14–17].

Let T be a decision table with n conditional attributes $f_1, \ldots, f_n$ having values from the set $\omega = \{0, 1, 2, \ldots\}$ in which rows are pairwise different, and each row is labeled with a decision from ω. For a given row of T, we should recognize the decision attached to

this row. To this end, we can use decision trees based on two types of queries. We can ask about the value of an attribute $f_i \in \{f_1, \ldots, f_n\}$ on the given row. We will obtain an answer of the kind $f_i = \delta$, where δ is the number in the intersection of the given row and the column f_i. We can also ask if a hypothesis $f_1 = \delta_1, \ldots, f_n = \delta_n$ is true, where $\delta_1, \ldots, \delta_n$ are numbers from the columns $f_1, \ldots, f_n$, respectively. Either this hypothesis will be confirmed or we obtain a counterexample in the form $f_i = \sigma$, where $f_i \in \{f_1, \ldots, f_n\}$, and σ is a number from the column f_i different from δ_i. The considered hypothesis is called proper if $(\delta_1, \ldots, \delta_n)$ is a row of the table T.

In this paper, we study four cost functions that characterize the complexity of decision trees: the depth, the number of realizable nodes relative to T, the number of realizable terminal nodes relative to T, and the number of working nodes. We consider the depth of a decision tree as its time complexity, which is equal to the maximum number of queries in a path from the root to a terminal node of the tree. The remaining three cost functions characterize the space complexity of decision trees. A node is called realizable relative to T if, for a row of T and some choice of counterexamples, the computation in the tree will pass through this node. Note that, in the considered trees, all working nodes are realizable.

Decision trees using hypotheses can be essentially more efficient than the decision trees using only attributes. Let us consider an example, the problem of computation of the conjunction $x_1 \wedge \cdots \wedge x_n$. The minimum depth of a decision tree solving this problem using the attributes $x_1, \ldots, x_n$ is equal to n. The minimum number of realizable nodes in such decision trees is equal to $2n + 1$, the minimum number of working nodes is equal to n, and the minimum number of realizable terminal nodes is equal to $n + 1$. However, the minimum depth of a decision tree solving this problem using proper hypotheses is equal to 1: it is enough to ask only about the hypothesis $x_1 = 1, \ldots, x_n = 1$. If it is true, then the considered conjunction is equal to 1. Otherwise, it is equal to 0. The obtained decision tree contains one working node and $n + 1$ realizable terminal nodes, altogether $n + 2$ realizable nodes.

We study the following five types of decision trees:

1. Decision trees that use only attributes.
2. Decision trees that use only hypotheses.
3. Decision trees that use both attributes and hypotheses.
4. Decision trees that use only proper hypotheses.
5. Decision trees that use both attributes and proper hypotheses.

For each cost function, we propose a dynamic programming algorithm that, for a given decision table and given type of decision trees, finds the minimum cost of a decision tree of the considered type for this table. Note that dynamic programming algorithms for the optimization of decision trees of the type 1 were studied in Reference [18] for decision tables with one-valued decisions and in Reference [19] for decision tables with many-valued decisions. The dynamic programming algorithms for the optimization of decision trees of all five types were studied in Reference [12,13] for the depth and for the number of realizable nodes.

It is interesting to consider not only specially chosen examples as the conjunction of n variables. For each cost function, we compute the minimum cost of a decision tree for each of the considered five types for eight decision tables from the UCI ML Repository [20]. We do the same for randomly generated Boolean functions with n variables, where $n = 3, \ldots, 6$.

From the obtained experimental results, it follows that, generally, the decision trees of the types 3 and 5 have less complexity than the decision trees of the type 1. Therefore, such decision trees can be useful as a means for knowledge representation. Decision trees of the types 2 and 4 have, generally, too many nodes.

Based on the experimental results, we formulate and prove the following hypothesis: for any decision table, we can construct a decision tree with the minimum number of realizable terminal nodes using only attributes.

The motivation for the work is related to the use of decision trees to represent knowledge: we try to reduce the complexity of decision trees (and improve their understandabil-

ity) by using hypotheses. The main achievements of the work are the following: (i) we have proposed dynamic programming algorithms for optimizing five types of decision trees relative to four cost functions, and (ii) we have shown cases, when the use of hypotheses leads to the decrease in the complexity of decision trees.

The rest of the paper is organized as follows. In Sections 2 and 3, we consider main notions. In Sections 4–8 we describe dynamic programming algorithms for the decision tree optimization. In Section 9, we prove the above hypothesis. Section 10 contains results of computer experiments, and Section 11 gives short conclusions.

2. Decision Tables

A decision table is a table T with $n \geq 1$ columns filled with numbers from the set $\omega = \{0, 1, 2, \ldots\}$. Columns of this table are labeled with conditional attributes $f_1, \ldots, f_n$. Rows of the table are pairwise different. Each row is labeled with a number from ω that is interpreted as a decision. Rows of the table are interpreted as tuples of values of the conditional attributes.

Each decision table can be represented by a word (sequence) over the alphabet $\{0, 1, ;, |\}$: numbers from ω are in binary representation, we use the symbol ";" to separate two numbers from ω, and we use the symbol "|" to separate two rows (for each row, we add corresponding decision as the last number in the row). The length of this word is called the size of the decision table.

A decision table T is called empty if it has no rows. The table T is called degenerate if it is empty or all rows of T are labeled with the same decision.

We denote $F(T) = \{f_1, \ldots, f_n\}$ and denote by $D(T)$ the set of decisions attached to the rows of T. For any conditional attribute $f_i \in F(T)$, 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 $|E(T, f_i)| \geq 2$.

A system of equations over T is an arbitrary equation system of the kind

$$\{f_{i_1} = \delta_1, \ldots, f_{i_m} = \delta_m\},$$

where $m \in \omega$, $f_{i_1}, \ldots, f_{i_m} \in F(T)$, and $\delta_1 \in E(T, f_{i_1}), \ldots, \delta_m \in E(T, f_{i_m})$ (if $m = 0$, then the considered equation system is empty).

Let T be a nonempty table. A subtable of T is a table obtained from T by removal of some rows. We correspond to each equation system S over T a subtable TS of the table T. If the system S is empty, then $TS = T$. Let S be nonempty and $S = \{f_{i_1} = \delta_1, \ldots, f_{i_m} = \delta_m\}$. Then, TS is the subtable of the table T containing the rows from T that, in the intersection with columns, $f_{i_1}, \ldots, f_{i_m}$ have numbers $\delta_1, \ldots, \delta_m$, 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.

3. Decision Trees

Let T be a nonempty decision table with n conditional attributes $f_1, \ldots, f_n$. We consider the decision trees with two types of queries. We can choose an attribute $f_i \in F(T) = \{f_1, \ldots, f_n\}$ and ask about its value. This query has the set of answers $A(f_i) = \{\{f_i = \delta\} : \delta \in E(T, f_i)\}$. We can formulate a hypothesis over T in the form of $H = \{f_1 = \delta_1, \ldots, f_n = \delta_n\}$, where $\delta_1 \in E(T, f_1), \ldots, \delta_n \in E(T, f_n)$, and ask about this hypothesis. This query has the set of answers $A(H) = \{H, \{f_1 = \sigma_1\}, \ldots, \{f_n = \sigma_n\} : \sigma_1 \in E(T, f_1) \setminus \{\delta_1\}, \ldots, \sigma_n \in E(T, f_n) \setminus \{\delta_n\}\}$. The answer H means that the hypothesis is true. Other answers are counterexamples. The hypothesis H is called proper for T if $(\delta_1, \ldots, \delta_n)$ is a row of the table T.

A decision tree over T is a marked finite directed tree with the root in which:

- Each terminal node is labeled with a number from the set $D(T) \cup \{0\}$.
- Each node, which is not terminal (such nodes are called working), is labeled with an attribute from the set $F(T)$ or with a hypothesis over T.

- If a working node is labeled with an attribute f_i from $F(T)$, then, for each answer from the set $A(f_i)$, there is exactly one edge labeled with this answer, which leave this node and there are no any other edges leaving this node.
- If a working node is labeled with a hypothesis $H = \{f_1 = \delta_1, \ldots, f_n = \delta_n\}$ over T, then, for each answer from the set $A(H)$, there is exactly one edge labeled with this answer, which leaves this node and there are no any other edges leaving this node.

Let Γ be a decision tree over T and v be a node of Γ. We now define an equation system $S(\Gamma, v)$ over T associated with the node v. We denote by ξ the directed path from the root of Γ to the node v. If there are no working nodes in ξ, then $S(\Gamma, v)$ is the empty system. Otherwise, $S(\Gamma, v)$ is the union of equation systems attached to the edges of the path ξ.

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

- The node v is terminal if and only if the subtable $TS(\Gamma, v)$ is degenerate.
- If v is a terminal node and the subtable $TS(\Gamma, v)$ is empty, then the node v is labeled with the decision 0.
- If v is a terminal node and the subtable $TS(\Gamma, v)$ is nonempty, then the node v is labeled with the decision attached to all rows of $TS(\Gamma, v)$.

A complete path in Γ is an arbitrary directed path from the root to a terminal node in Γ. As the time complexity of a decision tree, we consider its depth that is the maximum number of working nodes in a complete path in the tree or, which is the same, the maximum length of a complete path in the tree. We denote by $h(\Gamma)$ the depth of a decision tree Γ.

As the space complexity of the decision tree Γ, we consider the number of its realizable relative to T nodes. A node v of Γ is called realizable relative to T if and only if the subtable $TS(\Gamma, v)$ is nonempty. We denote by $L(T, \Gamma)$ the number of nodes in Γ that are realizable relative to T. We also consider two more cost functions relative to the space complexity: $L_t(T, \Gamma)$ — the number of terminal nodes in Γ that are realizable relative to T and $L_w(T, \Gamma)$ — the number of working nodes in Γ. Note that all working nodes of Γ are realizable relative to T.

We will use the following notation:

- For $k = 1, \ldots, 5$, $h^{(k)}(T)$ is the minimum depth of a decision tree of the type k for T.
- For $k = 1, \ldots, 5$, $L^{(k)}(T)$ is the minimum number of nodes realizable relative to T in a decision tree of the type k for T.
- For $k = 1, \ldots, 5$, $L_t^{(k)}(T)$ is the minimum number of terminal nodes realizable relative to T in a decision tree of the type k for T.
- For $k = 1, \ldots, 5$, $L_w^{(k)}(T)$ is the minimum number of working nodes in a decision tree of the type k for T.

4. Construction of Directed Acyclic Graph $\Delta(T)$

Let T be a nonempty decision table with n conditional attributes $f_1, \ldots, f_n$. We now describe an algorithm $\mathcal{A}_{DAG}$ for the construction of a directed acyclic graph (DAG) $\Delta(T)$ that will be used for the study of decision trees. Nodes of this graph are 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. This algorithm can be considered as a special case of the algorithm for DAG construction considered in Reference [18].

Algorithm $\mathcal{A}_{DAG}$ (construction of DAG $\Delta(T)$).

***Input*:** A nonempty decision table T with n 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 algorithm halts and returns the resulting graph as $\Delta(T)$. Otherwise, choose a node (table) Θ that has not been processed yet.

3. If Θ is degenerate, then mark the node Θ as processed and proceed to step 2.
4. If Θ is not degenerate, 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 systems of equations $\{f_i = a_1\}, \ldots, \{f_i = a_k\}$. 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.

The following statement about time complexity of the algorithm $\mathcal{A}_{DAG}$ follows immediately from Proposition 3.3 [18].

Proposition 1. *The time complexity of the algorithm $\mathcal{A}_{DAG}$ 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.*

Generally, the time complexity of the algorithm $\mathcal{A}_{DAG}$ is exponential, depending on the size of the input decision tables. Note that, in Section 3.4 of the book [18], classes of decision tables are described for each of which the number of separable subtables of decision tables from the class is bounded from above by a polynomial on the number of columns in the tables. For each of these classes, the time complexity of the algorithm $\mathcal{A}_{DAG}$ is polynomial depending on the size of the input decision tables.

Note that similar results can be obtained for the space complexity of the considered algorithm.

5. Minimizing the Depth

In this section, we consider some results obtained in Reference [12]. Let T be a nonempty decision table with n conditional attributes $f_1, \ldots, f_n$. We can use the DAG $\Delta(T)$ to compute values $h^{(1)}(T), \ldots, h^{(5)}(T)$. Let $k \in \{1, \ldots, 5\}$. To find the value $h^{(k)}(T)$, for each node Θ of the DAG $\Delta(T)$, we compute the value $h^{(k)}(\Theta)$. It will be convenient for us to consider not only subtables that are nodes of $\Delta(T)$ but also empty subtable Λ of T and subtables T_r that contain only one row r of T and are not nodes of $\Delta(T)$. We begin with these special subtables and terminal nodes of $\Delta(T)$ (nodes without leaving edges) that are degenerate separable subtables of T and step-by-step move to the table T.

Let Θ be a terminal node of $\Delta(T)$ or $\Theta = T_r$ for some row r of T. Then, $h^{(k)}(\Theta) = 0$: the decision tree that contains only one node labeled with the decision attached to all rows of Θ is a decision tree for Θ. If $\Theta = \Lambda$, then $h^{(k)}(\Theta) = 0$: the decision tree that contains only one node labeled with 0 will be considered as a decision tree for Λ.

Let Θ be a nonterminal node of $\Delta(T)$ such that, for each child Θ' of Θ, we already know the value $h^{(k)}(\Theta')$. Based on this information, we can find the minimum depth of a decision tree for Θ, which uses for the subtables corresponding to the children of the root decision trees of the type k and in which the root is labeled:

- With an attribute from $F(T)$ (we denote by $h_a^{(k)}(\Theta)$ the minimum depth of such a decision tree).
- With a hypothesis over T (we denote by $h_h^{(k)}(\Theta)$ the minimum depth of such a decision tree).
- With a proper hypothesis over T (we denote by $h_p^{(k)}(\Theta)$ the minimum depth of such a decision tree).

Since Θ is nondegenerate, the set $E(\Theta)$ is nonempty. We now describe three procedures for computing the values $h_a^{(k)}(\Theta)$, $h_h^{(k)}(\Theta)$, and $h_p^{(k)}(\Theta)$, respectively.

Let us consider a decision tree $\Gamma(f_i)$ for Θ in which the root is labeled with an attribute $f_i \in E(\Theta)$. For each $\delta \in E(T, f_i)$, there is an edge that leaves the root and enters a node $v(\delta)$. This edge is labeled with the equation system $\{f_i = \delta\}$. The node $v(\delta)$ is the root of a decision tree of the type k for $\Theta\{f_i = \delta\}$ for which the depth is equal to $h^{(k)}(\Theta\{f_i = \delta\})$. It is clear that

$$h(\Gamma(f_i)) = 1 + \max\{h^{(k)}(\Theta\{f_i = \delta\}) : \delta \in E(T, f_i)\}.$$

Since $h^{(k)}(\Theta\{f_i = \delta\}) = h^{(k)}(\Lambda) = 0$ for any $\delta \in E(T, f_i) \setminus E(\Theta, f_i)$,

$$h(\Gamma(f_i)) = 1 + \max\{h^{(k)}(\Theta\{f_i = \delta\}) : \delta \in E(\Theta, f_i)\}. \quad (1)$$

Evidently, for any $\delta \in E(\Theta, f_i)$, the subtable $\Theta\{f_i = \delta\}$ is a child of Θ in the DAG $\Delta(T)$, i.e., we know the value $h^{(k)}(\Theta\{f_i = \delta\})$.

One can show that $h(\Gamma(f_i))$ is the minimum depth of a decision tree for Θ in which the root is labeled with the attribute f_i and which uses for the subtables corresponding to the children of the root decision trees of the type k.

We should not consider attributes $f_i \in F(T) \setminus E(\Theta)$ since, for each such attribute, there is $\delta \in E(T, f_i)$ with $\Theta\{f_i = \delta\} = \Theta$, i.e., based on this attribute, we cannot construct an optimal decision tree for Θ. As a result, we have

$$h_a^{(k)}(\Theta) = \min\{h(\Gamma(f_i)) : f_i \in E(\Theta)\}. \quad (2)$$

Computation of $h_a^{(k)}(\Theta)$. Construct the set of attributes $E(\Theta)$. For each attribute $f_i \in E(\Theta)$, compute the value $h(\Gamma(f_i))$ using (1). Compute the value $h_a^{(k)}(\Theta)$ using (2).

Remark 2. *Let Θ be a nonterminal node of the DAG $\Delta(T)$ such that, for each child Θ' of Θ, we already know the value $h^{(k)}(\Theta')$. Then, the procedure of computation of the value $h_a^{(k)}(\Theta)$ has polynomial time complexity depending on the size of decision table T.*

A hypothesis $H = \{f_1 = \delta_1, \ldots, f_n = \delta_n\}$ over T is called admissible for Θ and an attribute $f_i \in F(T) = \{f_1, \ldots, f_n\}$ if, for any $\sigma \in E(T, f_i) \setminus \{\delta_i\}$, $\Theta\{f_i = \sigma\} \neq \Theta$. The hypothesis H is not admissible for Θ and an attribute $f_i \in F(T)$ if and only if $|E(\Theta, f_i)| = 1$ and $\delta_i \notin E(\Theta, f_i)$. The hypothesis H is called admissible for Θ if it is admissible for Θ and any attribute $f_i \in F(T)$.

Let us consider a decision tree $\Gamma(H)$ for Θ in which the root is labeled with an admissible for Θ hypothesis $H = \{f_1 = \delta_1, \ldots, f_n = \delta_n\}$. The set of answers for the query corresponding to the hypothesis H is equal to $A(H) = \{H, \{f_1 = \sigma_1\}, \ldots, \{f_n = \sigma_n\} : \sigma_1 \in E(T, f_1) \setminus \{\delta_1\}, ..., \sigma_n \in E(T, f_n) \setminus \{\delta_n\}\}$. For each $S \in A(H)$, there is an edge that leaves the root of $\Gamma(H)$ and enters a node $v(S)$. This edge is labeled with the equation system S. The node $v(S)$ is the root of a decision tree of the type k for ΘS, which depth is equal to $h^{(k)}(\Theta S)$. It is clear that

$$h(\Gamma(H)) = 1 + \max\{h^{(k)}(\Theta S) : S \in A(H)\}.$$

We have $\Theta H = \Lambda$ or $\Theta H = T_r$ for some row r of T. Therefore, $h^{(k)}(\Theta H) = 0$. Since H is admissible for Θ, $E(\Theta, f_i) \setminus \{\delta_i\} = \emptyset$ for any attribute $f \in F(T) \setminus E(\Theta)$. It is clear that $\Theta\{f_i = \sigma\} = \Lambda$ and $h^{(k)}(\Theta\{f_i = \sigma\}) = 0$ for any attribute $f_i \in E(\Theta)$ and any $\sigma \in E(T, f_i) \setminus \{\delta_i\}$ such that $\sigma \notin E(\Theta, f_i)$. Therefore,

$$h(\Gamma(H)) = 1 + \max\{h^{(k)}(\Theta\{f_i = \sigma\}) : f_i \in E(\Theta), \sigma \in E(\Theta, f_i) \setminus \{\delta_i\}\}. \quad (3)$$

It is clear that, for any $f_i \in E(\Theta)$ and any $\sigma \in E(\Theta, f_i) \setminus \{\delta_i\}$, the subtable $\Theta\{f_i = \sigma\}$ is a child of Θ in the DAG $\Delta(T)$, i.e., we know the value $h^{(k)}(\Theta\{f_i = \sigma\})$.

One can show that $h(\Gamma(H))$ is the minimum depth of a decision tree for Θ in which the root is labeled with the hypothesis H and which uses for the subtables corresponding to the children of the root decision trees of the type k.

We should not consider hypotheses that are not admissible for Θ since, for each such hypothesis H for corresponding query, there is an answer $S \in A(H)$ with $\Theta S = \Theta$, i.e., based on this hypothesis, we cannot construct an optimal decision tree for Θ.

Computation of $h_h^{(k)}(\Theta)$. First, we construct a hypothesis:

$$H_\Theta = \{f_1 = \delta_1, \ldots, f_n = \delta_n\}$$

for Θ. Let $f_i \in F(T) \setminus E(\Theta)$. Then, δ_i is equal to the only number in the set $E(\Theta, f_i)$. Let $f_i \in E(\Theta)$. Then, δ_i is the minimum number from $E(\Theta, f_i)$ for which $h^{(k)}(\Theta\{f_i = \delta_i\}) = \max\{h^{(k)}(\Theta\{f_i = \sigma\}) : \sigma \in E(\Theta, f_i)\}$. It is clear that H_Θ is admissible for Θ. Compute the value $h(\Gamma(H_\Theta))$ using (3). Simple analysis of (3) shows that $h(\Gamma(H_\Theta)) = h_h^{(k)}(\Theta)$.

Remark 3. *Let Θ be a nonterminal node of the DAG $\Delta(T)$ such that, for each child Θ' of Θ, we already know the value $h^{(k)}(\Theta')$. Then, the procedure of computation of the value $h_h^{(k)}(\Theta)$ has polynomial time complexity depending on the size of decision table T.*

Computation of $h_p^{(k)}(\Theta)$. For each row $r = (\delta_1, \ldots, \delta_n)$ of the decision table T, we check if the corresponding proper hypothesis $H_r = \{f_1 = \delta_1, \ldots, f_n = \delta_n\}$ is admissible for Θ. For each admissible for Θ proper hypothesis $H_r = \{f_1 = \delta_1, \ldots, f_n = \delta_n\}$, we compute the value $h(\Gamma(H_r))$ using (3). One can show that the minimum among the obtained numbers is equal to $h_p^{(k)}(\Theta)$.

Remark 4. *Let Θ be a nonterminal node of the DAG $\Delta(T)$ such that, for each child Θ' of Θ, we already know the value $h^{(k)}(\Theta')$. Then, the procedure of computation of the value $h_p^{(k)}(\Theta)$ has polynomial time complexity depending on the size of decision table T.*

We describe an algorithm $\mathcal{A}_h$ that, for a given nonempty decision table T and given $k \in \{1, \ldots, 5\}$, calculates the value $h^{(k)}(T)$, which is the minimum depth of a decision tree of the type k for the table T. During the work of this algorithm, we find for each node Θ of the DAG $\Delta(T)$ the value $h^{(k)}(\Theta)$.

Algorithm $\mathcal{A}_h$ (computation of $h^{(k)}(T)$).

***Input*:** A nonempty decision table T, the directed acyclic graph $\Delta(T)$, and number $k \in \{1, \ldots, 5\}$.

***Output*:** The value $h^{(k)}(T)$.

1. If a number is attached to each node of the DAG $\Delta(T)$, then return the number attached to the node T as $h^{(k)}(T)$ and halt the algorithm. Otherwise, choose a node Θ of the graph $\Delta(T)$ without attached number, which is either a terminal node of $\Delta(T)$ or a nonterminal node of $\Delta(T)$ for which all children have attached numbers.
2. If Θ is a terminal node, then attach to it the number $h^{(k)}(\Theta) = 0$ and proceed to step 1.
3. If Θ is not a terminal node, then, depending on the value k, do the following:
   - In the case $k = 1$, compute the value $h_a^{(1)}(\Theta)$ and attach to Θ the value $h^{(1)}(\Theta) = h_a^{(1)}(\Theta)$.
   - In the case $k = 2$, compute the value $h_h^{(2)}(\Theta)$ and attach to Θ the value $h^{(2)}(\Theta) = h_h^{(2)}(\Theta)$.
   - In the case $k = 3$, compute the values $h_a^{(3)}(\Theta)$ and $h_h^{(3)}(\Theta)$, and attach to Θ the value $h^{(3)}(\Theta) = \min\{h_a^{(3)}(\Theta), h_h^{(3)}(\Theta)\}$.
   - In the case $k = 4$, compute the value $h_p^{(4)}(\Theta)$ and attach to Θ the value $h^{(4)}(\Theta) = h_p^{(4)}(\Theta)$.
   - In the case $k = 5$, compute the values $h_a^{(5)}(\Theta)$ and $h_p^{(5)}(\Theta)$, and attach to Θ the value $h^{(5)}(\Theta) = \min\{h_a^{(5)}(\Theta), h_p^{(5)}(\Theta)\}$.

   Proceed to step 1.

Using Remarks 2–4, one can prove the following statement.

Proposition 5. *The time complexity of the algorithm $\mathcal{A}_h$ 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.*

A similar bound can be obtained for the space complexity of the considered algorithm.

6. Minimizing the Number of Realizable Nodes

In this section, we consider some results obtained in Reference [13]. Let T be a nonempty decision table with n conditional attributes $f_1, \ldots, f_n$. We can use the DAG $\Delta(T)$ to compute values $L^{(1)}(T), \ldots, L^{(5)}(T)$. Let $k \in \{1, \ldots, 5\}$. To find the value $L^{(k)}(T)$, we compute the value $L^{(k)}(\Theta)$ for each node Θ of the DAG $\Delta(T)$. We will consider not only subtables that are nodes of $\Delta(T)$ but also empty subtable Λ of T and subtables T_r that contain only one row r of T and are not nodes of $\Delta(T)$. We begin with these special subtables and terminal nodes of $\Delta(T)$ (nodes without leaving edges) that are degenerate separable subtables of T and step-by-step move to the table T.

Let Θ be a terminal node of $\Delta(T)$ or $\Theta = T_r$ for some row r of T. Then, $L^{(k)}(\Theta) = 1$: the decision tree that contains only one node labeled with the decision attached to all rows of Θ is a decision tree for Θ. The only node of this tree is realizable relative to Θ. If $\Theta = \Lambda$, then $L^{(k)}(\Theta) = 0$: the decision tree that contains only one node labeled with 0 will be considered as a decision tree for Λ. The only node of this tree is not realizable relative to Λ.

Let Θ be a nonterminal node of $\Delta(T)$ such that, for each child Θ' of Θ, we already know the value $L^{(k)}(\Theta')$. Based on this information, we can find the minimum number of realizable relative to Θ nodes in a decision tree for Θ, which uses for the subtables corresponding to the children of the root decision trees of the type k and in which the root is labeled

- With an attribute from $F(T)$ (we denote by $L_a^{(k)}(\Theta)$ the minimum number of realizable relative to Θ nodes in such a decision tree).
- With a hypothesis over T (we denote by $L_h^{(k)}(\Theta)$ the minimum number of realizable relative to Θ nodes in such a decision tree).
- With a proper hypothesis over T (we denote by $L_p^{(k)}(\Theta)$ the minimum number of realizable relative to Θ nodes in such a decision tree).

We now describe three procedures for computing the values $L_a^{(k)}(\Theta)$, $L_h^{(k)}(\Theta)$, and $L_p^{(k)}(\Theta)$, respectively. Since Θ is nondegenerate, the set $E(\Theta)$ is nonempty.

Let us consider a decision tree $\Gamma(f_i)$ for Θ in which the root is labeled with an attribute $f_i \in E(\Theta)$. For each $\delta \in E(T, f_i)$, there is an edge that leaves the root and enters a node $v(\delta)$. This edge is labeled with the equation system $\{f_i = \delta\}$. The node $v(\delta)$ is the root of a decision tree of the type k for $\Theta\{f_i = \delta\}$ for which the number of realizable relative to $\Theta\{f_i = \delta\}$ nodes is equal to $L^{(k)}(\Theta\{f_i = \delta\})$. It is clear that $L(\Theta, \Gamma(f_i)) = 1 + \sum_{\delta \in E(T, f_i)} L^{(k)}(\Theta\{f_i = \delta\})$. Since $L^{(k)}(\Theta\{f_i = \delta\}) = L^{(k)}(\Lambda) = 0$ for any $\delta \in E(T, f_i) \setminus E(\Theta, f_i)$,

$$L(\Theta, \Gamma(f_i)) = 1 + \sum_{\delta \in E(\Theta, f_i)} L^{(k)}(\Theta\{f_i = \delta\}). \quad (4)$$

Evidently, for any $\delta \in E(\Theta, f_i)$, the subtable $\Theta\{f_i = \delta\}$ is a child of Θ in the DAG $\Delta(T)$, i.e., we know the value $L^{(k)}(\Theta\{f_i = \delta\})$. One can show that $L(\Theta, \Gamma(f_i))$ is the minimum number of realizable relative to Θ nodes in a decision tree for Θ, which uses for the subtables corresponding to the children of the root decision trees of the type k and in which the root is labeled with the attribute f_i.

We should not consider attributes $f_i \in F(T) \setminus E(\Theta)$ since, for each such attribute, there is $\delta \in E(T, f_i)$ with $\Theta\{f_i = \delta\} = \Theta$, i.e., based on this attribute, we cannot construct an optimal decision tree for Θ. As a result, we have

$$L_a^{(k)}(\Theta) = \min\{L(\Theta, \Gamma(f_i)) : f_i \in E(\Theta)\}. \quad (5)$$

Computation of $L_a^{(k)}(\Theta)$. Construct the set of attributes $E(\Theta)$. For each attribute $f_i \in E(\Theta)$, compute the value $L(\Theta, \Gamma(f_i))$ using (4). Compute the value $L_a^{(k)}(\Theta)$ using (5).

Let us consider a decision tree $\Gamma(H)$ for Θ in which the root is labeled with an admissible for Θ hypothesis $H = \{f_1 = \delta_1, \ldots, f_n = \delta_n\}$. For each $S \in A(H)$, there is an edge that leaves the root of $\Gamma(H)$ and enters a node $v(S)$. This edge is labeled with the equation system S. The node $v(S)$ is the root of a decision tree of the type k for ΘS, for which the number of realizable relative to ΘS nodes is equal to $L^{(k)}(\Theta S)$. It is clear that $L(\Theta, \Gamma(H)) = 1 + \sum_{S \in A(H)} L^{(k)}(\Theta S)$.

Denote $r = (\delta_1, \ldots, \delta_n)$. It is easy to show that $\Theta H = \Lambda$ if r is not a row of Θ and $\Theta H = T_r$ if r is a row of Θ. Therefore,

$$L^{(k)}(\Theta H) = \begin{cases} 0, & \text{if } r \text{ is not a row of } \Theta, \\ 1, & \text{if } r \text{ is a row of } \Theta. \end{cases} \tag{6}$$

Since H is admissible for Θ, $E(\Theta, f_i) \setminus \{\delta_i\} = \emptyset$ for any attribute $f_i \in F(T) \setminus E(\Theta)$. It is clear that $\Theta\{f_i = \sigma\} = \Lambda$ and $L^{(k)}(\Theta\{f_i = \sigma\}) = 0$ for any attribute $f_i \in E(\Theta)$ and any $\sigma \in E(T, f_i) \setminus \{\delta_i\}$ such that $\sigma \notin E(\Theta, f_i)$. Therefore,

$$L(\Theta, \Gamma(H)) = L^{(k)}(\Theta H) + K(\Theta, H), \tag{7}$$

where

$$K(\Theta, H) = 1 + \sum_{f_i \in E(\Theta), \sigma \in E(\Theta, f_i) \setminus \{\delta_i\}} L^{(k)}(\Theta\{f_i = \sigma\}). \tag{8}$$

Evidently, for any $f_i \in E(\Theta)$ and any $\sigma \in E(\Theta, f_i) \setminus \{\delta_i\}$, the subtable $\Theta\{f_i = \sigma\}$ is a child of Θ in the DAG $\Delta(T)$, i.e., we know the value $L^{(k)}(\Theta\{f_i = \sigma\})$. It is easy to show that $L(\Theta, \Gamma(H))$ is the minimum number of realizable relative to Θ nodes in a decision tree for Θ, which uses for the subtables corresponding to the children of the root decision trees of the type k and in which the root is labeled with the admissible for Θ hypothesis H.

We should not consider hypotheses that are not admissible for Θ since, for each such hypothesis H for corresponding query, there is an answer $S \in A(H)$ with $\Theta S = \Theta$, i.e., based on this hypothesis, we cannot construct an optimal decision tree for Θ. As a result, we have

$$L_h^{(k)}(\Theta) = \min\{L(\Theta, \Gamma(H)) : H \in Adm(\Theta)\}, \tag{9}$$

where $Adm(\Theta)$ is the set of admissible hypotheses for Θ.

For each $f_i \in \{f_1, \ldots, f_n\}$, denote $a_i(\Theta) = \max\{L^{(k)}(\Theta\{f_i = \sigma\}) : \sigma \in E(\Theta, f_i)\}$ and $C(\Theta, f_i) = \{\sigma \in E(\Theta, f_i) : L^{(k)}(\Theta\{f_i = \sigma\}) = a_i(\Theta)\}$. Set $C(\Theta) = C(\Theta, f_1) \times \cdots \times C(\Theta, f_n)$. It is clear that, for each $\bar{\delta} = (\delta_1, \ldots, \delta_n) \in C(\Theta)$, the hypothesis $H_{\bar{\delta}} = \{f_1 = \delta_1, \ldots, f_n = \delta_n\}$ is admissible for Θ. Simple analysis of (8) shows that the set $\{H_{\bar{\delta}} : \bar{\delta} \in C(\Theta)\}$ coincides with the set of admissible for Θ hypotheses H that minimize the value $K(\Theta, H)$. Denote $K_{\min} = K(\Theta, H_{\bar{\delta}})$, where $\bar{\delta} \in C(\Theta)$.

Let there be a tuple $\bar{\delta} \in C(\Theta)$, which is not a row of Θ. Then, $L^{(k)}(\Theta H_{\bar{\delta}}) = 0$ and $L_h^{(k)}(\Theta) = K_{\min}$. Let all tuples from $C(\Theta)$ be rows of Θ. We now show that $L_h^{(k)}(\Theta) = 1 + K_{\min}$. For any $\bar{\delta} \in C(\Theta)$, we have $L(\Theta, \Gamma(H_{\bar{\delta}})) = 1 + K_{\min}$. Therefore, $L_h^{(k)}(\Theta) \leq 1 + K_{\min}$. Let us assume that $L_h^{(k)}(\Theta) < 1 + K_{\min}$. Then, by (9), there exists an admissible for Θ hypothesis $H = \{f_1 = \sigma_1, \ldots, f_n = \sigma_n\}$ for which $(\sigma_1, \ldots, \sigma_n) \notin C(\Theta)$ and $L(\Theta, \Gamma(H)) < 1 + K_{\min}$, but this is impossible since, according to (7), $L(\Theta, \Gamma(H)) \geq K(\Theta, H) \geq K_{\min} + 1$.

As a result, we have $L_h^{(k)}(\Theta) = K_{\min}$ if not all tuples from $C(\Theta)$ are rows of Θ, and $L_h^{(k)}(\Theta) = 1 + K_{\min}$ if all tuples from $C(\Theta)$ are rows of Θ.

Computation of $L_h^{(k)}(\Theta)$. For each $f_i \in \{f_1, \ldots, f_n\}$, we compute the value: $a_i(\Theta) = \max\{L^{(k)}(\Theta\{f_i = \sigma\}) : \sigma \in E(\Theta, f_i)\}$ and construct the set $C(\Theta, f_i) = \{\sigma \in E(\Theta, f_i) : L^{(k)}(\Theta\{f_i = \sigma\}) = a_i(\Theta)\}$. For a tuple $\bar{\delta} \in C(\Theta) = C(\Theta, f_1) \times \cdots \times C(\Theta, f_n)$, using (8), we compute the value $K_{\min} = K(\Theta, H_{\bar{\delta}})$. Then, we count the number N of rows from Θ

that belong to the set $C(\Theta)$ and compute the cardinality $|C(\Theta)|$ of the set $C(\Theta)$ that is equal to $|C(\Theta, f_1)| \cdot \ldots \cdot |C(\Theta, f_n)|$. As a result, we have $L_h^{(k)}(\Theta) = K_{\min}$ if $N < |C(\Theta)|$ and $L_h^{(k)}(\Theta) = 1 + K_{\min}$ if $N = |C(\Theta)|$.

Computation of $L_p^{(k)}(\Theta)$. For each row $r = (\delta_1, \ldots, \delta_n)$ of the decision table T, we check if the corresponding proper hypothesis $H_r = \{f_1 = \delta_1, \ldots, f_n = \delta_n\}$ is admissible for Θ. For each admissible for Θ proper hypothesis H_r, we compute the value $L(\Theta, \Gamma(H_r))$ using (6), (7), and (8). One can show that the minimum among the obtained numbers is equal to $L_p^{(k)}(\Theta)$.

We now consider an algorithm $\mathcal{A}_L$ that, for a given nonempty decision table T and number $k \in \{1, \ldots, 5\}$, calculates the value $L^{(k)}(T)$, which is the minimum number of nodes realizable relative to T in a decision tree of the type k for the table T. During the work of this algorithm, we find for each node Θ of the DAG $\Delta(T)$ the value $L^{(k)}(\Theta)$.

The description of the algorithm $\mathcal{A}_L$ is similar to the description of the algorithm $\mathcal{A}_h$. Instead of $h^{(k)}$, we should use $L^{(k)}$. For each $b \in \{a, h, p\}$, instead of $h_b^{(k)}$, we should use $L_b^{(k)}$. In particular, for each terminal node Θ, $L^{(k)}(\Theta) = 1$.

One can show that the procedures of computation of the values $L_a^{(k)}(\Theta)$, $L_h^{(k)}(\Theta)$, and $L_p^{(k)}(\Theta)$ have polynomial time complexity depending on the size of the decision table T. Using this fact, one can prove the following statement.

Proposition 6. *The time complexity of the algorithm $\mathcal{A}_L$ 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.*

A similar bound can be obtained for the space complexity of the considered algorithm.

7. Minimizing the Number of Realizable Terminal Nodes

The procedure considered in this section is similar to the procedure of the minimization of the number of realizable nodes. The main difference is that, in decision trees with the minimum number of realizable terminal nodes, it is possible to meet constant attributes and hypotheses that are not admissible. Fortunately, for any decision table and any type of decision trees, there is a decision tree of this type with the minimum number of realizable terminal nodes for the considered table that do not use such attributes and hypotheses. We will omit many details and describe main steps only.

Let T be a nonempty decision table with n conditional attributes $f_1, \ldots, f_n$ and $k \in \{1, \ldots, 5\}$. To find the value $L_t^{(k)}(T)$, we compute the value $L_t^{(k)}(\Theta)$ for each node Θ of the DAG $\Delta(T)$. We begin with terminal nodes of $\Delta(T)$ that are degenerate separable subtables of T and step-by-step move to the table T.

Let Θ be a terminal node of $\Delta(T)$. Then, $L_t^{(k)}(\Theta) = 1$: the decision tree that contains only one node labeled with the decision attached to all rows of Θ is a decision tree for Θ. The only node of this tree is a terminal node realizable relative to Θ.

Let Θ be a nonterminal node of $\Delta(T)$ such that, for each child Θ' of Θ, we already know the value $L_t^{(k)}(\Theta')$. Based on this information, we can find the minimum number of realizable relative to Θ terminal nodes in a decision tree for Θ, which uses for the subtables corresponding to children of the root decision trees of the type k and in which the root is labeled

- With an attribute from $F(T)$ (we denote by $L_{t,a}^{(k)}(\Theta)$ the minimum number of realizable relative to Θ terminal nodes in such a decision tree).
- With a hypothesis over T (we denote by $L_{t,h}^{(k)}(\Theta)$ the minimum number of realizable relative to Θ terminal nodes in such a decision tree).
- With a proper hypothesis over T (we denote by $L_{t,p}^{(k)}(\Theta)$ the minimum number of realizable relative to Θ terminal nodes in such a decision tree).

We now describe three procedures for computing the values $L^{(k)}_{t,a}(\Theta)$, $L^{(k)}_{t,h}(\Theta)$, and $L^{(k)}_{t,p}(\Theta)$, respectively. Since Θ is nondegenerate, the set $E(\Theta)$ is nonempty.

Computation of $L^{(k)}_{t,a}(\Theta)$. Construct the set of attributes $E(\Theta)$. For each attribute $f_i \in E(\Theta)$, compute the value:

$$L^{(k)}_{t,a}(\Theta, f_i) = \sum_{\delta \in E(\Theta, f_i)} L^{(k)}_t(\Theta\{f_i = \delta\}).$$

Then, compute the value:

$$L^{(k)}_{t,a}(\Theta) = \min\{L^{(k)}_{t,a}(\Theta, f_i) : f_i \in E(\Theta)\}.$$

Computation of $L^{(k)}_{t,h}(\Theta)$. For each $f_i \in \{f_1, \ldots, f_n\}$, we compute the value: $a_i(\Theta) = \max\{L^{(k)}_t(\Theta\{f_i = \sigma\}) : \sigma \in E(\Theta, f_i)\}$ and construct the set $C(\Theta, f_i) = \{\sigma \in E(\Theta, f_i) : L^{(k)}_t(\Theta\{f_i = \sigma\}) = a_i(\Theta)\}$. For a tuple $(\delta_1, \ldots, \delta_n) \in C(\Theta) = C(\Theta, f_1) \times \cdots \times C(\Theta, f_n)$, we compute the value:

$$K_{\min} = \sum_{f_i \in E(\Theta), \sigma \in E(\Theta, f_i) \setminus \{\delta_i\}} L^{(k)}_t(\Theta\{f_i = \sigma\}).$$

Then, we count the number N of rows from Θ that belong to the set $C(\Theta)$ and compute the cardinality $|C(\Theta)|$ of the set $C(\Theta)$ that is equal to $|C(\Theta, f_1)| \cdot \ldots \cdot |C(\Theta, f_n)|$. As a result, we have $L^{(k)}_{t,h}(\Theta) = K_{\min}$ if $N < |C(\Theta)|$ and $L^{(k)}_{t,h}(\Theta) = 1 + K_{\min}$ if $N = |C(\Theta)|$.

Computation of $L^{(k)}_{t,p}(\Theta)$. For each row $r = (\delta_1, \ldots, \delta_n)$ of the decision table T, we check if the corresponding proper hypothesis $H_r = \{f_1 = \delta_1, \ldots, f_n = \delta_n\}$ is admissible for Θ. For each admissible for Θ proper hypothesis H_r, we compute the value:

$$L^{(k)}_{t,p}(\Theta, H_r) = 1 + \sum_{f_i \in E(\Theta), \sigma \in E(\Theta, f_i) \setminus \{\delta_i\}} L^{(k)}_t(\Theta\{f_i = \sigma\}).$$

One can show that the minimum among the obtained numbers is equal to $L^{(k)}_{t,p}(\Theta)$.

We now consider an algorithm $\mathcal{A}_{L_t}$ that, for a given nonempty decision table T and number $k \in \{1, \ldots, 5\}$, calculates the value $L^{(k)}_t(T)$, which is the minimum number of terminal nodes realizable relative to T in a decision tree of the type k for the table T. During the work of this algorithm, we find for each node Θ of the DAG $\Delta(T)$ the value $L^{(k)}_t(\Theta)$.

The description of the algorithm $\mathcal{A}_{L_t}$ is similar to the description of the algorithm $\mathcal{A}_h$. Instead of $h^{(k)}$, we should use $L^{(k)}_t$. For each $b \in \{a, h, p\}$, instead of $h^{(k)}_b$, we should use $L^{(k)}_{t,b}$. In particular, for each terminal node Θ, $L^{(k)}_t(\Theta) = 1$.

One can show that the procedures of computation of the values $L^{(k)}_{t,a}(\Theta)$, $L^{(k)}_{t,h}(\Theta)$, and $L^{(k)}_{t,p}(\Theta)$ have polynomial time complexity depending on the size of the decision table T. Using this fact, one can prove the following statement.

Proposition 7. *The time complexity of the algorithm $\mathcal{A}_{L_t}$ 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.*

A similar bound can be obtained for the space complexity of the considered algorithm.

8. Minimizing the Number of Working Nodes

The procedure considered in this section is similar to the procedure of the minimization of the depth. We will omit many details and describe main steps only.

Let T be a nonempty decision table with n conditional attributes $f_1, \ldots, f_n$ and $k \in \{1, \ldots, 5\}$. To find the value $L_w^{(k)}(T)$, we compute the value $L_w^{(k)}(\Theta)$ for each node Θ of the DAG $\Delta(T)$. We begin with terminal nodes of $\Delta(T)$ that are degenerate separable subtables of T and step-by-step move to the table T.

Let Θ be a terminal node of $\Delta(T)$. Then, $L_t^{(k)}(\Theta) = 0$: the decision tree that contains only one node labeled with the decision attached to all rows of Θ is a decision tree for Θ. This tree has no working nodes.

Let Θ be a nonterminal node of $\Delta(T)$ such that, for each child Θ' of Θ, we already know the value $L_w^{(k)}(\Theta')$. Based on this information, we can find the minimum number of working nodes in a decision tree for Θ, which uses for the subtables corresponding to children of the root decision trees of the type k and in which the root is labeled

- With an attribute from $F(T)$ (we denote by $L_{w,a}^{(k)}(\Theta)$ the minimum number of working nodes in such a decision tree).
- With a hypothesis over T (we denote by $L_{w,h}^{(k)}(\Theta)$ the minimum number of working nodes in such a decision tree).
- With a proper hypothesis over T (we denote by $L_{w,p}^{(k)}(\Theta)$ the minimum number of working nodes in such a decision tree).

We now describe three procedures for computing the values $L_{w,a}^{(k)}(\Theta)$, $L_{w,h}^{(k)}(\Theta)$, and $L_{w,p}^{(k)}(\Theta)$, respectively. Since Θ is nondegenerate, the set $E(\Theta)$ is nonempty.

Computation of $L_{w,a}^{(k)}(\Theta)$. Construct the set of attributes $E(\Theta)$. For each attribute $f_i \in E(\Theta)$, compute the value:

$$L_{w,a}^{(k)}(\Theta, f_i) = \sum_{\delta \in E(\Theta, f_i)} L_w^{(k)}(\Theta\{f_i = \delta\}).$$

Then, compute the value:

$$L_{w,a}^{(k)}(\Theta) = \min\{L_{w,a}^{(k)}(\Theta, f_i) : f_i \in E(\Theta)\}.$$

Computation of $L_{w,h}^{(k)}(\Theta)$. First, we construct a hypothesis:

$$H_\Theta = \{f_1 = \delta_1, \ldots, f_n = \delta_n\}$$

for Θ. Let $f_i \in F(T) \setminus E(\Theta)$. Then, δ_i is equal to the only number in the set $E(\Theta, f_i)$. Let $f_i \in E(\Theta)$. Then, δ_i is the minimum number from $E(\Theta, f_i)$ for which $L_w^{(k)}(\Theta\{f_i = \delta_i\}) = \max\{L_w^{(k)}(\Theta\{f_i = \sigma\}) : \sigma \in E(\Theta, f_i)\}$. Then

$$L_{w,h}^{(k)}(\Theta) = 1 + \sum_{f_i \in E(\Theta), \sigma \in E(\Theta, f_i) \setminus \{\delta_i\}} L_w^{(k)}(\Theta\{f_i = \sigma\}).$$

Computation of $L_{w,p}^{(k)}(\Theta)$. For each row $r = (\delta_1, \ldots, \delta_n)$ of the decision table T, we check if the corresponding proper hypothesis $H_r = \{f_1 = \delta_1, \ldots, f_n = \delta_n\}$ is admissible for Θ. For each admissible for Θ proper hypothesis H_r, we compute the value:

$$L_{w,p}^{(k)}(\Theta, H_r) = 1 + \sum_{f_i \in E(\Theta), \sigma \in E(\Theta, f_i) \setminus \{\delta_i\}} L_w^{(k)}(\Theta\{f_i = \sigma\}).$$

One can show that the minimum among the obtained numbers is equal to $L_{w,p}^{(k)}(\Theta)$.

We now consider an algorithm $\mathcal{A}_{L_w}$ that, for a given nonempty decision table T and $k \in \{1, \ldots, 5\}$, calculates the value $L_w^{(k)}(T)$, which is the minimum number of working nodes in a decision tree of the type k for the table T. During the work of this algorithm, we find for each node Θ of the DAG $\Delta(T)$ the value $L_w^{(k)}(\Theta)$.

The description of the algorithm $\mathcal{A}_{L_w}$ is similar to the description of the algorithm $\mathcal{A}_h$. Instead of $h^{(k)}$, we should use $L_w^{(k)}$. For each $b \in \{a, h, p\}$, instead of $h_b^{(k)}$, we should use $L_{w,b}^{(k)}$. In particular, for each terminal node Θ, $L_w^{(k)}(\Theta) = 0$.

One can show that the procedures of computation of the values $L_{w,a}^{(k)}(\Theta)$, $L_{w,h}^{(k)}(\Theta)$, and $L_{w,p}^{(k)}(\Theta)$ have polynomial time complexity depending on the size of the decision table T. Using this fact, one can prove the following statement.

Proposition 8. *The time complexity of the algorithm $\mathcal{A}_{L_w}$ 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.*

A similar bound can be obtained for the space complexity of the considered algorithm.

9. On Number of Realizable Terminal Nodes

Based on the results of experiments, we formulated the following hypothesis: $L_t^{(1)}(T) = L_t^{(3)}(T) = L_t^{(5)}(T)$ for any decision table T. In this section, we prove it. First, we consider a simple lemma.

Lemma 9. *Let T be a decision table and T' be a subtable of the table T. Then, $L_t^{(3)}(T') \leq L_t^{(3)}(T)$.*

Proof. It is easy to prove the considered inequality if T' is degenerate. Let T' be nondegenerate and Γ be a decision tree of the type 3 for T with the minimum number of realizable relative to T terminal nodes. Then, the root r of Γ is a working node. It is clear that the table $T'S(\Gamma, r)$ is nondegenerate. For each working node v of Γ such that the table $T'S(\Gamma, v)$ is degenerate and the table $T'S(\Gamma, v')$ is nondegenerate, where v' is the parent of v, we do the following. We remove all nodes and edges of the subtree of Γ with the root v with the exception of the node v. If $T'S(\Gamma, v) = \Lambda$, then we label the node v with the number 0. If the subtable $T'S(\Gamma, v)$ is nonempty, then we label the node v with the decision attached to each row of this subtable. We denote by Γ' the obtained decision tree. One can show that Γ' is a decision tree of the type 3 for the table T' and $L_t^{(3)}(T', \Gamma') \leq L_t^{(3)}(T, \Gamma)$. Therefore, $L_t^{(3)}(T') \leq L_t^{(3)}(T)$. □

Proposition 10. *For any decision table T, the following equalities hold:*

$$L_t^{(1)}(T) = L_t^{(3)}(T) = L_t^{(5)}(T).$$

Proof. It is clear that $L_t^{(3)}(T) \leq L_t^{(5)}(T) \leq L_t^{(1)}(T)$ for any decision table T. To prove the considered statement, it is enough to show that $L_t^{(1)}(T) \leq L_t^{(3)}(T)$ for any decision table T. We will prove this inequality by induction on the number of attributes in the set $E(T)$.

We now show that $L_t^{(1)}(T) \leq L_t^{(3)}(T)$ for any decision table T with $|E(T)| = 0$. If $|E(T)| = 0$, then either the table T is empty or the table T contains one row. Let T be empty. In this case, the decision tree that contains only one node labeled with 0 is considered as a decision tree for T. The only node of this tree is not realizable relative to T. Therefore, $L_t^{(1)}(T) = L_t^{(3)}(T) = 0$. Let T contain one row. In this case, the decision tree that contains only one node labeled with the decision attached to the row of T is a decision tree for T. The only node of this tree is realizable relative to T. Therefore, $L_t^{(1)}(T) = L_t^{(3)}(T) = 1$.

Let $n \geq 1$ and, for any decision table T with $|E(T)| \leq n - 1$, the inequality $L_t^{(1)}(T) \leq L_t^{(3)}(T)$ hold. Let T be a decision table with $|E(T)| = n$ and T have $m \geq n$ columns labeled with the attributes $f_1, \ldots, f_m$. Let, for the definiteness, $E(T) = \{f_1, \ldots, f_n\}$. If T is a degenerate table, then, as it is easy to show, $L_t^{(1)}(T) = L_t^{(3)}(T) = 1$. Let T be nondegenerate.

We denote by Γ a decision tree of the type 3 for the table T for which $L_t(T,\Gamma) = L_t^{(3)}(T)$ and Γ has the minimum number of nodes among such decision trees. One can show that the root of Γ is either labeled with an attribute from $E(T)$ or with a hypothesis over T that is admissible for T. We now prove that the tree Γ can be transformed into a decision tree Γ^* of the type 1 for the table T such that $L_t(T,\Gamma^*) \leq L_t^{(3)}(T)$.

Let the root of Γ be labeled with an attribute $f_i \in E(T)$. Then, for each $\sigma \in E(T,f_i)$, the root of Γ has a child v_σ such that $TS(\Gamma,v_\sigma) = T\{f_i = \sigma\}$ and the root of Γ has no other children. Since $f_i \in E(T)$, $|E(T\{f_i = \sigma\}| \leq n-1$. Using the inductive hypothesis, we obtain that there is a decision tree Γ_σ of the type 1 for the table $T\{f_i = \sigma\}$ such that $L_t(T\{f_i = \sigma\},\Gamma_\sigma) \leq L_t^{(3)}(T\{f_i = \sigma\})$. For each child v_σ of the root of Γ, we replace the subtree of Γ with the root v_σ with the tree Γ_σ. As a result, we obtain a decision tree Γ^* of the type 1 for the table T such that $L_t(T,\Gamma^*) \leq L_t(T,\Gamma) = L_t^{(3)}(T)$.

Let the root of Γ be labeled with a hypothesis $H = \{f_1 = \delta_1, \ldots, f_m = \delta_m\}$ over T that is admissible for T; see Figure 1, which depicts a prefix of the tree Γ. The root of Γ has a child v_0 such that $TS(\Gamma,v_0) = TH = T\{f_1 = \delta_1, \ldots, f_n = \delta_n\}$. For each $f_i \in E(T)$ and each $\sigma_i \in E(T,f_i) \setminus \{\delta_i\}$, the root of Γ has a child v_{i,σ_i} such that $TS(\Gamma,v_{i,\sigma_i}) = T\{f_i = \sigma_i\}$. The root of Γ has no other children.

We transform the tree Γ into a decision tree Γ^* of the type 1 for the table T; see Figure 2, which depicts a prefix of the tree Γ^*. For the node u_0 of the considered prefix, $TS(\Gamma^*,u_0) = T\{f_1 = \delta_1, \ldots, f_n = \delta_n\} = TH$. For each $f_i \in E(T)$ and each $\sigma_i \in E(T,f_i) \setminus \{\delta_i\}$, the node of this prefix labeled with the attribute f_i has a child u_{i,σ_i} such that $TS(\Gamma^*,u_{i,\sigma_i}) = T\{f_1 = \delta_1, \ldots, f_{i-1} = \delta_{i-1}, f_i = \sigma_i\}$. It is clear that $TS(\Gamma^*,u_{i,\sigma_i})$ is a subtable of $TS(\Gamma,v_{i,\sigma_i})$. By Lemma 9, $L_t^{(3)}(TS(\Gamma^*,u_{i,\sigma_i})) \leq L_t^{(3)}(TS(\Gamma,v_{i,\sigma_i}))$. It is also clear that $|E(TS(\Gamma^*,u_{i,\sigma_i})| \leq n-1$. Using the inductive hypothesis, we obtain that there is a decision tree Γ_{i,σ_i} of the type 1 for the table $TS(\Gamma^*,u_{i,\sigma_i})$ such that $L_t(TS(\Gamma^*,u_{i,\sigma_i}),\Gamma_{i,\sigma_i}) \leq L_t^{(3)}(TS(\Gamma^*,u_{i,\sigma_i})) \leq L_t^{(3)}(TS(\Gamma,v_{i,\sigma_i}))$.

We now transform the prefix of a decision tree Γ^* depicted in Figure 2 into a decision tree Γ^* of the type 1 for the table T. First, we transform the node u_0 into a terminal node labeled with the number 0 if $(\delta_1, \ldots, \delta_n)$ is not a row of T and labeled with the decision attached to $(\delta_1, \ldots, \delta_n)$ if this tuple is a row of T. Next, for each $f_i \in E(T)$ and each $\sigma_i \in E(T,f_i) \setminus \{\delta_i\}$, we replace the node u_{i,σ_i} with the tree Γ_{i,σ_i}. It is clear that the obtained tree Γ^* is a decision tree of the type 1 for the decision table T and $L_t(T,\Gamma^*) \leq L_t(T,\Gamma) = L_t^{(3)}(T)$.

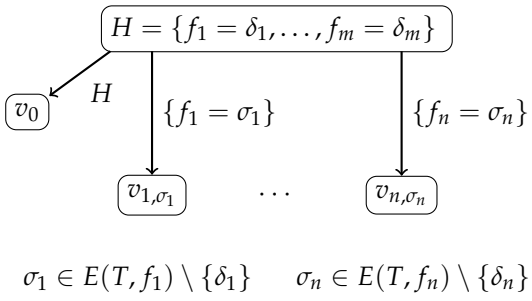

Figure 1. Prefix of decision tree Γ.

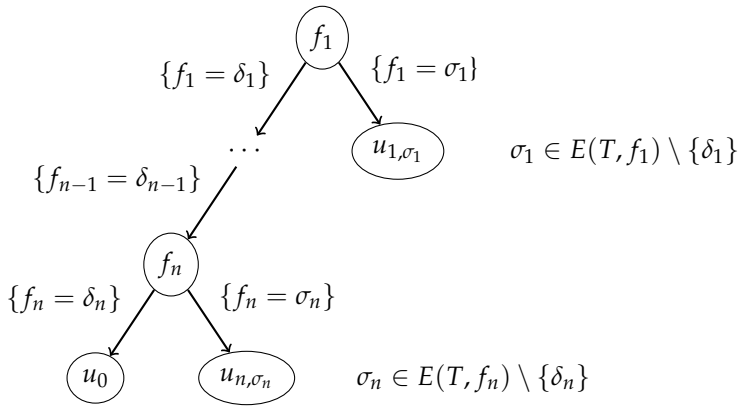

Figure 2. Prefix of decision tree Γ^*.

We proved that, for any decision table T, $L_t^{(1)}(T) \leq L_t^{(3)}(T)$; hence, $L_t^{(1)}(T) = L_t^{(3)}(T) = L_t^{(5)}(T)$. □

10. Results of Experiments

We conducted experiments with eight decision tables from the UCI ML Repository [20]. Table 1 contains information about each of these decision tables: its name, the number of rows, and the number of attributes. For each of the considered four cost functions, each of the considered five types of decision trees, and each of the considered eight decision tables, we find the minimum cost of a decision tree of the given type for the given table.

Table 1. Decision tables from Reference [20] used in experiments.

Decision Table	Number of Rows	Number of Attributes
BALANCE-SCALE	625	5
BREAST-CANCER	266	10
CARS	1728	7
HAYES-ROTH-DATA	69	5
NURSERY	12,960	9
SOYBEAN-SMALL	47	36
TIC-TAC-TOE	958	10
ZOO-DATA	59	17

For $n = 3, \ldots, 6$, we randomly generate 100 Boolean functions with n variables. We represent each Boolean function f with n variables $x_1, \ldots, x_n$ as a decision table T_f with n columns labeled with variables $x_1, \ldots, x_n$ considered as attributes and with 2^n rows that are all possible n-tuples of values of the variables. Each row is labeled with the decision that is the value of the function f on the corresponding n-tuple. We consider decision trees for the table T_f as decision trees computing the function f.

For each of the considered four cost functions, each of the considered five types of decision trees, and each of the generated Boolean functions, using its decision table representation, we find the minimum cost of a decision tree of the given type computing this function.

The following remarks clarify some experimental results considered later.

From Proposition 10, it follows that $L_t^{(1)}(T) = L_t^{(3)}(T) = L_t^{(5)}(T)$ for any decision table T.

Let f be a Boolean function with $n \geq 1$ variables. Since each hypothesis over the decision table T_f is proper, the following equalities hold:

$$h^{(2)}(T_f) = h^{(4)}(T_f), h^{(3)}(T_f) = h^{(5)}(T_f),$$
$$L^{(2)}(T_f) = L^{(4)}(T_f), L^{(3)}(T_f) = L^{(5)}(T_f),$$
$$L_t^{(2)}(T_f) = L_t^{(4)}(T_f), L_t^{(3)}(T_f) = L_t^{(5)}(T_f),$$
$$L_w^{(2)}(T_f) = L_w^{(4)}(T_f), L_w^{(3)}(T_f) = L_w^{(5)}(T_f).$$

10.1. Depth

In this section, we consider some results obtained in Reference [12]. Results of experiments with eight decision tables from Reference [20] and the depth are represented in Table 2. The first column contains the name of the considered decision table T. The last five columns contain values $h^{(1)}(T), \ldots, h^{(5)}(T)$ (minimum values for each decision table are in bold).

Table 2. Experimental results for decision tables from Reference [20]—depth.

Decision Table T	$h^{(1)}(T)$	$h^{(2)}(T)$	$h^{(3)}(T)$	$h^{(4)}(T)$	$h^{(5)}(T)$
BALANCE-SCALE	**4**	**4**	**4**	**4**	**4**
BREAST-CANCER	6	6	**5**	6	**5**
CARS	**6**	**6**	**6**	**6**	**6**
HAYES-ROTH-DATA	**4**	**4**	**4**	**4**	**4**
NURSERY	8	8	**7**	8	**7**
SOYBEAN-SMALL	**2**	4	**2**	6	**2**
TIC-TAC-TOE	6	6	**5**	8	6
ZOO-DATA	**4**	**4**	**4**	5	**4**
Average	5.00	5.25	4.63	5.88	4.75

Decision trees with the minimum depth using attributes (type 1) are optimal for 5 decision tables, using hypotheses (type 2) are optimal for 4 tables, using attributes and hypotheses (type 3) are optimal for 8 tables, using proper hypotheses (type 4) are optimal for 3 tables, using attributes and proper hypotheses (type 5) are optimal for 7 tables.

For the decision table SOYBEAN-SMALL, we must use attributes to construct an optimal decision tree. For this table, it is enough to use only attributes. For the decision tables BREAST-CANCER and NURSERY, we must use both attributes and hypotheses to construct optimal decision trees. For these tables, it is enough to use attributes and proper hypotheses. For the decision table TIC-TAC-TOE, we must use both attributes and hypotheses to construct optimal decision trees. For this table, it is not enough to use attributes and proper hypotheses.

Results of experiments with Boolean functions and the depth are represented in Table 3. The first column contains the number of variables in the considered Boolean functions. The last five columns contain information about values $h^{(1)}, \ldots, h^{(5)}$ in the format ${}_{min}Avg_{max}$.

Table 3. Experimental results for Boolean functions—depth.

Number of Variables n	$h^{(1)}$	$h^{(2)}$	$h^{(3)}$	$h^{(4)}$	$h^{(5)}$
3	${}_{2}2.82_{3}$	${}_{1}2.06_{3}$	${}_{1}1.89_{2}$	${}_{1}2.06_{3}$	${}_{1}1.89_{2}$
4	${}_{3}3.94_{4}$	${}_{2}3.05_{4}$	${}_{2}2.97_{3}$	${}_{2}3.05_{4}$	${}_{2}2.97_{3}$
5	${}_{4}4.95_{5}$	${}_{4}4.08_{5}$	${}_{3}3.99_{4}$	${}_{4}4.08_{5}$	${}_{3}3.99_{4}$
6	${}_{5}5.99_{6}$	${}_{5}5.01_{6}$	${}_{5}5.00_{5}$	${}_{5}5.01_{6}$	${}_{5}5.00_{5}$

From the obtained results, it follows that, generally, the decision trees of the types 2 and 4 are better than the decision trees of the type 1, and the decision trees of the types 3 and 5 are better than the decision trees of the types 2 and 4.

10.2. Number of Realizable Nodes

In this section, we consider some results obtained in Reference [13]. Results of experiments with eight decision tables from Reference [20] and the number of realizable nodes are represented in Table 4. The first column contains the name of the considered decision table T. The last five columns contain values $L^{(1)}(T), \ldots, L^{(5)}(T)$ (minimum values for each decision table are in bold).

Table 4. Experimental results for decision tables from Reference [20]—number of realizable nodes.

Decision Table T	$L^{(1)}(T)$	$L^{(2)}(T)$	$L^{(3)}(T)$	$L^{(4)}(T)$	$L^{(5)}(T)$
BALANCE-SCALE	501	5234	**499**	5234	**499**
BREAST-CANCER	161	18,061	**159**	24,226	**159**
CARS	396	65,250	**391**	65,250	**391**
HAYES-ROTH-DATA	**52**	317	**52**	338	**52**
NURSERY	1066	12,625,955	**1061**	12,625,955	**1061**
SOYBEAN-SMALL	**6**	4839	**6**	21,251	**6**
TIC-TAC-TOE	**244**	154,311	**244**	468,447	**244**
ZOO-DATA	**17**	1370	**17**	2847	**17**
Average	305	1,609,417	304	1,651,694	304

Decision trees with the minimum number of realizable nodes using attributes (type 1) are optimal for 4 decision tables, using hypotheses (type 2) are optimal for 0 tables, using attributes and hypotheses (type 3) are optimal for 8 tables, using proper hypotheses (type 4) are optimal for 0 tables, and using attributes and proper hypotheses (type 5) are optimal for 8 tables.

Decision trees of the types 3 and 5 can be a bit better than the decision trees of the type 1. Decision trees of the types 2 and 4 are far from the optimal.

For the decision tables HAYES-ROTH-DATA, SOYBEAN-SMALL, TIC-TAC-TOE, and ZOO-DATA, we must use attributes to construct optimal decision trees. For these tables, it is enough to use only attributes. For the rest of the considered decision tables, we must use both attributes and hypotheses to construct optimal decision trees. For these tables, it is enough to use attributes and proper hypotheses.

Results of experiments with Boolean functions and the number of realizable nodes are represented in Table 5. The first column contains the number of variables in the considered Boolean functions. The last five columns contain information about values $L^{(1)}, \ldots, L^{(5)}$ in the format ${}_{min}Avg_{max}$.

Table 5. Experimental results for Boolean functions—number of realizable nodes.

Number of Variables n	$L^{(1)}$	$L^{(2)}$	$L^{(3)}$	$L^{(4)}$	$L^{(5)}$
3	${}_{5}8.41_{13}$	${}_{5}12.38_{22}$	${}_{5}7.40_{12}$	${}_{5}12.38_{22}$	${}_{5}7.40_{12}$
4	${}_{9}16.26_{25}$	${}_{14}43.89_{66}$	${}_{8}14.59_{25}$	${}_{14}43.89_{66}$	${}_{8}14.59_{25}$
5	${}_{17}30.42_{41}$	${}_{113}201.95_{283}$	${}_{17}27.83_{39}$	${}_{113}201.95_{283}$	${}_{17}27.83_{39}$
6	${}_{49}58.94_{77}$	${}_{638}1057.16_{1406}$	${}_{46}54.13_{71}$	${}_{638}1057.16_{1406}$	${}_{46}54.13_{71}$

From the obtained results, it follows that, generally, the decision trees of the types 3 and 5 are slightly better than the decision trees of the type 1, and the decision trees of the types 2 and 4 are far from the optimal.

10.3. Number of Realizable Terminal Nodes

Results of experiments with eight decision tables from Reference [20] and the number of realizable terminal nodes are represented in Table 6. The first column contains the name of the considered decision table T. The last five columns contain values $L_t^{(1)}(T), \ldots, L_t^{(5)}(T)$ (minimum values for each decision table are in bold).

Table 6. Experimental results for decision tables from Reference [20]—number of realizable terminal nodes.

Decision Table T	$L_t^{(1)}(T)$	$L_t^{(2)}(T)$	$L_t^{(3)}(T)$	$L_t^{(4)}(T)$	$L_t^{(5)}(T)$
BALANCE-SCALE	**401**	4369	**401**	4369	**401**
BREAST-CANCER	**99**	15,577	**99**	21,208	**99**
CARS	**290**	50,260	**290**	50,260	**290**
HAYES-ROTH-DATA	**36**	240	**36**	259	**36**
NURSERY	**759**	9,643,103	**759**	9,643,103	**759**
SOYBEAN-SMALL	**4**	4508	**4**	19963	**4**
TIC-TAC-TOE	**155**	125,604	**155**	401,862	**155**
ZOO-DATA	**10**	1217	**10**	2560	**10**
Average	219	1,230,610	219	1,267,948	219

Decision trees of the types 1, 3, and 5 are optimal for each of the considered tables. Decision trees of the types 2 and 4 are far from the optimal.

Results of experiments with Boolean functions and the number of realizable terminal nodes are represented in Table 7. The first column contains the number of variables in the considered Boolean functions. The last five columns contain information about values $L_t^{(1)}, \ldots, L_t^{(5)}$ in the format ${}_{min}Avg_{max}$.

Table 7. Experimental results for Boolean functions—number of realizable terminal nodes.

Number of Variables n	$L_t^{(1)}$	$L_t^{(2)}$	$L_t^{(3)}$	$L_t^{(4)}$	$L_t^{(5)}$
3	${}_{3}4.70_{7}$	${}_{4}8.83_{14}$	${}_{3}4.70_{7}$	${}_{4}8.83_{14}$	${}_{3}4.70_{7}$
4	${}_{5}8.63_{13}$	${}_{11}31.53_{45}$	${}_{5}8.63_{13}$	${}_{11}31.53_{45}$	${}_{5}8.63_{13}$
5	${}_{9}15.71_{21}$	${}_{86}145.68_{200}$	${}_{9}15.71_{21}$	${}_{86}145.68_{200}$	${}_{9}15.71_{21}$
6	${}_{25}29.97_{39}$	${}_{485}770.52_{1005}$	${}_{25}29.97_{39}$	${}_{485}770.52_{1005}$	${}_{25}29.97_{39}$

From the obtained results, it follows that, generally, the decision trees of the types 1, 3, and 5 are optimal, and the decision trees of the types 2 and 4 are far from the optimal.

10.4. Number of Working Nodes

Results of experiments with eight decision tables from Reference [20] and the number of working nodes are represented in Table 8. The first column contains the name of the considered decision table T. The last five columns contain values $L_w^{(1)}(T), \ldots, L_w^{(5)}(T)$ (minimum values for each decision table are in bold).

Table 8. Experimental results for decision tables from Reference [20]—number of working nodes.

Decision Table T	$L_w^{(1)}(T)$	$L_w^{(2)}(T)$	$L_w^{(3)}(T)$	$L_w^{(4)}(T)$	$L_w^{(5)}(T)$
BALANCE-SCALE	100	865	**98**	865	**98**
BREAST-CANCER	49	2415	**45**	2926	**45**
CARS	104	14,981	**99**	14,981	**99**
HAYES-ROTH-DATA	16	77	**14**	77	**14**
NURSERY	286	2,980,719	**281**	2,980,719	**281**
SOYBEAN-SMALL	**2**	330	**2**	1281	**2**
TIC-TAC-TOE	88	27,867	**85**	65,104	86
ZOO-DATA	**7**	151	**7**	284	**7**
Average	82	378,426	79	383,280	79

Decision trees with the minimum number of working nodes using attributes (type 1) are optimal for 2 decision tables, using hypotheses (type 2) are optimal for 0 tables, using attributes and hypotheses (type 3) are optimal for 8 tables, using proper hypotheses (type 4) are optimal for 0 tables, using attributes and proper hypotheses (type 5) are optimal for 7 tables.

Decision trees of the types 3 and 5 can be a bit better than the decision trees of the type 1. Decision trees of the types 2 and 4 are far from the optimal.

For all decision tables with the exception of SOYBEAN-SMALL and ZOO-DATA, we must use both attributes and hypotheses to construct optimal decision trees. Moreover, for TIC-TAC-TOE, it is not enough to use attributes and proper hypotheses. For SOYBEAN-SMALL and ZOO-DATA, it is enough to use only attributes to construct optimal decision trees.

Results of experiments with Boolean functions and the number of working nodes are represented in Table 9. The first column contains the number of variables in the considered Boolean functions. The last five columns contain information about values $L_w^{(1)}, \ldots, L_w^{(5)}$ in the format ${}_{min}Avg_{max}$.

Table 9. Experimental results for Boolean functions—number of working nodes.

Number of Variables n	$L_w^{(1)}$	$L_w^{(2)}$	$L_w^{(3)}$	$L_w^{(4)}$	$L_w^{(5)}$
3	${}_{2}3.70_{6}$	${}_{1}3.55_{8}$	${}_{1}2.58_{4}$	${}_{1}3.55_{8}$	${}_{1}2.58_{4}$
4	${}_{4}7.63_{12}$	${}_{3}12.36_{21}$	${}_{3}5.62_{9}$	${}_{3}12.36_{21}$	${}_{3}5.62_{9}$
5	${}_{8}14.71_{20}$	${}_{27}56.25_{83}$	${}_{8}11.38_{15}$	${}_{27}56.25_{83}$	${}_{8}11.38_{15}$
6	${}_{24}28.97_{38}$	${}_{153}286.52_{401}$	${}_{19}22.76_{29}$	${}_{153}286.52_{401}$	${}_{19}22.76_{29}$

From the obtained results, it follows that, generally, the decision trees of the types 3 and 5 are better than the decision trees of the type 1, and the decision trees of the types 2 and 4 are far from the optimal.

We can now sum up the results of the experiments. Generally, the decision trees of the types 3 and 5 are slightly better than the decision trees of the type 1. Decision trees of the types 2 and 4 have, generally, too many nodes.

11. Conclusions

In this paper, we studied modified decision trees that use both queries based on one attribute each and queries based on hypotheses about values of all attributes. We designed dynamic programming algorithms for minimization of four cost functions for such decision trees and considered results of computer experiments. The main result of the paper is that the use of hypotheses can decrease the complexity of decision trees and make them more suitable for knowledge representation. In the future, we are planning to compare the length and coverage of decision rules derived from different types of decision trees constructed by the dynamic programming algorithms. Unfortunately, the considered algorithms cannot

work together to optimize more than one cost function. In the future, we are also planning to consider two extensions of these algorithms: (i) sequential optimization relative to a number of cost functions and (ii) bi-criteria optimization that allows us to construct for some pairs of cost functions the corresponding Pareto front.

Author Contributions: Conceptualization, all authors; methodology, all authors; software, I.C.; validation, I.C., M.A., and S.H.; formal analysis, all authors; investigation, M.A. and S.H.; resources, all authors; data curation, M.A. and S.H.; writing—original draft preparation, M.M.; writing—review and editing, all authors; visualization, M.A. and S.H.; supervision, I.C. and M.M.; project administration, M.M.; funding acquisition, M.M. All authors have read and agreed to the published version of the manuscript.

Funding: Research funded by King Abdullah University of Science and Technology.

Data Availability Statement: Data available in a publicly accessible repository that does not issue DOIs. Publicly available datasets were analyzed in this study. This data can be found here: http://archive.ics.uci.edu/ml (accessed on 26 May 2021)

Acknowledgments: Research reported in this publication was supported by King Abdullah University of Science and Technology (KAUST) including the provision of computing resources. The authors are greatly indebted to the anonymous reviewers for useful comments and suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Breiman, L.; Friedman, J.H.; Olshen, R.A.; Stone, C.J. *Classification and Regression Trees*, Wadsworth and Brooks; Springer: Berlin, Germany: 1984.
2. Moshkov, M. Time complexity of decision trees. In *Trans. Rough Sets III*; Lecture Notes in Computer Science; Peters, J.F., Skowron, A., Eds.; Springer: Berlin, Germany, 2005; Volume 3400, pp. 244–459.
3. Rokach, L.; Maimon, O. *Data Mining with Decision Trees—Theory and Applications*; Vol. 69, Series in Machine Perception and Artificial Intelligence; World Scientific: Singapore, 2007.
4. Chegis, I.A.; Yablonskii, S.V. Logical methods of control of work of electric schemes. *Trudy Mat. Inst. Steklov* **1958**, *51*, 270–360. (In Russian)
5. Pawlak, Z. Rough sets. *Int. J. Parallel Program.* **1982**, *11*, 341–356.
6. Pawlak, Z. *Rough Sets—Theoretical Aspects of Reasoning about Data*; Vol. 9, Theory and Decision Library: Series D; Kluwer: Dordrecht, the Netherlands, 1991.
7. Pawlak, Z.; Skowron, A. Rudiments of rough sets. *Inf. Sci.* **2007**, *177*, 3–27.
8. Angluin, D. Queries and concept learning. *Mach. Learn.* **1988**, *2*, 319–342.
9. Angluin, D. Queries revisited. *Theor. Comput. Sci.* **2004**, *313*, 175–194.
10. Angluin, D. Learning regular sets from queries and counterexamples. *Inf. Comput.* **1987**, *75*, 87–106.
11. Valiant, L.G. A theory of the learnable. *Commun. ACM* **1984**, *27*, 1134–1142.
12. Azad, M.; Chikalov, I.; Hussain, S.; Moshkov, M. Minimizing depth of decision trees with hypotheses (to appear). In Proceedings of the International Joint Conference on Rough Sets (IJCRS 2021), Bratislava, Slovakia, 19–24 September 2021.
13. Azad, M.; Chikalov, I.; Hussain, S.; Moshkov, M. Minimizing number of nodes in decision trees with hypotheses (to appear). In Proceedings of the 25th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES 2021), Szczecin, Poland, 8–10 September 2021.
14. Schmidhuber, J. Deep learning in neural networks: An overview. *Neural Netw.* **2015**, *61*, 85–117.
15. Ruiz-Garcia, A.; Schmidhuber, J.; Palade, V.; Took, C.C.; Mandic, D.P. Deep neural network representation and Generative Adversarial Learning. *Neural Netw.* **2021**, *139*, 199–200.
16. Shamshirband, S.; Fathi, M.; Chronopoulos, A.T.; Montieri, A.; Palumbo, F.; Pescapè, A. Computational intelligence intrusion detection techniques in mobile cloud computing environments: Review, taxonomy, and open research issues. *J. Inf. Secur. Appl.* **2020**, *55*, 102582.
17. Shamshirband, S.; Fathi, M.; Dehzangi, A.; Chronopoulos, A.T.; Alinejad-Rokny, H. A review on deep learning approaches in healthcare systems: Taxonomies, challenges, and open issues. *J. Biomed. Inform.* **2021**, *113*, 103627.
18. AbouEisha, H.; Amin, T.; Chikalov, I.; Hussain, S.; Moshkov, M. *Extensions of Dynamic Programming for Combinatorial Optimization and Data Mining*; Vol. 146, Intelligent Systems Reference Library; Springer: Berlin, Germany, 2019.
19. Alsolami, F.; Azad, M.; Chikalov, I.; Moshkov, M. *Decision and Inhibitory Trees and Rules for Decision Tables with Many-Valued Decisions*; Vol. 156, Intelligent Systems Reference Library; Springer: Berlin, Germany, 2020.
20. Dua, D.; Graff, C. UCI Machine Learning Repository. University of California, Irvine, School of Information and Computer Sciences. 2017. Available online: http://archive.ics.uci.edu/ml (accessed on 26 May 2021).