Decision and Inhibitory Trees for Decision Tables with Many-Valued Decisions

Thesis by
Mohammad Mohiuddin Azad

In Partial Fulfillment of the Requirements
For the Degree of
Doctor of Philosophy

King Abdullah University of Science and Technology
Thuwal, Kingdom of Saudi Arabia

June, 2018
The thesis of Mohammad Mohiuddin Azad is approved by the examination committee.

Committee Chairperson: Mikhail Moshkov
Committee Members: Vladimir Bajic, Xiangliang Zhang
External Committee Member: Endre Boros
ABSTRACT

Decision and Inhibitory Trees for Decision Tables with Many-Valued Decisions
Mohammad Mohiuddin Azad

Decision trees are one of the most commonly used tools in decision analysis, knowledge representation, machine learning, etc., for its simplicity and interpretability. We consider an extension of dynamic programming approach to process the whole set of decision trees for the given decision table which was previously only attainable by brute-force algorithms.

We study decision tables with many-valued decisions (each row may contain multiple decisions) because they are more reasonable models of data in many cases. To address this problem in a broad sense, we consider not only decision trees but also inhibitory trees where terminal nodes are labeled with “≠ decision”. Inhibitory trees can sometimes describe more knowledge from datasets than decision trees. As for cost functions, we consider depth or average depth to minimize time complexity of trees, and the number of nodes or the number of the terminal, or nonterminal nodes to minimize the space complexity of trees.

We investigate the multi-stage optimization of trees relative to some cost functions, and also the possibility to describe the whole set of strictly optimal trees. Furthermore, we study the bi-criteria optimization cost vs. cost and cost vs. uncertainty for decision trees, and cost vs. cost and cost vs. completeness for inhibitory trees.

The most interesting application of the developed technique is the creation of multi-pruning and restricted multi-pruning approaches which are useful for knowledge
representation and prediction. The experimental results show that decision trees constructed by these approaches can often outperform the decision trees constructed by the CART algorithm. Another application includes the comparison of 12 greedy heuristics for single- and bi-criteria optimization (cost vs. cost) of trees. We also study the three approaches (decision tables with many-valued decisions, decision tables with most common decisions, and decision tables with generalized decisions) to handle inconsistency of decision tables. We also analyze the time complexity of decision and inhibitory trees over arbitrary sets of attributes represented by information systems in the frameworks of local (when we can use in trees only attributes from problem description) and global (when we can use in trees arbitrary attributes from the information system) approaches.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examination Committee Page</td>
<td>2</td>
</tr>
<tr>
<td>Copyright</td>
<td>3</td>
</tr>
<tr>
<td>Abstract</td>
<td>4</td>
</tr>
<tr>
<td>List of Figures</td>
<td>11</td>
</tr>
<tr>
<td>List of Tables</td>
<td>13</td>
</tr>
<tr>
<td><strong>1 Introduction</strong></td>
<td>16</td>
</tr>
<tr>
<td>1.1 Related Works</td>
<td>22</td>
</tr>
<tr>
<td>1.1.1 Inconsistency</td>
<td>22</td>
</tr>
<tr>
<td>1.1.2 Various Learning Problems</td>
<td>23</td>
</tr>
<tr>
<td>1.2 Contribution</td>
<td>25</td>
</tr>
<tr>
<td>1.3 Organization of the Thesis</td>
<td>28</td>
</tr>
<tr>
<td><strong>2 Illustrative Examples</strong></td>
<td>30</td>
</tr>
<tr>
<td>2.1 Problems and Decision Tables</td>
<td>30</td>
</tr>
<tr>
<td>2.1.1 Problems with Many-Valued Decisions</td>
<td>30</td>
</tr>
<tr>
<td>2.1.2 Decision Tables Corresponding to Problems</td>
<td>33</td>
</tr>
<tr>
<td>2.2 Examples of Problems with Many-Valued Decisions</td>
<td>34</td>
</tr>
<tr>
<td>2.2.1 Problem of Three Post-Offices</td>
<td>34</td>
</tr>
<tr>
<td>2.2.2 Traveling Salesman Problem with Four Cities</td>
<td>36</td>
</tr>
<tr>
<td>2.2.3 Diagnosis of One-gate Circuit</td>
<td>39</td>
</tr>
<tr>
<td>2.2.4 Example of Data Table</td>
<td>40</td>
</tr>
<tr>
<td>2.3 Difference between Decision and Inhibitory Trees</td>
<td>42</td>
</tr>
<tr>
<td>2.3.1 Prediction Problem</td>
<td>42</td>
</tr>
<tr>
<td>2.3.2 Knowledge Representation Problem</td>
<td>43</td>
</tr>
<tr>
<td><strong>3 Preliminary Results for Decision and Inhibitory Trees</strong></td>
<td>48</td>
</tr>
<tr>
<td>3.1 Main Notions</td>
<td>48</td>
</tr>
</tbody>
</table>
3.1 Binary Decision Tables with Many-Valued Decisions . . . . . . 48
3.1.2 Decision Trees, Rule Systems and Tests . . . . . . . . . . . . . 49
3.1.3 Inhibitory Trees, Rule Systems, and Tests . . . . . . . . . . . . 51
3.1.4 Complementary Decision Table . . . . . . . . . . . . . . . . . 53
3.2 Relationships among Trees, Rule Systems, and Tests . . . . . . . . 56
3.2.1 Decision Interpretation . . . . . . . . . . . . . . . . . . . . . . 57
3.2.2 Inhibitory Interpretation . . . . . . . . . . . . . . . . . . . . . 58
3.3 Lower Bounds on Complexity of Trees . . . . . . . . . . . . . . . . 59
3.3.1 Decision Interpretation . . . . . . . . . . . . . . . . . . . . . . 60
3.3.2 Inhibitory Interpretation . . . . . . . . . . . . . . . . . . . . . 61
3.4 Upper Bounds on Complexity of Trees . . . . . . . . . . . . . . . . 65
3.4.1 Decision Interpretation . . . . . . . . . . . . . . . . . . . . . . 65
3.4.2 Inhibitory Interpretation . . . . . . . . . . . . . . . . . . . . . 65
3.5 Approximate Algorithms for Decision and Inhibitory Tree Optimization 66
3.5.1 Optimization of Decision Trees . . . . . . . . . . . . . . . . . . 66
3.5.2 Optimization of Inhibitory Trees . . . . . . . . . . . . . . . . . 68
3.6 Exact Algorithms for Optimization of Decision and Inhibitory Trees . 73
3.6.1 Decision Interpretation . . . . . . . . . . . . . . . . . . . . . . 73
3.6.2 Inhibitory Interpretation . . . . . . . . . . . . . . . . . . . . . 73

4 Decision Tables and Tools for Study of Pareto Optimal Points 74
4.1 Decision Tables . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 74
4.2 Uncertainty Measures . . . . . . . . . . . . . . . . . . . . . . . . . . . 76
4.3 Completeness Measures . . . . . . . . . . . . . . . . . . . . . . . . . . 77
4.4 Directed Acyclic Graph $\Delta_{U,\alpha}(T)$ . . . . . . . . . . . . . . . 78
4.5 Restricted Information Systems . . . . . . . . . . . . . . . . . . . . . 80
4.6 Time Complexity of Algorithms on $\Delta_{U,\alpha}(T)$ . . . . . . . . . . 84
4.7 Tools for Study of Pareto Optimal Points . . . . . . . . . . . . . . . 85

5 Decision and Inhibitory Trees 94
5.1 Different Kinds of Decision Trees . . . . . . . . . . . . . . . . . . . . . 94
5.1.1 Decision Trees for $T$ . . . . . . . . . . . . . . . . . . . . . . . . 94
5.1.2 $(U,\alpha)$-Decision Trees for $T$ . . . . . . . . . . . . . . . . . . 97
5.1.3 Cardinality of the Set $Tree(G,\Theta)$ . . . . . . . . . . . . . . . 99
5.1.4 $U^{\max}$-Decision Trees for $T$ . . . . . . . . . . . . . . . . . . 101
5.1.5 $U^{\sum}$-Decision Trees . . . . . . . . . . . . . . . . . . . . . . . 103
5.1.6 Cost Functions for Decision Trees . . . . . . . . . . . . . . . . . 104

7
5.2 Different Kinds of Inhibitory Trees .......................... 108

6 Multi-stage Optimization of Decision and Inhibitory Trees 114
   6.1 Multi-stage Optimization of Decision Trees  .................. 114
   6.2 Multi-stage Optimization of Inhibitory Trees  ................. 122
   6.3 Experimental Study of Totally Optimal Trees  .................. 125
      6.3.1 Decision Tables Used in Experiments  ...................... 125
      6.3.2 Totally Optimal Trees Relative to Two Cost Functions . 126
      6.3.3 Totally Optimal Trees Relative to Three Cost Functions . 127

7 Bi-criteria Optimization Problem for Decision and Inhibitory Trees:
   Cost vs. Cost 129
      7.1 Bi-criteria Optimization Problem for Decision Trees: Cost vs. Cost . 131
         7.1.1 Pareto Optimal Points: Cost vs. Cost ..................... 131
         7.1.2 Relationships between Two Cost Functions ................. 139
      7.2 Bi-criteria Optimization Problem for Inhibitory Trees: Cost vs. Cost 140
      7.3 Greedy Heuristics for Construction of Decision and Inhibitory Trees 142
         7.3.1 Uncertainty Measures  .................................. 143
         7.3.2 Impurity Types and Impurity Functions .................. 143
         7.3.3 Greedy Heuristics for Decision Tree Construction ....... 144
         7.3.4 Greedy Heuristics for Inhibitory Tree Construction .... 145
      7.4 Comparison of Greedy Heuristics  .............................. 145
      7.5 Decision Trees for Sorting .................................. 151
      7.6 Decision Trees for Knowledge Representation ................ 154

8 Bi-criteria Optimization Problem for Decision (Inhibitory) Trees:
   Cost vs. Uncertainty (Completeness) 157
      8.1 Bi-criteria Optimization Problem for Decision Trees:
         Cost vs. Uncertainty ........................................... 157
         8.1.1 Pareto Optimal Points: Cost vs. Uncertainty ............ 157
         8.1.2 Relationships between Cost and Uncertainty ............. 166
      8.2 Bi-criteria Optimization Problem for Inhibitory Trees: Cost vs. Completeness . 167
      8.3 Illustrative Examples ....................................... 169

9 Multi-pruning and Restricted Multi-pruning of Decision Trees 171
   9.1 Decision Tables ............................................... 174
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Problem of three post-offices</td>
<td>34</td>
</tr>
<tr>
<td>2.2</td>
<td>Distance to points $B_1$ and $B_2$</td>
<td>35</td>
</tr>
<tr>
<td>2.3</td>
<td>Decision table $T_1$ for problem of three post-offices</td>
<td>36</td>
</tr>
<tr>
<td>2.4</td>
<td>Decision tree with minimum depth for decision table $T_1$</td>
<td>36</td>
</tr>
<tr>
<td>2.5</td>
<td>Inhibitory tree with minimum depth for decision table $T'_1$</td>
<td>37</td>
</tr>
<tr>
<td>2.6</td>
<td>Traveling salesman problem with four cities</td>
<td>37</td>
</tr>
<tr>
<td>2.7</td>
<td>Decision table $T_2$ for traveling salesman problem with four cities</td>
<td>38</td>
</tr>
<tr>
<td>2.8</td>
<td>Decision tree with minimum depth for decision table $T_2$</td>
<td>38</td>
</tr>
<tr>
<td>2.9</td>
<td>Inhibitory tree with minimum depth for decision table $T'_2$</td>
<td>39</td>
</tr>
<tr>
<td>2.10</td>
<td>One-gate combinatorial circuit</td>
<td>39</td>
</tr>
<tr>
<td>2.11</td>
<td>Decision table $T_3$ for problem of diagnosis of one-gate circuit</td>
<td>40</td>
</tr>
<tr>
<td>2.12</td>
<td>Decision tree with minimum depth for decision table $T_3$</td>
<td>40</td>
</tr>
<tr>
<td>2.13</td>
<td>Inhibitory tree with minimum depth for decision table $T'_3$</td>
<td>41</td>
</tr>
<tr>
<td>2.14</td>
<td>Data table $D$</td>
<td>42</td>
</tr>
<tr>
<td>2.15</td>
<td>Decision table $T_4$ corresponding to data table $D$</td>
<td>42</td>
</tr>
<tr>
<td>2.16</td>
<td>Decision tree with minimum depth for decision table $T_4$</td>
<td>43</td>
</tr>
<tr>
<td>2.17</td>
<td>Inhibitory tree with minimum depth for decision table $T'_4$</td>
<td>43</td>
</tr>
<tr>
<td>2.18</td>
<td>Decision table $T_5$ with many-valued decisions</td>
<td>44</td>
</tr>
<tr>
<td>2.19</td>
<td>Inhibitory tree $\Gamma_1$ for decision table $T_5$</td>
<td>44</td>
</tr>
<tr>
<td>2.20</td>
<td>Inhibitory tree $\Gamma_2$ for decision table $T_5$</td>
<td>45</td>
</tr>
<tr>
<td>2.21</td>
<td>Data table (information system) $I$</td>
<td>45</td>
</tr>
<tr>
<td>2.22</td>
<td>Decision association tree $G_1$ for information system $I$</td>
<td>46</td>
</tr>
<tr>
<td>2.23</td>
<td>Decision association tree $G_2$ for information system $I$</td>
<td>46</td>
</tr>
<tr>
<td>2.24</td>
<td>Inhibitory association trees $\Gamma_1(a,b)$, $a,b \in {1,2}$, for informa-</td>
<td>47</td>
</tr>
<tr>
<td></td>
<td>tion system $I$</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.25</td>
<td>Inhibitory association trees $\Gamma_2(a,b)$, $a,b \in {1,2}$, for informa-</td>
<td>47</td>
</tr>
<tr>
<td></td>
<td>tion system $I$</td>
<td></td>
</tr>
<tr>
<td>7.1</td>
<td>Bi-criteria optimization of decision trees for LYMPH-5 relative to $h$ and $L$ - normalized POPs and normalized heuristic points</td>
<td>148</td>
</tr>
<tr>
<td>7.2</td>
<td>Decision table with many-valued decisions, $T_{sort}(3)$</td>
<td>153</td>
</tr>
</tbody>
</table>
7.3 Pareto optimal points for tables FLAGS-1 and LYMPH-5 for bi-criteria optimization problems relative to depth \( h \) and number of nodes \( L \), and relative to average depth \( h_{avg} \) and number of nodes \( L \).

8.1 Decision table CARS-1: depth vs. no. of misclassifications.

8.2 Decision table CARS-1: average depth vs. no. of misclassifications.

8.3 Decision table CARS-1: no. of nodes vs. no. of misclassifications.

9.1 Examples of decision tables.

9.2 Examples of decision tables with common decisions.

9.3 Pareto optimal points for decision tables with one-valued decisions and uncertainty measure \( gini \).

10.1 Transformation of inconsistent decision table \( T^0 \) into decision tables \( T^0_{MVP}, T^0_{GD}, \) and \( T^0_{MCD} \).
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1</td>
<td>Decision tables with many-valued decisions used in experiments</td>
<td>125</td>
</tr>
<tr>
<td>6.2</td>
<td>Existence of totally optimal decision and inhibitory trees for two cost functions</td>
<td>127</td>
</tr>
<tr>
<td>6.3</td>
<td>Existence of totally optimal decision trees for three cost functions</td>
<td>128</td>
</tr>
<tr>
<td>6.4</td>
<td>Existence of totally optimal inhibitory trees for three cost functions</td>
<td>128</td>
</tr>
<tr>
<td>7.1</td>
<td>Greedy heuristics</td>
<td>145</td>
</tr>
<tr>
<td>7.2</td>
<td>Decision tables with many-valued decisions used in experiments</td>
<td>146</td>
</tr>
<tr>
<td>7.3</td>
<td>Number of POPs</td>
<td>147</td>
</tr>
<tr>
<td>7.4</td>
<td>The minimum distance from normalized POPs to each normalized heuristic point for lymph-5 decision table for depth and number of nodes of decision trees</td>
<td>149</td>
</tr>
<tr>
<td>7.5</td>
<td>Average of ranks of heuristics for decision trees</td>
<td>149</td>
</tr>
<tr>
<td>7.6</td>
<td>Average of ranks of heuristics for inhibitory trees</td>
<td>150</td>
</tr>
<tr>
<td>7.7</td>
<td>Average relative difference (ARD)</td>
<td>150</td>
</tr>
<tr>
<td>7.8</td>
<td>Comparison of heuristics for decision trees</td>
<td>151</td>
</tr>
<tr>
<td>7.9</td>
<td>Comparison of heuristics for inhibitory trees</td>
<td>151</td>
</tr>
<tr>
<td>7.10</td>
<td>Minimum depth, minimum average depth, and minimum number of nodes in decision trees for decision table $T^2_{sort}(n)$, $n = 2, \ldots, 8$</td>
<td>152</td>
</tr>
<tr>
<td>7.11</td>
<td>Minimum depth, minimum average depth, and minimum number of nodes in decision trees for decision table $T^3_{sort}(n)$, $n = 2, \ldots, 6$</td>
<td>153</td>
</tr>
<tr>
<td>9.1</td>
<td>Decision tables with one-valued decisions from UCI ML Repository</td>
<td>182</td>
</tr>
<tr>
<td>9.2</td>
<td>Decision tables with many-valued decisions based on tables from UCI ML Repository</td>
<td>183</td>
</tr>
<tr>
<td>9.3</td>
<td>Average values of ranks for the average minimum distance from the origin to the Pareto optimal points for decision tables with one-valued decisions</td>
<td>186</td>
</tr>
<tr>
<td>9.4</td>
<td>Comparison of minimum distances from the origin to the Pareto optimal points for decision tables with one-valued decisions</td>
<td>187</td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>------</td>
</tr>
<tr>
<td>9.5</td>
<td>Comparison of minimum distance from the origin to the Pareto optimal points for MP and (2, 0) RMP algorithms for decision tables with one-valued decisions</td>
<td>187</td>
</tr>
<tr>
<td>9.6</td>
<td>Average values of ranks for the average minimum distance from the origin to the Pareto optimal points for decision tables with many-valued decisions</td>
<td>188</td>
</tr>
<tr>
<td>9.7</td>
<td>Comparison of minimum distances from the origin to the Pareto optimal points for decision tables with many-valued decisions</td>
<td>188</td>
</tr>
<tr>
<td>9.8</td>
<td>Comparison of minimum distance from the origin to the Pareto optimal points for MP and (2, 0) RMP algorithms for decision tables with many-valued decisions</td>
<td>189</td>
</tr>
<tr>
<td>9.9</td>
<td>Average values of ranks of classifiers based on $gini$ for decision tables with one-valued decisions</td>
<td>191</td>
</tr>
<tr>
<td>9.10</td>
<td>Comparison of average misclassification error rates of classifiers based on $gini$ for decision tables with one-valued decisions</td>
<td>192</td>
</tr>
<tr>
<td>9.11</td>
<td>Comparison of average time in seconds to construct and use classifiers based on $gini$ for decision tables with one-valued decisions</td>
<td>193</td>
</tr>
<tr>
<td>9.12</td>
<td>Average values of ranks of classifiers based on $abs$ for decision tables with one-valued decisions</td>
<td>194</td>
</tr>
<tr>
<td>9.13</td>
<td>Comparison of average misclassification error rates of classifiers based on $abs$ for decision tables with one-valued decisions</td>
<td>195</td>
</tr>
<tr>
<td>9.14</td>
<td>Comparison of average time in seconds to construct and use classifiers based on $abs$ for decision tables with one-valued decisions</td>
<td>195</td>
</tr>
<tr>
<td>9.15</td>
<td>Average values of ranks of classifiers based on $abs$ for decision tables with many-valued decisions</td>
<td>196</td>
</tr>
<tr>
<td>9.16</td>
<td>Average misclassification error rates of classifiers based on $abs$ for decision tables with many-valued decisions</td>
<td>196</td>
</tr>
<tr>
<td>9.17</td>
<td>Average time in seconds to construct and use of classifiers based on $abs$ for decision tables with many-valued decisions</td>
<td>197</td>
</tr>
<tr>
<td>10.1</td>
<td>Characteristics of decision tables obtained from UCI ML Repository data sets</td>
<td>202</td>
</tr>
<tr>
<td>10.2</td>
<td>Characteristics of decision tables from KEEL-Dataset Repository</td>
<td>202</td>
</tr>
<tr>
<td>10.3</td>
<td>Depth, average depth and number of nodes for decision trees constructed for tables $T_{MVD}$, $T_{MCD}$ and $T_{GD}$, UCI ML Repository</td>
<td>203</td>
</tr>
</tbody>
</table>
10.4 Depth, average depth and number of nodes for decision trees constructed for tables $T_{MVD}$, $T_{MCD}$ and $T_{GD}$, KEEL-Dataset Repository. 203

10.5 Greedy heuristics. 204

10.6 Average ranks of top three greedy heuristics. 206

10.7 Classification error rate (in %). 206
Chapter 1

Introduction

In usual decision tables, each row is labeled with one decision. We call such tables as decision tables with one-valued decisions. However, it is natural in some situations to consider decision tables in which each row is labeled with a set of decisions. We will study such tables and consider them as decision tables with many-valued decisions. Note that each decision table with one-valued decisions can be interpreted also as a decision table with many-valued decisions where each row is labeled with a set of decisions which has one element.

Decision tables with many-valued decisions can be found in problems of discrete optimization, pattern recognition, computational geometry, decision making, etc. [1, 2]. For example, finding a Hamiltonian circuit with the minimum length in the traveling salesman problem, finding the nearest post office in the post office problem [2]. In such cases, we give input with more than one optimal solutions. However, the main sources of decision tables with many-valued decisions are datasets with statistical or experimental data. It is common to have such tables in real life applications because of incomplete information about the domain. When we do not have enough attributes or features, we cannot completely separate multiple equal rows, and thus we have objects with equal values of conditional attributes but with different decisions.

Recently, it received significant attention in real life applications when we study, e.g., problem of semantic annotation of images [3] and videos [4] (news and movie clips), categorization of emotions from music [5], functional genomics [6] (gene and
protein functions), and text classification [7] (bookmarks, patents, news articles, emails).

In decision tables with many-valued decisions, we consider the following interpretation: for a given row (object), we should find a decision from the set of decisions attached to this row. Our main focus is on tree-based algorithms to discover knowledge from decision tables with many-valued decisions.

Decision trees are used as algorithms to solve problems given by decision tables, to represent knowledge from decision tables, etc. Depending on the goal, we should either minimize the time complexity of decision trees (depth or average depth), the space complexity of decision trees (number of nodes or number of terminal/nonterminal nodes), or both. It is well known that the considered problems are NP-hard (see, for example, [8], [2], [9]). However, it is possible to use dynamic programming for optimization of decision trees for medium-sized decision tables [10], [11], [12], [13]. The aim of these just cited papers was to find an optimal decision tree.

We consider not only decision but also inhibitory trees in which terminal (leaf) nodes are labeled with expressions of the kind “≠ decision”. Inhibitory trees sometimes can represent more information about decision tables than decision trees. We show that instead of inhibitory trees for a decision table we can study decision trees for the decision table complementary to the initial one.

This thesis is devoted to the study of multi-stage optimization, bi-criteria optimization (cost vs. cost and cost vs. uncertainty (completeness)) and time complexity analysis of the decision (inhibitory) trees for decision tables with many-valued decisions. Note that some of these results are extensions of the results obtained by Shahid Hussain for decision trees over decision tables with one-valued decisions [14] (see also [15]).

The first direction of our research is connected with the study of multi-stage (sequential) optimization of cost functions for decision and inhibitory trees. The
optimization technique considered here allows us not only to find an optimal tree but also to describe the set of all optimal trees or its subset by a directed acyclic graph (DAG). As a result, we can continue the process of optimization relative to another cost function, etc. If we have a number of criteria (cost functions) ordered from the most important to the least important, we can make multi-stage optimization relative to the considered sequence of cost functions. For example, if the number of nodes is the most important for us and the average depth is not quite as important, then we can first describe the set of all decision trees with the minimum number of nodes and, among these trees, we can describe all that have minimum average depth.

The second direction is connected with the study the relationships between cost and uncertainty of decision trees for decision tables with many-valued decisions. This is to understand the trade-off between, for example, number of nodes and number of misclassification and how we can correlate them. It can be used to represent the knowledge contained in data sets by exact or approximate decision trees. It also can be used for the classification purposes, i.e., to predict decisions for new instances. First, we need to create DAG on training data set and then construct the set of Pareto optimal points (non-dominated points) based on the relationship between the number of nodes and misclassification error. From each Pareto optimal point, we can derive a certain number of decision trees. After that, we can choose a single decision tree from them by evaluating its performance based on validation data set. We use testing data set to evaluate the accuracy of the constructed decision tree.

Using this strategy, we created a specific approach called multi-pruning where the DAG is constructed for the CART (Classification and Regression Trees)-like decision trees for the considered decision table. Such trees use binary splits based on the values of the conditional attributes. In contrast with the standard CART which uses the best (least impurity based on Gini index) splits among all attributes, CART-like trees use, additionally, the best splits for each attribute. The experimental
results show that decision trees constructed by the multi-pruning approach can often outperform the decision tree constructed by ‘CART’ method for decision tables with one-valued decision. But one flaw is that it is very time-consuming. Therefore, we create a restricted version of multi-pruning by restricting the number of used binary splits. The restricted multi-pruning is faster, and the accuracy of constructed classifiers is comparable with the regular multi-pruning. This approach is also extended to the case of decision tables with many-valued decisions.

Furthermore, we study the bi-criteria optimization of decision trees for decision tables with many-valued decisions relative to two cost functions. For this, we created algorithms for the construction of the sets of Pareto optimal points. It is well known that we can find all Pareto optimal points in a set of \( n \) points using \( O(n \log n) \) comparisons. Such algorithms can be used when \( n \) is a reasonable number. If \( n \) is too large, as, in the case of decision trees, we should use a special technique which is applicable when the number of the objects is huge. Such technique is based on the fact that, the number of Pareto optimal points for each subtable of the considered table is relatively small. For example, if one of the criteria is the depth of decision trees, then the number of Pareto optimal points for each subtable is at most one plus the number of conditional attributes in the decision table. We develop this technique and apply it to the cost functions for which the number of Pareto optimal points for subtables is comparable with the size of the decision table: depth, average depth, number of nodes, number of terminal nodes, and number of nonterminal nodes. The most part of the considered results is extended to the case of inhibitory trees.

Additionally, we study the approximate optimization approaches based on greedy heuristics. Since the above exact optimization problems are NP-hard, several researchers studied approximate optimization algorithms based on different approaches: genetic algorithms [19], ant colony algorithms [20], simulated annealing [21], etc. Most of the approximate algorithms for the generation of a decision tree are greedy: they
usually execute top-down partitioning of the table by minimizing some impurity at each step. Several impurity criteria are discussed in literature [22, 23, 24, 25, 26, 27, 28, 29]. In this thesis, we describe 12 greedy heuristics based on four uncertainty measures and three types of impurity functions. After that, we investigate the quality of the described greedy heuristics as algorithms for single- and bi-criteria optimization.

The similar technique is also used for inhibitory trees. We also consider two more applications of our method: experimental study of decision trees for sorting problem and study of two relationships for decision trees related to knowledge representation—number of nodes vs. depth and number of nodes vs. average depth.

We also consider a special case study of handling inconsistent decision tables that contain equal rows with different decisions. We investigate three approaches to deal with such tables:

a. Decision tables with many-valued decisions (MVD): in this approach, we keep one row from the group of equal rows and attach to this row the set of all decisions attached to rows from the group. Our goal is to find a single decision from the attached set of decisions.

b. Decision tables with most common decisions (MCD): in this approach, we keep one row from the group of equal rows and attach to this row the most common decision among decisions attached to the rows from the group.

c. Decision tables with generalized decisions (GD): in this approach, we keep one row from the group of equal rows and attach to this row a number (generalized decision) that encodes the set of all decisions attached to the rows from the group. Equal sets are encoded by the equal numbers, and different sets are encoded by the different numbers.

To compare the three approaches, we constructed decision trees in the frameworks of these approaches. Then we compared the depth, average depth, and a number of
nodes of the constructed decision trees. We also compared the accuracy of decision
tree classifiers based on these approaches. The experimental results show that the
decision trees constructed in the framework of MVD approach are usually simpler
than the trees constructed in the frameworks of MCD and GD approaches. The same
situation is with the accuracy of classifiers constructed in the frameworks of the three
approaches.

For experiments based on the above theoretical study, our group has created a
software tool DAGGER (from DAG) in KAUST which is based on extensions of dy-
namic programming. It is implemented in C++ and uses pthreads and MPI libraries
for managing threads and processes respectively. It runs on a single-processor com-
puter or multiprocessor system with shared memory [30]. This tool can be used to
work with the decision tables with many-valued decisions. It can construct exact
as well as approximate decision and inhibitory trees using both greedy and dynamic
programming algorithms.

Finally, we concentrate on the study of the time complexity of decision and in-
hibitory trees over arbitrary sets of attributes represented by information systems
(information system consists of a set of objects and a set of attributes defined on
these objects). For this, we study both local and global approaches. The local ap-
proach assumes that trees can use only attributes from the problem description. The
global approach allows us to use arbitrary attributes from the whole set of attributes.
Finally, for each information system and for each approach (local and global), we
investigate the behavior of two Shannon functions which characterize the growth in
the worst case of (i) minimum depth of decision trees and (ii) minimum depth of
inhibitory trees with the growth of number of attributes in the problem description.
1.1 Related Works

We have divided this section into two parts. In the first part, we discuss different research works that demonstrate how to overcome the inconsistency of decision tables. In the second part, we discuss the various learning problems from literature, and how our problem is different from others.

1.1.1 Inconsistency

In usual decision tables, a single decision is associated with each observation (row). However, inconsistent decision tables containing equal rows labeled with different decisions are possible. Such tables are studied in rough set theory [31, 32]. In this case, for each object, we know a set of decisions to which the decision belongs, but we do not have enough information to recognize the decision. Therefore, the notion of generalized decision has been used to handle inconsistency where we have to recognize the whole set of decision for a given row.

Note that, our interpretation (recognize a single decision from the set of decisions for the given row) is different from the generalized decision (recognize the whole set of decisions attached to the given group of equal rows). However, our interpretation makes sense also for inconsistent decision tables. For example, if we consider a decision table corresponding to an optimization problem, the choice of an arbitrary decision from the set attached to the row means that we would like to return not all optimal decisions but only one. In the case of inconsistent decision tables, for a given object, we return the decision for this object or the decision for an object indistinguishable from the given one. Our previous experiments [33, 34, 35, 36, 37] showed that decision trees constructed in our framework often have less complexity than decision trees built in the framework of the generalized decision.
1.1.2 Various Learning Problems

Traditional supervised learning approaches deal with the problems where a row is represented by an instance of the object and associated with a single decision. The goal is to correctly label the unseen rows.

But real-world problems are more complicated than the traditional simple supervised model, and each object may be associated with multiple concepts simultaneously. For example, an image can belong to several classes simultaneously, i.e., beach, sea, mountain; news can be categorized into sports, political, lifestyle; in medical diagnosis, a patient can suffer from diabetes and prostate cancer at the same time.

In literature, often, problems that are connected with multi-label data are considered in different ways: multi-label learning [38], multiple instances learning [39], multi-instance multi-label learning [40].

In multi-label learning, a row is represented by an instance of the object and associated with multiple decisions. There are two main approaches [38] to deal with multi-label learning: algorithm adaptation methods, and problem transformation methods. In algorithm adaptation method, it directly handles multi-label data by extending the known learning algorithms. Some examples are lazy learning, neural network, boosting, classification rules, decision trees, etc. [41]. On the other hand, the problem transformation methods transform the multi-label learning problem into one or more single-label learning problems. These single-label problems are solved with common single-label learning algorithms, and after that, the result is transformed back into the multi-label representation. There are two simple strategies for it: binary relevance method which is one-against-all strategy and pairwise method which is one-against-one strategy [41]. The authors of the paper [41] also used another group of methods named ensemble methods. Some examples are RAkEL [42], ensembles of classification chains (ECC) [43], random forests of predictive clustering trees [44], and random forests of multi-label C4.5 trees [45].
In multiple instance learning, the task is to learn a concept given positive and negative bags of instances. A bag is labeled negatively if all the instances in it are negative [39]. The goal is to label unseen bags correctly. Note that although the training bags are labeled, the labels of their instances are unknown [40].

Multi-instance multi-label (MIML) learning is a combination of both multiple instances and multi-label learning problem. Multi-instance learning studies the ambiguity in the input space (or instance space), where an object has many alternative input descriptions, i.e., instances; multi-label learning studies the ambiguity in the output space (or label space), where an object has many alternative output descriptions, i.e., labels; while MIML considers the ambiguities in both the input and output spaces simultaneously [40]. It considers the problem where a row is described by multiple instances and associated with multiple decisions.

There is also semi-supervised learning [46] where some rows are labeled, but some are not labeled. The study of semi-supervised learning is motivated by two factors: its practical value in building better computer algorithms, and its theoretical value in understanding learning in machines and humans. Semi-supervised learning has tremendous practical value. In many tasks, there is a scarcity of labeled data. The labels may be difficult to obtain because they require human annotators, special devices, or expensive and slow experiments [46].

There is another learning problem which is mentioned in different names in literature: partial learning [47], ambiguous learning [48], and multiple label learning [49]. In this learning problem, each row is associated with multiple decisions but only one decision is correct, and all others are incorrect. The goal is to find out which decision is correct. In [47, 49], the authors show probabilistic methods to solve the learning problem whereas, in [48], the author used standard heuristic approach to exploit inductive bias to disambiguate decision information.

In this thesis, we consider mainly the problem of knowledge representation and op-
timization of the data model. We choose decision and inhibitory trees as a data model to represent knowledge from decision tables with many-valued decisions. Also, our goal is to optimize the tree structure (number of nodes, number of terminal/nonterminal nodes, depth, average depth) to efficiently manage the knowledge in such tables. However, when we consider learning problem (in the case of multi-pruning), we see each row is represented by an instance of the object and associated with multiple decisions. We assume that all decisions in the set of decision attached to this row are correct for this row. Our goal is to find for each row a single (arbitrary) decision from the set of decisions attached to it and predict a single decision for the new (unlabeled) row.

1.2 Contribution

The main novelty of this thesis lies in the study of extensions of the dynamic programming for the decision tables with many-valued decisions. The contribution of this thesis can be directed into six categories:

- The first contribution of this thesis is the introduction of different uncertainty (completeness) measures which are used to define approximate decision (inhibitory) trees for the decision tables with many-valued decisions. Uncertainty measures are also used for the design of greedy algorithms for decision and inhibitory tree construction.

- The second contribution is the study of multi-stage optimization of the decision and inhibitory trees for decision tables with many-valued decisions relative to different cost functions such as depth, number of nodes, average depth. In [50], it was proven that the result of the optimization of decision trees relative to a strictly increasing cost function (for example, the number of nodes or average depth) is the set of optimal trees and that the result of the optimization of decision trees relative to an increasing cost function (for example, depth) is a
subset of the set of optimal trees. Here, we introduce the notion of a strictly optimal decision tree. For each node of such a tree, the subtree with its root in this node is optimal for the subtable corresponding to this node. We prove that the result of optimization of decision trees relative to an increasing cost function is the set of strictly optimal decision trees and, for a strictly increasing cost function, the set of optimal decision trees coincides with the set of strictly optimal decision trees.

We use this tool for the study of totally optimal decision trees (decision trees that are simultaneously optimal relative to two or more given cost functions) for decision tables with many-valued decisions [16]. The same technique can be used for inhibitory trees as well. A similar technique was used earlier to study of totally optimal decision trees for Boolean functions [9]. Previously, the existence of such trees was proven in rare cases using nontrivial techniques (for example, the existence of decision trees for sorting six elements that are simultaneously optimal relative to depth, average depth, and a number of nodes (Knuth, 1998)).

- The third contribution is the study of relationships between cost function and uncertainty (completeness) measure and relationships between two different cost functions for decision (inhibitory) trees for decision tables with many-valued decisions. The novelty in this category lies in (i) the development of bi-criteria optimization (cost vs. cost and cost vs. uncertainty) of decision trees for decision tables with many-valued decisions, (ii) the extension of the obtained results to the inhibitory trees, and (iii) the comparison of 12 greedy heuristics as algorithms for single- and bi-criteria optimization (cost vs. cost) of decision and inhibitory trees for decision tables with many-valued decisions.

- The fourth contribution is the applications to machine learning problems using
multi-pruning and restricted multi-pruning approaches. This is one of the most interesting contributions of this thesis where the bi-criteria optimization for cost and uncertainty plays a major role not only in knowledge representation but also in the classification of new instances. The first part of this work is based on decision tables with one-valued decision which is joint work with Shahid Hussain, and initial results were published in [17]. After that, these results have been extended to the case of decision tables with many-valued decisions using the new uncertainty measure ‘abs’.

- The fifth contribution is the study of the three approaches to handle inconsistency in decision tables. We developed three different approaches to handle inconsistency: decision tables with many-valued decisions, decision tables with most common decisions, and decision tables with generalized decisions. We compare the complexity of the constructed decision trees and accuracy of the constructed decision tree classifiers for the three approaches.

- The sixth contribution is the study of the time complexity of decision and inhibitory trees over arbitrary information system in the frameworks of both local and global approaches. For each information system and for each approach (local and global), we investigate the behavior of two Shannon functions which characterize in the worst case the growth of (i) minimum depth of decision trees and (ii) minimum depth of inhibitory trees, with the growth of the number of attributes in the problem description.

We did some of our experiments on a workstation with 128 GB RAM and 16 cores while others on a high-performance computing system with 1.5 TB RAM and 60 cores.
1.3 Organization of the Thesis

The remainder of the thesis contains nine chapters:

Chapter 2 This chapter uses examples to explain the notion of the problems connected with decision tables with many-valued decisions, e.g., fault diagnosis, combinatorial optimization, computational geometry, and analysis of experimental data. We also discuss two examples which explain the situation when decision trees might not give complete information compared to inhibitory trees.

Chapter 3 This chapter contains basic notations and definitions regarding binary decision tables with many-valued decisions, decision, and inhibitory trees, rules and tests. It also contains the important definition of a complementary decision table. It describes the relationships among tests, rule systems, and trees. It also discusses the lower and upper bounds on the complexity of trees as well as approximate and exact algorithms for decision and inhibitory tree optimizations.

Chapter 4 This chapter contains the essential notation and definitions: the notions of the decision table, directed acyclic graph for this table, uncertainty and completeness measures, and restricted information system. It also discusses the theoretical and algorithmic results regarding Pareto optimality. The most part of the definitions and results considered in this chapter were obtained jointly with our group members Hassan AbouEisha, Fawaz Alsolami, Talha Amin, and Shahid Hussain (see also [15]).

Chapter 5 This chapter contains the discussion of different kinds of decision and inhibitory trees as well the cost functions for trees.

Chapter 6 This chapter contains the study of multi-stage optimization of the decision and inhibitory trees relative to a sequence of cost functions, and an appli-
cation of this technique: the study of totally optimal (simultaneously optimal relative to a number of cost functions) decision and inhibitory trees.

**Chapter 7** This chapter contains the study of bi-criteria optimization problem cost vs. cost for decision and inhibitory trees. It also shows the comparison among 12 greedy heuristics for construction of decision and inhibitory trees as single-criterion and bi-criteria optimization algorithms.

**Chapter 8** This chapter contains the study of bi-criteria optimization problems cost vs. uncertainty for decision trees and cost vs. completeness for inhibitory trees, and consider illustrative examples.

**Chapter 9** This chapter contains the discussion of multi-pruning and restricted multi-pruning approaches to constructing decision trees which can be used for knowledge representation and classification.

**Chapter 10** This chapter contains the study of three approaches to handle inconsistency in decision tables. We compare decision tree complexity and classification accuracy for these three approaches.

**Chapter 11** This chapter contains the study of time complexity of decision and inhibitory trees over arbitrary sets of attributes represented by information systems.
Chapter 2

Illustrative Examples

In this chapter, we discuss briefly main notions: problems with many-valued decisions, decision tables corresponding to these problems, and decision and inhibitory trees. We consider here only the depth of trees.

After that, we concentrate on consideration of simple examples of problems with many-valued decisions from different areas of applications: fault diagnosis, computational geometry, combinatorial optimization, and analysis of data.

In the end, we discuss two examples which explain why we study not only decision but also inhibitory trees.

Note that the decision tables with many-valued decisions for the problem of three post-offices and for the traveling salesman problem with four cities were prepared jointly with Fawaz Alsolami. He also took part in the development of the last two examples.

2.1 Problems and Decision Tables

In this section, we discuss the notions of problems, decision tables, and decision and inhibitory trees.

2.1.1 Problems with Many-Valued Decisions

We begin with a simple model of a problem. Let $A$ be a set (set of inputs or the universe). It is possible that $A$ is an infinite set. Let $f_1, \ldots, f_n$ be attributes, each
of which is a function from $A$ to $B$ where $B$ is a nonempty finite set. Attributes $f_1, \ldots, f_n$ divide the set $A$ into some domains in each of which values of attributes are constant. These domains are labeled with nonempty finite subsets of the set $\omega = \{0, 1, 2, \ldots\}$ of nonnegative integers. We will interpret these subsets as sets of decisions.

More formally, a problem is a tuple $z = (\nu, f_1, \ldots, f_n)$ where $\nu$ is a mapping from $B^n$ to the set of all nonempty finite subsets of the set $\omega$. Each domain corresponds to the nonempty set of solutions over $A$ of a system of equations of the kind

$$\{ f_1(x) = \delta_1, \ldots, f_n(x) = \delta_n \}$$

where $\delta_1, \ldots, \delta_n \in B$. Denote $D(z) = \bigcup \nu(\delta_1, \ldots, \delta_n)$ where union is considered over all $(\delta_1, \ldots, \delta_n) \in B^n$ for which $\{ f_1(x) = \delta_1, \ldots, f_n(x) = \delta_n \}$ has a solution over $A$.

For a given $a \in A$, denote $z(a) = \nu(f_1(a), \ldots, f_n(a))$.

We will consider two interpretations of the problem $z$: decision and inhibitory.

Decision interpretation: for a given $a \in A$, we should find a number from the set $z(a)$.

Inhibitory interpretation: for a given $a \in A$, we should find a number from the set $D(z) \setminus z(a)$ (we will assume here that $D(z) \neq z(a)$ for any $a \in A$).

In the case of decision interpretation, we study decision trees.

A decision tree over $z$ is a finite directed tree with the root in which each terminal node is labeled with a number from $\omega$ (decision), each nonterminal node (such nodes will be called working nodes) is labeled with an attribute from the set $\{ f_1, \ldots, f_n \}$. Edges starting in a working node are labeled with distinct elements from $B$.

Let $\Gamma$ be a decision tree over $z$. For a given element $a \in A$, the work of the tree starts in the root. If the current node $v$ is terminal then $\Gamma$ finishes its work in this node. Let the current node $v$ be a working node labeled with an attribute $f_i$. If there
is an edge starting at \( v \) and labeled with \( f_i(a) \), then the computation passes along this edge, etc. Otherwise, \( \Gamma \) finishes its work in \( v \).

We will say that \( \Gamma \) solves the problem \( z \) in the decision interpretation if, for any \( a \in A \), \( \Gamma \) finishes its work in a terminal node labeled with a number belonging to \( z(a) \).

As time complexity of \( \Gamma \) we will consider the depth \( h(\Gamma) \) of \( \Gamma \) which is the maximum length of a path from the root to a terminal node of \( \Gamma \). We denote by \( h(z) \) the minimum depth of a decision tree over \( z \) which solves the problem \( z \) in decision interpretation.

In the case of inhibitory interpretation, we study inhibitory trees.

An inhibitory tree over \( z \) is a finite directed tree with the root in which each terminal node is labeled with an expression \( \neq t \), \( t \in \omega \), each nonterminal node (such nodes will be called working nodes) is labeled with an attribute from the set \( \{f_1, \ldots, f_n\} \). Edges starting in a working node are labeled with distinct elements from \( B \).

Let \( \Gamma \) be an inhibitory tree over \( z \). For a given element \( a \in A \), the work of the tree starts in the root. If the current node \( v \) is terminal then \( \Gamma \) finishes its work in this node. Let the current node \( v \) be a working node labeled with an attribute \( f_i \). If there is an edge starting at \( v \) and labeled with \( f_i(a) \), then the computation passes along this edge, etc. Otherwise, \( \Gamma \) finishes its work in \( v \).

We will say that \( \Gamma \) solves the problem \( z \) in inhibitory interpretation if, for any \( a \in A \), \( \Gamma \) finishes its work in a terminal node labeled with an expression \( \neq t \) such that \( t \in D(z) \setminus z(a) \).

As time complexity of \( \Gamma \) we will consider the depth \( h(\Gamma) \) of \( \Gamma \) which is the maximum length of a path from the root to a terminal node of \( \Gamma \). We denote by \( ih(z) \) the minimum depth of an inhibitory tree over \( z \) which solves the problem \( z \) in inhibitory interpretation.
2.1.2 Decision Tables Corresponding to Problems

We associate a decision table \( T = T(z) \) with the considered problem \( z \).

This table is a rectangular table with \( n \) columns labeled with attributes \( f_1, \ldots, f_n \). A tuple \( (\delta_1, \ldots, \delta_n) \in B^n \) is a row of \( T \) if and only if the system of equations

\[
\{ f_1(x) = \delta_1, \ldots, f_n(x) = \delta_n \}
\]

is consistent (has a solution over the set \( A \)). This row is labeled with the set \( \nu(\delta_1, \ldots, \delta_n) \). Let \( \text{Row}(T) \) be the set of rows of \( T \). Denote

\[
D(T) = \bigcup_{(\delta_1, \ldots, \delta_n) \in \text{Row}(T)} \nu(\delta_1, \ldots, \delta_n).
\]

It is clear that \( D(T) = D(z) \).

We can formulate the notion of decision tree over \( T \), and describe the work of a decision tree over \( T \) on a row of \( T \) in a natural way. We will say that a decision tree \( \Gamma \) over \( T \) is a decision tree for \( T \) if, for any row \( (\delta_1, \ldots, \delta_n) \) of \( T \), \( \Gamma \) finishes its work in a terminal node labeled with a decision \( t \) such that \( t \in \nu(\delta_1, \ldots, \delta_n) \). It is not difficult to show that the set of decision trees for \( T \) coincides with the set of decision trees over \( z \) solving the problem \( z \) in decision interpretation. We denote by \( h(T) \) the minimum depth of decision tree for the table \( T = T(z) \). Then \( h(z) = h(T(z)) \).

When we study inhibitory trees for \( T \) we assume that

\[
\nu(\delta_1, \ldots, \delta_n) \neq D(T)
\]

for any row \( (\delta_1, \ldots, \delta_n) \) of \( T \).

We can formulate the notion of an inhibitory tree over \( T \), and describe the work of an inhibitory tree over \( T \) on a row of \( T \) in a natural way. We will say that an inhibitory tree \( \Gamma \) over \( T \) is an inhibitory tree for \( T \) if, for any row \( (\delta_1, \ldots, \delta_n) \) of
Figure 2.1: Problem of three post-offices

$T$, $\Gamma$ finishes its work in a terminal node labeled with an expression $\neq t$ such that $t \in D(T) \setminus \nu(\delta_1, \ldots, \delta_n)$. It is not difficult to show that the set of inhibitory trees for $T$ coincides with the set of inhibitory trees over $z$ solving the problem $z$ in inhibitory interpretation. We denote by $ih(T)$ the minimum depth of inhibitory tree for the table $T = T(z)$. Then $ih(z) = ih(T(z))$.

As a result, instead of the problem $z$ in the decision or inhibitory interpretation we can study the decision table $T(z)$.

2.2 Examples of Problems with Many-Valued Decisions

There are two sources of problems with many-valued decisions and corresponding decision tables: classes of exactly formulated problems and experimental data. We consider examples of exactly formulated problems with many-valued decisions from the following areas: computational geometry, combinatorial optimization, diagnosis of faults in combinatorial circuits.

We also consider an example of the data table with experimental data.

2.2.1 Problem of Three Post-Offices

Let three post-offices $P_1, P_2$ and $P_3$ exist (see Fig. 2.1 from [2]). Let new client appear. Then this client will be served by a nearest post-office.
Let we have two points $B_1$ and $B_2$. We join these points by segment (of a straight line) and draw the perpendicular through the center of this segment (see Fig. 2.2 from [2]).

All points which lie from the left of this perpendicular are nearer to $B_1$, all points that lie on the perpendicular have the same distance to $B_1$ and to $B_2$, and all points which lie from the right of the perpendicular are nearer to the point $B_2$. This reasoning allows us to construct attributes for the problem of three post-offices which will be denoted $z_1$.

We joint all pairs of post-offices $P_1, P_2, P_3$ by segments (these segments are invisible in Fig. 2.1) and draw perpendiculars through centers of these segments. These perpendiculars (lines) correspond to three attributes $f_1, f_2, f_3$. Each such attribute takes value $-1$ from the left of the considered line, takes value 0 on the line, and takes value $+1$ from the right of the considered line (arrow points to the right). These three straight lines divide the plane into 13 regions. We correspond to each region the set of numbers of post-offices which are nearest to points of this region and construct the decision table $T_1$ for the considered problem (see Fig. 2.3).

One can show that the tree depicted in Fig. 2.4 is a decision tree with minimum depth for the table $T_1$.

For the consideration of inhibitory trees, we should remove the central point which is the intersection of lines corresponding to attributes $f_1, f_2, f_3$ (see Fig. 2.1) since the whole set of decisions $\{1, 2, 3\}$ corresponds to this point. We denote the obtained problem $z'_1$. We should also remove from the table $T_1$ the row $(0, 0, 0)$ labeled with
\[ T_1 = \begin{array}{ccc} f_1 & f_2 & f_3 \\ +1 & +1 & +1 & \{3\} \\ 0 & +1 & +1 & \{2, 3\} \\ -1 & +1 & +1 & \{2\} \\ -1 & 0 & +1 & \{2\} \\ -1 & -1 & +1 & \{2\} \\ -1 & -1 & 0 & \{1, 2\} \\ -1 & -1 & -1 & \{1\} \\ 0 & -1 & -1 & \{1\} \\ +1 & -1 & -1 & \{1\} \\ +1 & 0 & -1 & \{1, 3\} \\ +1 & +1 & -1 & \{3\} \\ +1 & +1 & 0 & \{3\} \\ 0 & 0 & 0 & \{1, 2, 3\} \end{array} \]

Figure 2.3: Decision table \( T_1 \) for problem of three post-offices

Figure 2.4: Decision tree with minimum depth for decision table \( T_1 \)

The whole set of decisions \( \{1, 2, 3\} \). We denote the obtained decision table \( T'_1 \).

One can show that the tree depicted in Fig. 2.5 is an inhibitory tree with minimum depth for the table \( T'_1 \).

The considered problem is an example of problems studied in computational geometry.

### 2.2.2 Traveling Salesman Problem with Four Cities

Let we have a complete unordered graph with four nodes in which each edge is labeled with a real number – the length of this edge (see Fig. 2.6 from [2]).

A Hamiltonian circuit is a closed path which passes through each node exactly one
time. We should find a Hamiltonian circuit which has the minimum length. There are three Hamiltonian circuits:

\( H_1 \): 12341 or, which is the same, 14321,

\( H_2 \): 12431 or 13421,

\( H_3 \): 13241 or 14231.

For \( i = 1, 2, 3 \), we denote by \( L_i \) the length of \( H_i \). Then

\[
\begin{align*}
\alpha & \quad \beta \\
L_1 &= x_{12} + x_{23} + x_{34} + x_{14} = (x_{12} + x_{34}) + (x_{23} + x_{14}), \\
\alpha & \quad \gamma \\
L_2 &= x_{12} + x_{24} + x_{34} + x_{13} = (x_{12} + x_{34}) + (x_{24} + x_{13}), \\
\gamma & \quad \beta \\
L_3 &= x_{13} + x_{23} + x_{24} + x_{14} = (x_{24} + x_{13}) + (x_{23} + x_{14}).
\end{align*}
\]

In the capacity of attributes we will use three functions

\[ f_1 = \text{sign}(L_1 - L_2), \]

\[ f_2 = \text{sign}(L_2 - L_3), \]

\[ f_3 = \text{sign}(L_1 - L_3). \]
If $\alpha < \beta < \gamma$ then $L_1 < L_2 < L_3$
If $\alpha = \beta < \gamma$ then $L_1 < L_2 = L_3$
If $\alpha < \beta = \gamma$ then $L_1 = L_2 < L_3$
If $\alpha = \beta = \gamma$ then $L_1 = L_2 = L_3$
If $\alpha < \gamma < \beta$ then $L_2 < L_1 < L_3$
If $\alpha = \gamma < \beta$ then $L_2 < L_1 = L_3$
If $\beta < \alpha < \gamma$ then $L_1 < L_3 < L_2$
If $\beta < \alpha = \gamma$ then $L_1 = L_3 < L_2$
If $\beta < \gamma < \alpha$ then $L_3 < L_1 < L_2$
If $\beta = \gamma < \alpha$ then $L_3 < L_1 = L_2$
If $\gamma < \alpha < \beta$ then $L_2 < L_3 < L_1$
If $\gamma < \alpha = \beta$ then $L_2 = L_3 < L_1$
If $\gamma < \beta < \alpha$ then $L_3 < L_2 < L_1$

<table>
<thead>
<tr>
<th>$f_1$</th>
<th>$f_2$</th>
<th>$f_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>+1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>+1</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
<td>+1</td>
</tr>
<tr>
<td>-1</td>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>-1</td>
<td>+1</td>
<td>+1</td>
</tr>
<tr>
<td>0</td>
<td>+1</td>
<td>+1</td>
</tr>
<tr>
<td>+1</td>
<td>+1</td>
<td>0</td>
</tr>
<tr>
<td>+1</td>
<td>+1</td>
<td>+1</td>
</tr>
</tbody>
</table>

Figure 2.7: Decision table $T_2$ for traveling salesman problem with four cities

\[ f_2 = \text{sign}(L_1 - L_3), \text{ and } f_3 = \text{sign}(L_2 - L_3) \text{ where } \text{sign}(x) = -1 \text{ if } x < 0, \text{ sign}(x) = 0 \text{ if } x = 0, \text{ and } \text{sign}(x) = +1 \text{ if } x > 0. \]

Values $L_1$, $L_2$ and $L_3$ are linearly ordered. It is clear that values of $\alpha$, $\beta$ and $\gamma$ can be chosen independently. Based on this fact, we can show that any order of $L_1$, $L_2$ and $L_3$ is possible, and we can construct the decision table $T_2$ for the considered problem (see Fig. 2.7).

One can show that the tree depicted in Fig. 2.8 is a decision tree with minimum depth for the table $T_2$.

For the consideration of inhibitory trees, we should remove from the table $T_2$ the row $(0,0,0)$ labeled with the whole set of decisions \{1,2,3\}. We denote the obtained
One can show that the tree depicted in Fig. 2.9 is an inhibitory tree with minimum depth for the table $T'_2$.

It was an example of \textit{combinatorial optimization} problem.

2.2.3 Diagnosis of One-gate Circuit

This example was discussed in [2]. We add here only results about inhibitory trees.

Consider the combinatorial circuit $S$ depicted in Fig. 2.10 from [2].

Each input of $S$ can work correctly or can have constant fault 1. Let us assume that at least one such fault exists in $S$. We should find an input with fault. To this end, we can use attributes from the set $\{0, 1\}^3$. We give a tuple from this set of inputs of $S$ and observe the value of the output of $S$ which is the value of the considered attribute. It is clear that the circuit $S$ with at least one fault 1 on an input implements one of the functions from the set \{1, $x, y, z, xy, xz, yz$\} (we write
One can show that the tree depicted in Fig. 2.12 is a decision tree with minimum depth for the table $T_3$.

For the consideration of inhibitory trees, we should remove from the table $T_3$ the first row labeled with the whole set of decisions \{x, y, z\}. We denote the obtained decision table $T_3'$.

One can show that the tree depicted in Fig. 2.13 is an inhibitory tree with minimum depth for the table $T_3'$.

It was an example of fault diagnosis problem.

### 2.2.4 Example of Data Table

The primary source of decision tables with many-valued decisions is data tables with experimental data. It is possible that in such a table there are equal rows with
different decisions. We keep one row from a group of equal rows and attach to this row all the most common decisions among the decisions attached to rows from the group (we use this way in the considered example).

Note that there are other ways to form a set of decisions attached to a row. For example, we can include in this set all decisions attached to equal rows, or the first $k$ most frequent decisions for equal rows, etc.

The considered here example was discussed in [2]. We add only results about inhibitory trees.

Let we have the data table $D$ depicted in Fig. 2.14 from [2].

All variables $x_1$, $x_2$, $x_3$ and $y$ are discrete. We must predict the value of $y$ using variables $x_1$, $x_2$, $x_3$ or, more exactly, values of attributes $f_1 = x_1$, $f_2 = x_2$, and $f_3 = x_3$. Corresponding decision table $T_4$ with many-valued decisions is depicted in Fig. 2.15 from [2].

One can show that the tree depicted in Fig. 2.16 is a decision tree with minimum depth for the table $T_4$.

One can show that the tree depicted in Fig. 2.17 is an inhibitory tree with minimum depth for the table $T_4$. 

Figure 2.13: Inhibitory tree with minimum depth for decision table $T'_3$
\[
D = \begin{array}{cccc}
0 & 0 & 1 & 1 \\
0 & 0 & 1 & 2 \\
0 & 0 & 1 & 2 \\
0 & 0 & 1 & 3 \\
0 & 0 & 1 & 3 \\
1 & 0 & 0 & 1 \\
1 & 0 & 0 & 1 \\
1 & 0 & 0 & 3 \\
1 & 0 & 0 & 3 \\
1 & 0 & 1 & 2 \\
1 & 0 & 1 & 3 \\
0 & 1 & 1 & 1 \\
0 & 1 & 1 & 2 \\
0 & 1 & 1 & 2 \\
1 & 1 & 0 & 1 \\
1 & 1 & 0 & 1 \\
1 & 1 & 0 & 3 \\
\end{array}
\]

Figure 2.14: Data table \(D\)

\[
T_4 = \begin{array}{ccc|c}
\hline
f_1 & f_2 & f_3 & \{2,3\} \\
0 & 0 & 1 & \{2,3\} \\
1 & 0 & 0 & \{1,3\} \\
1 & 0 & 1 & \{2,3\} \\
0 & 1 & 1 & \{2\} \\
1 & 1 & 0 & \{1\} \\
\hline
\end{array}
\]

Figure 2.15: Decision table \(T_4\) corresponding to data table \(D\)

### 2.3 Difference between Decision and Inhibitory Trees

We consider here two examples which show that the use of inhibitory trees can give us additional possibilities in comparison with decisions trees.

#### 2.3.1 Prediction Problem

Let us consider the decision table \(T_5\) with many-valued decisions depicted in Fig. 2.18.

Let us discuss the problem of prediction of decision value for a new object given by values of condition attributes \(f_1 = 0\) and \(f_2 = 0\). We will consider only decision
and inhibitory trees for $T_5$ such that, for each terminal node of the tree, there is a row of $T_5$ for which the tree work finishes in this node.

It is clear that there is no a decision tree for $T_5$ the work of which, for a new object given by values of condition attributes $f_1 = 0$ and $f_2 = 0$, finishes in a terminal node. It means that the decision trees will not give us any information about decisions corresponding to this new object. However, the inhibitory trees $\Gamma_1$ and $\Gamma_2$ for the decision table $T_5$ (see Figs. 2.19 and 2.20) will restrict the set of possible decisions to \{2, 4\}.

### 2.3.2 Knowledge Representation Problem

Let us consider the data table (information system) $I$ depicted in Fig. 2.21.

We will compare now knowledge that can be derived from $I$ by inhibitory and decision association rules which are true for $I$ and realizable for at least one row of $I$. Decision association rules have on the right-hand side expressions of the kind $f_i = a$ where $a \in V_I(f_i)$ and $V_I(f_i)$ is the set of values of the attribute $f_i$ in the information system $I$. Inhibitory association rules have on the right-hand side expressions of the kind $f_i \neq a$ where $a \in V_I(f_i)$.
For the information system $I$, $V_I(f_1) = V_I(f_2) = \{0, 1, 2\}$. We will consider the set

$$V_I(f_1) \times V_I(f_2) = \{(0, 0), (0, 1), (0, 2), (1, 0), (1, 1), (1, 2), (2, 0), (2, 1), (2, 2)\}$$

which contains all 2-tuples from $I$ and some additional 2-tuples, for example, 2-tuple $(0, 0)$.

The question under consideration is about the possibility to remove from the set $V_I(f_1) \times V_I(f_2)$ all 2-tuples that do not belong to $I$ using decision (inhibitory) association rules that are true for $I$ and realizable for at least one 2-tuple (row) from $I$.

One can show that the set of decision association rules that are true for $I$ and realizable for at least one 2-tuple from $I$ is the following:

$$\{(f_1 = 1) \rightarrow (f_2 = 0), (f_1 = 2) \rightarrow (f_2 = 0),$$

$$ (f_2 = 1) \rightarrow (f_1 = 0), (f_2 = 2) \rightarrow (f_1 = 0)\} . \quad (2.1)$$

Based on these rules we can remove all 2-tuples from $V_I(f_1) \times V_I(f_2)$ that do not
belong to \( I \) with the exception of \((0, 0)\). For example, the tuple \((2, 2)\) can be removed by the rule \(f_2 = 2 \rightarrow f_1 = 0\). We cannot remove the 2-tuple \((0, 0)\) since no one rule is realizable for this 2-tuple. So we have that the information derived from \( I \) by decision association rules is in some sense incomplete. The existence of such information systems was discovered in [51, 52].

One can show that the set of inhibitory association rules that are true for \( I \) and realizable for at least one 2-tuple from \( I \) is the following:

\[
\begin{align*}
(f_1 = 0) & \rightarrow (f_2 \neq 0), (f_2 = 1) \rightarrow (f_1 \neq 1), \\
(f_1 = 1) & \rightarrow (f_2 \neq 1), (f_2 = 1) \rightarrow (f_1 \neq 2), \\
(f_1 = 1) & \rightarrow (f_2 \neq 2), (f_2 = 0) \rightarrow (f_1 \neq 0), \\
(f_1 = 2) & \rightarrow (f_2 \neq 1), (f_2 = 2) \rightarrow (f_1 \neq 1), \\
(f_1 = 2) & \rightarrow (f_2 \neq 2), (f_2 = 2) \rightarrow (f_1 \neq 2)
\end{align*}
\]

Based on these rules we can remove all 2-tuples from \( V_I(f_1) \times V_I(f_2) \) that do not belong to \( I \). For example, the 2-tuple \((2, 2)\) can be removed by the rule \(f_1 = 2 \rightarrow f_2 \neq 2\), and the 2-tuple \((0, 0)\) can be removed by the rule \(f_1 = 0 \rightarrow f_2 \neq 0\).
So we have that the information derived from $I$ by inhibitory association rules is in some sense complete. This fact was proven for the arbitrary information system in [53].

We can fix sequentially each attribute of the information system as the decision attribute and consider the decision and inhibitory trees for the obtained decision tables as decision and inhibitory association trees for the initial information system. In some cases, these trees can have empty terminal nodes.

For the considered information system $I$, there are two decision association trees $G_1$ and $G_2$ depicted in Figs. 2.22 and 2.23 respectively. The tree $G_1$ contains an empty terminal node corresponding to the case when $f_1 = 0$. In this case, the attribute $f_2$ can have two values 1 and 2, so there is no definite value of $f_2$. The same situation is with the tree $G_2$. Note that the set of decision association rules corresponding to paths in $G_1$ and $G_2$ from the roots to nonempty terminal nodes is equal to the set (2.1).

For the considered information system $I$, all eight inhibitory association trees can be represented as trees $\Gamma_1(a,b)$ and $\Gamma_2(a,b)$ depicted in Figs. 2.24 and 2.25 respectively, where $a,b \in \{1,2\}$. Note that the set of inhibitory association rules cor-
responding to paths in $\Gamma_1(a, b)$ and $\Gamma_2(a, b)$, $a, b \in \{1, 2\}$, from the roots to nonempty terminal nodes is equal to the set $\{2, 2\}$.

So we have that the information derived from $I$ by decision association trees is incomplete, but the information derived from $I$ by inhibitory association trees is complete.
Chapter 3

Preliminary Results for Decision and Inhibitory Trees

In the book [2], some relatively simple results were considered for binary decision tables with many-valued decisions: relationships among decision trees, rules, and tests, bounds on their complexity, greedy algorithms for the construction of decision trees, rules and tests, and dynamic programming algorithms for minimization of tree depth and rule length.

In this chapter, we mention results from [2] related mainly to decision trees without proofs and extend them to inhibitory trees over binary decision tables with many-valued decisions.

Note that some part of this chapter related to relationships among inhibitory trees, rules, and tests, and bounds on their complexity was developed jointly with Fawaz Alsolami.

3.1 Main Notions

Now we consider formal definitions of notions corresponding to binary decision tables with many-valued decisions including decision and inhibitory trees, tests, rules, and rule systems, and complementary decision tables.

3.1.1 Binary Decision Tables with Many-Valued Decisions

A binary decision table with many-valued decisions is a rectangular table $T$ filled by numbers from the set $\{0, 1\}$. Columns of this table are labeled with attributes
$f_1, \ldots, f_n$. Rows of the table are pairwise different, and each row $r$ is labeled with a nonempty finite set $D(r) = D_T(r)$ of numbers from $\omega = \{0, 1, 2, \ldots\}$ (set of decisions). Let $\text{Row}(T)$ be the set of rows of $T$ and $D(T) = \bigcup_{r \in \text{Row}(T)} D(r)$. We denote by $N(T)$ the number of rows in the table $T$. When we consider inhibitory trees, rules and tests, we assume that $D(r) \neq D(T)$ for any $r \in \text{Row}(T)$.

A table obtained by removal of some rows of $T$ is called a subtable of $T$. Let $f_{i_1}, \ldots, f_{i_m} \in \{f_1, \ldots, f_n\}$ and $\delta_1, \ldots, \delta_m \in \{0, 1\}$. We denote by

$$T(f_{i_1}, \delta_1) \ldots (f_{i_m}, \delta_m)$$

a subtable of the table $T$ which contains only rows that at the intersection with columns $f_{i_1}, \ldots, f_{i_m}$ have numbers $\delta_1, \ldots, \delta_m$, respectively. We denote by $\text{SEP}(T)$ the set of all nonempty subtables of the table $T$ including the table $T$ which can be represented in the form $T(f_{i_1}, \delta_1) \ldots (f_{i_m}, \delta_m)$. We will call such tables separable subtables of the table $T$.

We will consider two interpretations of the decision table $T$: decision and inhibitory. In the case of decision interpretation, for a given row $r$ of $T$, we should find a decision from the set $D(r)$. In the case of inhibitory interpretation, for a given row $r$ of $T$, we should find a decision from the set $D(T) \setminus D(r)$.

### 3.1.2 Decision Trees, Rule Systems and Tests

We begin from the consideration of decision interpretation.

A decision tree over $T$ is a finite tree with root in which each terminal node is labeled with a decision (a number from $\omega$), each nonterminal node (such nodes will be called working) is labeled with an attribute from the set $\{f_1, \ldots, f_n\}$. Two edges start in each working node. These edges are labeled with 0 and 1, respectively.

Let $\Gamma$ be a decision tree over $T$. For a given row $r$ of $T$, this tree works in the
following way. We begin the work at the root of \( \Gamma \). If the considered node is terminal then the result of \( \Gamma \) work is the number attached to this node. Let the considered node be a working node which is labeled with an attribute \( f_i \). If the value of \( f_i \) in the considered row is 0, then we pass along the edge which is labeled with 0. Otherwise, we pass along the edge which is labeled with 1, etc.

We will say that a decision tree \( \Gamma \) over the decision table \( T \) is a decision tree for \( T \) if, for any row \( r \) of \( T \), the work of \( \Gamma \) finishes in a terminal node which is labeled with a number from the set \( D(r) \) attached to the row \( r \).

We denote by \( h(\Gamma) \) the depth of \( \Gamma \) which is the maximum length of a path from the root to a terminal node. We denote by \( h(T) \) the minimum depth of a decision tree for the table \( T \).

A decision rule over \( T \) is an expression of the kind

\[
(f_{i_1} = b_1) \land \ldots \land (f_{i_m} = b_m) \rightarrow t
\]

where \( f_{i_1}, \ldots, f_{i_m} \in \{f_1, \ldots, f_n\}, b_1, \ldots, b_m \in \{0, 1\}, \) and \( t \in \omega \). We denote this rule \( \rho \). The number \( m \) is called the length of the rule \( \rho \) and is denoted \( l(\rho) \). The decision \( t \) is called the right-hand side of the rule \( \rho \). This rule is called realizable for a row \( r = (\delta_1, \ldots, \delta_n) \) if

\[
\delta_{i_1} = b_1, \ldots, \delta_{i_m} = b_m.
\]

The rule \( \rho \) is called true for \( T \) if, for any row \( r \) of \( T \) such that the rule \( \rho \) is realizable for row \( r \), \( t \in D(r) \). We denote by \( l(T, r) \) the minimum length of a decision rule over \( T \) which is true for \( T \) and realizable for \( r \). We will say that the considered rule is a rule for \( T \) and \( r \) if this rule is true for \( T \) and realizable for \( r \).

A nonempty finite set \( S \) of decision rules over \( T \) is called a complete decision rule system for \( T \) if each rule from \( S \) is true for \( T \) and, for every row of \( T \), there exists a rule from \( S \) which is realizable for this row. We denote by \( l(S) \) the maximum length of
a rule from $S$ (we will call $l(S)$ the length of $S$), and by $l(T)$ we denote the minimum value of $l(S)$ among all complete decision rule systems $S$ for $T$.

We will say that $T$ is a degenerate table if either $T$ has no rows, or the intersection of sets of decisions attached to rows of $T$ is nonempty (in this case, we will say that $T$ has a common decision).

A decision test for the table $T$ is a subset of columns $\{f_{i_1}, \ldots, f_{i_m}\}$ such that, for any numbers $\delta_1, \ldots, \delta_m \in \{0, 1\}$, the subtable $T(f_{i_1}, \delta_1) \ldots T(f_{i_m}, \delta_m)$ is a degenerate table. Empty set is a decision test for $T$ if and only if $T$ is a degenerate table.

A decision reduct for the table $T$ is a decision test for $T$ for which each proper subset is not a test. It is clear that each decision test has a decision reduct as a subset. We denote by $R(T)$ the minimum cardinality of a decision reduct for $T$.

### 3.1.3 Inhibitory Trees, Rule Systems, and Tests

We consider now the inhibitory interpretation of the decision table $T$. We will assume that $D_T(r) \neq D(T)$ for any $r \in \text{Row}(T)$.

An inhibitory tree over $T$ is a finite tree with root in which each terminal node is labeled with an expression $\not= t$ where $t$ is a decision (a number from $\omega$), each nonterminal node (such nodes will be called working) is labeled with an attribute from the set $\{f_1, \ldots, f_n\}$. Two edges start in each working node. These edges are labeled with 0 and 1, respectively.

Let $\Gamma$ be an inhibitory tree over $T$. For a given row $r$ of $T$, this tree works in the following way. We begin the work at the root of $\Gamma$. If the considered node is terminal then the result of $\Gamma$ work is the expression attached to this node. Let the considered node be a working node which is labeled with an attribute $f_i$. If the value of $f_i$ in the considered row is 0 then we pass along the edge which is labeled with 0. Otherwise, we pass along the edge which is labeled with 1, etc.

We will say that an inhibitory tree $\Gamma$ over the decision table $T$ is an inhibitory
tree for $T$ if, for any row $r$ of $T$, the work of $\Gamma$ finishes in a terminal node which is labeled with an expression $\neq t$ where $t \in D(T) \setminus D(r)$.

We denote by $h(\Gamma)$ the depth of $\Gamma$ which is the maximum length of a path from the root to a terminal node. We denote by $ih(T)$ the minimum depth of an inhibitory tree for the table $T$.

An inhibitory rule over $T$ is an expression of the kind

$$(f_{i_1} = b_1) \land \ldots \land (f_{i_m} = b_m) \rightarrow \neq t$$

where $f_{i_1}, \ldots, f_{i_m} \in \{f_1, \ldots, f_n\}$, $b_1, \ldots, b_m \in \{0, 1\}$, and $t \in \omega$. We denote this rule $\rho$. The number $m$ is called the length of the rule $\rho$ and is denoted by $l(\rho)$. The expression $\neq t$ is called the right-hand side of the rule $\rho$. This rule is called realizable for a row $r = (\delta_1, \ldots, \delta_n)$ if

$$\delta_{i_1} = b_1, \ldots, \delta_{i_m} = b_m.$$

The rule $\rho$ is called true for $T$ if, for any row $r$ of $T$ such that the rule $\rho$ is realizable for row $r$, $t \in D(T) \setminus D(r)$. We denote by $il(T, r)$ the minimum length of an inhibitory rule over $T$ which is true for $T$ and realizable for $r$. We will say that the considered rule is a rule for $T$ and $r$ if this rule is true for $T$ and realizable for $r$.

A nonempty finite set $S$ of inhibitory rules over $T$ is called a complete inhibitory rule system for $T$ if each rule from $S$ is true for $T$ and, for every row of $T$, there exists a rule from $S$ which is realizable for this row. We denote by $l(S)$ the maximum length of a rule from $S$ (we will call $l(S)$ the length of $S$), and by $il(T)$ we denote the minimum value of $l(S)$ among all complete inhibitory rule systems $S$ for $T$.

Let $\Theta$ be a subtable of $T$. We will say that $\Theta$ is incomplete relative to $T$ if $D(\Theta) \subset D(T)$.

An inhibitory test for the table $T$ is a subset of columns $\{f_{i_1}, \ldots, f_{i_m}\}$ such that, for
any numbers $\delta_1, \ldots, \delta_m \in \{0, 1\}$, the subtable $T(f_{i_1}, \delta_1) \ldots T(f_{i_m}, \delta_m)$ is incomplete relative to $T$.

An inhibitory reduct for the table $T$ is an inhibitory test for $T$ for which each proper subset is not an inhibitory test. It is clear that each inhibitory test has an inhibitory reduct as a subset. We denote by $iR(T)$ the minimum cardinality of an inhibitory reduct for $T$.

### 3.1.4 Complementary Decision Table

Let $T$ be a nondegenerate binary decision table with many-valued decisions such that $D_T(r) \neq D(T)$ for any $r \in Row(T)$. We denote by $T^C$ complementary to $T$ decision table obtained from the table $T$ by changing, for each row $r \in Row(T)$, the set $D(r)$ with the set $D(T) \setminus D(r)$, i.e., $D_{T^C}(r) = D(T) \setminus D_T(r)$ for any $r \in Row(T)$. It is clear that $Row(T) = Row(T^C)$, $D(T) = D(T^C)$, and $D_T(r) = D(T) \setminus D_{T^C}(r)$ for any $r \in Row(T)$. In particular, we have $T^{CC} = T$.

Let $\Gamma_1$ be a decision tree over $T^C$ and $\Gamma_2$ be an inhibitory tree over $T$. We denote by $\Gamma_1^-$ an inhibitory tree over $T$ obtained from $\Gamma_1$ by changing expressions attached to terminal nodes: if a terminal node in $\Gamma_1$ is labeled with $t$ then the corresponding node in $\Gamma_1^-$ is labeled with $\neq t$. We denote by $\Gamma_2^+$ a decision tree over $T^C$ obtained from $\Gamma_2$ by changing expressions attached to terminal nodes: if a terminal node in $\Gamma_2$ is labeled with $\neq t$ then the corresponding node in $\Gamma_2^+$ is labeled with $t$.

Let $\rho_1$ be a decision rule over $T^C$ and $\rho_2$ be an inhibitory rule over $T$. We denote by $\rho_1^-$ an inhibitory rule over $T$ obtained from $\rho_1$ by changing the right-hand side of $\rho_1$: if the right-hand side of $\rho_1$ is $t$ then the right-hand side of $\rho_1^-$ is $\neq t$. We denote by $\rho_2^+$ a decision rule over $T^C$ obtained from $\rho_2$ by changing the right-hand side of $\rho_2$: if the right-hand side of $\rho_2$ is $\neq t$ then the right-hand side of $\rho_2^+$ is $t$.

Let $S_1$ be a nonempty finite set of decision rules over $T^C$ and $S_2$ be a nonempty finite system of inhibitory rules over $T$. We denote $S_1^- = \{\rho^- : \rho \in S_1\}$ and $S_2^+ = \{\rho^+ : \rho \in S_2\}$.
\[ \{ \rho^+ : \rho \in S_2 \} \].

**Proposition 1.** Let \( T \) be a nondegenerate binary decision table with many-valued decisions containing \( n \) columns labeled with attributes \( f_1, \ldots, f_n \), and \( T^C \) be complementary to \( T \) decision table. Then

1. A decision tree \( \Gamma \) over \( T^C \) is a decision tree for \( T^C \) if and only if \( \Gamma^- \) is an inhibitory tree for \( T \);

2. A decision rule \( \rho \) over \( T^C \) is true for \( T^C \) if and only if the inhibitory rule \( \rho^- \) is true for \( T \);

3. A nonempty finite set \( S \) of decision rules over \( T^C \) is a complete system of decision rules for \( T^C \) if and only if \( S^- \) is a complete system of inhibitory rules for \( T \);

4. A subset \( \{ f_{i_1}, \ldots, f_{i_m} \} \) of the set \( \{ f_1, \ldots, f_n \} \) is a decision test for \( T^C \) if and only if the subset \( \{ f_{i_1}, \ldots, f_{i_m} \} \) is an inhibitory test for \( T \).

**Proof.** 1. Let \( \Gamma \) be a decision tree over \( T^C \). Let us assume that \( \Gamma \) is a decision tree for \( T^C \). Then, for any row \( r \) of \( T^C \), the work of \( \Gamma \) finishes in a terminal node which is labeled with a number \( t \in D_{T^C}(r) \). From here it follows that, for any row \( r \) of \( T \), the work of \( \Gamma^- \) finishes in a terminal node which is labeled with an expression \( \neq t \) where \( t \in D_{T^C}(r) = D(T) \setminus D_T(r) \). Therefore \( \Gamma^- \) is an inhibitory tree for \( T \).

Let us assume now that \( \Gamma^- \) is an inhibitory tree for \( T \). Then, for any row \( r \) of \( T \), the work of \( \Gamma^- \) finishes in a terminal node which is labeled with an expression \( \neq t \) where \( t \in D(T) \setminus D_T(r) \). From here it follows that, for any row \( r \) of \( T^C \), the work of \( \Gamma \) finishes in a terminal node which is labeled with a number \( t \) where \( t \in D(T) \setminus D_T(r) = D_{T^C}(r) \). Therefore \( \Gamma \) is a decision tree for \( T^C \).
2. Let \( \rho \) be a decision rule over \( T^C \) and \( \rho \) be equal to \( f_{i_1} = b_1 \land \ldots \land f_{i_m} = b_m \rightarrow t \). Then \( \rho^- \) is equal to \( f_{i_1} = b_1 \land \ldots \land f_{i_m} = b_m \rightarrow \neq t \). Denote

\[
T^C_* = T^C(f_{i_1}, b_1) \ldots (f_{i_m}, b_m)
\]

and \( T_* = T(f_{i_1}, b_1) \ldots (f_{i_m}, b_m) \). One can show that \( \rho \) is true for \( T^C \) if and only if \( t \in D_{T^C}(r) \) for any \( r \in Row(T^C) \), and \( \rho^- \) is true for \( T \) if and only if \( t \in D(T) \setminus D_{T^C}(r) \) for any \( r \in Row(T) \). We know that Row\( (T^C_*) = Row(T_*) \) and \( D_{T^C}(r) = D(T) \setminus D_{T}(r) \) for any \( r \in Row(T) \). From here it follows that \( \rho \) is true for \( T^C \) if and only if \( \rho^- \) is true for \( T \).

3. Let \( S \) be a nonempty finite set of decision rules over \( T^C \). From statement 2 it follows that each rule \( \rho \) from \( S \) is true for \( T^C \) if and only if each rule \( \rho^- \) from \( S^- \) is true for \( T \). It is clear that a rule \( \rho \) from \( S \) is realizable for a row \( r \) from Row\( (T^C) = Row(T) \) if and only if \( \rho^- \) is realizable for \( r \). From here it follows that \( S \) is a complete system of decision rules for \( T^C \) if and only if \( S^- \) is a complete system of inhibitory rules for \( T \).

4. Let \( \{f_{i_1}, \ldots, f_{i_m}\} \) be a subset of the set \( \{f_1, \ldots, f_n\} \) and \( \delta_1, \ldots, \delta_m \in \{0, 1\} \). One can show that \( T^C(f_{i_1}, \delta_1) \ldots (f_{i_m}, \delta_m) \) is degenerate if and only if

\[
T(f_{i_1}, \delta_1) \ldots (f_{i_m}, \delta_m)
\]

is incomplete relative to \( T \). Therefore the subset \( \{f_{i_1}, \ldots, f_{i_m}\} \) is a decision test for \( T^C \) if and only the subset \( \{f_{i_1}, \ldots, f_{i_m}\} \) is an inhibitory test for \( T \).

\[\blacksquare\]

**Corollary 1.** Let \( T \) be a nondegenerate binary decision table with many-valued decisions, \( T^C \) be complementary to \( T \) decision table, and \( r \) be a row of \( T \). Then

\[ih(T) = h(T^C), \quad il(T, r) = l(T^C, r), \quad il(T) = l(T^C), \quad \text{and} \quad iR(T) = R(T^C).\]

**Proof.** 1. Let \( \Gamma_1 \) be a decision tree for \( T^C \) such that \( h(\Gamma_1) = h(T^C) \). Then, by
Proposition 1. \( \Gamma_1^- \) is an inhibitory tree for \( T \). It is clear that \( h(\Gamma_1) = h(\Gamma_1^-) \). Therefore \( ih(T) \leq h(T^C) \). Let \( \Gamma_2 \) be an inhibitory tree for \( T \) such that \( h(\Gamma_2) = ih(T) \). We now consider a decision tree \( \Gamma_2^+ \) over \( T^C \). It is clear that \( (\Gamma_2^+)^- = \Gamma_2 \). By Proposition 1, \( \Gamma_2^+ \) is a decision tree for \( T^C \). It is clear that \( h(\Gamma_2) = h(\Gamma_2^+) \). Therefore \( h(T^C) \leq ih(T) \) and \( h(T^C) = ih(T) \).

2. Let \( \rho_1 \) be a decision rule for \( T^C \) and \( r \) such that \( l(\rho_1) = l(T^C, r) \). It is clear that \( \rho_1 \) is realizable for \( r \) if and only if the rule \( \rho_1^- \) is realizable for \( r \). Then, by Proposition 1, \( \rho_1^- \) is an inhibitory rule for \( T \) and \( r \). It is clear that \( l(\rho_1) = l(\rho_1^-) \). Therefore \( il(T, r) \leq l(T^C, r) \). Let \( \rho_2 \) be an inhibitory rule for \( T \) and \( r \) such that \( l(\rho_2) = il(T, r) \). We now consider a decision rule \( \rho_2^+ \) over \( T^C \). It is clear that \( (\rho_2^+)^- = \rho_2 \), and \( \rho_2 \) is realizable for \( r \) if and only if the rule \( \rho_2^+ \) is realizable for \( r \). By Proposition 1, \( \rho_2^+ \) is a decision rule for \( T^C \). It is clear that \( l(\rho_2) = l(\rho_2^+) \). Therefore \( l(T^C, r) \leq il(T, r) \) and \( l(T^C, r) = il(T, r) \).

3. Let \( S_1 \) be a complete system of decision rules for \( T^C \) such that \( l(S_1) = l(T^C) \). Then, by Proposition 1, \( S_1^- \) is a complete system of inhibitory rules for \( T \). It is clear that \( l(S_1) = l(S_1^-) \). Therefore \( il(T) \leq l(T^C) \). Let \( S_2 \) be a complete system of inhibitory rules for \( T \) such that \( l(S_2) = il(T) \). We now consider a set of decision rules \( S_2^+ \) over \( T^C \). It is clear that \( (S_2^+)^- = S_2 \). By Proposition 1, \( S_2^+ \) is a complete system of decision rules for \( T^C \). It is clear that \( l(S_2) = l(S_2^+) \). Therefore \( l(T^C) \leq il(T) \) and \( l(T^C) = il(T) \).

4. The equality \( iR(T) = R(T^C) \) follows directly from Proposition 1.

3.2 Relationships among Trees, Rule Systems, and Tests

In this section, we consider relationships among decision (inhibitory) trees, rule systems, and tests.
3.2.1 Decision Interpretation

We consider now relationships among decision trees, rule systems, and tests.

**Theorem 1.** Let $T$ be a binary decision table with many-valued decisions.

1. If $\Gamma$ is a decision tree for $T$ then the set of attributes attached to working nodes of $\Gamma$ is a decision test for the table $T$.

2. Let $\{f_{i_1}, \ldots, f_{i_m}\}$ be a decision test for $T$. Then there exists a decision tree $\Gamma$ for $T$ which uses only attributes from $\{f_{i_1}, \ldots, f_{i_m}\}$ and for which $h(\Gamma) = m$.

**Corollary 2.** Let $T$ be a binary decision table with many-valued decisions. Then

$$h(T) \leq R(T).$$

Let $\Gamma$ be a decision tree for $T$ and $\tau$ be a path in $\Gamma$ from the root to a terminal node in which working nodes are labeled with attributes $f_{i_1}, \ldots, f_{i_m}$, edges are labeled with numbers $b_1, \ldots, b_m$, and the terminal node of $\tau$ is labeled with the decision $t$. We correspond to $\tau$ the decision rule

$$(f_{i_1} = b_1) \land \ldots \land (f_{i_m} = b_m) \rightarrow t.$$

**Theorem 2.** Let $T$ be a binary decision table with many-valued decisions, $\Gamma$ be a decision tree for $T$, and $S$ be the set of decision rules corresponding to paths in $\Gamma$ from the root to terminal nodes. Then $S$ is a complete system of decision rules for $T$, and $l(S) = h(\Gamma)$.

**Corollary 3.** Let $T$ be a binary decision table with many-valued decisions. Then

$$l(T) \leq h(T).$$
3.2.2 Inhibitory Interpretation

We consider now relationships among inhibitory trees, rule systems, and tests. Here and later, under the consideration of inhibitory trees, rules or tests for a decision table $T$, we will assume that $D_T(r) \neq D(T)$ for any row $r$ of $T$.

**Theorem 3.** Let $T$ be a nondegenerate binary decision table with many-valued decisions.

1. If $\Gamma$ is an inhibitory tree for $T$ then the set of attributes attached to working nodes of $\Gamma$ is an inhibitory test for the table $T$.

2. Let $\{f_{i_1}, \ldots, f_{i_m}\}$ be an inhibitory test for $T$. Then there exists an inhibitory tree for $T$ which uses only attributes from $\{f_{i_1}, \ldots, f_{i_m}\}$ and which depth is equal to $m$.

**Proof.** 1. Let $\Gamma$ be an inhibitory tree for $T$ and $F(\Gamma)$ be the set of attributes attached to working nodes of $\Gamma$. We now consider a decision tree $\Gamma^+$ over $T^C$. It is clear that $F(\Gamma)$ is the set of attributes attached to working nodes of $\Gamma^+$. By Proposition 1, $\Gamma^+$ is a decision tree for $T^C$. Using Theorem 1, we obtain that $F(\Gamma)$ is a decision test for $T^C$. By Proposition 1, $F(\Gamma)$ is an inhibitory test for $T$.

2. Let $\{f_{i_1}, \ldots, f_{i_m}\}$ be an inhibitory test for $T$. By Proposition 1, $\{f_{i_1}, \ldots, f_{i_m}\}$ is a decision test for $T^C$. Then, by Theorem 1, there exists a decision tree $\Gamma$ for $T^C$ which uses only attributes from $\{f_{i_1}, \ldots, f_{i_m}\}$ and for which $h(\Gamma) = m$. By Proposition 1, $\Gamma^-$ is an inhibitory tree for $T$. It is clear that $\Gamma^-$ uses only attributes from $\{f_{i_1}, \ldots, f_{i_m}\}$, and $h(\Gamma^-) = m$.

**Corollary 4.** Let $T$ be a nondegenerate binary decision table with many-valued decisions. Then

\[ \text{ih}(T) \leq \text{iR}(T). \]
Let $\Gamma$ be an inhibitory tree for $T$ and $\tau$ be a path in $\Gamma$ from the root to a terminal node in which working nodes are labeled with attributes $f_{i_1}, \ldots, f_{i_m}$, edges are labeled with numbers $b_1, \ldots, b_m$, and the terminal node of $\tau$ is labeled with the expression $\neq t$. We correspond to $\tau$ the inhibitory rule

$$(f_{i_1} = b_1) \land \ldots \land (f_{i_m} = b_m) \rightarrow \neq t. $$

**Theorem 4.** Let $T$ be a nondegenerate binary decision table with many-valued decisions, $\Gamma$ be an inhibitory tree for $T$, and $S$ be the set of inhibitory rules corresponding to paths in $\Gamma$ from the root to terminal nodes. Then $S$ is a complete system of inhibitory rules for $T$, and $l(S) = h(\Gamma)$.

**Proof.** Let $\Gamma$ be an inhibitory tree for $T$, and $S$ be the set of inhibitory rules corresponding to paths in $\Gamma$ from the root to terminal nodes. Let us consider a decision tree $\Gamma^+$ over $T^C$. By Proposition 1, $\Gamma^+$ is a decision tree for $T^C$. It is clear that $S^+$ is the set of decision rules corresponding to paths in $\Gamma^+$ from the root to terminal nodes. Using Theorem 2 we obtain that $S^+$ is a complete system of decision rules for $T^C$. By Proposition 1, $(S^+)^- = S$ is a complete system of inhibitory rules for $T$. $\square$

**Corollary 5.** Let $T$ be a binary decision table with many-valued decisions. Then

$$il(T) \leq ih(T).$$

### 3.3 Lower Bounds on Complexity of Trees

In this section, we consider lower bounds on the complexity of decision and inhibitory trees.
3.3.1 Decision Interpretation

Let $T$ be a nonempty binary decision table with many-valued decisions. A nonempty subset $B$ of the set $D(T)$ is called a system of representatives for the table $T$ if, for each row $r$ of $T$, $B \cap D_T(r) \neq \emptyset$. We denote by $S(T)$ the minimum cardinality of a system of representatives for the table $T$.

**Theorem 5.** Let $T$ be a nonempty binary decision table with many-valued decisions. Then

$$h(T) \geq \log_2 S(T).$$

**Theorem 6.** Let $T$ be a binary decision table with many-valued decisions. Then

$$h(T) \geq \log_2 (R(T) + 1).$$

Let $T$ be a binary decision table with many-valued decisions which has $n$ columns labeled with attributes $f_1, \ldots, f_n$, and $\bar{\delta} = (\delta_1, \ldots, \delta_n) \in \{0, 1\}^n$. We define the parameters $M(T, \bar{\delta})$ and $M(T)$. If $T$ is a degenerate table then $M(T, \bar{\delta}) = 0$ and $M(T) = 0$. Let now $T$ be a nondegenerate table. Then $M(T, \bar{\delta})$ is the minimum natural $m$ such that there exist attributes $f_{i_1}, \ldots, f_{i_m} \in \{f_1, \ldots, f_n\}$ for which $T(f_{i_1}, \delta_{i_1}) \ldots (f_{i_m}, \delta_{i_m})$ is a degenerate table. We denote $M(T) = \max\{M(T, \bar{\delta}) : \bar{\delta} \in \{0, 1\}^n\}$.

**Theorem 7.** Let $T$ be a binary decision table with many-valued decisions. Then

$$h(T) \geq M(T).$$

Let $T$ be a binary decision table with many-valued decisions which has $n$ columns labeled with attributes $\{f_1, \ldots, f_n\}$, $m$ be a natural number, and $m \leq n$.

A $(T, m)$-proof-tree is a finite directed tree $G$ with the root in which the length of each path from the root to a terminal node is equal to $m - 1$. Nodes of this tree are
not labeled. In each nonterminal node, exactly \( n \) edges start. These edges are labeled with pairs of the kind \((f_1, \delta_1), \ldots, (f_n, \delta_n)\), respectively, where \( \delta_1, \ldots, \delta_n \in \{0, 1\} \).

Let \( v \) be an arbitrary terminal node of \( G \) and \((f_{i_1}, \delta_1), \ldots, (f_{i_m-1}, \delta_{m-1})\) be pairs attached to edges in the path from the root of \( G \) to the terminal node \( v \). Denote \( T(v) = T(f_{i_1}, \delta_1) \ldots (f_{i_{m-1}}, \delta_{m-1}) \).

We will say that \( G \) is a proof-tree for the bound \( h(T) \geq m \) if, for any terminal node \( v \), the subtable \( T(v) \) is not degenerate. The following statement allows us to use proof-trees to obtain lower bounds on \( h(T) \).

**Theorem 8.** [2] Let \( T \) be a nondegenerate binary decision table with many-valued decisions having \( n \) columns, and \( m \) be a natural number such that \( m \leq n \). Then a proof-tree for the bound \( h(T) \geq m \) exists if and only if the inequality \( h(T) \geq m \) holds.

From Corollary 3 it follows that \( l(T) \leq h(T) \). So each lower bound on \( l(T) \) is also a lower bound on \( h(T) \).

**Theorem 9.** [2] Let \( T \) be a binary decision table with many-valued decisions and \( \text{Row}(T) \) be the set of rows of \( T \). Then \( l(T, \bar{\delta}) = M(T, \bar{\delta}) \) for any row \( \bar{\delta} \in \text{Row}(T) \) and \( l(T) = \max\{M(T, \bar{\delta}) : \bar{\delta} \in \text{Row}(T)\} \).

### 3.3.2 Inhibitory Interpretation

Let \( T \) be a nondegenerate binary decision table with many-valued decisions. A nonempty subset \( B \) of the set \( D(T) \) is called a system of non-representatives for the table \( T \) if, for each row \( r \) of \( T \), \( B \cap (D(T) \setminus D_T(r)) \neq \emptyset \). We denote by \( iS(T) \) the minimum cardinality of a system of non-representatives for the table \( T \).

**Theorem 10.** Let \( T \) be a nondegenerate binary decision table with many-valued decisions. Then

\[
i_h(T) \geq \log_2(iS(T)) .
\]
Proof. Let $T^C$ be complementary to $T$ decision table. Since $T$ is a nondegenerate table, $D(T) = D(T^C)$. One can show that a nonempty subset $B$ of the set $D(T)$ is a system of representatives for the table $T^C$ if and only if $B$ is a system of non-representatives for the table $T$. Therefore $iS(T) = S(T^C)$. From Corollary \ref{corollary1} and from Theorem \ref{theorem5} it follows that $ih(T) = h(T^C)$ and $h(T^C) \geq \log_2 S(T^C)$. Therefore $ih(T) \geq \log_2(iS(T))$. \hfill \qed

**Theorem 11.** Let $T$ be a nondegenerate binary decision table with many-valued decisions. Then

$$ih(T) \geq \log_2(iR(T) + 1).$$

*Proof. Let $T^C$ be complementary to $T$ decision table. From Corollary \ref{corollary1} it follows that $ih(T) = h(T^C)$ and $iR(T) = R(T^C)$. By Theorem \ref{theorem6} $h(T^C) \geq \log_2(R(T^C) + 1)$. Therefore $ih(T) \geq \log_2(iR(T) + 1)$. \hfill \qed*

Let $T$ be a nondegenerate binary decision table with many-valued decisions which has $n$ columns labeled with attributes $f_1, \ldots, f_n$ and $\bar{\delta} = (\delta_1, \ldots, \delta_n) \in \{0, 1\}^n$. We define now the parameters $iM(T, \bar{\delta})$ and $iM(T)$. Denote $iM(T, \bar{\delta})$ the minimum natural $m$ such that there exist attributes $f_{i_1}, \ldots, f_{i_m} \in \{f_1, \ldots, f_n\}$ for which subtable $T(f_{i_1}, \delta_{i_1}) \cdots (f_{i_m}, \delta_{i_m})$ is incomplete relative to $T$. We denote $iM(T) = \max\{iM(T, \bar{\delta}) : \bar{\delta} \in \{0, 1\}^n\}$.

**Lemma 1.** Let $T$ be a nondegenerate binary decision table with many-valued decisions and $T^C$ be complementary to $T$ decision table. Then

$$iM(T) = M(T^C).$$

*Proof. Let $T$ have $n$ columns labeled with attributes $f_1, \ldots, f_n$. One can show that, for any $f_{i_1}, \ldots, f_{i_m} \in \{f_1, \ldots, f_n\}$ and any $\delta_1, \ldots, \delta_m \in \{0, 1\}$, the subtable
$T(f_{i_1}, \delta_1) \ldots (f_{i_m}, \delta_m)$ is incomplete relative to $T$ if and only if the subtable

$$T^C(f_{i_1}, \delta_1) \ldots (f_{i_m}, \delta_m)$$

is degenerate. From here it follows that $iM(T) = M(T^C)$. \qed

**Theorem 12.** Let $T$ be a nondegenerate binary decision table with many-valued decisions. Then

$$ih(T) \geq iM(T).$$

**Proof.** By Corollary 1, $ih(T) = h(T^C)$. From Lemma 1 it follows that $iM(T) = M(T^C)$. By Theorem 7, $h(T^C) \geq M(T^C)$. Therefore $ih(T) \geq iM(T)$. \qed

Let $T$ be a binary decision table with many-valued decisions which has $n$ columns labeled with attributes $\{f_1, \ldots, f_n\}$, $m$ be a natural number, and $m \leq n$.

A $(T, m)$-proof-tree is a finite directed tree $G$ with the root in which the length of each path from the root to a terminal node is equal to $m - 1$. Nodes of this tree are not labeled. In each nonterminal node, exactly $n$ edges start. These edges are labeled with pairs of the kind $(f_1, \delta_1), \ldots, (f_n, \delta_n)$, respectively, where $\delta_1, \ldots, \delta_n \in \{0, 1\}$.

Let $v$ be an arbitrary terminal node of $G$ and $(f_{i_1}, \delta_1), \ldots, (f_{i_{m-1}}, \delta_{m-1})$ be pairs attached to edges in the path from the root of $G$ to the terminal node $v$. Denote $T(v) = T(f_{i_1}, \delta_1) \ldots (f_{i_{m-1}}, \delta_{m-1})$.

We will say that $G$ is a proof-tree for the bound $ih(T) \geq m$ if, for any terminal node $v$, the subtable $T(v)$ is not incomplete relative to $T$. The following statement allows us to use proof-trees to obtain lower bounds on $ih(T)$.

**Theorem 13.** Let $T$ be a nondegenerate binary decision table with many-valued decisions having $n$ columns, and $m$ be a natural number such that $m \leq n$. Then a proof-tree for the bound $ih(T) \geq m$ exists if and only if the inequality $ih(T) \geq m$ holds.
Proof. Let $T^C$ be complementary to $T$ decision table, $G$ be a $(T, m)$-proof-tree, and $v$ be a terminal node of $G$. One can show that the subtable $T(v)$ is not incomplete relative to $T$ if and only if the subtable $T^C(v)$ is not degenerate. From here it follows that $G$ is a proof-tree for the bound $ih(T) \geq m$ if and only if $G$ is a proof-tree for the bound $h(T^C) \geq m$. Using Theorem 8 we obtain that a proof-tree for the bound $ih(T) \geq m$ exists if and only if the inequality $h(T^C) \geq m$ holds. By Corollary 1, $h(T^C) \geq m$ if and only if $ih(T) \geq m$.

From Corollary 5 it follows that $il(T) \leq ih(T)$. So each lower bound on $il(T)$ is also a lower bound on $ih(T)$.

**Theorem 14.** Let $T$ be a nondegenerate binary decision table with many-valued decisions and $Row(T)$ be the set of rows of $T$. Then $il(T, \bar{\delta}) = iM(T, \bar{\delta})$ for any row $\bar{\delta} \in Row(T)$, and $il(T) = \max\{iM(T, \bar{\delta}) : \bar{\delta} \in Row(T)\}$.

**Proof.** Let $T$ have $n$ columns labeled with attributes $f_1, \ldots, f_n$,

$$\bar{\delta} = (\delta_1, \ldots, \delta_n) \in Row(T),$$

and $T^C$ be complementary to $T$ decision table. It is clear that $Row(T) = Row(T^C)$. One can show that, for any $f_{i_1}, \ldots, f_{i_m} \in \{f_1, \ldots, f_n\}$, the subtable

$$T(f_{i_1}, \delta_{i_1}) \ldots (f_{i_m}, \delta_{i_m})$$

is incomplete relative to $T$ if and only if the subtable $T^C(f_{i_1}, \delta_{i_1}) \ldots (f_{i_m}, \delta_{i_m})$ is degenerate. From here it follows that $iM(T, \bar{\delta}) = M(T^C, \bar{\delta})$. By Corollary 1 $il(T, \bar{\delta}) = l(T^C, \bar{\delta})$ and $il(T) = l(T^C)$. By Theorem 9 $l(T^C, \bar{\delta}) = M(T^C, \bar{\delta})$ for any row $\bar{\delta} \in Row(T^C)$. Therefore $il(T, \bar{\delta}) = iM(T, \bar{\delta})$ for any row $\bar{\delta} \in Row(T)$. By Theorem 9 $l(T^C) = \max\{M(T^C, \bar{\delta}) : \bar{\delta} \in Row(T^C)\}$. Therefore $il(T) = \max\{iM(T, \bar{\delta}) : \bar{\delta} \in Row(T)\}$. \hfill $\square$
3.4 Upper Bounds on Complexity of Trees

In this section, we consider upper bounds on the complexity of decision and inhibitory trees.

3.4.1 Decision Interpretation

We know that \( h(T) \leq R(T) \) (see Corollary \[\ref{corollary:decision-bound}\]). Therefore each upper bound on \( R(T) \) is also an upper bound on \( h(T) \).

**Theorem 15.** \[\ref{theorem:decision-bound}\] Let \( T \) be a nonempty binary decision table with many-valued decisions. Then

\[
R(T) \leq N(T) - 1.
\]

**Theorem 16.** \[\ref{theorem:decision-bound}\] Let \( T \) be a nonempty binary decision table with many-valued decisions. Then

\[
h(T) \leq M(T) \log_2 N(T).
\]

3.4.2 Inhibitory Interpretation

We know that \( ih(T) \leq iR(T) \) (see Corollary \[\ref{corollary:inhibitory-bound}\]). Therefore each upper bound on \( iR(T) \) is also an upper bound on \( ih(T) \).

**Theorem 17.** Let \( T \) be a nondegenerate binary decision table with many-valued decisions. Then

\[
iR(T) \leq N(T) - 1.
\]

**Proof.** Let \( T^C \) be complementary to \( T \) decision table. From Corollary \[\ref{corollary:inhibitory-bound}\] it follows that \( iR(T) = R(T^C) \). It is clear that \( N(T) = N(T^C) \). By Theorem \[\ref{theorem:decision-bound}\] \( R(T^C) \leq N(T^C) - 1 \). Therefore \( iR(T) \leq N(T) - 1 \). \(\square\)
Theorem 18. Let $T$ be a nondegenerate binary decision table with many-valued decisions. Then

$$ih(T) \leq iM(T) \log_2 N(T).$$

Proof. Let $T$ have $n$ columns labeled with attributes $f_1, \ldots, f_n$, and $T^C$ be complementary to $T$ decision table. From Corollary [1] it follows that $ih(T) = h(T^C)$. It is clear that $N(T) = N(T^C)$. From Lemma [1] it follows that $iM(T) = M(T^C)$. By Theorem [16] $h(T^C) \leq M(T^C) \log_2 N(T^C)$ Therefore $ih(T) \leq iM(T) \log_2 N(T)$.

3.5 Approximate Algorithms for Decision and Inhibitory Tree Optimization

In this section, we consider approximate polynomial algorithms for problems of minimization of the decision and inhibitory tree depth.

3.5.1 Optimization of Decision Trees

Let $T$ be a binary decision table with many-valued decisions. A subtable $T'$ of $T$ is called boundary subtable if $T'$ is not degenerate but each proper subtable of $T'$ is degenerate.

We denote by $B(T)$ the number of boundary subtables of the table $T$. We denote by $Tab(t)$, where $t$ is a natural number, the set of decision tables with many-valued decisions such that each row in the table is labeled with a set of decisions which cardinality is at most $t$.

Proposition 2. Let $T'$ be a boundary subtable with $m$ rows. Then each row of $T'$ is labeled with a set of decisions which cardinality is at least $m - 1$.

Corollary 6. Each boundary subtable of a table $T \in Tab(t)$ has at most $t + 1$ rows.
Therefore, for tables from $\text{Tab}(t)$, there exists a polynomial algorithm for computation of the parameter $B(T)$, and for construction of the set of boundary subtables of the table $T$.

Let $T$ be a binary decision table with many-valued decisions. It is clear that $T$ is a degenerate table if and only if $B(T) = 0$.

We now describe an algorithm $U$ which constructs a decision tree $U(T)$ for the binary decision table $T$ with many-valued decisions. Let $T$ have $n$ columns labeled with attributes $f_1, \ldots, f_n$.

**Step 1:** Construct a tree consisting of a single node labeled with the table $T$ and proceed to the second step.

Suppose $t \geq 1$ steps have been made already. The tree obtained at the step $t$ will be denoted by $G$.

**Step $(t + 1)$:** If no one node of the tree $G$ is labeled with a table then we denote by $U(T)$ the tree $G$. The work of the algorithm $U$ is completed.

Otherwise, we choose certain node $v$ in the tree $G$ which is labeled with a subtable of the table $T$. Let the node $v$ be labeled with the table $\Theta$. If $\Theta$ is a degenerate table and $d$ is the minimum decision such that $d \in D(r)$ for any row $r$ of $\Theta$, then instead of $\Theta$ we mark the node $v$ by the number $d$ and proceed to the step $(t + 2)$. Let $\Theta$ be a nondegenerate table. Then, for $i = 1, \ldots, n$, we compute the value

$$Q(f_i) = \max\{B(\Theta(f_i, 0)), B(\Theta(f_i, 1))\}.$$  

We mark the node $v$ by the attribute $f_{i_0}$ where $i_0$ is the minimum $i$ for which $Q(f_i)$ has minimum value. For each $\delta \in \{0, 1\}$, we add to the tree $G$ the node $v(\delta)$, mark this node by the table $\Theta(f_{i_0}, \delta)$, draw the edge from $v$ to $v(\delta)$, and mark this edge by $\delta$. Proceed to the step $(t + 2)$.

**Theorem 19.** Let $T$ be a nondegenerate binary decision table with many-valued decisions.
decisions. Then

\[ h(U(T)) \leq M(T) \ln B(T) + 1. \]

**Corollary 7.** [2] For any nondegenerate binary decision table \( T \) with many-valued decisions,

\[ h(U(T)) \leq h(T) \ln B(T) + 1. \]

The next two statements characterize the complexity of algorithms for minimization of decision tree depth.

**Proposition 3.** [2] The problem of minimization of decision tree depth for nondegenerate binary decision tables with many-valued decisions is \( \text{NP} \)-hard.

**Theorem 20.** [2] If \( \text{NP} \not\subseteq \text{DTIME}(n^{O(\log \log n)}) \) then, for any \( \varepsilon > 0 \), there is no polynomial algorithm which, for a given nondegenerate binary decision table \( T \) with many-valued decisions, constructs a decision tree for \( T \) which depth is at most

\[ (1 - \varepsilon)h(T) \ln B(T). \]

### 3.5.2 Optimization of Inhibitory Trees

Let \( T \) be a nondegenerate binary decision table with many-valued decisions. A subtable \( T' \) of \( T \) is called \emph{i-boundary} subtable if \( T' \) is not incomplete relative to \( T \) but each proper subtable of \( T' \) is incomplete relative to \( T \).

Let \( \Theta \) be a subtable of \( T \). We denote by \( iB(T, \Theta) \) the number of i-boundary subtables of the table \( T \) which are subtables of \( \Theta \). We denote by \( iTab(t) \), where \( t \) is a natural number, the set of binary decision tables \( T \) with many-valued decisions such that \( |D(T)| \leq t \).

**Proposition 4.** Let \( T' \) be an i-boundary subtable of the table \( T \), and \( T' \) contain \( m \) rows. Then \( |D(T)| \geq m \).
Proof. According to the definition of i-boundary subtable, each row \( r \) of \( T' \) should have in the set \( D(r) \) a decision \( d(r) \) such that

\[
d(r) \notin \bigcup_{r' \in \text{Row}(T) \setminus \{r\}} D(r') .
\]

From here it follows that \( |D(T)| \geq m \). □

**Corollary 8.** Each i-boundary subtable of a table \( T \in i\text{Tab}(t) \) has at most \( t \) rows.

Therefore, for tables from \( i\text{Tab}(t) \), there exists a polynomial algorithm for computation of the parameter \( iB(T) \), and for construction of the set of i-boundary subtables of the table \( T \).

Let \( T \) be a nondegenerate binary decision table with many-valued decisions and \( \Theta \) be a subtable of \( T \). It is clear that \( \Theta \) is incomplete relative to \( T \) if and only if \( iB(T, \Theta) = 0 \).

**Lemma 2.** Let \( T \) be a nondegenerate binary decision table with many-valued decisions, \( T^C \) be complementary to \( T \) decision table, \( T_1 \) be a subtable of \( T \), \( T_2 \) be a subtable of \( T^C \), and \( \text{Row}(T_1) = \text{Row}(T_2) \). Then

\[
iB(T, T_1) = B(T_2) .
\]

Proof. Let \( \Theta_1 \) be a subtable of \( T \), \( \Theta_2 \) be a subtable of \( T^C \), and \( \text{Row}(\Theta_1) = \text{Row}(\Theta_2) \). One can show that \( \Theta_1 \) is an i-boundary subtable of \( T \) if and only if \( \Theta_2 \) is a boundary subtable of \( T^C \). From here it follows that \( iB(T, T_1) = B(T_2) \). □

We now describe an algorithm \( iU \) which, for a nondegenerate binary decision table with many-valued decisions \( T \), constructs an inhibitory tree \( iU(T) \) for the table \( T \). Let \( T \) have \( n \) columns labeled with attributes \( f_1, \ldots, f_n \).

**Step 1:** Construct a tree consisting of a single node labeled with the table \( T \) and proceed to the second step.
Suppose \( t \geq 1 \) steps have been made already. The tree obtained at the step \( t \) will be denoted by \( G \).

**Step \((t + 1)\):** If no one node of the tree \( G \) is labeled with a table then we denote by \( iU(T) \) the tree \( G \). The work of the algorithm \( iU \) is completed.

Otherwise, we choose certain node \( v \) in the tree \( G \) which is labeled with a subtable of the table \( T \). Let the node \( v \) be labeled with the table \( \Theta \). If \( \Theta \) is an incomplete subtable relative to \( T \) and \( d \) is the minimum decision such that \( d \in D(T) \setminus D(\Theta) \), then instead of \( \Theta \) we mark the node \( v \) by the expression \( \not= d \) and proceed to the step \((t + 2)\). Let \( \Theta \) be not incomplete relative to \( T \). Then, for \( j = 1, \ldots, n \), we compute the value

\[
iQ(f_j) = \max\{iB(T, \Theta(f_j, 0)), iB(T, \Theta(f_j, 1))\}.
\]

We mark the node \( v \) by the attribute \( f_{j_0} \) where \( j_0 \) is the minimum \( j \) for which \( iQ(f_j) \) has minimum value. For each \( \delta \in \{0, 1\} \), we add to the tree \( G \) the node \( v(\delta) \), mark this node by the table \( \Theta(f_{j_0}, \delta) \), draw the edge from \( v \) to \( v(\delta) \), and mark this edge by \( \delta \). Proceed to the step \((t + 2)\).

**Theorem 21.** Let \( T \) be a nondegenerate binary decision table with many-valued decisions. Then

\[
h(iU(T)) \leq iM(T) \ln(iB(T, T)) + 1.
\]

**Proof.** Let \( T \) have \( n \) columns labeled with attributes \( f_1, \ldots, f_n \),

\[
f_{i_1}, \ldots, f_{i_m} \in \{f_1, \ldots, f_n\},
\]

\( \delta_1, \ldots, \delta_m \in \{0, 1\} \), and \( \alpha = (f_{i_1}, \delta_1) \cdots (f_{i_m}, \delta_m) \). One can show that \( T\alpha \) is incomplete relative to \( T \) if and only if \( T^c\alpha \) is degenerate. By Lemma 2 \( B(T^c\alpha) = iB(T, T\alpha) \). It is not difficult to show that \( D(T) \setminus D(T\alpha) = \bigcap_{r \in \text{Row}(T^c\alpha)} D_{T^c}(r) \). Using these facts it is not difficult to prove that \( iU(T) = U(T^c)^- \) and \( h(U(T^c)) = h(iU(T)) \). From
Theorem 19 it follows that \( h(U(T^C)) \leq M(T^C) \ln B(T^C) + 1 \). From Lemmas 1 and 2 it follows that \( B(T^C) = iB(T, T) \) and \( M(T^C) = iM(T) \). Therefore \( h(iU(T)) \leq iM(T) \ln(iB(T, T)) + 1 \).

\[ \text{Corollary 9. For any nondegenerate binary decision table } T \text{ with many-valued decisions,} \]
\[ h(iU(T)) \leq ih(T) \ln(iB(T, T)) + 1. \]

Proof. By Theorem 12, \( iM(T) \leq ih(T) \). From here and from Theorem 21 it follows that \( h(iU(T)) \leq ih(T) \ln(iB(T, T)) + 1 \).

The next two statements characterize the complexity of algorithms for minimization of inhibitory tree depth.

Proposition 5. The problem of minimization of inhibitory tree depth for binary decision tables with many-valued decisions is \( NP \)-hard.

Proof. Let \( T \) be a nondegenerate binary decision table with many-valued decisions. We denote \( \Theta = T^C \). It is clear that \( \Theta \) is nondegenerate and \( \Theta^C = T \). From Proposition 1 it follows that a decision tree \( \Gamma \) over \( \Theta^C = T \) is a decision tree for \( \Theta^C = T \) if and only if the inhibitory tree \( \Gamma^- \) over \( \Theta = T^C \) is an inhibitory tree for \( \Theta = T^C \). By Corollary 1, \( ih(\Theta) = h(\Theta^C) \). So we have a polynomial time reduction of the problem of minimization of decision tree depth for nondegenerate binary decision table \( T \) with many-valued decisions to the problem of minimization of inhibitory tree depth for nondegenerate binary decision table \( T^C \) with many-valued decisions. By Proposition 3, the problem of minimization of decision tree depth for nondegenerate binary decision tables with many-valued decisions is \( NP \)-hard. Therefore the problem of minimization of inhibitory tree depth for nondegenerate binary decision tables with many-valued decisions is \( NP \)-hard.
Theorem 22. If $NP \not\subseteq DTIME(n^{O(\log \log n)})$ then, for any $\varepsilon > 0$, there is no polynomial algorithm which, for a given nondegenerate binary decision table $T$ with many-valued decisions, constructs an inhibitory tree for $T$ which depth is at most

$$(1 - \varepsilon)ih(T) \ln(iB(T, T)) .$$

Proof. Let us assume that, for some $\varepsilon > 0$, there exists a polynomial algorithm $A$ which, for a given nondegenerate binary decision table $T$ with many-valued decisions, constructs an inhibitory tree for $T$ which depth is at most

$$(1 - \varepsilon)ih(T) \ln(iB(T, T)) .$$

Let $T$ be a nondegenerate binary decision table with many-valued decisions. We denote $\Theta = T^C$. It is clear that $\Theta$ is nondegenerate and $\Theta^C = T$. We apply the algorithm $A$ to the table $\Theta = T^C$. As a result, we obtain an inhibitory tree $\Gamma$ for $\Theta$ such that $h(\Gamma) \leq (1 - \varepsilon)ih(\Theta) \ln(iB(\Theta, \Theta))$. Let us consider a decision tree $\Gamma^+$ over $\Theta^C = T$. It is clear that $(\Gamma^+)^- = \Gamma$. From Proposition 1 it follows that $\Gamma^+$ is a decision tree for $\Theta^C = T$. We have $h(\Gamma^+) \leq (1 - \varepsilon)ih(\Theta) \ln(iB(\Theta, \Theta))$. By Corollary 1 $ih(\Theta) = h(\Theta^C)$. By Lemma 2 $iB(\Theta, \Theta) = B(\Theta^C)$. Therefore $h(\Gamma^+) \leq (1 - \varepsilon)h(T) \ln B(T)$.

As a result, there is a polynomial algorithm which, for a given nondegenerate binary decision table $T$ with many-valued decisions, constructs a decision tree $\Gamma$ for $T$ such that $h(\Gamma) \leq (1 - \varepsilon)h(T) \ln B(T)$ which is impossible if $NP \not\subseteq DTIME(n^{O(\log \log n)})$ (see Theorem 20). Thus, if $NP \not\subseteq DTIME(n^{O(\log \log n)})$ then, for any $\varepsilon > 0$, there is no polynomial algorithm that, for a given nondegenerate binary decision table $T$ with many-valued decisions, constructs an inhibitory tree for $T$ which depth is at most $(1 - \varepsilon)ih(T) \ln(iB(T, T))$. 

72
3.6 Exact Algorithms for Optimization of Decision and Inhibitory Trees

In this section, we discuss exact algorithms for minimization of decision and inhibitory tree depth.

3.6.1 Decision Interpretation

A dynamic programming algorithm $W$ for minimization of decision tree depth was described in [2].

**Theorem 23.** [2] For any nondegenerate binary decision table $T$ with many-valued decisions, the algorithm $W$ constructs a decision tree $W(T)$ for the table $T$ such that $h(W(T)) = h(T)$, and makes exactly $2|SEP(T)| + 3$ steps. The time of the algorithm $W$ work is bounded from below by $|SEP(T)|$ and bounded from above by a polynomial on $|SEP(T)|$ and the number of columns in the table $T$.

3.6.2 Inhibitory Interpretation

In the next chapters, we show that there exists a dynamic programming algorithm for minimization of inhibitory tree depth which time complexity is polynomial depending on the size of the input table $T$ and the number of separable subtables of $T$. 
Chapter 4

Decision Tables and Tools for Study of Pareto Optimal Points

In this chapter, we consider some notions connected with decision tables with many-valued decisions (the notions of the table, directed acyclic graph for this table, uncertainty and completeness measures, and restricted information system) and discuss tools for the work with Pareto optimal points.

The most of the definitions and results considered in this chapter were obtained jointly with Hassan AbouEisha, Fawaz Alsolami, Talha Amin, and Shahid Hussain.

4.1 Decision Tables

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

We denote by $T^C$ complementary to $T$ decision table obtained from the table $T$ by changing, for each row $r \in \text{Row}(T)$, the set $D(r)$ with the set $D(T) \setminus D(r)$. When we consider complementary to $T$ table $T^C$ or when we study inhibitory trees for $T$, we will assume that, for any row $r$ of $T$, $D(r) \neq D(T)$. All results for decision trees
continue to be true if in the considered decision table $T$ there are rows $r$ such that $D(r) = D(T)$.

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

A decision table is called empty if it has no rows. We denote by $T$ the set of all decision tables with many-valued decisions and by $T^+$ the set of nonempty decision tables with many valued decisions. Let $T \in T$. The table $T$ is called degenerate if it is empty or has a common decision – a decision $d \in D(T)$ such that $d \in D(r)$ for any row $r$ of $T$. We denote by $\text{dim}(T)$ the number of columns (conditional attributes) in $T$. We denote by $N(T)$ the number of rows in the table $T$ and, for any $d \in \omega$, we denote by $N_d(T)$ the number of rows $r$ of $T$ such that $d \in D(r)$. By $\text{mcd}(T)$ we denote the most common decision for $T$ which is the minimum decision $d_0$ from $D(T)$ such that $N_{d_0}(T) = \max\{N_d(T) : d \in D(T)\}$. If $T$ is empty then $\text{mcd}(T) = 0$. A nonempty decision table is called diagnostic if $D(r_1) \cap D(r_2) = \emptyset$ for any rows $r_1, r_2$ of $T$ such that $r_1 \neq r_2$.

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 $\text{range}(T) = \max\{|E(T, f_i)| : i = 1, \ldots, n\}$.

Let $T$ be a nonempty decision table. A subtable of $T$ is a table obtained from $T$ by removal of some rows. We denote by $\text{Word}(T)$ the set of all finite words over the alphabet $\{(f_i, a) : f_i \in \{f_1, \ldots, f_n\}, a \in \omega\}$ including the empty word $\lambda$. Let $\alpha \in \text{Word}(T)$. We define now a subtable $T\alpha$ of the table $T$. If $\alpha = \lambda$ then $T\alpha = T$. If $\alpha \neq \lambda$ and $\alpha = (f_{i_1}, a_1) \ldots (f_{i_m}, a_m)$ then $T\alpha = T(f_{i_1}, a_1) \ldots (f_{i_m}, a_m)$.

75
is 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 $T$. Note that $N(T) \leq |SEP(T)|$. It is clear that the size of each subtable of $T$ is at most the size of $T$.

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

### 4.2 Uncertainty Measures

Let $\mathbb{R}$ be the set of real numbers and $\mathbb{R}_+$ be the set of all nonnegative real numbers. An uncertainty measure is a function $U : \mathcal{T} \rightarrow \mathbb{R}$ 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 $T$, the value of each of the considered functions is equal to 0) are uncertainty measures:

- Misclassification error: $\text{me}(T) = N(T) - N_{\text{med}(T)}(T)$.

- Relative misclassification error: $\text{rme}(T) = (N(T) - N_{\text{med}(T)}(T))/N(T)$.

- Absence: $\text{abs}(T) = \prod_{d \in D(T)} (1 - N_{d}(T)/N(T))$.

We assume that each of the following numerical operations (we call these operations basic) has time complexity $O(1)$: $\max(x, y)$, $x + y$, $x \times y$, $x \div y$, $\log_2 x$. This assumption is reasonable for computations with floating-point numbers. Under this assumption, each of the considered three uncertainty measures has polynomial time complexity depending on the size of decision tables.
4.3 Completeness Measures

A completeness measure is a function $W$ defined on the pairs nonempty decision table $T$ and its subtable $\Theta$ such that $W(T, \Theta) \geq 0$ for any $T \in \mathcal{T}$ and its subtable $\Theta$, and $W(T, \Theta) = 0$ if and only if $\Theta$ is incomplete relative to $T$. One can show that the following functions (we assume that, for any empty subtable $\Theta$, the value of each of the considered functions is equal to 0) are completeness measures:

- **Inhibitory misclassification error**: $ime(T, \Theta) = N_{\text{lcd}(T, \Theta)}(\Theta)$.

- **Inhibitory relative misclassification error**: $irme(T, \Theta) = \frac{N_{\text{lcd}(T, \Theta)}(\Theta)}{N(\Theta)}$.

- **Inhibitory absence**: $iabs(T, \Theta) = \prod_{d \in D(T)}(N_d(\Theta)/N(\Theta))$.

Each of the considered three completeness measures has polynomial time complexity depending on the size of decision tables.

It is not difficult to prove the following statement.

**Lemma 3.** Let $T$ be a nondegenerate decision table and $\alpha \in \text{Word}(T)$. Then

1. $D(T) = D(T^C)$;

2. $N(T\alpha) = N(T^C\alpha)$;

3. $T\alpha$ is incomplete relative to $T$ if and only if $T^C\alpha$ is degenerate;

4. For any $d \in D(T)$, $N_d(T\alpha) = N(T^C\alpha) - N_d(T^C\alpha)$;

5. $\text{lcd}(T, T\alpha) = \text{mcd}(T^C\alpha)$.

Let $U$ be an uncertainty measure and $W$ be a completeness measure. We will say that $W$ and $U$ are dual if, for any nondegenerate $T \in \mathcal{T}$ and any $\alpha \in \text{Word}(T)$, $W(T, T\alpha) = U(T^C\alpha)$. 
Proposition 6. The following pairs of completeness and uncertainty measures are dual: $ime$ and $me$, $irme$ and $rme$, $iabs$ and $abs$.

Proof. Let $T$ be a nondegenerate decision table and $\alpha \in \text{Word}(T)$. From Lemma 3 it follows that

$$ime(T, T\alpha) = N_{\text{lcd}(T,T\alpha)}(T\alpha) = N(T^C\alpha) - N_{\text{med}(T^C\alpha)}(T^C\alpha) = me(T^C\alpha),$$

$$irme(T, T\alpha) = N_{\text{lcd}(T,T\alpha)}(T\alpha)/N(T\alpha) = (N(T^C\alpha) - N_{\text{med}(T^C\alpha)}(T^C\alpha))/N(T\alpha) = rme(T^C\alpha),$$

$$iabs(T, T\alpha) = \prod_{d \in D(T)} (N_d(T\alpha)/N(T\alpha)) = \prod_{d \in D(T^C)} (1-N_d(T^C\alpha)/N(T^C\alpha)) = abs(T^C\alpha).$$

\[\square\]

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

Let $T$ be a nonempty decision table with $n$ conditional attributes $f_1, \ldots, f_n$, $U$ be an uncertainty measure, and $\alpha \in \mathbb{R}_+$. We now consider an 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.

Algorithm $\mathcal{A}_1$

**Input:** A nonempty decision table $T$ with $n$ conditional attributes $f_1, \ldots, f_n$, an uncertainty measure $U$, and a number $\alpha \in \mathbb{R}_+$.

**Output:** Directed acyclic graph $\Delta_{U,\alpha}(T)$. 

78
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) $\Theta$ that has not been processed yet.

3. If $U(\Theta) \leq \alpha$ mark the node $\Theta$ as processed and proceed to step 2.

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

We now analyze time complexity of the algorithm $A_1$. By $L(\Delta_{U,\alpha}(T))$ we denote the number of nodes in the graph $\Delta_{U,\alpha}(T)$.

**Proposition 7.** Let the algorithm $A_1$ use an uncertainty measure $U$ which has polynomial time complexity depending on the size of the input decision table $T$. Then the time complexity of the algorithm $A_1$ is bounded from above by a polynomial on the size of the input table $T$ and the number $|SEP(T)|$ of different separable subtables of $T$.

**Proof.** Since the uncertainty measure, $U$ has polynomial time complexity depending on the size of decision tables, each step of the algorithm $A_1$ has polynomial time complexity depending on the size of the table $T$ and the number $L(\Delta_{U,\alpha}(T))$. The number of steps is $O(L(\Delta_{U,\alpha}(T)))$. Therefore the time complexity of the algorithm...
$\mathcal{A}_1$ is bounded from above by a polynomial on the size of the input table $T$ and the number $L(\Delta_{U,\alpha}(T))$. The number $L(\Delta_{U,\alpha}(T))$ is bounded from above by the number $|SEP(T)|$ of different separable subtables of $T$.

In Section 4.3, we describe classes of decision tables such that the number of separable subtables in tables from the class is bounded from above by a polynomial on the number of conditional attributes in the table, and the size of tables is bounded from above by a polynomial on the number of conditional attributes. For each such class, the time complexity of the algorithm $\mathcal{A}_1$ is polynomial depending on the number of conditional attributes in decision tables.

**Remark 1.** Note that, for any decision table $T$, the graph $\Delta_{U,0}(T)$ does not depend on the uncertainty measure $U$. We denote this graph $\Delta(T)$. Note also that $L(\Delta(T)) = |SEP(T)|$ for any diagnostic decision table $T$.

A node of the directed graph is called *terminal* if there are no edges starting in this node. A *bundle-preserving subgraph* of the graph $\Delta_{U,\alpha}(T)$ is a graph $G$ obtained from $\Delta_{U,\alpha}(T)$ by removal of some bundles of edges such that each nonterminal node of $\Delta_{U,\alpha}(T)$ keeps at least one bundle of edges starting in this node. By definition, $\Delta_{U,\alpha}(T)$ is a bundle-preserving subgraph of $\Delta_{U,\alpha}(T)$. A node $\Theta$ of the graph $G$ is terminal if and only if $U(\Theta) \leq \alpha$. We denote by $L(G)$ the number of nodes in the graph $G$.

### 4.5 Restricted Information Systems

In this section, we describe classes of decision tables for which algorithms that deal with the graphs $\Delta_{U,\alpha}(T)$ have polynomial time complexity depending on the number of conditional attributes in the input table $T$.

Let $A$ be a nonempty set and $F$ be a nonempty set of non-constant functions from $A$ to $E_k = \{0, ..., k - 1\}$, $k \geq 2$. Functions from $F$ are called *attributes*, and the
triple $\mathcal{U} = (A, E_k, F)$ is called a $k$-valued information system. With the exception of Chapter 11, we will consider only such information systems. They are close enough to the information systems proposed by Zdzisław Pawlak [54].

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

$$\{f_1(x) = \delta_1, \ldots, f_n(x) = \delta_n\}$$

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

Let us consider the function

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

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

Let $r$ be a natural number. The information system $\mathcal{U}$ will be called $r$-restricted if, for each consistent system of equations of the kind

$$\{f_1(x) = \delta_1, \ldots, f_m(x) = \delta_m\}$$

where $m \in \omega \setminus \{0\}$, $f_1, \ldots, f_m \in F$ and $\delta_1, \ldots, \delta_m \in E_k$, there exists a subsystem of this system which has the same set of solutions over the set $A$ and contains at most $r$
equations. The information system \( \mathcal{U} \) will be called \textit{restricted} if it is \( r \)-restricted for some natural \( r \).

\textbf{Theorem 24.} \cite{55} Let \( \mathcal{U} = (A, E_k, F) \) be a \( k \)-valued information system. Then the following statements hold:

1. If \( \mathcal{U} \) is an \( r \)-restricted information system for some natural \( r \) then \( SEP_\mathcal{U}(n) \leq (nk)^r + 1 \) for any natural \( n \).

2. If \( \mathcal{U} \) is not a restricted information system then \( SEP_\mathcal{U}(n) \geq 2^n \) for any natural \( n \).

We now evaluate the time complexity of the algorithm \( A_1 \) for decision tables over a restricted information system \( \mathcal{U} \) under the assumption that the uncertainty measure \( U \) used by \( A_1 \) has polynomial time complexity.

\textbf{Lemma 4.} Let \( \mathcal{U} \) be a restricted information system. Then, for decision tables from \( \mathcal{T}(\mathcal{U}) \), both the size and the number of separable subtables are bounded from above by polynomials on the number of conditional attributes.

\textbf{Proof.} Let \( \mathcal{U} \) be a \( k \)-valued information system which is \( r \)-restricted. For any decision table \( T \in \mathcal{T}(\mathcal{U}) \), each value of each conditional attribute is at most \( k \), the value of each decision is at most \( k^{\dim(T)} \), the cardinality of each set of decisions attached to rows of \( T \) is at most \( \dim(T)^k \) and, by Theorem \ref{24}, \( N(T) \leq |SEP(T)| \leq (\dim(T)k)^r + 1 \). From here it follows that the size of decision tables from \( \mathcal{T}(\mathcal{U}) \) is bounded from above by a polynomial on the number of conditional attributes in decision tables. By Theorem \ref{24}, the number of separable subtables for decision tables from \( \mathcal{T}(\mathcal{U}) \) is bounded from above by a polynomial on the number of conditional attributes in decision tables. \( \square \)

\textbf{Proposition 8.} Let \( \mathcal{U} \) be a restricted information system, and the uncertainty measure \( U \) used by the algorithm \( A_1 \) have polynomial time complexity depending on the
size of decision tables. Then the algorithm $A_1$ has polynomial time complexity for decisions from $T(U)$ depending on the number of conditional attributes.

Proof. By Proposition $[7]$ the time complexity of the algorithm $A_1$ is bounded from above by a polynomial on the size of the input table $T$ and the number $|SEP(T)|$ of different separable subtables of $T$. From Lemma $[4]$ it follows that, for decision tables from $T(U)$, both the size and the number of separable subtables are bounded from above by polynomials on the number of conditional attributes. \qed

Remark 2. Let $U$ be an information system which is not restricted. Using Remark $[7]$ and Theorem $[24]$ one can show that there is no an algorithm which constructs the graph $\Delta(T)$ for decision tables $T \in T(U)$ and which time complexity is bounded from above by a polynomial on the number of conditional attributes in the considered decision tables.

Let us consider a family of restricted information systems. Let $d$ and $t$ be natural numbers, $f_1, \ldots, f_t$ be functions from $\mathbb{R}^d$ to $\mathbb{R}$, and $s$ be a function from $\mathbb{R}$ to $\{0, 1\}$ such that $s(x) = 0$ if $x < 0$ and $s(x) = 1$ if $x \geq 0$. Then the 2-valued information system $(\mathbb{R}^d, E_2, F)$ where $F = \{s(f_i + c) : i = 1, \ldots, t, c \in \mathbb{R}\}$ is a $2t$-restricted information system.

If $f_1, \ldots, f_t$ are linear functions then we deal with attributes corresponding to $t$ families of parallel hyperplanes in $\mathbb{R}^d$. This is usual for decision trees constructed by CART $[18]$ for datasets with $t$ numerical attributes only (in this case, $d = t$).

We consider now a class of so-called linear information systems for which all restricted systems are known. Let $P$ be the set of all points in the two-dimensional Euclidean plane and $l$ be a straight line (line in short) in the plane. This line divides the plane into two open half-planes $H_1$ and $H_2$, and the line $l$. Two attributes correspond to the line $l$. The first attribute takes value 0 on points from $H_1$, and value 1 on points from $H_2$ and $l$. The second one takes value 0 on points from $H_2$, and
and value 1 on points from $H_1$ and $l$. We denote by $\mathcal{L}$ the set of all attributes corresponding to lines in the plane. Information systems of the kind $(P, E_2, F)$ where $F \subseteq \mathcal{L}$, will be called linear information systems. We describe all restricted linear information systems.

Let $l$ be a line in the plane. Let us denote by $\mathcal{L}(l)$ the set of all attributes corresponding to lines which are parallel to $l$. Let $p$ be a point in the plane. We denote by $\mathcal{L}(p)$ the set of all attributes corresponding to lines which pass through $p$. A set $C$ of attributes from $\mathcal{L}$ is called a clone if $C \subseteq \mathcal{L}(l)$ for some line $l$ or $C \subseteq \mathcal{L}(p)$ for some point $p$.

**Theorem 25.** [56] A linear information system $(P, E_2, F)$ is restricted if and only if $F$ is the union of a finite number of clones.

### 4.6 Time Complexity of Algorithms on $\Delta_{U,\alpha}(T)$

In this thesis, we consider some algorithms which deal with the graph $\Delta_{U,\alpha}(T)$. To evaluate time complexity of these algorithms, we will count the number of elementary operations made by the algorithms. These operations can either be basic numerical operations or computations of numerical parameters of decision tables. We assume, as we already mentioned, that each basic numerical operation ($\max(x, y)$, $x + y$, $x - y$, $x \times y$, $x \div y$, $\log_2 x$) has time complexity $O(1)$. This assumption is reasonable for computations with floating-point numbers. Furthermore, computing the considered parameters of decision tables usually has polynomial time complexity depending on the size of the decision table.

**Proposition 9.** Let, for some algorithm $A$ working with decision tables, the number of elementary operations (basic numerical operations and computations of numerical parameters of decision tables) be polynomial depending on the size of the input table $T$ and on the number of separable subtables of $T$, and the computations of parameters
of decision tables used by the algorithm \( A \) have polynomial time complexity depending on the size of decision tables. Then, for any restricted information system \( U \), the algorithm \( A \) has polynomial time complexity for decision tables from \( \mathcal{T}(U) \) depending on the number of conditional attributes.

**Proof.** Let \( U \) be a restricted information system. From Lemma 4, it follows that the size and the number of separable subtables for decision tables from \( \mathcal{T}(U) \) are bounded from above by polynomials on the number of conditional attributes in the tables. From here it follows that the algorithm \( A \) has polynomial time complexity for decision tables from \( \mathcal{T}(U) \) depending on the number of conditional attributes. \( \square \)

### 4.7 Tools for Study of Pareto Optimal Points

In this section, we consider tools (statements and algorithms) which are used for the study of Pareto optimal points for bi-criteria optimization problems.

Let \( \mathbb{R}^2 \) be the set of pairs of real numbers (points). We consider a partial order \( \leq \) on the set \( \mathbb{R}^2 \): \((c,d) \leq (a,b)\) if \( c \leq a \) and \( d \leq b \). Two points \( \alpha \) and \( \beta \) are comparable if \( \alpha \leq \beta \) or \( \beta \leq \alpha \). A subset of \( \mathbb{R}^2 \) 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 \( \alpha \) and \( \beta \) 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 \( \mathbb{R}^2 \). A point \( \alpha \in A \) is called a Pareto optimal point for \( A \) if there is no a point \( \beta \in A \) such that \( \beta < \alpha \). We denote by \( \text{Par}(A) \) the set of Pareto optimal points for \( A \). It is clear that \( \text{Par}(A) \) is an antichain.

**Lemma 5.** Let \( A \) be a nonempty finite subset of the set \( \mathbb{R}^2 \). Then, for any point \( \alpha \in A \), there is a point \( \beta \in \text{Par}(A) \) such that \( \beta \leq \alpha \).

**Proof.** Let \( \beta = (a,b) \) be a point from \( A \) such that \((a,b) \leq \alpha \) and \( a + b = \min\{c + d : (c,d) \in A, (c,d) \leq \alpha \} \). Then \((a,b) \in \text{Par}(A)\). \( \square \)
Lemma 6. Let $A, B$ be nonempty finite subsets of the set $\mathbb{R}^2$, $A \subseteq B$, and, for any $\beta \in B$, there exists $\alpha \in A$ such that $\alpha \leq \beta$. Then $\text{Par}(A) = \text{Par}(B)$.

**Proof.** Let $\beta \in \text{Par}(B)$. Then there exists $\alpha \in A$ such that $\alpha \leq \beta$. By Lemma 5, there exists $\gamma \in \text{Par}(A)$ such that $\gamma \leq \alpha$. Therefore $\gamma \leq \beta$ and $\gamma = \beta$ since $\beta \in \text{Par}(B)$. Hence $\text{Par}(B) \subseteq \text{Par}(A)$.

Let $\alpha \in \text{Par}(A)$. By Lemma 5, there exists $\beta \in \text{Par}(B)$ such that $\beta \leq \alpha$. We know that there exists $\gamma \in A$ such that $\gamma \leq \beta$. Therefore $\gamma \leq \alpha$ and $\gamma = \alpha$ since $\alpha \in \text{Par}(A)$. As a result, we have $\beta = \alpha$ and $\text{Par}(A) \subseteq \text{Par}(B)$. Hence $\text{Par}(A) = \text{Par}(B)$. \qed

Lemma 7. Let $A$ be a nonempty finite subset of $\mathbb{R}^2$. Then

$$|\text{Par}(A)| \leq \min \left( |A^{(1)}|, |A^{(2)}| \right)$$

where $A^{(1)} = \{a : (a, b) \in A\}$ and $A^{(2)} = \{b : (a, b) \in A\}$.

**Proof.** Let $(a, b), (c, d) \in \text{Par}(A)$ and $(a, b) \neq (c, d)$. Then $a \neq c$ and $b \neq d$ (otherwise, $(a, b)$ and $(c, d)$ are comparable). Therefore $|\text{Par}(A)| \leq \min \left( |A^{(1)}|, |A^{(2)}| \right)$. \qed

Points from $\text{Par}(A)$ can be ordered in the following way: $(a_1, b_1), \ldots, (a_t, b_t)$ where $a_1 < \ldots < a_t$. Since points from $\text{Par}(A)$ are incomparable, $b_1 > \ldots > b_t$. We will refer to the sequence $(a_1, b_1), \ldots, (a_t, b_t)$ as the normal representation of the set $\text{Par}(A)$.

We now describe an algorithm which, for a given non-empty finite subset $A$ of the set $\mathbb{R}^2$, constructs the normal representation of the set $\text{Par}(A)$. We assume that $A$ is a multiset containing, possibly, repeating elements. The cardinality $|A|$ of $A$ is the total number of elements in $A$.

Algorithm $A_2$
**Input:** A nonempty finite subset \( A \) of the set \( \mathbb{R}^2 \) containing, possibly, repeating elements (multiset).

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

1. Set \( P \) equal to the empty sequence.

2. Construct a sequence \( B \) of all points from \( A \) ordered according to the first coordinate in the ascending order.

3. If there is only one point in the sequence \( B \), then 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 \( B \).

4. If \( \alpha \) and \( \beta \) are comparable then remove \( \alpha \) and \( \beta \) from \( B \), add the point \( \min(\alpha, \beta) \) to the beginning of \( B \), and proceed to step 3.

5. If \( \alpha \) and \( \beta \) are not comparable (in this case \( \alpha_1 < \beta_1 \) and \( \alpha_2 > \beta_2 \) ) then remove \( \alpha \) from \( B \), add the point \( \alpha \) to the end of \( P \), and proceed to step 3.

**Proposition 10.** Let \( A \) be a nonempty finite subset of the set \( \mathbb{R}^2 \) containing, possibly, repeating elements (multiset). Then the algorithm \( A_2 \) returns the normal representation of the set \( Par(A) \) of Pareto optimal points for \( A \) and makes \( O(|A| \log |A|) \) comparisons.

**Proof.** The step 2 of the algorithm requires \( O(|A| \log |A|) \) comparisons. Each call to step 3 (with the exception of the last one) leads to two comparisons. The number of calls to step 3 is at most \( |A| \). Therefore the algorithm \( A_2 \) makes \( O(|A| \log |A|) \) comparisons.

Let the output sequence \( P \) be equal to \((a_1, b_1), \ldots, (a_t, b_t)\) and let us set \( Q = \{(a_1, b_1), \ldots, (a_t, b_t)\} \). It is clear that \( a_1 < \ldots < a_t \), \( b_1 > \ldots > b_t \) and, for any \( \alpha \in A \),
\(\alpha \notin Q\), there exists \(\beta \in Q\) such that \(\beta < \alpha\). From here it follows that \(\text{Par}(A) \subseteq Q\) and \(Q\) is an antichain. Let us assume that there exists \(\gamma \in Q\) which does not belong to \(\text{Par}(A)\). 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 \(\beta \leq \alpha\). This results in two different points \(\beta\) and \(\gamma\) from \(Q\) being comparable, which is impossible. Therefore \(Q = \text{Par}(A)\) and \(P\) is the normal representation of the set \(\text{Par}(A)\).

**Remark 3.** Let \(A\) be a nonempty finite subset of \(\mathbb{R}^2\), \((a_1, b_1), \ldots, (a_t, b_t)\) be the normal representation of the set \(\text{Par}(A)\), and \(\text{rev}(A) = \{(b, a) : (a, b) \in A\}\). Then \(\text{Par}(\text{rev}(A)) = \text{rev}(\text{Par}(A))\) and \((b_t, a_t), \ldots, (b_1, a_1)\) is the normal representation of the set \(\text{Par}(\text{rev}(A))\).

**Lemma 8.** Let \(A\) be a nonempty finite subset of \(\mathbb{R}^2\), \(B \subseteq A\), and \(\text{Par}(A) \subseteq B\). Then \(\text{Par}(B) = \text{Par}(A)\).

**Proof.** It is clear that \(\text{Par}(A) \subseteq \text{Par}(B)\). Let us assume that, for some \(\beta, \beta \in \text{Par}(B)\) and \(\beta \notin \text{Par}(A)\). Then there exists \(\alpha \in A\) such that \(\alpha < \beta\). By Lemma 5, there exists \(\gamma \in \text{Par}(A) \subseteq B\) such that \(\gamma \leq \alpha\). Therefore \(\gamma < \beta\) and \(\beta \notin \text{Par}(B)\). Hence \(\text{Par}(B) = \text{Par}(A)\).

**Lemma 9.** Let \(A_1, \ldots, A_k\) be nonempty finite subsets of \(\mathbb{R}^2\). Then \(\text{Par}(A_1 \cup \ldots \cup A_k) \subseteq \text{Par}(A_1) \cup \ldots \cup \text{Par}(A_k)\).

**Proof.** Let \(\alpha \in (A_1 \cup \ldots \cup A_k) \setminus (\text{Par}(A_1) \cup \ldots \cup \text{Par}(A_k))\). Then there is \(i \in \{1, \ldots, k\}\) such that \(\alpha \in A_i\) but \(\alpha \notin \text{Par}(A_i)\). Therefore there is \(\beta \in A_i\) such that \(\beta < \alpha\). Hence \(\alpha \notin \text{Par}(A_1 \cup \ldots \cup A_k)\), and \(\text{Par}(A_1 \cup \ldots \cup A_k) \subseteq \text{Par}(A_1) \cup \ldots \cup \text{Par}(A_k)\).

A function \(f : \mathbb{R}^2 \to \mathbb{R}\) is called **increasing** if \(f(x_1, y_1) \leq f(x_2, y_2)\) for any \(x_1, x_2, y_1, y_2 \in \mathbb{R}\) such that \(x_1 \leq x_2\) and \(y_1 \leq y_2\). For example, \(\text{sum}(x, y) = x + y\) and \(\text{max}(x, y)\) are increasing functions.

Let \(f, g\) be increasing functions from \(\mathbb{R}^2\) to \(\mathbb{R}\), and \(A, B\) be nonempty finite subsets of the set \(\mathbb{R}^2\). We denote by \(A \langle fg \rangle B\) the set \(\{(f(a, c), g(b, d)) : (a, b) \in A, (c, d) \in B\}\).
Lemma 10. Let $A, B$ be nonempty finite subsets of $\mathbb{R}^2$, and $f, g$ be increasing functions from $\mathbb{R}^2$ to $\mathbb{R}$. Then $\text{Par}(A, \langle fg \rangle B) \subseteq \text{Par}(A) \langle fg \rangle \text{Par}(B)$.

Proof. Let $\beta \in \text{Par}(A, \langle fg \rangle B)$ and $\beta = (f(a, c), g(b, d))$ where $(a, b) \in A$ and $(c, d) \in B$. Then, by Lemma 5, there exist $(a', b') \in \text{Par}(A)$ and $(c', d') \in \text{Par}(B)$ such that $(a', b') \leq (a, b)$ and $(c', d') \leq (c, d)$. It is clear that $\alpha = (f(a', c'), g(b', d')) \leq (f(a, c), g(b, d)) = \beta$, and $\alpha \in \text{Par}(A) \langle fg \rangle \text{Par}(B)$. Since $\beta \in \text{Par}(A, \langle fg \rangle B)$, we have $\beta = \alpha$. Therefore $\text{Par}(A, \langle fg \rangle B) \subseteq \text{Par}(A) \langle fg \rangle \text{Par}(B)$.

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

Algorithm $A_3$

Input: Increasing functions $f, g$ from $\mathbb{R}^2$ to $\mathbb{R}$, and sets $\text{Par}(P_1), \ldots, \text{Par}(P_t)$ for some nonempty finite subsets $P_1, \ldots, P_t$ of $\mathbb{R}^2$.

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

1. Set $B_1 = \text{Par}(P_1)$ and $i = 2$.

2. Construct the multiset

$$A_i = B_{i-1} \langle fg \rangle \text{Par}(P_i) = \{(f(a, c), g(b, d)) : (a, b) \in B_{i-1}, (c, d) \in \text{Par}(P_i)\}$$

– we will not remove equal pairs from the constructed set.

3. Using algorithm $A_2$, construct the set $B_i = \text{Par}(A_i)$.

4. If $i = t$ then return $B_i$ and finish the work of the algorithm. Otherwise, set $i = i + 1$ and proceed to step 2.
Proposition 11. Let \( f,g \) be increasing functions from \( \mathbb{R}^2 \) to \( \mathbb{R} \), \( P_1,\ldots,P_t \) be non-empty finite subsets of \( \mathbb{R}^2 \), \( Q_1 = P_1 \), and, for \( i = 2,\ldots,t \), \( Q_i = Q_{i-1} \langle fg \rangle P_i \). Then the algorithm \( \mathcal{A}_3 \) returns the set \( \text{Par}(Q_t) \).

Proof. 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 \( \text{Par}(Q_i) \). Since \( B_1 = \text{Par}(P_1) \) and \( Q_1 = P_1 \), we have \( B_1 = \text{Par}(Q_1) \). Let for some \( i-1, 2 \leq i \leq t \), the considered statement hold, i.e., \( B_{i-1} = \text{Par}(Q_{i-1}) \). Then \( B_i = \text{Par}(B_{i-1} \langle fg \rangle \text{Par}(P_i)) = \text{Par}(\text{Par}(Q_{i-1}) \langle fg \rangle \text{Par}(P_i)) \). We know that \( Q_i = Q_{i-1} \langle fg \rangle P_i \). By Lemma \([10]\), \( \text{Par}(Q_i) \subseteq \text{Par}(Q_{i-1}) \langle fg \rangle \text{Par}(P_i) \). By Lemma \([8]\),

\[
\text{Par}(Q_i) = \text{Par}(\text{Par}(Q_{i-1}) \langle fg \rangle \text{Par}(P_i)) .
\]

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

Proposition 12. Let \( f,g \) be increasing functions from \( \mathbb{R}^2 \) to \( \mathbb{R} \),

\[
f \in \{ x + y, \max(x,y) \} , \]

\( P_1,\ldots,P_t \) be nonempty finite subsets of \( \mathbb{R}^2 \), \( Q_1 = P_1 \), and, for \( i = 2,\ldots,t \), \( Q_i = Q_{i-1} \langle fg \rangle P_i \). Let \( P^{(1)}_i = \{ a : (a,b) \in P_i \} \) for \( i = 1,\ldots,t \), \( m \in \omega \), and

\[
P^{(1)}_i \subseteq \{ 0,1,\ldots,m \}
\]

for \( i = 1,\ldots,t \). Then, during the construction of the set \( \text{Par}(Q_t) \), the algorithm \( \mathcal{A}_3 \) makes

\[
O(t^2m^2 \log(tm))
\]
elementary operations (computations of \( f, g \) and comparisons) if \( f = x + y \), and
\[
O(tm^2 \log m)
\]
elementary operations (computations of \( f, g \) and comparisons) if \( f = \max(x, y) \). If \( f = x + y \) then \(|\text{Par}(Q_i)| \leq tm + 1\), and if \( f = \max(x, y) \) then \(|\text{Par}(Q_i)| \leq m + 1\).

**Proof.** For \( i = 1, \ldots, t \), we denote \( p_i = |\text{Par}(P_i)| \) and \( q_i = |\text{Par}(Q_i)| \). Let \( i \in \{2, \ldots, t\} \). To construct the multiset \( A_i = \text{Par}(Q_{i-1}) \langle fg \rangle \text{Par}(P_i) \), we need to compute the values of \( f \) and \( g \) a number of times equal to \( q_{i-1}p_i \). The cardinality of \( A_i \) is equal to \( q_{i-1}p_i \). We apply to \( A_i \) the algorithm \( A_2 \) which makes \( O(q_{i-1}p_i \log(q_{i-1}p_i)) \) comparisons. As a result, we find the set \( \text{Par}(A_i) = \text{Par}(Q_i) \). To construct the sets \( \text{Par}(Q_1), \ldots, \text{Par}(Q_t) \), the algorithm \( A_3 \) makes \( \sum_{i=2}^t q_{i-1}p_i \) computations of \( f \), \( \sum_{i=2}^t q_{i-1}p_i \) computations of \( g \), and \( O(\sum_{i=2}^t q_{i-1}p_i \log(q_{i-1}p_i)) \) comparisons.

We know that \( P_i^{(1)} \subseteq \{0, 1, \ldots, m\} \) for \( i = 1, \ldots, t \). Then, by Lemma 7, \( p_i \leq m + 1 \) for \( i = 1, \ldots, t \).

Let \( f = x + y \). Then, for \( i = 1, \ldots, t \), \( Q_i^{(1)} = \{a : (a, b) \in Q_i\} \subseteq \{0, 1, \ldots, im\} \) and by Lemma 7, \( q_i \leq im + 1 \). In this case, to construct the sets \( \text{Par}(Q_1), \ldots, \text{Par}(Q_t) \) the algorithm \( A_3 \) makes \( O(t^2m^2) \) computations of \( f \), \( O(t^2m^2) \) computations of \( g \), and \( O(t^2m^2 \log(tm)) \) comparisons, i.e.,
\[
O(t^2m^2 \log(tm))
\]
elementary operations (computations of \( f, g \), and comparisons). Since \( q_t \leq tm + 1 \), \(|\text{Par}(Q_t)| \leq tm + 1\).

Let \( f = \max(x, y) \). Then, for \( i = 1, \ldots, t \), \( Q_i^{(1)} = \{a : (a, b) \in Q_i\} \subseteq \{0, 1, \ldots, m\} \) and by Lemma 7, \( q_i \leq m + 1 \). In this case, to construct the sets \( \text{Par}(Q_1), \ldots, \text{Par}(Q_t) \) the algorithm \( A_3 \) makes \( O(tm^2) \) computations of \( f \), \( O(tm^2) \) computations of \( g \), and
$O(tm^2 \log m)$ comparisons, i.e.,

$$O(tm^2 \log m)$$

elementary operations (computations of $f$, $g$, and comparisons). Since $q_t \leq m + 1$, $|Par(Q_t)| \leq m + 1$.

Similar analysis can be done for the sets $P_i^{(2)} = \{b : (a,b) \in P_i\}$, $Q_i^{(2)} = \{b : (a,b) \in Q_i\}$, and the function $g$.

Let $A$ be a nonempty finite subset of $\mathbb{R}^2$. We correspond to $A$ a partial function $F_A : \mathbb{R} \to \mathbb{R}$ defined in the following way: $F_A(x) = \min\{b : (a,b) \in A, a \leq x\}$.

**Lemma 11.** Let $A$ be a nonempty finite subset of $\mathbb{R}^2$, and $(a_1,b_1), \ldots, (a_t,b_t)$ be the normal representation of the set $Par(A)$. Then, for any $x \in \mathbb{R}$, $F_A(x) = F(x)$ where

$$F(x) = \begin{cases} 
\text{undefined}, & x < a_1 \\
b_1, & a_1 \leq x < a_2 \\
\vdots & \vdots \\
b_{t-1}, & a_{t-1} \leq x < a_t \\
b_t, & a_t \leq x 
\end{cases}$$

**Proof.** One can show that $a_1 = \min\{a : (a,b) \in A\}$. Therefore the value $F_A(x)$ is undefined if $x < a_1$. Let $x \geq a_1$. Then both values $F(x)$ and $F_A(x)$ are defined. It is easy to check that $F(x) = F_{par(A)}(x)$. Since $Par(A) \subseteq A$, we have $F_A(x) \leq F(x)$.

By Lemma 5 for any point $(a,b) \in A$, there is a point $(a_i,b_i) \in Par(A)$ such that $(a_i,b_i) \leq (a,b)$. Therefore $F(x) \leq F_A(x)$ and $F_A(x) = F(x)$.

**Remark 4.** We can consider not only function $F_A$ but also function $F_{rev(A)} : \mathbb{R} \to \mathbb{R}$ defined in the following way:

$$F_{rev(A)}(x) = \min\{a : (b,a) \in rev(A), b \leq x\} = \min\{a : (a,b) \in A, b \leq x\}.$$
From Remark 3 and Lemma 11 it follows that

\[ F_{\text{rev}(A)}(x) = \begin{cases} 
\text{undefined}, & x < b_t \\
a_t, & b_t \leq x < b_{t-1} \\
... & ... \\
a_2, & b_2 \leq x < b_1 \\
a_1, & b_1 \leq x 
\end{cases} \]
Chapter 5

Decision and Inhibitory Trees

In this chapter, we consider various types of decision and inhibitory trees. We discuss the notion of a cost function for trees, the notion of decision tree uncertainty, and the notion of inhibitory tree completeness. We also consider an algorithm for counting the number of trees represented by the directed acyclic graph.

We extend here some definitions from the book [15] to the case of the decision tables with many-valued decisions.

5.1 Different Kinds of Decision Trees

In this section, we discuss main notions connected with decision trees.

5.1.1 Decision Trees for $T$

Let $T$ be a decision table with many-valued decisions containing $n$ 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 $\omega$, and, for each nonterminal node, edges starting from this node are labeled with pairwise different numbers from $\omega$.

Let $\Gamma$ be a decision tree over $T$ and $v$ be a node of $\Gamma$. We denote by $\Gamma(v)$ the subtree of $\Gamma$ 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 $\Gamma$ then $T(v) = T$. Let $v$ be not the root of $\Gamma$ and
\(v_1, e_1, \ldots, v_m, e_m, v_{m+1} = v\) be the directed path from the root of \(\Gamma\) 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 \(T(v) = T(f_{i_1}, a_1) \ldots (f_{i_m}, a_m)\).

A decision tree \(\Gamma\) over \(T\) is called a decision tree for \(T\) if, for any node \(v\) of \(\Gamma\),

- 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 \(T\).

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 \(\omega\), 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

\[
DT(T, f_i) = \{tree(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t) : \Gamma_j \in DT(T(f_i, a_j)), j = 1, \ldots, t\}.
\]

**Proposition 13.** Let \(T\) be a decision table. Then \(DT(T) = \{tree(mcd(T))\}\) if \(T\) is degenerate, and \(DT(T) = \{tree(mcd(T))\} \cup \bigcup_{f_i \in E(T)} DT(T, f_i)\) if \(T\) is nondegenerate.

**Proof.** Let \(T\) be a degenerate decision table. From the definition of a decision tree for \(T\) it follows that \(tree(mcd(T))\) is the only decision tree for \(T\).
Let $T$ be a nondegenerate decision table. It is clear that $\text{tree}(mcd(T))$ is the only decision tree for $T$ with one node.

Let $\Gamma \in DT(T)$ and $\Gamma$ have more than one node. Then, by definition of the set $DT(T)$, $\Gamma = \text{tree}(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t)$ where $f_i \in E(T)$ and $\{a_1, \ldots, a_t\} = E(T, f_i)$. Using the fact that $\Gamma \in DT(T)$ it is not difficult to show that $\Gamma_j \in DT(T(f_i, a_j))$ for $j = 1, \ldots, t$. From here it follows that $\Gamma \in DT(T, f_i)$ for $f_i \in E(T)$. Therefore $DT(T) \subseteq \{\text{tree}(mcd(T))\} \cup \bigcup_{f_i \in E(T)} DT(T, f_i)$.

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

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

where $\{a_1, \ldots, a_t\} = E(T, f_i)$ and $\Gamma_j \in DT(T(f_i, a_j))$ for $j = 1, \ldots, t$. Using these facts it is not difficult to show that $\Gamma \in DT(T)$. Therefore

$$\{\text{tree}(mcd(T))\} \cup \bigcup_{f_i \in E(T)} DT(T, f_i) \subseteq DT(T) .$$


For each node $\Theta$ of the directed acyclic graph $\Delta(T)$, we define the set

$$Tree^*(\Delta(T), \Theta)$$

of decision trees in the following way. If $\Theta$ is a terminal node of $\Delta(T)$ (in this case $\Theta$ is degenerate) then $Tree^*(\Delta(T), \Theta) = \{\text{tree}(mcd(\Theta))\}$. Let $\Theta$ be a nonterminal node of $\Delta(T)$ (in this case $\Theta$ is nondegenerate), $f_i \in E(T)$, and $E(\Theta, f_i) = \{a_1, \ldots, a_t\}$. We denote by $Tree^*(\Delta(T), \Theta, f_i)$ the set of decision trees $\{\text{tree}(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t) :$
\[ j \in Tree^*(\Delta(T), \Theta(f_i, a_j)), j = 1, \ldots, t \]. Then

\[
Tree^*(\Delta(T), \Theta) = \{tree(mcd(\Theta))\} \cup \bigcup_{f_i \in E(\Theta)} Tree^*(\Delta(T), \Theta, f_i).
\]

**Proposition 14.** For any decision table \( T \) and any node \( \Theta \) of the graph \( \Delta(T) \), the equality \( Tree^*(\Delta(T), \Theta) = DT(\Theta) \) holds.

**Proof.** We prove this statement by induction on nodes of \( \Delta(T) \). Let \( \Theta \) be a terminal node of \( \Delta(T) \). Then \( Tree^*(\Delta(T), \Theta) = \{tree(mcd(\Theta))\} = DT(\Theta) \). Let now \( \Theta \) be a nonterminal node of \( \Delta(T) \), and let us assume that \( Tree^*(\Delta(T), \Theta(f_i, a_j)) = DT(\Theta(f_i, a_j)) \) for any \( f_i \in E(\Theta) \) and \( a_j \in E(\Theta, f_i) \). Then, for any \( f_i \in E(\Theta) \), we have \( Tree^*(\Delta(T), \Theta, f_i) = DT(\Theta, f_i) \). Using Proposition 13, we obtain

\[
Tree^*(\Delta(T), \Theta) = DT(\Theta).
\]

\[ \square \]

### 5.1.2 \((U, \alpha)\)-Decision Trees for \( T \)

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

- 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 \((U, \alpha)\)-decision trees for \( T \). It is easy to show that \( DT_{U,\alpha}(T) \subseteq DT(T) \). Let \( f_i \in E(T) \) and \( E(T, f_i) = \{a_1, \ldots, a_t\} \). We denote

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

97
Proposition 15. Let \( U \) be an uncertainty measure, \( \alpha \in \mathbb{R}_+ \), and \( T \) be a decision table. Then \( DT_{U,\alpha}(T) = \{ \text{tree}(mcd(T)) \} \) if \( U(T) \leq \alpha \), and \( DT_{U,\alpha}(T) = \bigcup_{f_i \in E(T)} DT_{U,\alpha}(T, f_i) \) if \( U(T) > \alpha \).

Proof. Let \( U(T) \leq \alpha \). Then \( \text{tree}(mcd(T)) \) is the only \((U, \alpha)\)-decision tree for \( T \). Let \( U(T) > \alpha \) and \( \Gamma \in DT_{U,\alpha}(T) \). Then, by definition, \( \Gamma = \text{tree}(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t) \) 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 \( \Gamma = \text{tree}(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t) \) 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 \( \bigcup_{f_i \in E(T)} DT_{U,\alpha}(T, f_i) \subseteq DT_{U,\alpha}(T) \). \( \Box \)

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

\[
\text{Tree}(G, \Theta) = \bigcup_{f_i \in E_G(\Theta)} \text{Tree}(G, \Theta, f_i).
\]

Proposition 16. Let \( U \) be an uncertainty measure, \( \alpha \in \mathbb{R}_+ \), and \( T \) be a decision table. Then, for any node \( \Theta \) of the graph \( \Delta_{U,\alpha}(T) \), the following equality holds: \( \text{Tree}(\Delta_{U,\alpha}(T), \Theta) = DT_{U,\alpha}(\Theta) \).

Proof. We prove this statement by induction on nodes of \( \Delta_{U,\alpha}(T) \). Let \( \Theta \) be a terminal
node of $\Delta_{U,\alpha}(T)$. Then $\text{Tree}(\Delta_{U,\alpha}(T), \Theta) = \{\text{tree}(\text{med}(\Theta))\} = DT_{U,\alpha}(\Theta)$. Let now $\Theta$ be a nonterminal node of $\Delta_{U,\alpha}(T)$, and let us assume that, for any $f_i \in E(\Theta)$ and $\alpha_j \in E(\Theta, f_i)$, $\text{Tree}(\Delta_{U,\alpha}(T), \Theta(f_i, a_j)) = DT_{U,\alpha}(\Theta(f_i, a_j))$. Then, for any $f_i \in E(\Theta)$, we have $\text{Tree}(\Delta_{U,\alpha}(T), \Theta, f_i) = DT_{U,\alpha}(\Theta, f_i)$. Using Proposition 15, we obtain $\text{Tree}(\Delta_{U,\alpha}(T), \Theta) = DT_{U,\alpha}(\Theta)$. □

5.1.3 Cardinality of the Set $\text{Tree}(G, \Theta)$

Let $U$ be an uncertainty measure, $\alpha \in \mathbb{R}_+$, $T$ be a decision table with $n$ attributes $f_1, \ldots, f_n$, and $G$ be a bundle-preserving subgraph of the graph $\Delta_{U,\alpha}(T)$. We describe now an algorithm which counts, for each node $\Theta$ of the graph $G$, the cardinality $C(\Theta)$ of the set $\text{Tree}(G, \Theta)$, and returns the number $C(T) = |\text{Tree}(G, T)|$.

Algorithm $A_4$

Input: A bundle-preserving subgraph $G$ of the graph $\Delta_{U,\alpha}(T)$ for some decision table $T$, uncertainty measure $U$, and number $\alpha \in \mathbb{R}_+$.

Output: The number $|\text{Tree}(G, T)|$.

1. If all nodes of the graph $G$ are processed then return the number $C(T)$ and finish the work of the algorithm. Otherwise, choose a node $\Theta$ of 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 each $f_i \in E_G(T)$ and $a_j \in E(\Theta, f_i)$, the node $\Theta(f_i, a_j)$ is processed.

2. If $\Theta$ is a terminal node then set $C(\Theta) = 1$, mark the node $\Theta$ as processed, and proceed to step 1.

3. If $\Theta$ is a nonterminal node then set

$$C(\Theta) = \sum_{f_i \in E_G(\Theta)} \prod_{a_j \in E(\Theta, f_i)} C(\Theta(f_i, a_j)) ,$$

99
mark the node $\Theta$ as processed, and proceed to step 1.

**Proposition 17.** Let $U$ be an uncertainty measure, $\alpha \in \mathbb{R}_+$, $T$ be a decision table with $n$ attributes $f_1, \ldots, f_n$, and $G$ be a bundle-preserving subgraph of the graph $\Delta_{U,\alpha}(T)$. Then the algorithm $\mathcal{A}_4$ returns the number $|\text{Tree}(G,T)|$ and makes at most

$$nL(G)\text{range}(T)$$

operations of addition and multiplication.

**Proof.** We prove by induction on the nodes of $G$ that $C(\Theta) = |\text{Tree}(G, \Theta)|$ for each node $\Theta$ of $G$. Let $\Theta$ be a terminal node of $G$. Then $\text{Tree}(G, \Theta) = \{\text{tree}(\text{med}(\Theta))\}$ and $|\text{Tree}(G, \Theta)| = 1$. Therefore the considered statement holds for $\Theta$. Let now $\Theta$ be a nonterminal node of $G$ such that the considered statement holds for its children. By definition, $\text{Tree}(G, \Theta) = \bigcup_{f_i \in E_G(\Theta)} \text{Tree}(G, \Theta, f_i)$, and, for $f_i \in E_G(\Theta)$,

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

where $\{a_1, \ldots, a_t\} = E(\Theta, f_i)$. One can show that, for any $f_i \in E_G(\Theta)$,

$$|\text{Tree}(G, \Theta, f_i)| = \prod_{a_j \in E(\Theta, f_i)} |\text{Tree}(G, \Theta(f_i, a_j))| ,$$

and $|\text{Tree}(G, \Theta)| = \sum_{f_i \in E_G(\Theta)} |\text{Tree}(G, \Theta, f_i)|$. By the induction hypothesis,

$$C(\Theta(f_i, a_j)) = |\text{Tree}(G, \Theta(f_i, a_j))|$$

for any $f_i \in E_G(\Theta)$ and $a_j \in E(\Theta, f_i)$. Therefore $C(\Theta) = |\text{Tree}(G, \Theta)|$. Hence, the considered statement holds. From here it follows that $C(T) = |\text{Tree}(G, T)|$, and the Algorithm $\mathcal{A}_4$ returns the cardinality of the set $\text{Tree}(G,T)$. 

100
It is easy to see that the considered algorithm makes at most \( nL(G)\text{range}(T) \) operations of addition and multiplication where \( L(G) \) is the number of nodes in the graph \( G \) and \( \text{range}(T) = \max\{|E(T, f_i)| : i = 1, \ldots, n\} \).

**Proposition 18.** Let \( U \) be a restricted information system. Then the algorithm \( A_4 \) has polynomial time complexity for decision tables from \( \mathcal{T}(U) \) depending on the number of conditional attributes in these tables.

**Proof.** All operations made by the algorithm \( A_4 \) are basic numerical operations. From Proposition 17 it follows that the number of these operations is bounded from above by a polynomial depending on the size of input table \( T \) and on the number of separable subtables of \( T \).

According to Proposition 9 the algorithm \( A_4 \) has polynomial time complexity for decision tables from \( \mathcal{T}(U) \) depending on the number of conditional attributes in these tables.

5.1.4 \( U^{\text{max}} \)-Decision Trees for \( T \)

Let \( U \) be an uncertainty measure, \( T \) be a decision table, and \( \Gamma \) be a decision tree for \( T \). We denote by \( V_t(\Gamma) \) the set of terminal nodes of \( \Gamma \), and by \( V_n(\Gamma) \) we denote the set of nonterminal nodes of \( \Gamma \). We denote by \( U^{\text{max}}(T, \Gamma) \) the number \( \max\{U(T(v)) : v \in V_t(\Gamma)\} \) which will be interpreted as a kind of uncertainty of \( \Gamma \).

Let \( V_n(\Gamma) \neq \emptyset \). For a nonterminal node \( v \) of \( \Gamma \), we denote by \( \Gamma^v \) a decision tree for \( T \) which is obtained from \( \Gamma \) by removal all nodes and edges of the subtree \( \Gamma(v) \) with the exception of \( v \). Instead of an attribute we attach to \( v \) the number \( mcd(T(v)) \).

The operation of transformation of \( \Gamma \) into \( \Gamma^v \) will be called pruning of the subtree \( \Gamma(v) \). Let \( v_1, \ldots, v_m \) be nonterminal nodes of \( \Gamma \) such that, for any \( i, j \in \{1, \ldots, m\} \), \( i \neq j \), the node \( v_i \) does not belong to the subtree \( \Gamma(v_j) \). We denote by \( \Gamma^{v_1, \ldots, v_m} \) the tree obtained from \( \Gamma \) by sequential pruning of subtrees \( \Gamma(v_1), \ldots, \Gamma(v_m) \).
A decision tree $\Gamma$ for $T$ is called a $U_{\text{max}}$-decision tree for $T$ if either $V_n(\Gamma) = \emptyset$ or $U_{\text{max}}(T, \Gamma^v) > U_{\text{max}}(T, \Gamma)$ for any node $v \in V_n(\Gamma)$. We denote by $DT_{U_{\text{max}}}(T)$ the set of $U_{\text{max}}$-decision trees for $T$. These trees can be considered as irredundant decision trees for $T$ relative to the uncertainty $U_{\text{max}}$ of decision trees. According to the definition, $DT_{U_{\text{max}}}(T) \subseteq DT(T)$.

**Proposition 19.** Let $U$ be an uncertainty measure and $T$ be a decision table. Then $DT_{U_{\text{max}}}(T) = \bigcup_{\alpha \in \mathbb{R}_+} DT_{U, \alpha}(T)$.

**Proof.** From Propositions 13 and 15 it follows that $\text{tree}(\text{mcd}(T))$ is the only decision tree with one node in $DT_{U_{\text{max}}}(T)$ and the only decision tree with one node in $\bigcup_{\alpha \in \mathbb{R}_+} DT_{U, \alpha}(T)$. We now consider decision trees $\Gamma$ from $DT(T)$ that contain more than one node.

Let $\Gamma \in DT_{U_{\text{max}}}(T)$ and $\alpha = U_{\text{max}}(T, \Gamma)$. Let $v \in V_n(\Gamma)$. Since $U_{\text{max}}(T, \Gamma^v) > \alpha$, we have $U(T(v)) > \alpha$. So, for each terminal node $v$ of $\Gamma$, $U(T(v)) \leq \alpha$ and, for each nonterminal node $v$ of $\Gamma$, $U(T(v)) > \alpha$. Taking into account that $\Gamma \in DT(T)$, we obtain $\Gamma \in DT_{U, \alpha}(T)$.

Let $\Gamma \in DT_{U, \alpha}(T)$ for some $\alpha \in \mathbb{R}_+$. Then $U_{\text{max}}(T, \Gamma) \leq \alpha$ and, for each node $v \in V_n(\Gamma)$, the inequality $U(T(v)) > \alpha$ holds. Therefore, for each node $v \in V_n(\Gamma)$, we have $U_{\text{max}}(T, \Gamma^v) > U_{\text{max}}(T, \Gamma)$, and $\Gamma \in DT_{U, \alpha}(T)$. \qed

**Proposition 20.** Let $U$ be an uncertainty measure, $T$ be a decision table, and $\Gamma \in DT(T) \setminus DT_{U_{\text{max}}}(T)$. Then there exist nodes $v_1, \ldots, v_m \in V_n(\Gamma)$ such that, for any $i, j \in \{1, \ldots, m\}$, $i \neq j$, the node $v_i$ does not belong to the subtree $\Gamma(v_j)$, $\Gamma^{v_1 \ldots v_m} \in DT_{U_{\text{max}}}(T)$ and $U_{\text{max}}(T, \Gamma^{v_1 \ldots v_m}) \leq U_{\text{max}}(T, \Gamma)$.

**Proof.** Let $\alpha = U_{\text{max}}(T, \Gamma)$ and $v_1, \ldots, v_m$ be all nonterminal nodes $v$ in $\Gamma$ such that $U(T(v)) \leq \alpha$ and there is no a nonterminal node $v'$ such that $v \neq v'$, $U(T(v')) \leq \alpha$ and $v$ belongs to $\Gamma(v')$. One can show that, for any $i, j \in \{1, \ldots, m\}$, $i \neq j$, the node
v_i does not belong to the subtree Γ(v_j), Γ^{v_1...v_m} ∈ DT_{U}^{\text{max}}(T) and U^{\text{max}}(T, Γ^{v_1...v_m}) ≤ α.

5.1.5 $U^{\text{sum}}$-Decision Trees

Let $U$ be an uncertainty measure, $T$ be a decision table, and $Γ$ be a decision tree for $T$. We denote by $U^{\text{sum}}(T, Γ)$ the number $\sum_{v \in V_t(Γ)} U(T(v))$ which will be interpreted as a kind of uncertainty of $Γ$.

A decision tree $Γ$ for $T$ is called a $U^{\text{sum}}$-decision tree for $T$ if either $V_n(Γ) = \emptyset$ or $U^{\text{sum}}(T, Γ^v) > U^{\text{sum}}(T, Γ)$ for any node $v \in V_n(Γ)$. We denote by $DT_{U}^{\text{sum}}(T)$ the set of $U^{\text{sum}}$-decision trees for $T$. These trees can be considered as irredundant decision trees for $T$ relative to the uncertainty $U^{\text{sum}}$ of decision trees. According to the definition, $DT_{U}^{\text{sum}}(T) \subseteq DT(T)$.

**Proposition 21.** Let $U$ be an uncertainty measure, $T$ be a decision table, and $Γ \in DT(T) \setminus DT_{U}^{\text{sum}}(T)$. Then there exist nodes $v_1, \ldots, v_m \in V_n(Γ)$ such that, for any $i, j \in \{1, \ldots, m\}$, $i \neq j$, the node $v_i$ does not belong to the subtree $Γ(v_j)$, $Γ^{v_1...v_m} ∈ DT_{U}^{\text{sum}}(T)$ and $U^{\text{sum}}(T, Γ^{v_1...v_m}) ≤ U^{\text{sum}}(T, Γ)$.

**Proof.** Let $v_1, \ldots, v_m$ be all nonterminal nodes $v$ in $Γ$ such that

$$U(T(v)) ≤ \sum_{u \in V_t(Γ(v))} U(T(u))$$

and there is no a nonterminal node $v'$ such that $U(T(v')) ≤ \sum_{u \in V_t(Γ(v'))} U(T(u))$, $v \neq v'$, and $v$ belongs to $Γ(v')$. It is clear that, for any $i, j \in \{1, \ldots, m\}$, $i \neq j$, the node $v_i$ does not belong to the subtree $Γ(v_j)$ and, for any nonterminal node $v$ of the tree $Γ' = Γ^{v_1...v_m}$, $\sum_{u \in V_t(Γ'(v))} U(T(u)) ≤ \sum_{u \in V_t(Γ(v))} U(T(u))$. Using this fact, one can show that $Γ^{v_1...v_m} ∈ DT_{U}^{\text{sum}}(T)$ and $U^{\text{sum}}(T, Γ^{v_1...v_m}) ≤ U^{\text{sum}}(T, Γ)$.
5.1.6 Cost Functions for Decision Trees

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

Let $f$ be a function from $\mathbb{R}_+^2$ to $\mathbb{R}_+$. 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 $f(x_1, \ldots, x_n) = f(f(x_1, \ldots, x_{n-1}), x_n)$.

**Proposition 22.** Let $f$ be an increasing function from $\mathbb{R}_+^2$ to $\mathbb{R}_+$. Then, for any natural $n$, the function $f(x_1, \ldots, x_n)$ is increasing.

**Proof.** We prove the considered statement by induction on $n$. If $n = 1$ then, evidently, the function $f(x_1) = x_1$ is increasing. We know that the function $f(x_1, x_2)$ is increasing. Let us assume that, for some $n \geq 2$, the function $f(x_1, \ldots, x_n)$ is increasing. We now show that the function $f(x_1, \ldots, x_{n+1})$ is increasing. Let $x = (x_1, \ldots, x_{n+1}), y = (y_1, \ldots, y_{n+1}) \in \mathbb{R}^{n+1}_+$ and $x \leq y$. By induction hypothesis,

$$f(x_1, \ldots, x_n) \leq f(y_1, \ldots, y_n).$$

Since $x_{n+1} \leq y_{n+1}$, we obtain $f(f(x_1, \ldots, x_n), x_{n+1}) \leq f(f(y_1, \ldots, y_n), y_{n+1})$. Therefore the function $f(x_1, \ldots, x_{n+1})$ is increasing. \qed

**Proposition 23.** Let $f$ be a strictly increasing function from $\mathbb{R}_+^2$ to $\mathbb{R}_+$. Then, for any natural $n$, the function $f(x_1, \ldots, x_n)$ is strictly increasing.

**Proof.** We prove the considered statement by induction on $n$. If $n = 1$ then the function $f(x_1) = x_1$ is strictly increasing. We know that the function $f(x_1, x_2)$ is
strictly increasing. Let us assume that, for some \( n \geq 2 \), the function \( f(x_1, \ldots, x_n) \) is strictly increasing. We now show that the function \( f(x_1, \ldots, x_{n+1}) \) is strictly increasing. Let \( x = (x_1, \ldots, x_{n+1}), y = (y_1, \ldots, y_{n+1}) \in \mathbb{R}_{+}^{n+1}, x \leq y \) and \( x \neq y \). It is clear that \( x' = (x_1, \ldots, x_n) \leq y' = (y_1, \ldots, y_n) \). If \( x' \neq y' \) then, by induction hypothesis, \( f(x_1, \ldots, x_n) < f(y_1, \ldots, y_n) \) and, since \( x_{n+1} \leq y_{n+1} \), \( f(f(x_1, \ldots, x_n), x_{n+1}) ) < f(f(y_1, \ldots, y_n), y_{n+1}) \). Let now \( x' = y' \). Then \( x_{n+1} < y_{n+1} \) and

\[
f(f(x_1, \ldots, x_n), x_{n+1}) < f(f(y_1, \ldots, y_n), y_{n+1}) .
\]

Therefore the function \( f(x_1, \ldots, x_{n+1}) \) is strictly increasing.

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

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

- If \( \Gamma = \text{tree}(\text{mcd}(T)) \) then \( \psi(T, \Gamma) = \psi^0(T) \).

- If \( \Gamma = \text{tree}(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t) \) 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 \( \psi \) is called increasing if \( F \) is an increasing function. The cost function \( \psi \) is called strictly increasing if \( F \) is a strictly increasing function. The cost function \( \psi \) is called integral if, for any \( T \in \mathcal{T} \) and any \( x, y \in \omega = \{0, 1, 2, \ldots\} \), \( \psi^0(T) \in \omega \), \( F(x, y) \in \omega \), and \( w(T) \in \omega \).

We now consider examples of cost functions for decision trees:

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

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

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

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

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

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

Note that the functions \( U_{\text{sum}}(T, \Gamma) \) and \( U_{\text{max}}(T, \Gamma) \) where \( U \) is an uncertainty measure can be represented in the form of cost functions for decision trees:

- For \( U_{\text{max}}(T, \Gamma) \), \( \psi^0(T) = U(T), F(x, y) = \max(x, y), \) and \( w(T) = 0 \).

- For \( U_{\text{sum}}(T, \Gamma) \), \( \psi^0(T) = U(T), F(x, y) = x + y, \) and \( w(T) = 0 \).
We will say that a cost function \( \psi \) is \textit{bounded} if, for any decision table \( T \) and any decision tree \( \Gamma \) for \( T \), \( \psi(T, \text{tree}(\text{mcd}(T))) \leq \psi(T, \Gamma) \).

**Lemma 12.** The cost functions \( h, \text{tpl}, L, L_t, \text{and} L_n \) are bounded cost functions for decision trees.

**Proof.** If \( T \) is a degenerate decision table then there is only one decision tree for \( T \) which is equal to \( \text{tree}(\text{mcd}(T)) \). Let \( T \) be a nondegenerate decision table, \( \Gamma_0 = \text{tree}(\text{mcd}(T)) \) and \( \Gamma \) be a decision tree for \( T \) such that \( \Gamma \neq \Gamma_0 \). Since \( \Gamma \) is a decision tree for \( T \) different from \( \Gamma_0 \), the root of \( \Gamma \) is a nonterminal node, and there are at least two edges starting from the root. From here it follows that \( h(\Gamma) \geq 1 \), \( \text{tpl}(T, \Gamma) \geq 2 \), \( L(\Gamma) \geq 3 \), \( L_n(\Gamma) \geq 1 \), and \( L_t(\Gamma) \geq 2 \). Since \( h(\Gamma_0) = 0 \), \( \text{tpl}(T, \Gamma_0) = 0 \), \( L(\Gamma_0) = 1 \), \( L_n(\Gamma_0) = 0 \), and \( L_t(\Gamma_0) = 1 \), we have \( h \), \( \text{tpl} \), \( L \), \( L_t \), and \( L_n \) are bounded cost functions.

Let \( \psi \) be an integral cost function and \( T \) be a decision table. We denote

\[
\text{ub}(\psi, T) = \max\{\psi(\Theta, \Gamma) : \Theta \in SEP(T), \Gamma \in DT(\Theta)\}.
\]

It is clear that, for any separable subtable \( \Theta \) of \( T \) and for any decision tree \( \Gamma \) for \( \Theta \), \( \psi(\Theta, \Gamma) \in \{0, \ldots, \text{ub}(\psi, T)\} \).

**Lemma 13.** Let \( T \) be a decision table with \( n \) conditional attributes. Then \( \text{ub}(h, T) \leq n \), \( \text{ub}(\text{tpl}, T) \leq nN(T) \), \( \text{ub}(L, T) \leq 2N(T) \), \( \text{ub}(L_n, T) \leq N(T) \), and \( \text{ub}(L_t, T) \leq N(T) \).

**Proof.** Let \( \Theta \) be a separable subtable of the table \( T \) and \( \Gamma \) be a decision tree for \( \Theta \). From the definition of a decision tree for a decision table it follows that, for any node \( v \) of \( \Gamma \), the subtable \( \Theta_{\Gamma}(v) \) is nonempty, for any nonterminal node \( v \) of \( \Gamma \), at least two edges start in \( v \), and, in any path from the root to a terminal node of \( \Gamma \), nonterminal nodes are labeled with pairwise different attributes. Therefore
\[ h(\Theta, \Gamma) \leq n, \ L(\Theta, \Gamma) \leq N(\Theta) \leq N(T), \ tpl(\Theta, \Gamma) \leq nN(\Theta) \leq nN(T), \ L_n(\Theta, \Gamma) \leq N(\Theta) \leq N(T), \ \text{and} \ L(\Theta, \Gamma) \leq 2N(\Theta) \leq 2N(T). \]  

**Proposition 24.** Let \( U \) be an uncertainty measure, \( \psi \) be bounded and increasing cost function for decision trees, \( T \) be a decision table, and \( \Gamma \in DT(T) \setminus DT^{\text{max}}_U(T) \). Then there is a decision tree \( \Gamma' \in DT^{\text{max}}_U(T) \) such that \( U^{\text{max}}(T, \Gamma') \leq U^{\text{max}}(T, \Gamma) \) and \( \psi(T, \Gamma') \leq \psi(T, \Gamma) \).

**Proof.** By Proposition 20, there are nodes \( v_1, \ldots, v_m \in V_n(\Gamma) \) such that the decision tree \( \Gamma' = \Gamma^{v_1 \ldots v_m} \) belongs to \( DT^{\text{max}}_U(T) \) and \( U^{\text{max}}(T, \Gamma') \leq U^{\text{max}}(T, \Gamma) \). Since \( \psi \) is bounded and increasing, \( \psi(T, \Gamma') \leq \psi(T, \Gamma) \).

**Proposition 25.** Let \( U \) be an uncertainty measure, \( \psi \) be bounded and increasing cost function for decision trees, \( T \) be a decision table, and \( \Gamma \in DT(T) \setminus DT^{\text{sum}}_U(T) \). Then there is a decision tree \( \Gamma' \in DT^{\text{sum}}_U(T) \) such that \( U^{\text{sum}}(T, \Gamma') \leq U^{\text{sum}}(T, \Gamma) \) and \( \psi(T, \Gamma') \leq \psi(T, \Gamma) \).

**Proof.** By Proposition 21, there are nodes \( v_1, \ldots, v_m \in V_n(\Gamma) \) such that the decision tree \( \Gamma' = \Gamma^{v_1 \ldots v_m} \) belongs to \( DT^{\text{sum}}_U(T) \) and \( U^{\text{sum}}(T, \Gamma') \leq U^{\text{sum}}(T, \Gamma) \). Since \( \psi \) is bounded and increasing, \( \psi(T, \Gamma') \leq \psi(T, \Gamma) \).

### 5.2 Different Kinds of Inhibitory Trees

In this section, we discuss main notions connected with inhibitory trees.

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

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

An inhibitory tree $\Gamma$ over $T$ is called an inhibitory tree for $T$ if, for any node $v$ of $\Gamma$,

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

- If $T(v)$ is not an incomplete table relative to $T$ then either $v$ is a terminal node labeled with $\neq \text{lcd}(T, T(v))$, 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 $\text{IT}(T)$ the set of inhibitory trees for $T$.

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

We now consider examples of cost functions for inhibitory trees:

- Depth $h(T, \Gamma) = h(\Gamma)$ of an inhibitory tree $\Gamma$ for a decision table $T$ is the maximum length of a path in $\Gamma$ from the root to a terminal node.

- Total path length $\text{tpl}(T, \Gamma)$ of an inhibitory tree $\Gamma$ for a decision table $T$ is equal to $\sum_{r \in \text{Row}(T)} l_{\Gamma}(r)$ where $\text{Row}(T)$ is the set of rows of $T$, and $l_{\Gamma}(r)$ is the length of a path in $\Gamma$ from the root to a terminal node $v$ such that the row $r$ belongs to $T_{\Gamma}(v)$. Let $T$ be a nonempty decision table. The value $\text{ tpl}(T, \Gamma)/N(T)$ is the average depth $h_{\text{avg}}(T, \Gamma)$ of an inhibitory tree $\Gamma$ for a decision table $T$.

- Number of nodes $L(T, \Gamma) = L(\Gamma)$ of an inhibitory tree $\Gamma$ for a decision table $T$. 

109
• Number of nonterminal nodes $L_n(T, \Gamma) = L_n(\Gamma)$ of an inhibitory tree $\Gamma$ for a decision table $T$.

• Number of terminal nodes $L_t(T, \Gamma) = L_t(\Gamma)$ of an inhibitory tree $\Gamma$ for a decision table $T$.

Let $W$ be a completeness measure and $\alpha \in \mathbb{R}_+$. An inhibitory tree $\Gamma$ over $T$ is called a $(W, \alpha)$-inhibitory tree for $T$ if, for any node $v$ of $\Gamma$,

- If $W(T, T_\Gamma(v)) \leq \alpha$ then $v$ is a terminal node which is labeled with $\neq \text{lcd}(T, T(v))$.
- If $W(T, T_\Gamma(v)) > \alpha$ then $v$ is a nonterminal node labeled with an attribute $f_i \in E(T_\Gamma(v))$, and if $E(T_\Gamma(v), f_i) = \{a_1, \ldots, a_t\}$ then $t$ edges start from the node $v$ which are labeled with $a_1, \ldots, a_t$, respectively.

We denote by $IT_{W,\alpha}(T)$ the set of $(W, \alpha)$-inhibitory trees for $T$. It is easy to show that $IT_{W,\alpha}(T) \subseteq IT(T)$.

Let $W$ be a completeness measure, $T$ be a decision table, and $\Gamma$ be an inhibitory tree for $T$. We denote by $V_t(\Gamma)$ the set of terminal nodes of $\Gamma$, and by $V_n(\Gamma)$ we denote the set of nonterminal nodes of $\Gamma$. We denote by $W^{\text{max}}(T, \Gamma)$ the number $\max\{W(T, T_\Gamma(v)) : v \in V_t(\Gamma)\}$ which will be interpreted as a kind of completeness of $\Gamma$. We denote by $W^{\text{sum}}(T, \Gamma)$ the number $\sum_{v \in V_t(\Gamma)} W(T, T_\Gamma(v))$ which will be also interpreted as a kind of completeness of $\Gamma$.

Let $\Gamma$ be an inhibitory tree for $T$ and $V_n(\Gamma) \neq \emptyset$. For a nonterminal node $v$ of $\Gamma$, we denote by $\Gamma^v$ an inhibitory tree for $T$ which is obtained from $\Gamma$ by removal all nodes and edges of the subtree $\Gamma(v)$ with the exception of $v$. Instead of an attribute we attach to $v$ the expression $\neq \text{lcd}(T, T_\Gamma(v))$. The operation of transformation of $\Gamma$ into $\Gamma^v$ will be called pruning of the subtree $\Gamma(v)$.

An inhibitory tree $\Gamma$ for $T$ is called a $W^{\text{max}}$-inhibitory tree for $T$ if either $V_n(\Gamma) = \emptyset$ or $W^{\text{max}}(T, \Gamma^v) > W^{\text{max}}(T, \Gamma)$ for any node $v \in V_n(\Gamma)$. We denote by $IT^{\text{max}}_W(T)$ the
set of $W^{\text{max}}$-inhibitory trees for $T$. These trees can be considered as irredundant inhibitory trees for $T$ relative to the completeness $W^{\text{max}}$ of inhibitory trees. According to the definition, $IT^{\text{max}}_W(T) \subseteq IT(T)$.

An inhibitory tree $\Gamma$ for $T$ is called a $W^{\text{sum}}$-inhibitory tree for $T$ if either $V_n(\Gamma) = \emptyset$ or $W^{\text{sum}}(T, \Gamma^v) > W^{\text{sum}}(T, \Gamma)$ for any node $v \in V_n(\Gamma)$. We denote by $IT^{\text{sum}}_W(T)$ the set of $W^{\text{sum}}$-inhibitory trees for $T$. These trees can be considered as irredundant inhibitory trees for $T$ relative to the completeness $W^{\text{sum}}$ of inhibitory trees. According to the definition, $IT^{\text{sum}}_W(T) \subseteq IT(T)$.

Let $\Gamma_1$ be an inhibitory tree over $T$, and $\Gamma_2$ be a decision tree over $T^C$. We denote by $\Gamma_1^+$ a decision tree over $T^C$ obtained from $\Gamma_1$ by changing expressions attached to terminal nodes: if a terminal node in $\Gamma_1$ is labeled with $\neq t$, then the corresponding node in $\Gamma_1^+$ is labeled with $t$. We denote by $\Gamma_2^-$ an inhibitory tree over $T$ obtained from $\Gamma_2$ by changing expressions attached to terminal nodes: if a terminal node in $\Gamma_2$ is labeled with $t$ than the corresponding node in $\Gamma_2^-$ is labeled with $\neq t$. It is clear that $(\Gamma_1^+)^- = \Gamma_1$ and $(\Gamma_2^-)^+ = \Gamma_2$. Let $A$ be a set of inhibitory trees over $T$. We denote $A^+ = \{\Gamma^+ : \Gamma \in A\}$. Let $B$ be a set of decision trees over $T^C$. We denote $B^- = \{\Gamma^- : \Gamma \in B\}$.

**Proposition 26.** Let $T$ be a nondegenerate decision table with attributes $f_1, \ldots, f_n$, $\Gamma$ be a decision tree over $T^C$, $W$ be a completeness measure, $U$ be an uncertainty measure, $W$ and $U$ are dual, and $\alpha \in \mathbb{R}_+$. Then

1. $W^H(T, \Gamma^-) = U^H(T^C, \Gamma)$ for any $H \in \{\text{sum}, \text{max}\}$;
2. $\Gamma \in DT(T^C)$ if and only if $\Gamma^- \in IT(T)$;
3. For any $H \in \{\text{sum}, \text{max}\}$, $\Gamma \in DT^H_U(T^C)$ if and only if $\Gamma^- \in IT^H_W(T)$;
4. $\Gamma \in DT_{U,\alpha}(T^C)$ if and only if $\Gamma^- \in IT_{W,\alpha}(T)$;
5. If $\Gamma \in DT(T^C)$ then $\psi(T, \Gamma^-) = \psi(T^C, \Gamma)$ for any $\psi \in \{h, \text{tpl}, L, L_n, L_t\}$.  

111
Proof. Let \( v \) be a node of \( \Gamma \) (we keep the notation \( v \) for corresponding to \( v \) node in \( \Gamma^- \)). Then there exists a word \( \alpha \in Word(T) \) such that \( T_{\Gamma^-}(v) = T\alpha \) and \( T_{\Gamma^C}(v) = T^{C}\alpha \). Since \( W \) and \( U \) are dual, \( W(T, T\alpha) = U(T^C\alpha) \). From here it follows that the statement 1 of the proposition holds. By Lemma 3, \( mcd(T^C\alpha) = lcd(T, T\alpha) \). It is clear that \( E(T\alpha) = E(T^C\alpha) \) and \( E(T\alpha, f_i) = E(T^C\alpha, f_i) \) for any \( f_i \in \{f_1, \ldots, f_n\} \). Since \( W \) and \( U \) are dual, \( W(T, T\alpha) = 0 \) if and only if \( U(T^C\alpha) = 0 \), i.e., \( T\alpha \) is incomplete if and only if \( T^C\alpha \) is degenerate. Using these facts it is not difficult to show that statements 2, 3 and 4 hold. If \( \Gamma \in DT(T) \) and \( \psi \in \{h, tpl, L, L_n, L_t\} \) then it is easy to show that \( \psi(T, \Gamma^-) = \psi(T^C, \Gamma) \), i.e., the statement 5 holds. \( \square \)

Corollary 10. Let \( T \) be a nondegenerate decision table, \( W \) be a completeness measure, \( U \) be an uncertainty measure, \( W \) and \( U \) are dual, and \( \alpha \in \mathbb{R}_+ \). Then

1. \( IT(T) = DT(T^C)^-; \)

2. For any \( H \in \{\text{sum, max}\} \), \( IT^H_W(T) = DT^H_U(T^C)^-; \)

3. \( IT_{W,\alpha}(T) = DT_{U,\alpha}(T^C)^-. \)

Proof. Let \( \Gamma \in DT(T^C) \). Then, by the statement 2 of Proposition \( \square \), \( \Gamma^- \in IT(T) \). Therefore \( DT(T^C)^- \subseteq IT(T) \). Let \( \Gamma \in IT(T) \). Then \( \Gamma^+ \) is a decision tree over \( T^C \) and \( \Gamma = (\Gamma^+)^- \). By the statement 2 of Proposition \( \square \), \( \Gamma^+ \in DT(T^C) \). Therefore \( \Gamma \in DT(T^C)^- \) and \( DT(T^C)^- \supseteq IT(T) \). Hence the statement 1 of the corollary holds. The statements 2 and 3 can be proven similarly. \( \square \)

Let \( G \) be a bundle-preserving subgraph of the graph \( \Delta_{U,\alpha}(T^C) \). The algorithm \( A_4 \) described in Section \( \square \) allows us to find the cardinality of the set \( Tree(G, T^C) \) containing some \((U, \alpha)\)-decision trees for \( T^C \) and, at the same time, the cardinality of the set \( Tree(G, T^C)^- \) containing some \((W, \alpha)\)-inhibitory trees for \( T \). It can be, for example, the set of all \((W, \alpha)\)-inhibitory trees for \( T \) with a minimum number of nodes.

112
We will discuss this, and other interesting cases in Section 6.2 devoted to multi-stage optimization of inhibitory trees.
Chapter 6

Multi-stage Optimization of Decision and Inhibitory Trees

In this chapter, we consider multi-stage optimization of decision (Sect. 6.1) and inhibitory (Sect. 6.2) trees relative to a sequence of cost functions, and an application of this technique: the study of totally optimal (simultaneously optimal relative to some cost functions) decision and inhibitory trees (Sect. 6.3).

Multi-stage optimization approach was created for usual decision tables (decision tables with one-valued decisions) in [57, 58]. The first results in this direction for decision tables with many-valued decisions were obtained in [50]. Totally optimal decision trees for Boolean functions were studied in [9].

Note that the paper [16] contains the most of theoretical and experimental results considered in this chapter.

6.1 Multi-stage Optimization of Decision Trees

In this section, we discuss how to optimize \((U, \alpha)\)-decision trees represented by a bundle-preserving subgraph of the graph \(\Delta_{U, \alpha}(T)\) relative to a cost function for decision trees. We explain also possibilities of multi-stage optimization of decision trees for different cost functions and consider the notion of a totally optimal decision tree relative to some cost functions. This is a decision tree which is optimal simultaneously for each of the considered cost functions.

Let \(\psi\) be an increasing cost function for decision trees given by the triple of functions \(\psi^0, F\) and \(w\), \(U\) be an uncertainty measure, \(\alpha \in \mathbb{R}_+\), \(T\) be a decision table
with \( n \) conditional attributes \( f_1, \ldots, f_n \), and \( G \) be a bundle-preserving subgraph of the graph \( \Delta_{U,\alpha}(T) \).

In Section 5.1.2 for each nonterminal node \( \Theta \) of the graph \( G \), we denoted by \( E_G(\Theta) \) the set of attributes \( f_i \) from \( E(\Theta) \) such that \( f_i \)-bundle of edges starts from \( \Theta \) in \( G \). For each node \( \Theta \) of the graph \( G \), we defined a set \( \text{Tree}(G, \Theta) \) of \((U, \alpha)\)-decision trees for \( \Theta \) in the following way. If \( \Theta \) is a terminal node of \( G \), then

\[
\text{Tree}(G, \Theta) = \{ \text{tree}(\text{mcd}(\Theta)) \}.
\]

Let \( \Theta \) be a nonterminal node of \( G \), \( f_i \in E_G(\Theta) \), and \( E(\Theta, f_i) = \{ a_1, \ldots, a_t \} \). Then

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

and

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

Let \( \Theta \) be a node of \( G \), \( \Gamma \in \text{Tree}(G, \Theta) \), and \( v \) be a node of \( \Gamma \). In Section 5.1.1 we defined a subtree \( \Theta_{\Gamma}(v) \) of \( \Gamma \) for which \( v \) is the root, and a subtable \( \Theta_{\Gamma}(v) \) of \( \Theta \). If \( v \) is the root of \( \Gamma \) then \( \Theta_{\Gamma}(v) = \Theta \). Let \( v \) not be the root of \( \Gamma \) and \( v_1, e_1, \ldots, v_m, e_m, v_{m+1} = v \) be the directed path from the root of \( \Gamma \) 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 \( \Theta_{\Gamma}(v) = \Theta(f_{i_1}, a_1) \ldots (f_{i_m}, a_m) \). One can show that the decision tree \( \Gamma(v) \) belongs to the set \( \text{Tree}(G, \Theta_{\Gamma}(v)) \).

A decision tree \( \Gamma \) from \( \text{Tree}(G, \Theta) \) is called an \textit{optimal decision tree for } \( \Theta \) \text{ relative to } \psi \text{ and } G \text{ if } \psi(\Theta, \Gamma) = \min \{ \psi(\Theta, \Gamma') : \Gamma' \in \text{Tree}(G, \Theta) \} \).

A decision tree \( \Gamma \) from \( \text{Tree}(G, \Theta) \) is called a \textit{strictly optimal decision tree for } \( \Theta \) \text{ relative to } \psi \text{ and } G \text{ if, for any node } v \text{ of } \Gamma \text{, the decision tree } \Gamma(v) \text{ is an optimal decision tree for } \Theta_{\Gamma}(v) \text{ relative to } \psi \text{ and } G \).

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

**Proposition 27.** Let \( \psi \) be a strictly increasing cost function for decision trees, \( U \) be an uncertainty measure, \( \alpha \in \mathbb{R}_+ \), \( T \) be a decision table, and \( G \) be a bundle-preserving
subgraph of the graph $\Delta_{U,\alpha}(T)$. Then, for any node $\Theta$ of the graph $G$, $\text{Tree}_{\psi}^{\text{opt}}(G, \Theta) = \text{Tree}_{\psi}^{s-\text{opt}}(G, \Theta)$.

**Proof.** It is clear that $\text{Tree}_{\psi}^{s-\text{opt}}(G, \Theta) \subseteq \text{Tree}_{\psi}^{\text{opt}}(G, \Theta)$. Let $\Gamma \in \text{Tree}_{\psi}^{\text{opt}}(G, \Theta)$ and let us assume that $\Gamma \notin \text{Tree}_{\psi}^{s-\text{opt}}(G, \Theta)$. Then there is a node $v$ of $\Gamma$ such that $\Gamma(v) \notin \text{Tree}_{\psi}^{\text{opt}}(G, \Theta_{\Gamma(v)})$. Let $\Gamma_{0} \in \text{Tree}_{\psi}^{\text{opt}}(G, \Theta_{\Gamma(v)})$ and $\Gamma'$ be the decision tree obtained from $\Gamma$ by replacing $\Gamma(v)$ with $\Gamma_{0}$. One can show that $\Gamma' \in \text{Tree}(G, \Theta)$. Since $\psi$ is strictly increasing and $\psi(\Theta_{\Gamma(v)}, \Gamma_{0}) < \psi(\Theta_{\Gamma(v)}, \Gamma(v))$, we have $\psi(\Theta, \Gamma') < \psi(\Theta, \Gamma)$. Therefore $\Gamma \notin \text{Tree}_{\psi}^{\text{opt}}(G, \Theta)$ which is impossible. Thus $\text{Tree}_{\psi}^{\text{opt}}(G, \Theta) \subseteq \text{Tree}_{\psi}^{s-\text{opt}}(G, \Theta)$. \hfill \qed

We describe now an algorithm $A_{5}$ (a procedure of optimization relative to the cost function $\psi$). The algorithm $A_{5}$ attaches to each node $\Theta$ of $G$ the number $c(\Theta) = \min\{\psi(\Theta, \Gamma) : \Gamma \in \text{Tree}(G, \Theta)\}$ and, probably, remove some $f_{i}$-bundles of edges starting from nonterminal nodes of $G$. As a result, we obtain a bundle-preserving subgraph $G^{\psi}$ of the graph $G$. It is clear that $G^{\psi}$ is also a bundle-preserving subgraph of the graph $\Delta_{U,\alpha}(T)$.

**Algorithm $A_{5}$**

**Input:** A bundle-preserving subgraph $G$ of the graph $\Delta_{U,\alpha}(T)$ for some decision table $T$, uncertainty measure $U$, and a number $\alpha \in \mathbb{R}_{+}$, and an increasing cost function $\psi$ for decision trees given by the triple of functions $\psi^{0}$, $F$ and $w$.

**Output:** The bundle-preserving subgraph $G^{\psi}$ of the graph $G$.

1. If all nodes of the graph $G$ are processed then return the obtained graph as $G^{\psi}$ and finish the work of the algorithm. Otherwise, choose a node $\Theta$ of the graph $G$ which is not processed yet and which is either a terminal node of $G$ or a nonterminal node of $G$ for which all children are processed.

2. If $\Theta$ is a terminal node then set $c(\Theta) = \psi^{0}(\Theta)$, mark node $\Theta$ as processed and proceed to step 1.
3. If Θ is a nonterminal node then, for each \( f_i \in E_G(\Theta) \), compute the value
\[
c(\Theta, f_i) = F(c(\Theta(f_i, a_1)), \ldots, c(\Theta(f_i, a_t))) + w(\Theta)
\]
where
\[
\{a_1, \ldots, a_t\} = E(\Theta, f_i)
\]
and set \( c(\Theta) = \min\{c(\Theta, f_i) : f_i \in E_G(\Theta)\} \). Remove all \( f_i \)-bundles of edges starting from \( \Theta \) for which \( c(\Theta) < c(\Theta, f_i) \). Mark the node \( \Theta \) as processed and proceed to step 1.

**Proposition 28.** Let \( G \) be a bundle-preserving subgraph of the graph \( \Delta_{U,\alpha}(T) \) for some decision table \( T \) with \( n \) conditional attributes \( f_1, \ldots, f_n \), uncertainty measure \( U \), and a number \( \alpha \in \mathbb{R}_+ \), and \( \psi \) be an increasing cost function for decision trees given by the triple of functions \( \psi^0, F \) and \( w \). Then, to construct the graph \( G^\psi \), the algorithm \( A_5 \) makes
\[
O(nL(G)\text{range}(T))
\]
for decision tables from \( T(U) \) depending on the number of conditional attributes in these tables.

**Proof.** In each terminal node of the graph \( G \), the algorithm \( A_5 \) computes the value of \( \psi^0 \). In each nonterminal node of \( G \), the algorithm \( A_5 \) computes the value of \( F \) (as a function with two variables) at most \( \text{range}(T)n \) times, where \( \text{range}(T) = \max\{|E(T, f_i)| : i = 1, \ldots, n\} \), and the value of \( w \) at most \( n \) times, makes at most \( n \) additions and at most \( 2n \) comparisons. Therefore the algorithm \( A_5 \) makes
\[
O(nL(G)\text{range}(T))
\]
for decision tables from \( T(U) \) depending on the number of conditional attributes in these tables.

**Proposition 29.** Let \( \psi \in \{h, tpl, L, L_n, L_t\} \) and \( U \) be a restricted information system. Then the algorithm \( A_5 \) has polynomial time complexity for decision tables from \( T(U) \) depending on the number of conditional attributes in these tables.
Proof. Since $\psi \in \{h, t_{pl}, L, L_n, L_t\}$, $\psi^0$ is a constant, $F$ is either $\max(x, y)$ or $x + y$, and $w$ is either a constant or $N(T)$. Therefore the elementary operations used by the algorithm $A_5$ are either basic numerical operations or computations of numerical parameters of decision tables which have polynomial time complexity depending on the size of decision tables. From Proposition 28 it follows that the number of elementary operations is bounded from above by a polynomial depending on the size of input table $T$ and on the number of separable subtables of $T$.

According to Proposition 9, the algorithm $A_5$ has polynomial time complexity for decision tables from $\mathcal{T}(U)$ depending on the number of conditional attributes in these tables. \qed

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

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

Lemma 14. Let $G$ be a bundle-preserving subgraph of the graph $\Delta_{U, \alpha}(T)$ for some decision table $T$ with $n$ conditional attributes, uncertainty measure $U$, and number $\alpha \in \mathbb{R}_+$, and $\psi$ be an increasing cost function for decision trees given by the triple of functions $\psi^0$, $F$ and $w$. Then, for any node $\Theta$ of the graph $G$ and for any attribute $f_i \in E_G(\Theta)$, the algorithm $A_5$ computes values $c(\Theta) = \psi_G(\Theta)$ and $c(\Theta, f_i) = \psi_G(\Theta, f_i)$.

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

Let now $\Theta$ be a nonterminal node of $G$ such that the considered statement holds for each node $\Theta(f_i, a_j)$ with $f_i \in E_G(\Theta)$ and $a_j \in E(\Theta, f_i)$. By definition, $\text{Tree}(G, \Theta) = \bigcup_{f_i \in E_G(\Theta)} \text{Tree}(G, \Theta, f_i)$ and, for each $f_i \in E_G(\Theta)$, $\text{Tree}(G, \Theta, f_i) = \text{Tree}(G, \Theta, f_i)$.
\{\text{tree}(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t) : \Gamma_j \in \text{Tree}(G, \Theta(f_i, a_j)), j = 1, \ldots, t\} \quad \text{where} \\
\{a_1, \ldots, a_t\} = E(\Theta, f_i).

Since \(\psi\) is an increasing cost function,

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

where \(\{a_1, \ldots, a_t\} = E(\Theta, f_i)\). It is clear that \(\psi_G(\Theta) = \min\{\psi_G(\Theta, f_i) : f_i \in E_G(\Theta)\}\).

By the induction hypothesis, \(\psi_G(\Theta(f_i, a_j)) = c(\Theta(f_i, a_j))\) for each \(f_i \in E_G(\Theta)\) and \(a_j \in E(\Theta, f_i)\). Therefore \(c(\Theta, f_i) = \psi_G(\Theta, f_i)\) for each \(f_i \in E_G(\Theta)\), and \(c(\Theta) = \psi_G(\Theta)\). 

**Theorem 26.** Let \(\psi\) be an increasing cost function for decision trees, \(U\) be an uncertainty measure, \(\alpha \in \mathbb{R}_+\), \(T\) be a decision table, and \(G\) be a bundle-preserving subgraph of the graph \(\Delta_{U, \alpha}(T)\). Then, for any node \(\Theta\) of the graph \(G^\psi\), the following equality holds: \(\text{Tree}(G^\psi, \Theta) = \text{Tree}^{s-opt}_{\psi}(G, \Theta)\).

**Proof.** We prove the considered statement by induction on nodes of \(G^\psi\). We use Lemma 14 which shows that, for any node \(\Theta\) of the graph \(G\) and for any \(f_i \in E_G(\Theta)\), \(c(\Theta) = \psi_G(\Theta)\) and \(c(\Theta, f_i) = \psi_G(\Theta, f_i)\).

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

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

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

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

We know that \( E_{G^\psi}(\Theta) = \{f_i : f_i \in E_G(\Theta), \psi_G(\Theta, f_i) = \psi_G(\Theta)\} \). Let \( f_i \in E_{G^\psi}(\Theta) \) and \( \Gamma \in \text{Tree}(G^\psi, \Theta, f_i) \). Then \( \Gamma = \text{tree}(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t) \), where

\[ \{a_1, \ldots, a_t\} = E(\Theta, f_i) \]

and \( \Gamma_j \in \text{Tree}(G^\psi, \Theta(f_i, a_j)) \) for \( j = 1, \ldots, t \). According to the induction hypothesis, \( \text{Tree}(G^\psi, \Theta(f_i, a_j)) = \text{Tree}_{\psi}^{s-opt}(G, \Theta(f_i, a_j)) \) and \( \Gamma_j \in \text{Tree}_{\psi}^{s-opt}(G^\psi, \Theta(f_i, a_j)) \) for \( j = 1, \ldots, t \). In particular, \( \psi(\Theta(f_i, a_j), \Gamma_j) = \psi_G(\Theta(f_i, a_j)) \) for \( j = 1, \ldots, t \). Since \( \psi_G(\Theta, f_i) = \psi_G(\Theta) \) we have \( F(\psi_G(\Theta(f_i, a_1)), \ldots, \psi_G(\Theta(f_i, a_t))) + w(\Theta) = \psi_G(\Theta) \) and \( \psi(\Theta, \Gamma) = \psi_G(\Theta) \). Therefore \( \Gamma \in \text{Tree}_{\psi}^{s-opt}(G, \Theta) \), \( \Gamma \in \text{Tree}_{\psi}^{s-opt}(G, \Theta) \) and \( \text{Tree}(G^\psi, \Theta) \subseteq \text{Tree}_{\psi}^{s-opt}(G, \Theta) \).

Let \( \Gamma \in \text{Tree}_{\psi}^{s-opt}(G, \Theta) \). Since \( \Theta \) is a nonterminal node, \( \Gamma \) can be represented in the form \( \Gamma = \text{tree}(f_i, a_1, \ldots, a_t, \Gamma_1, \ldots, \Gamma_t) \), where \( f_i \in E_G(\Theta) \), \( \{a_1, \ldots, a_t\} = E(\Theta, f_i) \), and \( \Gamma_j \in \text{Tree}_{\psi}^{s-opt}(G, \Theta(f_i, a_j)) \) for \( j = 1, \ldots, t \). Since

\[ \Gamma \in \text{Tree}_{\psi}^{s-opt}(G, \Theta), \]

\( \psi_G(\Theta, f_i) = \psi_G(\Theta) \) and \( f_i \in E_{G^\psi}(T) \). According to the induction hypothesis,

\[ \text{Tree}(G^\psi, \Theta(f_i, a_j)) = \text{Tree}_{\psi}^{s-opt}(G, \Theta(f_i, a_j)) \]

for \( j = 1, \ldots, t \). Therefore \( \Gamma \in \text{Tree}(G^\psi, \Theta, f_i) \subseteq \text{Tree}(G^\psi, \Theta) \). As a result, we have \( \text{Tree}_{\psi}^{s-opt}(G, \Theta) \subseteq \text{Tree}(G^\psi, \Theta) \).

\[ \square \]

**Corollary 11.** Let \( \psi \) be a strictly increasing cost function, \( U \) be an uncertainty measure, \( \alpha \in \mathbb{R}_+ \), \( T \) be a decision table, and \( G \) be a bundle-preserving subgraph of the graph \( \Delta_{U, \alpha}(T) \). Then, for any node \( \Theta \) of the graph \( G^\psi \), \( \text{Tree}(G^\psi, \Theta) = \text{Tree}_{\psi}^{opt}(G, \Theta) \).

This corollary follows immediately from Proposition 27 and Theorem 26.
We can make multi-stage optimization of \((U, \alpha)\)-decision trees for \(T\) relative to a sequence of strictly increasing cost functions \(\psi_1, \psi_2, \ldots\). We begin with the graph \(G = \Delta_{U, \alpha}(T)\) and apply to it the procedure of optimization relative to the cost function \(\psi_1\) (the algorithm \(A_5\)). As a result, we obtain a bundle-preserving subgraph \(G^{\psi_1}\) of the graph \(G\).

By Proposition 16, the set \(\text{Tree}(G, T)\) is equal to the set \(\mathcal{D}T_{U, \alpha}(T)\) of all \((U, \alpha)\)-decision trees for \(T\). Using Corollary 11, we obtain that the set \(\text{Tree}(G^{\psi_1}, T)\) coincides with the set \(\text{Tree}^{\text{opt}}_{\psi_1}(G, T)\) of all decision trees from \(\text{Tree}(G, T)\) which have minimum cost relative to \(\psi_1\) among all trees from the set \(\text{Tree}(G, T)\). Next we apply to \(G^{\psi_1}\) the procedure of optimization relative to the cost function \(\psi_2\). As a result, we obtain a bundle-preserving subgraph \(G^{\psi_1, \psi_2}\) of the graph \(G^{\psi_1}\) (and of the graph \(G = \Delta_{\alpha}(T)\)). By Corollary 11, the set \(\text{Tree}(G^{\psi_1, \psi_2}, T)\) coincides with the set \(\text{Tree}^{\text{opt}}_{\psi_2}(G^{\psi_1}, T)\) of all decision trees from \(\text{Tree}(G^{\psi_1}, T)\) which have minimum cost relative to \(\psi_2\) among all trees from \(\text{Tree}(G^{\psi_1}, T)\), etc.

If one of the cost functions \(\psi_i\) is increasing and not strictly increasing then the set \(\text{Tree}(G^{\psi_1, \ldots, \psi_i}, T)\) coincides with the set \(\text{Tree}^{\text{opt}}_{\psi_i}(G^{\psi_1, \ldots, \psi_{i-1}}, T)\) which is a subset of the set of all decision trees from \(\text{Tree}(G^{\psi_1, \ldots, \psi_{i-1}}, T)\) that have minimum cost relative to \(\psi_i\) among all trees from \(\text{Tree}(G^{\psi_1, \ldots, \psi_{i-1}}, T)\).

For a cost function \(\psi\), we denote \(\psi^{U, \alpha}(T) = \min\{\psi(T, \Gamma) : \Gamma \in \mathcal{D}T_{U, \alpha}(T)\}\), i.e., \(\psi^{U, \alpha}(T)\) is the minimum cost of a \((U, \alpha)\)-decision tree for \(T\) relative to the cost function \(\psi\). Let \(\psi_1, \ldots, \psi_m\) be cost functions and \(m \geq 2\). A \((U, \alpha)\)-decision tree \(\Gamma\) for \(T\) is called a totally optimal \((U, \alpha)\)-decision tree for \(T\) relative to the cost functions \(\psi_1, \ldots, \psi_m\) if \(\psi_1(T, \Gamma) = \psi_1^{U, \alpha}(T), \ldots, \psi_m(T, \Gamma) = \psi_m^{U, \alpha}(T)\), i.e., \(\Gamma\) is optimal relative to \(\psi_1, \ldots, \psi_m\) simultaneously.

Assume that \(\psi_1, \ldots, \psi_{m-1}\) are strictly increasing cost functions and \(\psi_m\) is increasing or strictly increasing. We now describe how to recognize the existence of a \((U, \alpha)\)-decision tree for \(T\) which is a totally optimal \((U, \alpha)\)-decision tree for \(T\) relative
to the cost functions \(\psi_1, \ldots, \psi_m\).

First, we construct the graph \(G = \Delta_{U,\alpha}(T)\) using the algorithm \(A_1\). For \(i = 1, \ldots, m\), we apply to \(G\) the procedure of optimization relative to \(\psi_i\) (the Algorithm \(A_5\)). As a result, we obtain for \(i = 1, \ldots, m\), the graph \(G^{\psi_i}\) and the number \(\psi^{U,\alpha}_i(T)\) attached to the node \(T\) of \(G^{\psi_i}\). Next, we apply to \(G\) sequentially the procedures of optimization relative to the cost functions \(\psi_1, \ldots, \psi_m\). As a result, we obtain graphs \(G^{\psi_1}, G^{\psi_1, \psi_2}, \ldots, G^{\psi_1, \ldots, \psi_m}\) and numbers \(\varphi_1, \varphi_2, \ldots, \varphi_m\) attached to the node \(T\) of these graphs. It is clear that \(\varphi_1 = \psi^{U,\alpha}_1(T)\). For \(i = 2, \ldots, m\), \(\varphi_i = \min\{\psi_i(T, \Gamma) : \Gamma \in \text{Tree}(G^{\psi_1, \ldots, \psi_{i-1}, T})\}\). One can show that a totally optimal \((U, \alpha)\)-decision tree for \(T\) relative to the cost functions \(\psi_1, \ldots, \psi_m\) exists if and only if \(\varphi_i = \psi^{U,\alpha}_i(T)\) for \(i = 1, \ldots, m\).

### 6.2 Multi-stage Optimization of Inhibitory Trees

In this section, we consider possibilities of optimization of inhibitory trees including multi-stage optimization relative to a sequence if cost functions. We also discuss the notion of a totally optimal inhibitory tree.

Let \(T\) be a nondegenerate decision table with \(n\) conditional attributes \(f_1, \ldots, f_n\), \(T^C\) be the decision table complementary to \(T\), \(U\) be an uncertainty measure, \(W\) be a completeness measure, \(U\) is dual to \(W\), and \(\alpha \in \mathbb{R}_+\).

Let \(G\) be a bundle-preserving subgraph of the graph \(\Delta_{U,\alpha}(T^C)\). We correspond to the node \(T\) of \(G\) a set \(\text{Tree}(G, T^C)\) of \((U, \alpha)\)-decision trees for \(T^C\). If \(G = \Delta_{U,\alpha}(T^C)\) then, by Proposition \(16\), the set \(\text{Tree}(G, T^C)\) is equal to the set \(\text{DT}_{U,\alpha}(T^C)\) of all \((U, \alpha)\)-decision trees for \(T\). In general case, \(\text{Tree}(G, T^C) \subseteq \text{DT}_{U,\alpha}(T^C)\).

Let us consider the set \(\text{Tree}(G, T^C)^-\). By Corollary \(10\), \(\text{IT}_{W,\alpha}(T) = \text{DT}_{U,\alpha}(T^C)^-\). Therefore, if \(G = \Delta_{U,\alpha}(T^C)\) then \(\text{Tree}(G, T^C)^- = \text{IT}_{W,\alpha}(T)\). In general case,

\[\text{Tree}(G, T^C)^- \subseteq \text{DT}_{U,\alpha}(T^C)^- = \text{IT}_{W,\alpha}(T)\]
In Section 6.1 the algorithm $A_5$ is considered which, for the graph $G$ and increasing cost function $\psi$ for decision trees, constructs bundle-preserving subgraph $G^\psi$ of the graph $G$.

Let $\psi \in \{tpl, L, L_n, L_t\}$. Then $\psi$ is strictly increasing and, by Corollary 11, the set $\text{Tree}(G^\psi, T^C)$ is equal to the set of all trees from $\text{Tree}(G, T^C)$ which have minimum cost relative to $\psi$ among all decision trees from the set $\text{Tree}(G, T^C)$. From Proposition 26 it follows that, for any $\Gamma \in \text{Tree}(G, T^C)$, $\psi(T^C, \Gamma) = \psi(T, \Gamma^-)$. Therefore, the set $\text{Tree}(G^\psi, T^C)^-$ is equal to the set of all trees from $\text{Tree}(G, T^C)^-$ which have minimum cost relative to $\psi$ among all inhibitory trees from the set $\text{Tree}(G, T^C)^-$.

Let $\psi = h$. Then $\psi$ is increasing and, by Theorem 26, the set $\text{Tree}(G^\psi, T^C)$ is a subset of the set of all trees from $\text{Tree}(G, T^C)$ which have minimum cost relative to $\psi$ among all decision trees from the set $\text{Tree}(G, T^C)$. From here it follows that the set $\text{Tree}(G^\psi, T^C)^-$ is a subset of the set of all trees from $\text{Tree}(G, T^C)^-$ which have minimum cost relative to $\psi$ among all inhibitory trees from the set $\text{Tree}(G, T^C)^-$. More accurate analysis can be done if we consider the notion of a strictly optimal inhibitory tree.

We can make multi-stage optimization of inhibitory trees relative to a sequence of cost functions $\psi_1, \psi_2, \ldots$ from $\{h, tpl, L, L_n, L_t\}$. We begin from the graph $G = \Delta_{U, \alpha}(T^C)$ and apply to it the procedure of optimization relative to the cost function $\psi_1$ (the Algorithm $A_5$). As a result, we obtain a bundle-preserving subgraph $G^{\psi_1}$ of the graph $G$. The set $\text{Tree}(G, T^C)^-$ is equal to the set $\text{IT}_{W, \alpha}(T)$ of all $(W, \alpha)$-inhibitory trees for $T$. If $\psi_1 \neq h$ then the set $\text{Tree}(G^{\psi_1}, T^C)^-$ coincides with the set of all inhibitory trees from $\text{Tree}(G, T^C)^-$ which have minimum cost relative to $\psi_1$ among all trees from the set $\text{Tree}(G, T^C)^-$. Next we apply to $G^{\psi_1}$ the procedure of optimization relative to the cost function $\psi_2$. As a result, we obtain a bundle-preserving subgraph $G^{\psi_1, \psi_2}$ of the graph $G^{\psi_1}$. If $\psi_2 \neq h$ then the set $\text{Tree}(G^{\psi_1, \psi_2}, T^C)^-$ coincides with the set of all inhibitory trees from $\text{Tree}(G^{\psi_1}, T^C)^-$ which have minimum cost relative to
If one of the cost functions $\psi_i$ is equal to $h$ then the set $\text{Tree}(G^{\psi_1,\ldots,\psi_i}, T_C)^-$ is a subset of the set of all inhibitory trees from $\text{Tree}(G^{\psi_1,\ldots,\psi_{i-1}}, T_C)^-$ that have minimum cost relative to $h$ among all trees from $\text{Tree}(G^{\psi_1,\ldots,\psi_{i-1}}, T_C)^-$.

We can study also totally optimal inhibitory trees relative to various combinations of cost functions. For a cost function $\psi$, we denote $\psi^{W,\alpha}(T) = \min \{ \psi(T, \Gamma) : \Gamma \in I_{W,\alpha}(T) \}$, i.e., $\psi^{W,\alpha}(T)$ is the minimum cost of a $(W,\alpha)$-inhibitory tree for $T$ relative to the cost function $\psi$. Let $\psi_1,\ldots,\psi_m$ be cost functions and $m \geq 2$. A $(W,\alpha)$-inhibitory tree $\Gamma$ for $T$ is called a totally optimal $(W,\alpha)$-inhibitory tree for $T$ relative to the cost functions $\psi_1,\ldots,\psi_m$ if $\psi_1(T, \Gamma) = \psi_1^{W,\alpha}(T),\ldots,\psi_m(T, \Gamma) = \psi_m^{W,\alpha}(T)$, i.e., $\Gamma$ is optimal relative to $\psi_1,\ldots,\psi_m$ simultaneously.

Assume that $\psi_1,\ldots,\psi_{m-1} \in \{\text{tpl}, L, L_n, L_t\}$ and $\psi_m \in \{h, \text{tpl}, L, L_n, L_t\}$. We now describe how to recognize the existence of a $(W,\alpha)$-inhibitory tree for $T$ which is a totally optimal $(W,\alpha)$-inhibitory tree for $T$ relative to the cost functions $\psi_1,\ldots,\psi_m$.

First, we construct the graph $G = \Delta_{\psi_1}(T_C)$ using the Algorithm $A_1$. For $i = 1,\ldots,m$, we apply to $G$ the procedure of optimization relative to $\psi_i$ (the Algorithm $A_5$). As a result, we obtain, for $i = 1,\ldots,m$, the graph $G^{\psi_i}$ and the number $\psi_i^{U,\alpha}(T_C)$ attached to the node $T_C$ of $G^{\psi_i}$. Next, we apply to $G$ sequentially the procedures of optimization relative to the cost functions $\psi_1,\ldots,\psi_m$. As a result, we obtain graphs $G^{\psi_1}, G^{\psi_1,\psi_2},\ldots, G^{\psi_1,\ldots,\psi_m}$ and numbers $\varphi_1, \varphi_2,\ldots, \varphi_m$ attached to the node $T_C$ of these graphs. We know (see Section 6.1) that a totally optimal $(U,\alpha)$-decision tree for $T_C$ relative to the cost functions $\psi_1,\ldots,\psi_m$ exists if and only if $\varphi_i = \psi_i^{U,\alpha}(T_C)$ for $i = 1,\ldots,m$. Using Proposition 26 one can show that a totally optimal $(W,\alpha)$-inhibitory tree for $T$ relative to the cost functions $\psi_1,\ldots,\psi_m$ exists if and only if a totally optimal $(U,\alpha)$-decision tree for $T_C$ relative to the cost functions $\psi_1,\ldots,\psi_m$ exists.
We did experiments to study the existence of totally optimal decision and inhibitory trees relative to the depth, average depth, and the number of nodes. Instead of the investigation of inhibitory trees for a decision table with many-valued decisions $T$, we studied decision trees for the table $T^C$ complementary to $T$.

6.3.1 Decision Tables Used in Experiments

We took datasets (decision tables) from UCI ML Repository [59] and removed one or more conditional attributes from them; as a result, for some tables, there are multiple rows that have equal values of conditional attributes but different decisions which are then merged into a single row labeled with the set of decisions from the group of equal rows. Before the experiment work, some preprocessing procedures are performed. An
attribute is removed if it has a unique value for each row. The missing value for an attribute is filled up with the most common value for that attribute.

In Table 6.1, the first column ‘Decision table T’ refers to the name of the new decision table T (that we get after removing attributes from the data set from UCI ML Repository), second column ‘Original data set - Columns’ refers to the name of the original data set along with the indexes of attributes removed from the original data set, the column ‘Rows’ refers to the number of rows, the column ‘Attr’ refers to the number of attributes, the column ‘|D(T)|’ refers to the total number of decisions in T, and the column ‘Spectrum’ refers to a sequence #1, #2, #3, ..., where #i means the number of rows in T that are labeled with sets of decisions containing i decisions.

6.3.2 Totally Optimal Trees Relative to Two Cost Functions

In Table 6.2, we have shown the results for the existence of totally optimal decision and inhibitory trees relative to two cost functions. The results are grouped according to the three pairs of cost functions: (average depth, depth), (number of nodes, depth), and (number of nodes, average depth).

For the case of decision trees, for each pair of cost functions, more than half of the considered decision tables do not have totally optimal decision trees. On the other hand, for the case of inhibitory trees, we have totally optimal inhibitory trees for all the decision tables for the pair (average depth, depth), for almost all the decision tables except two cases for the pair (number of nodes, depth), and for ten cases for the pair (number of nodes, average depth).
Table 6.2: Existence of totally optimal decision and inhibitory trees for two cost functions

### 6.3.3 Totally Optimal Trees Relative to Three Cost Functions

In Tables 6.3 and 6.4 we have shown the results of the experiments with multi-stage optimization for the three cost functions $L, h_{avg}$, and $h$. We have listed first the values of the cost functions for non-sequential, i.e., individual cost optimization, then the values of cost functions for sequential, i.e., multi-stage optimization, and then whether the decision tables have totally optimal trees or not. A totally optimal tree exists if and only if the values after non-sequential optimization are equal to the values after sequential optimization.

We can see that, for the case of decision trees, there are only two decision tables that have totally optimal trees. On the other hand, for the case of inhibitory trees, there are ten decision tables that have totally optimal trees. Moreover, the optimal inhibitory trees have usually smaller depth, average depth and number of nodes compared to optimal decision trees.
<table>
<thead>
<tr>
<th>Decision table</th>
<th>Non-sequential</th>
<th>Sequential</th>
<th>Has totally optimal decision trees?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$L$</td>
<td>$h_{avg}$</td>
<td>$h$</td>
</tr>
<tr>
<td>CARS-1</td>
<td>28</td>
<td>1.96</td>
<td>4</td>
</tr>
<tr>
<td>FLAGS-5</td>
<td>152</td>
<td>3.6</td>
<td>10</td>
</tr>
<tr>
<td>FLAGS-4</td>
<td>149</td>
<td>3.73</td>
<td>6</td>
</tr>
<tr>
<td>FLAGS-3</td>
<td>151</td>
<td>3.64</td>
<td>5</td>
</tr>
<tr>
<td>FLAGS-1</td>
<td>112</td>
<td>2.77</td>
<td>5</td>
</tr>
<tr>
<td>LYMPH-5</td>
<td>56</td>
<td>3.65</td>
<td>5</td>
</tr>
<tr>
<td>LYMPH-4</td>
<td>67</td>
<td>3.73</td>
<td>5</td>
</tr>
<tr>
<td>NURSERY-4</td>
<td>9</td>
<td>1.33</td>
<td>2</td>
</tr>
<tr>
<td>NURSERY-1</td>
<td>117</td>
<td>2.127</td>
<td>7</td>
</tr>
<tr>
<td>POKER-5A</td>
<td>230</td>
<td>2.62</td>
<td>4</td>
</tr>
<tr>
<td>POKER-5B</td>
<td>226</td>
<td>2.529</td>
<td>5</td>
</tr>
<tr>
<td>POKER-5C</td>
<td>69</td>
<td>2.33</td>
<td>5</td>
</tr>
<tr>
<td>ZOO-5</td>
<td>17</td>
<td>2.77</td>
<td>4</td>
</tr>
<tr>
<td>ZOO-4</td>
<td>23</td>
<td>3.34</td>
<td>5</td>
</tr>
<tr>
<td>ZOO-2</td>
<td>13</td>
<td>2.72</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 6.3: Existence of totally optimal decision trees for three cost functions

<table>
<thead>
<tr>
<th>Decision table</th>
<th>Non-sequential</th>
<th>Sequential</th>
<th>Has totally optimal inhibitory trees?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$L$</td>
<td>$h_{avg}$</td>
<td>$h$</td>
</tr>
<tr>
<td>CARS-1</td>
<td>14</td>
<td>1.47</td>
<td>4</td>
</tr>
<tr>
<td>FLAGS-5</td>
<td>17</td>
<td>1.9</td>
<td>3</td>
</tr>
<tr>
<td>FLAGS-4</td>
<td>17</td>
<td>1.81</td>
<td>3</td>
</tr>
<tr>
<td>FLAGS-3</td>
<td>17</td>
<td>1.83</td>
<td>3</td>
</tr>
<tr>
<td>FLAGS-1</td>
<td>9</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>LYMPH-5</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>LYMPH-4</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>NURSERY-4</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>NURSERY-1</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>POKER-5A</td>
<td>13</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>POKER-5B</td>
<td>13</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>POKER-5C</td>
<td>13</td>
<td>1.5</td>
<td>2</td>
</tr>
<tr>
<td>ZOO-5</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>ZOO-4</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>ZOO-2</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 6.4: Existence of totally optimal inhibitory trees for three cost functions
Chapter 7

Bi-criteria Optimization Problem for Decision and Inhibitory Trees: Cost vs. Cost

In this chapter, we study bi-criteria optimization problem cost vs. cost for decision (Sect. 7.1) and inhibitory (Sect. 7.2) trees. We consider an algorithm which constructs the set of Pareto optimal points for bi-criteria optimization problem for decision trees, and show how the constructed set can be transformed into the graphs of functions which describe the relationships between the studied cost functions. We extend the obtained results to the case of inhibitory trees.

As the first application of the created methods, we compare 12 greedy heuristics for construction of decision and inhibitory trees (Sects. 7.3 and 7.4) as single-criterion and bi-criteria optimization algorithms. For single-criterion optimization, we not only rank the heuristics based on the cost of constructed trees, but we also find the relative difference between the cost of trees constructed by heuristics and the cost of optimal trees. For bi-criteria optimization, we rank heuristics based on the minimum distance from the set of Pareto optimal points to the heuristic point coordinates of which are values of the two cost functions for the tree constructed by the heuristic.

Algorithms for analysis of relationships between various pairs of cost functions for decision trees and decision tables with one-valued decisions were previously described in papers such as [58, 60, 61]. Some initial results regarding the comparison of greedy heuristics for construction of decision trees for decision tables with one-valued decisions can be found in [23, 24, 62]. Papers [11, 12, 13, 36] contain initial results related to comparison of greedy heuristics for construction of decision trees for decision tables.
with many-valued decisions.

The second application is the study of decision trees for sorting problem. Usually, the problem under consideration is to sort \( n \) pairwise different elements \( x_1, \ldots, x_n \) from a linearly ordered set \([63, 64, 65]\). In this first case, there is only one permutation \((p_1, \ldots, p_n)\) of the set \(\{1, \ldots, n\}\) such that \(x_{p_1} < \ldots < x_{p_n}\), and each comparison \(x_i : x_j\) of two elements has only two results: \(x_i < x_j\) and \(x_i > x_j\). We consider also the second case when it is possible to have equal elements in the sequence \(x_1, \ldots, x_n\). In this case, each comparison \(x_i : x_j\) of two elements can have three possible results \(x_i < x_j\), \(x_i = x_j\), and \(x_i > x_j\), it can be more than one permutation \((p_1, \ldots, p_n)\) such that \(x_{p_1} \leq \ldots \leq x_{p_n}\), and we should find one of these permutations. For \(n = 2, \ldots, 6\), we compare the minimum depth, average depth, and number of nodes of decision trees for the considered two cases. The minimum depth is the same for both cases, the minimum average depth for \(n > 2\) is less for the second case, and the minimum number of nodes is less for the first case.

The third application is related to knowledge representation. When decision trees are used for knowledge representation, the usual goal is to minimize the number of nodes in the tree. However, the depth and the average depth are also important: we need to understand conjunctions of conditions corresponding to paths in the tree from the root to terminal nodes. We construct the sets of Pareto optimal points and study trade offs number of nodes vs. depth and number of nodes vs. average depth for some decision tables with many-valued decisions. We show that, at the cost of a minor increase in the number of nodes, we can essentially decrease the depth or the average depth of decision trees.

Note that the paper “Bi-criteria optimization of decision and inhibitory trees for decision tables with many-valued decisions” by Mohammad Azad and Mikhail Moshkov submitted to Discrete Applied Mathematics contains theoretical and experimental results similar to considered in this chapter.
7.1 Bi-criteria Optimization Problem for Decision Trees: Cost vs. Cost

In this section, we consider an algorithm which constructs the sets of Pareto optimal points for bi-criteria optimization problems for decision trees relative to two cost functions. We also show how the constructed set of Pareto optimal points can be transformed into the graphs of functions which describe the relationships between the considered cost functions.

7.1.1 Pareto Optimal Points: Cost vs. Cost

We begin with the consideration of an algorithm for construction of the set of Pareto optimal points.

Let $\psi$ and $\varphi$ be increasing cost functions for decision trees given by triples of functions $\psi^0, F, w$ and $\varphi^0, H, u$, respectively, $U$ be an uncertainty measure, $\alpha \in \mathbb{R}_+$, $T$ be a decision table with $n$ conditional attributes $f_1, \ldots, f_n$, and $G$ be a bundle-preserving subgraph of the graph $\Delta_{U,\alpha}(T)$ (it is possible that $G = \Delta_{U,\alpha}(T)$). Interesting cases are when $G = \Delta_{U,\alpha}(T)$ or $G$ is a result of application of the procedure of optimization of decision trees (algorithm $A_5$) relative to cost functions different from $\psi$ and $\varphi$ to the graph $\Delta_{U,\alpha}(T)$.

For each node $\Theta$ of the graph $G$, we denote $t_{\psi,\varphi}(G, \Theta) = \{(\psi(\Theta, \Gamma), \varphi(\Theta, \Gamma)) : \Gamma \in Tree(G, \Theta)\}$. Note that, by Proposition 16, if $G = \Delta_{U,\alpha}(T)$ then the set $Tree(G, \Theta)$ is equal to the set of $(U, \alpha)$-decision trees for $\Theta$. 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 $A_6$ which constructs the set $Par(t_{\psi,\varphi}(G, T))$. In fact, this algorithm constructs, for each node $\Theta$ of the graph $G$, the set $B(\Theta) = Par(t_{\psi,\varphi}(G, \Theta))$.

*Algorithm $A_6$. 

131
**Input:** Increasing cost functions for decision trees $\psi$ and $\varphi$ given by triples of functions $\psi^0, F, w$ and $\varphi^0, H, u$, respectively, a decision table $T$ with $n$ conditional attributes $f_1, \ldots, f_n$, and a bundle-preserving subgraph $G$ of the graph $\Delta_{U,\alpha}(T)$ where $U$ is an uncertainty measure and $\alpha \in \mathbb{R}_+$.

**Output:** The set $Par(t_{\psi,\varphi}(G,T))$ of Pareto optimal points for the set of pairs $t_{\psi,\varphi}(G,T) = \{(\psi(T,\Gamma), \varphi(T,\Gamma)) : \Gamma \in Tree(G,T)\}$.

1. If all nodes in $G$ are processed, then return the set $B(T)$. Otherwise, choose a node $\Theta$ 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 $f_i \in E_G(\Theta)$ 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 $\Theta$ is a terminal node, then set $B(\Theta) = \{(\psi^0(\Theta), \varphi^0(\Theta))\}$. Mark the node $\Theta$ as processed and proceed to step 1.

3. If $\Theta$ is a nonterminal node then, for each $f_i \in E_G(\Theta)$, apply the algorithm $A_3$ 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 $A_3$ and

   \[
   B(\Theta, f_i) = C(\Theta, f_i) \langle ++ \rangle \{(w(\Theta), u(\Theta))\}
   = \{(a + w(\Theta), b + u(\Theta)) : (a, b) \in C(\Theta, f_i)\}.
   \]

4. Construct the multiset $A(\Theta) = \bigcup_{f_i \in E_G(\Theta)} B(\Theta, f_i)$ by simple transcription of elements from the sets $B(\Theta, f_i), f_i \in E_G(\Theta)$. Apply to the obtained multiset $A(\Theta)$ the algorithm $A_2$ which constructs the set $Par(A(\Theta))$. Set $B(\Theta) = Par(A(\Theta))$. Mark the node $\Theta$ as processed and proceed to step 1.
**Proposition 30.** Let $\psi$ and $\varphi$ be increasing cost functions for decision trees given by triples of functions $\psi^0, F, w$ and $\varphi^0, H, u$, respectively, $U$ be an uncertainty measure, $\alpha \in \mathbb{R}^+$, $T$ be a decision table with $n$ conditional attributes $f_1, \ldots, f_n$, and $G$ be a bundle-preserving subgraph of the graph $\Delta_{U, \alpha}(T)$. Then, for each node $\Theta$ of the graph $G$, the algorithm $A_6$ constructs the set $B(\Theta) = \text{Par}(t_{\psi, \varphi}(G, \Theta))$.

**Proof.** We prove the considered statement by induction on nodes of $G$. Let $\Theta$ be a terminal node of $G$. Then $\text{Tree}(G, \Theta) = \{\text{tree}(\text{mcd}(\Theta))\}$,

$$t_{\psi, \varphi}(G, \Theta) = \text{Par}(t_{\psi, \varphi}(G, \Theta)) = \{((\psi^0(\Theta), \varphi^0(\Theta))\},$$

and $B(\Theta) = \text{Par}(t_{\psi, \varphi}(G, \Theta))$.

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

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

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

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

If we apply the algorithm $A_3$ to the functions $F, H$ and the sets

$$\text{Par}(P_1), \ldots, \text{Par}(P_t),$$

we obtain the set $\text{Par}(Q_i)$ where $Q_1 = P_1$, and, for $j = 2, \ldots, t$, $Q_j = Q_{j-1}(FH)P_j$ (see Proposition 11). It is not difficult to show that

$$P(f_i) = Q_i \{++\} \{(w(\Theta), u(\Theta))\} = \{(a + w(\Theta), b + u(\Theta)) : (a, b) \in Q_i\}$$

133
and $\text{Par}(P(f_i)) = \text{Par}(Q_i) \{++\} \{(w(\Theta), u(\Theta))\}$.

According to the induction hypothesis, $B(\Theta(f_i, a_j)) = \text{Par}(P_j)$ for $j = 1, \ldots, t$. Therefore $C(\Theta, f_i) = \text{Par}(Q_i)$ and $B(\Theta, f_i) = \text{Par}(P(f_i))$.

One can show that $t_{\psi, \phi}(G, \Theta) = \bigcup_{f_i \in E_G(\Theta)} P(f_i)$. By Lemma 9

$$\text{Par}(t_{\psi, \phi}(G, \Theta)) = \text{Par}\left(\bigcup_{f_i \in E_G(\Theta)} P(f_i)\right).$$

Using Lemma 8 we obtain $\text{Par}(t_{\psi, \phi}(G, \Theta)) = \text{Par}\left(\bigcup_{f_i \in E_G(\Theta)} \text{Par}(P(f_i))\right)$. Since $B(\Theta, f_i) = \text{Par}(P(f_i))$ for any $f_i \in E_G(\Theta)$, $\text{Par}(t_{\psi, \phi}(G, \Theta)) = \text{Par}(A(\Theta)) = B(\Theta)$.

We now analyze the number of elementary operations made by the algorithm $\mathcal{A}_6$ during the construction of the set $\text{Par}(t_{\psi, \phi}(G, T))$ for integral cost functions, for the cases when $F(x, y) = \max(x, y)$ ($h$ is an example of such cost function) and when $F(x, y) = x + y$ ($\text{tpl}, L, L_n$, and $L_t$ are examples of such cost functions).

Let us recall that $\text{range}(T) = \max\{|E(T, f_i)| : i = 1, \ldots, n\}$,

$$\text{ub}(\psi, T) = \max\{\psi(\Theta, \Gamma) : \Theta \in \text{SEP}(T), \Gamma \in \text{DT}(\Theta)\},$$

and $\psi(\Theta, \Gamma) \in \{0, 1, \ldots, \text{ub}(\psi, T)\}$ for any separable subtable $\Theta$ of $T$ and for any decision tree $\Gamma$ for $\Theta$. Upper bounds on the value $\text{ub}(\psi, T)$ for $\psi \in \{h, \text{tpl}, L, L_n, L_t\}$ are given in Lemma 13: $\text{ub}(h, T) \leq n$, $\text{ub}(\text{tpl}, T) \leq n N(T)$, $\text{ub}(L, T) \leq 2 N(T)$, $\text{ub}(L_n, T) \leq N(T)$, and $\text{ub}(L_t, T) \leq N(T)$.

**Proposition 31.** Let $\psi$ and $\varphi$ be increasing integral cost functions for decision trees given by triples of functions $\psi^0, F, w$ and $\varphi^0, H, u$, respectively, $F \in \{\max(x, y), x+y\}$, $U$ be an uncertainty measure, $\alpha \in \mathbb{R}_+$, $T$ be a decision table with $n$ conditional attributes $f_1, \ldots, f_n$, and $G$ be a bundle-preserving subgraph of the graph $\Delta_{U, \alpha}(T)$. 134
Then, to construct the set \( \text{Par}(t_{\psi,\varphi}(G,T)) \), the algorithm \( A_6 \) makes

\[
O(L(G)\text{range}(T)\text{ub}(\psi,T)^2n \log(\text{ub}(\psi,T)n))
\]

elementary operations (computations of \( F, H, w, u, \psi^0, \varphi^0 \), additions, and comparisons) if \( F = \max(x,y) \), and

\[
O(L(G)\text{range}(T)^2\text{ub}(\psi,T)^2n \log(\text{range}(T)\text{ub}(\psi,T)n))
\]

elementary operations (computations of \( F, H, w, u, \psi^0, \varphi^0 \), additions, and comparisons) if \( F = x + y \).

**Proof.** It is clear that, for any node \( \Theta \) of the graph \( G \), \( t_{\psi,\varphi}(G,\Theta)^{(1)} = \{a : (a,b) \in t_{\psi,\varphi}(G,\Theta) \} \subseteq \{0,\ldots,\text{ub}(\psi,T)\} \). From Proposition 30 it follows that \( B(\Theta) = \text{Par}(t_{\psi,\varphi}(G,\Theta)) \) for any node \( \Theta \) of the graph \( G \). Therefore, for any node \( \Theta \) of the graph \( G \), \( B(\Theta)^{(1)} = \{a : (a,b) \in B(\Theta) \} \subseteq \{0,\ldots,\text{ub}(\psi,T)\} \).

Let \( F(x,y) = \max(x,y) \). To process a terminal node \( \Theta \), the algorithm \( A_6 \) makes two elementary operations (computations of \( \psi^0 \) and \( \varphi^0 \)).

We now consider the processing of a nonterminal node \( \Theta \) (see description of the algorithm \( A_6 \)). We know that \( B(\Theta(f_i,a_j))^{(1)} \subseteq \{0,\ldots,\text{ub}(\psi,T)\} \) for \( j = 1,\ldots,t \), and \( t \leq \text{range}(T) \). From Proposition 12 it follows that \( |C(\Theta,f_i)| \leq \text{ub}(\psi,T) + 1 \), and to construct the set \( C(\Theta,f_i) \), the algorithm \( A_3 \) makes

\[
O(\text{range}(T)\text{ub}(\psi,T)^2 \log \text{ub}(\psi,T))
\]

elementary operations (computations of \( F, H \) and comparisons). To construct the set \( B(\Theta,f_i) \) from the set \( C(\Theta,f_i) \), the algorithm makes computations of \( w \) and \( u \), and at most \( 2\text{ub}(\psi,T) + 2 \) additions. From here it follows that, to construct the set
\( B(\Theta, f_i) \), the algorithm makes
\[
O(\text{range}(T)ub(\psi, T)^2 \log ub(\psi, T))
\]
elementary operations. It is clear that \(|E_G(\Theta)| \leq n\). Therefore, to construct the set \( B(\Theta, f_i) \) for each \( f_i \in E_G(\Theta) \), the algorithm makes
\[
O(\text{range}(T)ub(\psi, T)^2 n \log(ub(\psi, T)))
\]
elementary operations.

Since \(|C(\Theta, f_i)| \leq ub(\psi, T) + 1\), we have \(|B(\Theta, f_i)| \leq ub(\psi, T) + 1\). Since
\[
|E_G(\Theta)| \leq n,
\]
we have \(|A(\Theta)| \leq n(ub(\psi, T) + 1)\) where \(A(\Theta) = \bigcup_{f_i \in E(\Theta)} B(\Theta, f_i)\). From Proposition 10 it follows that, to construct the set \( B(\Theta) = \text{Par}(A(\Theta)) \), the algorithm \( A_2 \) makes \( O(ub(\psi, T)n \log(ub(\psi, T)n)) \) comparisons. So, to process a nonterminal node \( \Theta \) (to construct \( B(\Theta) = \text{Par}(t_{\psi, \phi}(G, \Theta)) \) if
\[
B(\Theta(f_i, a_j)) = \text{Par}(t_{\psi, \phi}(G, \Theta(f_i, a_j)))
\]
is known for all \( f_i \in E_G(\Theta) \) and \( a_j \in E(\Theta, f_i) \), the algorithm \( A_6 \) makes
\[
O(\text{range}(T)ub(\psi, T)^2 n \log(ub(\psi, T)n))
\]
elementary operations.

To construct the set \( \text{Par}(t_{\psi, \phi}(G, T)) \) for given decision table \( T \) with \( n \) conditional
attributes and the graph $G$, it is enough to make

$$O(L(G)\text{range}(T)\text{ub}(\psi, T)^2n \log(\text{ub}(\psi, T)n))$$

elementary operations (computations of $F$, $H$, $w$, $u$, $\psi^0$, $\varphi^0$, additions, and comparisons).

Let $F(x,y) = x + y$. To process a terminal node $\Theta$, the algorithm $A_6$ makes two elementary operations (computations of $\psi^0$ and $\varphi^0$).

We now consider the processing of a nonterminal node $\Theta$ (see description of the algorithm $A_6$). We know that $B(\Theta(f_i,a_j))^{(1)} \subseteq \{0,\ldots,\text{ub}(\psi, T)\}$ for $j = 1,\ldots,t$, and $t \leq \text{range}(T)$. From Proposition 12 it follows that $|C(\Theta,f_i)| \leq t \times \text{ub}(\psi, T) + 1 \leq \text{range}(T)\text{ub}(\psi, T) + 1$, and to construct the set $C(\Theta,f_i)$, the algorithm $A_3$ makes

$$O(\text{range}(T)^2\text{ub}(\psi, T)^2 \log(\text{range}(T)\text{ub}(\psi, T)))$$

elementary operations (computations of $F$, $H$ and comparisons). To construct the set $B(\Theta,f_i)$ from the set $C(\Theta,f_i)$, the algorithm makes computations of $w$ and $u$, and at most $2\text{range}(T)\text{ub}(\psi, T) + 2$ additions. From here it follows that, to construct the set $B(\Theta,f_i)$, the algorithm makes

$$O(\text{range}(T)^2\text{ub}(\psi, T)^2 \log(\text{range}(T)\text{ub}(\psi, T)))$$

elementary operations. It is clear that $|E_G(\Theta)| \leq n$. Therefore, to construct the set $B(\Theta,f_i)$ for each $f_i \in E_G(\Theta)$, the algorithm makes

$$O(\text{range}(T)^2\text{ub}(\psi, T)^2 n \log(\text{range}(T)\text{ub}(\psi, T)))$$

elementary operations.
Since $|C(\Theta, f_i)| \leq \text{range}(T)\text{ub}(\psi, T) + 1$, we have

$$|B(\Theta, f_i)| \leq \text{range}(T)\text{ub}(\psi, T) + 1.$$ 

Since $|E_G(\Theta)| \leq n$, we have $|A(\Theta)| \leq n(\text{range}(T)\text{ub}(\psi, T) + 1)$ where $A(\Theta) = \bigcup_{f_i \in E_G(\Theta)} B(\Theta, f_i)$. From Proposition 10 it follows that, to construct the set $B(\Theta) = \text{Par}(A(\Theta))$, the algorithm $\mathcal{A}_2$ makes

$$O(\text{range}(T)\text{ub}(\psi, T)n \log(\text{range}(T)\text{ub}(\psi, T)n))$$

comparisons. So, to process a nonterminal node $\Theta$ (to construct

$$B(\Theta) = \text{Par}(t_{\psi, \varphi}(G, \Theta))$$

if $B(\Theta(f_i, a_j)) = \text{Par}(t_{\psi, \varphi}(G, \Theta(f_i, a_j)))$ is known for all $f_i \in E_G(\Theta)$ and $a_j \in E(\Theta, f_i)$, the algorithm $\mathcal{A}_6$ makes

$$O(\text{range}(T)^2\text{ub}(\psi, T)^2n \log(\text{range}(T)\text{ub}(\psi, T)n))$$

elementary operations.

To construct the set $\text{Par}(t_{\psi, \varphi}(G, T))$ for given decision table $T$ with $n$ conditional attributes and the graph $G$, it is enough to make

$$O(\text{L}(G)\text{range}(T)^2\text{ub}(\psi, T)^2n \log(\text{range}(T)\text{ub}(\psi, T)n))$$

elementary operations (computations of $F$, $H$, $w$, $u$, $\psi^0$, $\varphi^0$, additions, and comparisons).

Note that similar results can be obtained if $H \in \{\max(x, y), x + y\}$. 

138
Proposition 32. Let $\psi$ and $\varphi$ be cost functions for decision trees given by triples of functions $\psi^0, F, w$ and $\varphi^0, H, u$, respectively, $\psi, \varphi \in \{h, \text{tpl}, L, L_n, L_t\}$, and $\mathcal{U}$ be a restricted information system. Then the algorithm $\mathcal{A}_6$ has polynomial time complexity for decision tables from $\mathcal{T}(\mathcal{U})$ depending on the number of conditional attributes in these tables.

Proof. Since $\psi, \varphi \in \{h, \text{tpl}, L, L_n, L_t\}$, $\psi^0$ and $\varphi^0$ are constants, each of the functions $F, H$ is either $\max(x, y)$ or $x + y$, and each of the functions $w, u$ is either a constant or $N(T)$. From Proposition 31 and Lemma 13 it follows that, for the algorithm $\mathcal{A}_6$, the number of elementary operations (computations of $F, H, w, u, \psi^0, \varphi^0$, additions, and comparisons) is bounded from above by a polynomial depending on the size of input table $T$ and on the number of separable subtables of $T$. All operations with numbers are basic ones. The computations of numerical parameters of decision tables used by the algorithm $\mathcal{A}_6$ (constants and $N(T)$) have polynomial time complexity depending on the size of decision tables.

According to Proposition 31 the algorithm $\mathcal{A}_6$ has polynomial time complexity for decision tables from $\mathcal{T}(\mathcal{U})$ depending on the number of conditional attributes in these tables. \hfill \qed

7.1.2 Relationships between Two Cost Functions

We now show how to construct graphs describing relationships between two cost functions.

Let $\psi$ and $\varphi$ be increasing cost functions for decision trees, $U$ be an uncertainty measure, $\alpha \in \mathbb{R}_+$, $T$ be a decision table, and $G$ be a bundle-preserving subgraph of the graph $\Delta_{U,\alpha}(T)$ (it is possible that $G = \Delta_{U,\alpha}(T)$).

To study relationships between cost functions $\psi$ and $\varphi$ on the set of decision trees $\text{Tree}(G, T)$ we consider partial functions $\mathcal{T}^{\psi, \varphi}_{G, T} : \mathbb{R} \to \mathbb{R}$ and $\mathcal{T}^{\varphi, \psi}_{G, T} : \mathbb{R} \to \mathbb{R}$ defined as
follows:

\[
\mathcal{T}^\psi_{{G,T}}(x) = \min\{\varphi(T, \Gamma) : \Gamma \in \text{Tree}(G, T), \psi(T, \Gamma) \leq x\},
\]

\[
\mathcal{T}^\varphi_{{G,T}}(x) = \min\{\psi(T, \Gamma) : \Gamma \in \text{Tree}(G, T), \varphi(T, \Gamma) \leq x\}.
\]

Let \((a_1, b_1), \ldots, (a_k, b_k)\) be the normal representation of the set \(\text{Par}(t_{{\psi,\varphi}}(G, T))\) where \(a_1 < \ldots < a_k\) and \(b_1 > \ldots > b_k\). By Lemma 11 and Remark 4, for any \(x \in \mathbb{R}\),

\[
\mathcal{T}^\psi_{{G,T}}(x) = \begin{cases} 
\text{undefined}, & x < a_1 \\
\quad b_1, & a_1 \leq x < a_2 \\
\quad \ldots & \ldots \\
\quad b_{k-1}, & a_{k-1} \leq x < a_k \\
\quad b_k, & a_k \leq x
\end{cases}
\]

\[
\mathcal{T}^\varphi_{{G,T}}(x) = \begin{cases} 
\text{undefined}, & x < b_k \\
\quad a_k, & b_k \leq x < b_{k-1} \\
\quad \ldots & \ldots \\
\quad a_2, & b_2 \leq x < b_1 \\
\quad a_1, & b_1 \leq x
\end{cases}
\]

### 7.2 Bi-criteria Optimization Problem for Inhibitory Trees: Cost vs. Cost

In this section, we consider bi-criteria optimization problem cost vs. cost for inhibitory trees.

Let \(T\) be a nondegenerate decision table with \(n\) conditional attributes \(f_1, \ldots, f_n\), \(T^C\) be the decision table complementary to \(T\), \(U\) be an uncertainty measure, \(W\) be a completeness measure, \(U\) and \(W\) are dual, \(\alpha \in \mathbb{R}_+\), and \(\psi, \varphi \in \{h, tpl, L, L_n, L_t\}\).
Let $G$ be a bundle-preserving subgraph of the graph $\Delta_{U,\alpha}(T^C)$ and

$$t_{\psi,\varphi}(G, T^C) = \{(\psi(T^C, \Gamma), \varphi(T^C, \Gamma)) : \Gamma \in \text{Tree}(G, T^C)\}.$$ 

In Section 7.1 the Algorithm $A_6$ is described which constructs the set

$$\text{Par}(t_{\psi,\varphi}(G, T^C))$$

of Pareto optimal points for the set of pairs $t_{\psi,\varphi}(G, T^C)$.

In Section 6.2 we show that $\text{Tree}(G, T^C)^- \subseteq IT_{W,\alpha}(T)$. In particular, if $G = \Delta_{U,\alpha}(T^C)$ then $\text{Tree}(G, T^C)^- = IT_{W,\alpha}(T)$. If $G = \Delta_{U,\alpha}(T^C)^{\eta}$ where

$$\eta \in \{\text{tpl}, L, L_t, L_n\}$$

then $\text{Tree}(G, T^C)^-$ is equal to the set of all trees from $IT_{W,\alpha}(T)$ which have minimum cost relative to $\eta$ among all inhibitory trees from the set $IT_{W,\alpha}(T)$.

Denote $it_{\psi,\varphi}(G, T) = \{(\psi(T, \Gamma^-), \varphi(T, \Gamma^-)) : \Gamma^- \in \text{Tree}(G, T^C)^-\}$. From Proposition 26 it follows that $(\psi(T, \Gamma^-), \varphi(T, \Gamma^-)) = (\psi(T^C, \Gamma), \varphi(T^C, \Gamma))$ for any $\Gamma \in \text{Tree}(G, T^C)$. Therefore $t_{\psi,\varphi}(G, T^C) = it_{\psi,\varphi}(G, T)$ and

$$\text{Par}(t_{\psi,\varphi}(G, T^C)) = \text{Par}(it_{\psi,\varphi}(G, T)).$$

To study relationships between cost functions $\psi$ and $\varphi$ on the set of inhibitory trees $\text{Tree}(G, T^C)^-$ we consider partial functions $IT_{G,T}^{\psi,\varphi} : \mathbb{R} \to \mathbb{R}$ and $IT_{G,T}^{\varphi,\psi} : \mathbb{R} \to \mathbb{R}$ defined as follows:

$$IT_{G,T}^{\psi,\varphi}(x) = \min \{\varphi(T, \Gamma^-) : \Gamma^- \in \text{Tree}(G, T^C)^-, \psi(T, \Gamma^-) \leq x\},$$

$$IT_{G,T}^{\varphi,\psi}(x) = \min \{\psi(T, \Gamma^-) : \Gamma^- \in \text{Tree}(G, T^C)^-, \varphi(T, \Gamma^-) \leq x\}.$$
Let \((a_1, b_1), \ldots, (a_k, b_k)\) be the normal representation of the set
\[
\text{Par}(\text{it}_{\psi, \varphi}(G, T)) = \text{Par}(\text{t}_{\psi, \varphi}(G, T^C))
\]
where \(a_1 < \ldots < a_k\) and \(b_1 > \ldots > b_k\). By Lemma 11 and Remark 4, for any \(x \in \mathbb{R}\),
\[
\text{IT}_{G,T}^{\psi, \varphi}(x) = \begin{cases}
\text{undefined}, & x < a_1 \\
b_1, & a_1 \leq x < a_2 \\
\ldots & \ldots \\
b_{k-1}, & a_{k-1} \leq x < a_k \\
b_k, & a_k \leq x
\end{cases},
\]
\[
\text{IT}_{G,T}^{\varphi, \psi}(x) = \begin{cases}
\text{undefined}, & x < b_k \\
a_k, & b_k \leq x < b_{k-1} \\
\ldots & \ldots \\
a_2, & b_2 \leq x < b_1 \\
a_1, & b_1 \leq x
\end{cases}.
\]

### 7.3 Greedy Heuristics for Construction of Decision and Inhibitory Trees

In this section, we consider four uncertainty measures and three types of impurity functions. Each pair (uncertainty measure, type of impurity function) defines an impurity function which is used to choose attributes attached to nodes of decision trees. The obtained 12 greedy heuristics for construction of decision trees are also extended to the construction of inhibitory trees.
7.3.1 Uncertainty Measures

Uncertainty measure $U$ is a function from the set of decision tables with many-valued decisions to the set of real numbers such that $U(T) \geq 0$ for any decision table $T$, and $U(T) = 0$ if and only if $T$ is degenerate.

Let $T$ be a nonempty decision table with many-valued decisions, $D(T) = \{d_1, \ldots, d_m\}$ and $p_i = \frac{N_{d_i}(T)}{N(T)}$ for $i = 1, \ldots, m$. Let $d_1, \ldots, d_m$ be ordered such that $p_1 \geq \ldots \geq p_m$. For $i = 1, \ldots, m$, we denote by $N'_{d_i}(T)$ the number of rows in $T$ such that the set of decisions attached to row contains $d_i$, and if $i > 1$ then this set does not contain $d_1, \ldots, d_{i-1}$, and $p'_i = \frac{N'_{d_i}(T)}{N(T)}$. We consider the following four uncertainty measures (we assume $0 \log_2 0 = 0$, and for any empty table $T$, the value of each of the considered functions is equal to 0):

- Misclassification error: $me(T) = N(T) - N_{med}(T)$.
- Sorted entropy: $entSort(T) = -\sum_{i=1}^m p'_i \log_2 p'_i$ (see [48]).
- Multi-label entropy: $entML(T) = 0$, if and only if $T$ is degenerate, otherwise, it is equal to $-\sum_{i=1}^m (p_i \log_2 p_i + q_i \log_2 q_i)$, where, $q_i = 1 - p_i$ (see [45]).
- Absence: $abs(T) = \prod_{i=1}^m q_i$, where $q_i = 1 - p_i$.

7.3.2 Impurity Types and Impurity Functions

Let $f_i \in E(T)$, 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 three types which gives “impurity” of this partition:

- weighted max (wm): $I(T, f_i) = \max_{1 \leq j \leq t} U(T_j)N(T_j)$.
- weighted sum (ws): $I(T, f_i) = \sum_{j=1}^t U(T_j)N(T_j)$.
multiplied weighted sum ($Mult_{ws}$): $I(T, f_i) = \left( \sum_{j=1}^t U(T_j)N(T_j) \right) \cdot \log_2 t$.

As a result, we have 12 (for each of 3 types and each of 4 uncertainty measures) impurity functions.

7.3.3 Greedy Heuristics for Decision Tree Construction

For each impurity function $I$, we describe a greedy heuristic $H_I$ which, for a given nonempty decision table $T$, constructs a decision tree for the table $T$.

Greedy heuristic $H_I$.

\textit{Input}: A nonempty decision table with many-valued decisions $T$.

\textit{Output}: Decision tree $H_I(T)$ for $T$.

1. Construct the tree $G$ consisting of a single node labeled with the table $T$.

2. If no node of the tree $G$ is labeled with a table then denote the tree $G$ by $H_I(T)$, and return.

3. Otherwise, choose a node $v$ in $G$ which is labeled with a subtable $T'$ of the table $T$.

4. Let $U$ be the uncertainty measure for $I$. If $U(T') = 0$ then, instead of $T'$, mark the node $v$ with the common decision for $T'$. Return to step 2.

5. If $U(T') \neq 0$ then, for each $f_i \in E(T')$, we compute the value of the impurity function $I(T', f_i)$. Choose the attribute $f_{i_0} \in E(T')$, where $i_0$ is the minimum $i$ for which $I(T', f_i)$ has the minimum value. Instead of $T'$, mark the node $v$ with the attribute $f_{i_0}$. For each $\delta \in E(T', f_{i_0})$, add to the tree $G$ the node $v_\delta$ and mark this node with the subtable $T'(f_{i_0}, \delta)$. Draw an edge from $v$ to $v_\delta$ and mark this edge with $\delta$. Return to step 2.
We consider 12 greedy heuristics (see Table 7.1). Each heuristic is specified by an uncertainty measure and a type of impurity function. The time complexities of these heuristics are polynomial depending on the size of the input table.

### 7.3.4 Greedy Heuristics for Inhibitory Tree Construction

The described heuristics can be extended to the construction of inhibitory trees. Let $T$ be a decision table, $H_I$ be one of the heuristics described in Table 7.1 and $\Gamma$ be a decision tree for $T^C$ constructed by the heuristic $H_I$. Then the tree $\Gamma^-$ will be considered as an inhibitory tree for $T$ constructed by the heuristic $H_I$.

### 7.4 Comparison of Greedy Heuristics

In this section, we compare 12 greedy heuristics for construction of decision and inhibitory trees as single-criterion and bi-criteria optimization algorithms.

We took decision tables (datasets) from UCI ML Repository [59] and removed one or more conditional attributes from them. As a result, for some tables, there are multiple rows that have equal values of conditional attributes but different decisions which are then merged into a single row labeled with the set of decisions from the group of equal rows. Before the experimental work, some preprocessing procedures are

<table>
<thead>
<tr>
<th>Name of heuristic</th>
<th>Uncertainty measure</th>
<th>Type of impurity</th>
</tr>
</thead>
<tbody>
<tr>
<td>wm_entML</td>
<td>entML</td>
<td>wm</td>
</tr>
<tr>
<td>wm_entSort</td>
<td>entSort</td>
<td>wm</td>
</tr>
<tr>
<td>wm_abs</td>
<td>abs</td>
<td>wm</td>
</tr>
<tr>
<td>wm_me</td>
<td>me</td>
<td>wm</td>
</tr>
<tr>
<td>ws_entML</td>
<td>entML</td>
<td>ws</td>
</tr>
<tr>
<td>ws_entSort</td>
<td>entSort</td>
<td>ws</td>
</tr>
<tr>
<td>ws_abs</td>
<td>abs</td>
<td>ws</td>
</tr>
<tr>
<td>ws_me</td>
<td>me</td>
<td>ws</td>
</tr>
<tr>
<td>Mult_ws_entML</td>
<td>entML</td>
<td>Mult_ws</td>
</tr>
<tr>
<td>Mult_ws_entSort</td>
<td>entSort</td>
<td>Mult_ws</td>
</tr>
<tr>
<td>Mult_ws_abs</td>
<td>abs</td>
<td>Mult_ws</td>
</tr>
<tr>
<td>Mult_ws_me</td>
<td>me</td>
<td>Mult_ws</td>
</tr>
</tbody>
</table>

Table 7.1: Greedy heuristics
### Table 7.2: Decision tables with many-valued decisions used in experiments

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

An attribute is removed if it has a unique value for each row. The missing value for an attribute is filled up with the most common value for that attribute.

In Table 7.2, the first column ‘Decision table $T$’ refers to the name of the new decision table $T$ with many-valued decisions that we get after removing attributes from a dataset from UCI ML Repository. The second column ‘Original data set - removed columns’ refers to the name of the original data set along with the indexes of attributes removed from the original dataset. The column ‘Rows’ refers to the number of rows in $T$, and the column ‘Attr’ refers to the number of conditional attributes. The column ‘$|D(T)|$’ refers to the total number of decisions in $T$, and the column ‘Spectrum’ refers to a sequence #1, #2, #3, ..., where #i means the number of rows in $T$ that are labeled with sets of decisions containing exactly i decisions.

The first part of experiments is the construction of the sets of Pareto optimal points (POPs) for different pairs of cost functions for both decision and inhibitory trees. We use the obtained results for comparison of heuristics as algorithms for single-criterion and bi-criteria optimization of the decision and inhibitory trees.

In Table 7.3 we have shown the total number of POPs for each pair of cost functions for each decision tables for both decision and inhibitory trees. We can see that often there are more POPs in the case of decision trees than in the case
of inhibitory trees. Probably, it is because the complexity of decision trees is often larger than the complexity of inhibitory trees. Note that the number of POPs is equal to one if and only if there is a tree which is optimal relative to both cost functions simultaneously.

The second part of the experiments is the application of each heuristic to each decision table $T$ and to its complimentary table $T^C$ and transforming the decision tree constructed for $T^C$ into an inhibitory tree for $T$. For each of the constructed trees and for each of the cost functions, we find the cost of the tree. For each pair of cost functions and for each heuristic, we find a heuristic point whose coordinates are two costs of the constructed tree.

For any pair of cost functions, we can draw a two-dimensional graph to show the positions of POPs and corresponding heuristic points. As we might have different maximum values for the different coordinates of POPs, we normalize the coordinates of all points by the maximum values of POP coordinates. For example, we have shown in Figure 7.1 the positions of normalized POPs and all 12 normalized heuristic points for the decision table lymph-5 for depth and number of nodes of decision trees. For
Figure 7.1: Bi-criteria optimization of decision trees for Lymph-5 relative to $h$ and $L$ – normalized POPs and normalized heuristic points

For each heuristic, we took the minimum distance from all POPs to the heuristic point. We call it ‘Min_Distance’. An example has been shown in Table 7.4 for the decision table Lymph-5. It is clear that the minimum distance is smallest in the case of Mult.ws_entML, Mult.ws_abs and Mult.ws_entSort heuristics.

After getting all ‘Min_Distance’s for all decision tables and pairs of cost functions, we can compare the heuristics as bi-criteria optimization algorithms by their average of ranks based on the ‘Min_Distance’. We assign the best performing heuristic the rank of 1, the second best rank 2, and so on. We break ties by computing the average of ranks. Let $r_{ij}$ be the rank of the $j$-th of 12 heuristics on the decision table $T_i$ ($i = 1, \ldots, 15$). For $j = 1, \ldots, 12$, we correspond to the heuristic $H_j$ the average of rank $R_j = \frac{1}{15} \cdot \sum_{i=1}^{15} r_{ij}$.

The results are shown in Tables 7.5 and 7.6 where we list average of ranks for single and bi-criteria optimization (total six possible combinations). The three top-ranked heuristics are highlighted in bold. For single-criterion optimization, the ranking is based on the values of the cost functions for the constructed trees.

For single-criterion optimization relative to a cost function $\psi$, we also consider
Table 7.4: The minimum distance from normalized POPs to each normalized heuristic point for lymph-5 decision table for depth and number of nodes of decision trees

<table>
<thead>
<tr>
<th>POPs</th>
<th>Heuristics</th>
<th>Points</th>
<th>Min_Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>h</td>
<td>L</td>
</tr>
<tr>
<td>0.56</td>
<td>1.00</td>
<td>0.78</td>
<td>1.27</td>
</tr>
<tr>
<td>0.67</td>
<td>0.89</td>
<td>0.78</td>
<td>1.27</td>
</tr>
<tr>
<td>1.00</td>
<td>0.88</td>
<td>0.67</td>
<td>1.39</td>
</tr>
<tr>
<td></td>
<td>wm_me</td>
<td>0.78</td>
<td>1.27</td>
</tr>
<tr>
<td></td>
<td>ws_me</td>
<td>0.78</td>
<td>1.20</td>
</tr>
<tr>
<td></td>
<td>ws_entML</td>
<td>0.89</td>
<td>1.17</td>
</tr>
<tr>
<td></td>
<td>ws_abs</td>
<td>0.78</td>
<td>1.14</td>
</tr>
<tr>
<td></td>
<td>ws_entSort</td>
<td>0.89</td>
<td>1.16</td>
</tr>
<tr>
<td></td>
<td>Mult_ws_me</td>
<td>0.78</td>
<td>1.23</td>
</tr>
<tr>
<td></td>
<td>Mult_ws_entML</td>
<td>1.00</td>
<td>1.06</td>
</tr>
<tr>
<td></td>
<td>Mult_ws_abs</td>
<td>1.00</td>
<td>1.06</td>
</tr>
<tr>
<td></td>
<td>Mult_ws_entSort</td>
<td>1.00</td>
<td>1.06</td>
</tr>
</tbody>
</table>

Table 7.5: Average of ranks of heuristics for decision trees

average relative difference (ARD) for each of 12 heuristics. For a heuristic and a decision table, the relative difference is $\frac{\psi_{\text{greedy}} - \psi_{\text{opt}}}{\psi_{\text{opt}}} \times 100\%$, where $\psi_{\text{greedy}}$ is the cost of the tree constructed by the heuristic, and $\psi_{\text{opt}}$ is the minimum cost of the tree for the considered table. ARD is the average value of relative differences among all the considered 15 decision tables. The results are shown in Table 7.7. The top three or two heuristics with minimum ARD has been highlighted. For decision trees, we can see that some heuristics can give less than 5% ARD for the average depth. Interestingly, for inhibitory trees, we can see smaller or even zero ARD.

To compare all heuristics, we make Tables 7.8 and 7.9 containing the positions of
the heuristics based on the corresponding values from Tables 7.5, 7.6 and 7.7 (total nine possible combinations). Here, the columns “h”, “\(h_{\text{avg}}\)”, and “\(L\)” refer to the position based on average of rank and “\(h'\)”, “\(h'_{\text{avg}}\)”, and “\(L'\)” refer to the position based on ARD.

The positions based on average of ranks are usually close to the positions based on ARD. There is no a heuristic which is good for all nine possible combinations both for decision and inhibitory trees. However \(ws_{\text{entSort}}\) heuristic looks not bad. It is interesting that there exist heuristics which are good for bi-criteria optimization but bad for single-criterion optimization. An example is the heuristic \(Mult_{ws_{\text{me}}}\) which

<table>
<thead>
<tr>
<th>Heuristics</th>
<th>(h)</th>
<th>(h_{\text{avg}})</th>
<th>(L)</th>
<th>(h, h_{\text{avg}})</th>
<th>(h, L)</th>
<th>(h_{\text{avg}}, L)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(wm_{\text{entML}})</td>
<td>6.33</td>
<td>7.10</td>
<td>8.10</td>
<td>7.10</td>
<td>8.10</td>
<td>8.10</td>
</tr>
<tr>
<td>(wm_{\text{entSort}})</td>
<td>6.33</td>
<td>6.87</td>
<td>7.87</td>
<td>6.87</td>
<td>7.87</td>
<td>7.87</td>
</tr>
<tr>
<td>(wm_{\text{abs}})</td>
<td>6.33</td>
<td>6.83</td>
<td>6.63</td>
<td>6.77</td>
<td>6.87</td>
<td>6.87</td>
</tr>
<tr>
<td>(wm_{\text{me}})</td>
<td>6.33</td>
<td>6.87</td>
<td>7.87</td>
<td>6.87</td>
<td>7.87</td>
<td>7.87</td>
</tr>
<tr>
<td>(ws_{\text{entML}})</td>
<td><strong>6.27</strong></td>
<td><strong>5.50</strong></td>
<td><strong>5.70</strong></td>
<td><strong>5.77</strong></td>
<td><strong>5.57</strong></td>
<td><strong>5.97</strong></td>
</tr>
<tr>
<td>(ws_{\text{entSort}})</td>
<td><strong>5.93</strong></td>
<td><strong>5.33</strong></td>
<td><strong>5.53</strong></td>
<td><strong>5.33</strong></td>
<td><strong>5.83</strong></td>
<td><strong>5.87</strong></td>
</tr>
<tr>
<td>(ws_{\text{abs}})</td>
<td>6.33</td>
<td>6.37</td>
<td>6.10</td>
<td>6.30</td>
<td>6.33</td>
<td>6.33</td>
</tr>
<tr>
<td>(ws_{\text{me}})</td>
<td>6.33</td>
<td>6.00</td>
<td>6.43</td>
<td>6.00</td>
<td>6.63</td>
<td>6.67</td>
</tr>
<tr>
<td>(Mult_{ws_{\text{entML}}})</td>
<td>7.37</td>
<td>7.23</td>
<td><strong>5.17</strong></td>
<td>7.23</td>
<td>6.23</td>
<td><strong>5.83</strong></td>
</tr>
<tr>
<td>(Mult_{ws_{\text{entSort}}})</td>
<td>7.00</td>
<td>6.90</td>
<td>5.97</td>
<td>6.90</td>
<td><strong>5.23</strong></td>
<td><strong>5.17</strong></td>
</tr>
<tr>
<td>(Mult_{ws_{\text{abs}}})</td>
<td>7.50</td>
<td>7.17</td>
<td>6.60</td>
<td>7.03</td>
<td><strong>5.23</strong></td>
<td><strong>5.30</strong></td>
</tr>
<tr>
<td>(Mult_{ws_{\text{me}}})</td>
<td><strong>5.93</strong></td>
<td><strong>5.83</strong></td>
<td>6.03</td>
<td><strong>5.83</strong></td>
<td>6.23</td>
<td>6.17</td>
</tr>
</tbody>
</table>

Table 7.6: Average of ranks of heuristics for inhibitory trees

<table>
<thead>
<tr>
<th>Heuristics</th>
<th>Decision trees</th>
<th>Inhibitory trees</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(h)</td>
<td>(h_{\text{avg}})</td>
</tr>
<tr>
<td>(wm_{\text{entML}})</td>
<td><strong>13.78</strong></td>
<td>13.42</td>
</tr>
<tr>
<td>(wm_{\text{entSort}})</td>
<td>16.44</td>
<td>11.16</td>
</tr>
<tr>
<td>(wm_{\text{abs}})</td>
<td><strong>15.67</strong></td>
<td>13.13</td>
</tr>
<tr>
<td>(wm_{\text{me}})</td>
<td>18.44</td>
<td>13.72</td>
</tr>
<tr>
<td>(ws_{\text{entML}})</td>
<td>31.56</td>
<td><strong>5.58</strong></td>
</tr>
<tr>
<td>(ws_{\text{entSort}})</td>
<td>28.89</td>
<td><strong>5.39</strong></td>
</tr>
<tr>
<td>(ws_{\text{abs}})</td>
<td>33.11</td>
<td>9.42</td>
</tr>
<tr>
<td>(ws_{\text{me}})</td>
<td><strong>14.89</strong></td>
<td>4.96</td>
</tr>
<tr>
<td>(Mult_{ws_{\text{entML}}})</td>
<td>68.33</td>
<td>42.00</td>
</tr>
<tr>
<td>(Mult_{ws_{\text{entSort}}})</td>
<td>57.44</td>
<td>40.50</td>
</tr>
<tr>
<td>(Mult_{ws_{\text{abs}}})</td>
<td>82.44</td>
<td>53.56</td>
</tr>
<tr>
<td>(Mult_{ws_{\text{me}}})</td>
<td>32.89</td>
<td>22.34</td>
</tr>
</tbody>
</table>

Table 7.7: Average relative difference (ARD)
is the best from the point of view of optimization of average depth and number of nodes for decision trees.

### 7.5 Decision Trees for Sorting

In theoretical investigations (see [63, 64, 65]), the problem of sorting usually means to sort a sequence of \( n \) pairwise different elements \( x_1, \ldots, x_n \) from a linearly ordered set. In this case, there is only one permutation \((p_1, \ldots, p_n)\) of the set \( \{1, \ldots, n\} \) such that \( x_{p_1} < \ldots < x_{p_n} \), and each comparison \( x_i : x_j \) of two elements has only two possible results: \( x_i < x_j \) and \( x_i > x_j \). Our aim is to find, for a given sequence \( x_1, \ldots, x_n \), the permutation \((p_1, \ldots, p_n)\) such that \( x_{p_1} < \ldots < x_{p_n} \).

For a given \( n \), we can construct the conventional decision table (decision table with one-valued decisions) \( T^2_{\text{sort}}(n) \), in which columns correspond to attributes \( x_i : x_j \),
$1 \leq i < j \leq n$, and rows are all possible tuples of values of these attributes for sequences of pairwise different elements $x_1, \ldots, x_n$. Each row is labeled with the corresponding permutation. The index 2 in the notation $T_{\text{sort}}^2(n)$ means that we consider two-valued attributes.

We denote by $h(T_{\text{sort}}^2(n))$ the minimum depth of a decision tree for the decision table $T_{\text{sort}}^2(n)$, by $h_{\text{avg}}(T_{\text{sort}}^2(n))$ – the minimum average depth of a decision tree for the decision table $T_{\text{sort}}^2(n)$, and by $L(T_{\text{sort}}^2(n))$ – the minimum number of nodes in a decision tree for the decision table $T_{\text{sort}}^2(n)$. Using results obtained in [63, 66, 67, 64, 65], we can fill in Table 7.10. Moreover, for $n = 2, \ldots, 8$, there exists a decision tree $\Gamma_n$ for the table $T_{\text{sort}}^2(n)$ such that $h(\Gamma_n) = h(T_{\text{sort}}^2(n))$, $h_{\text{avg}}(T_{\text{sort}}^2(n), \Gamma_n) = h_{\text{avg}}(T_{\text{sort}}^2(n))$, and $L(\Gamma_n) = L(T_{\text{sort}}^2(n))$. This decision tree is optimal relative to the depth, average depth, and number of nodes simultaneously.

<table>
<thead>
<tr>
<th>$n$</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h(T_{\text{sort}}^2(n))$</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>10</td>
<td>13</td>
<td>16</td>
</tr>
<tr>
<td>$h_{\text{avg}}(T_{\text{sort}}^2(n))$</td>
<td>1</td>
<td>2.6667</td>
<td>4.6667</td>
<td>6.9333</td>
<td>9.5778</td>
<td>12.384</td>
<td>15.381</td>
</tr>
<tr>
<td>$L(T_{\text{sort}}^2(n))$</td>
<td>3</td>
<td>11</td>
<td>47</td>
<td>239</td>
<td>1439</td>
<td>10079</td>
<td>80639</td>
</tr>
</tbody>
</table>

Table 7.10: Minimum depth, minimum average depth, and minimum number of nodes in decision trees for decision table $T_{\text{sort}}^2(n)$, $n = 2, \ldots, 8$

We now consider the case when it is possible to have equal elements in the sequence $x_1, \ldots, x_n$. In this case, it can be more than one permutation $(p_1, \ldots, p_n)$ such that $x_{p_1} \leq \ldots \leq x_{p_n}$, and each comparison $x_i : x_j$ of two elements can have three possible results: $x_i < x_j$, $x_i = x_j$, and $x_i > x_j$. Our aim is to find, for a given sequence $x_1, \ldots, x_n$, a permutation $(p_1, \ldots, p_n)$ such that $x_{p_1} \leq \ldots \leq x_{p_n}$.

For a given $n$, we can construct the decision table with many-valued decisions $T_{\text{sort}}^3(n)$, in which columns correspond to attributes $x_i : x_j$, $1 \leq i < j \leq n$, rows are all possible tuples of values of these attributes for sequences of elements $x_1, \ldots, x_n$ which can contain equal elements (decision table $T_{\text{sort}}^3(3)$ is depicted in Fig. 7.2).
Figure 7.2: Decision table with many-valued decisions, $T_{\text{sort}}^3(3)$

Each row is labeled with the set of corresponding permutations. The index 3 in the notation $h(T_{\text{sort}}^3(n))$ means that we consider three-valued attributes.

We denote by $h(T_{\text{sort}}^3(n))$ the minimum depth of a decision tree for the decision table $T_{\text{sort}}^3(n)$, by $h_{\text{avg}}(T_{\text{sort}}^3(n))$ – the minimum average depth of a decision tree for the decision table $T_{\text{sort}}^3(n)$, and by $L(T_{\text{sort}}^3(n))$ – the minimum number of nodes in a decision tree for the decision table $T_{\text{sort}}^3(n)$. The considered parameters for $n = 2, \ldots, 6$ can be found in Table 7.11.

<table>
<thead>
<tr>
<th>$n$</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h(T_{\text{sort}}^3(n))$</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td>$h_{\text{avg}}(T_{\text{sort}}^3(n))$</td>
<td>1</td>
<td>2.4615</td>
<td>4.1733</td>
<td>6.1479</td>
<td>8.3850</td>
</tr>
<tr>
<td>$L(T_{\text{sort}}^3(n))$</td>
<td>4</td>
<td>19</td>
<td>112</td>
<td>811</td>
<td>7024</td>
</tr>
</tbody>
</table>

Table 7.11: Minimum depth, minimum average depth, and minimum number of nodes in decision trees for decision table $T_{\text{sort}}^3(n)$, $n = 2, \ldots, 6$

To find these parameters, we studied bi-criteria optimization problem for the table $T_{\text{sort}}^3(n)$ for each pair of cost functions from the set $\{h, h_{\text{avg}}, L\}$. In each case, we obtained only one Pareto optimal point. It means that, for each pair of the considered cost functions, there exists a decision tree which is optimal relative to these functions.
simultaneously. Note that the directed acyclic graph \( \Delta(T^3_{sort}(6)) \) contains 209,527 nodes and 3,945,465 edges.

Comparing Table 7.10 (the first case) and Table 7.11 (the second case) we obtain that, for the second case, the minimum depth is the same as for the first case, the minimum average depth is less for \( n > 2 \), and the minimum number of nodes is greater than for the first case.

### 7.6 Decision Trees for Knowledge Representation

Decision trees are often used for knowledge representation. In this case, the usual goal is to minimize the number of nodes in the tree to make it more understandable. However, the depth and the average depth of the tree are also important: we need to understand conjunctions of conditions corresponding to paths in the tree from the root to terminal nodes.

To study trade-offs number of nodes vs. depth and number of nodes vs. average depth, we can construct the sets of Pareto optimal points for corresponding bi-criteria optimization problems.

We consider two decision tables with many-valued decisions \textsc{flags}-1 and \textsc{lymph}-5 (see Table 7.2). For these tables, we construct the sets of Pareto optimal points for bi-criteria optimization problems relative to the depth and number of nodes, and relative to the average depth and number of nodes.

When we consider bi-criteria optimization problem relative to the depth and number of nodes (see Fig. 7.3 (a)) for the table \textsc{flags}-1, we find three Pareto optimal points \((5,116)\), \((6,113)\), and \((7,112)\). The minimum depth of a decision tree with the minimum number of nodes 112 is equal to 7. Instead of such a tree, it is more reasonable to choose a decision tree which depth is equal to 5 and the number of nodes is equal to 116.

When we consider bi-criteria optimization problem relative to the average depth
and number of nodes (see Fig. 7.3 (b)) for the table flags-1, we find 29 Pareto optimal points. The minimum average depth of a decision tree with the minimum number of nodes 112 is equal to 3.9. Instead of such a tree, it is more reasonable to choose a decision tree which average depth is equal to 3.552632 and the number of nodes is equal to 113. There are other interesting possibilities. For example, a decision tree which average depth is equal to 2.878947 and the number of nodes is equal to 125. Note that the minimum possible average depth is equal to 2.768421 but, in this case, the minimum number of nodes is equal to 142.

When we consider bi-criteria optimization problem relative to the depth and number of nodes (see Fig. 7.3 (c)) for the table lymph-5, we find three Pareto optimal
points (5,64), (6,57), and (9,56). The minimum depth of a decision tree with the minimum number of nodes 56 is equal to 9. Instead of such a tree, it is more reasonable to choose a decision tree which depth is equal to 6 and the number of nodes is equal to 57.

When we consider bi-criteria optimization problem relative to the average depth and number of nodes (see Fig. 7.3 (d)) for the table LYMHP-5, we find 12 Pareto optimal points. The minimum average depth of a decision tree with the minimum number of nodes 56 is equal to 4.721312. Instead of such a tree, it is more reasonable to choose a decision tree which average depth is equal to 4.172131 and the number of nodes is equal to 57. We even can use a decision tree with the minimum average depth 3.647541. For such a tree, the minimum number of nodes is equal to 69.

Therefore, at the cost of a minor increase in the number of nodes, we can essentially decrease the depth or the average depth of decision trees.
Chapter 8

Bi-criteria Optimization Problem for Decision (Inhibitory) Trees: Cost vs. Uncertainty (Completeness)

In this chapter, we study bi-criteria optimization problems cost vs. uncertainty for decision trees (Sect. 8.1) and cost vs. completeness for inhibitory trees (Sect. 8.2), and consider illustrative examples (Sect. 8.3). The created tools allow us to understand complexity vs. accuracy trade-off for decision and inhibitory trees and to choose appropriate trees.

8.1 Bi-criteria Optimization Problem for Decision Trees: Cost vs. Uncertainty

In this section, we consider an algorithm which constructs the sets of Pareto optimal points for bi-criteria optimization problems for decision trees relative to cost and uncertainty. We also show how the constructed set of Pareto optimal points can be transformed into the graphs of functions which describe the relationships between the considered cost function and uncertainty measure. Some of the initial results in this direction for decision tables with one-valued decisions were obtained in [68, 69, 70].

8.1.1 Pareto Optimal Points: Cost vs. Uncertainty

First, we consider an algorithm for construction of the set of Pareto optimal points. Let $\psi$ be an increasing cost function for decision trees given by the triple of functions $\psi^0, F, w$, $U$ be an uncertainty measure, $T$ be a decision table with $n$ conditional
attributes $f_1, \ldots, f_n$, and $H \in \{\text{max}, \text{sum}\}$ where $\text{max} = \max(x, y)$ and $\text{sum} = x + y$.

For each node $\Theta$ of the graph $\Delta(T)$, we denote

$$t_{U,H,\psi}(\Theta) = \{(U^H(\Theta, \Gamma), \psi(\Theta, \Gamma)) : \Gamma \in \text{Tree}^*(\Delta(T), \Theta)\}$$

where $\text{Tree}^*(\Delta(T), \Theta)$ is, by Proposition[14] the set of decision trees for $\Theta$. We denote by $\text{Par}(t_{U,H,\psi}(\Theta))$ the set of Pareto optimal points for $t_{U,H,\psi}(\Theta)$.

We now describe an algorithm $\mathcal{A}_7$ which constructs the set $\text{Par}(t_{U,H,\psi}(T))$. In fact, this algorithm constructs, for each node $\Theta$ of the graph $\Delta(T)$, the set $B(\Theta) = \text{Par}(t_{U,H,\psi}(\Theta))$.

Algorithm $\mathcal{A}_7$.

Input: Increasing cost function for decision trees $\psi$ given by triple of functions $\psi^0, F, w$, an uncertainty measure $U$, a function $H \in \{\text{max}, \text{sum}\}$, a decision table $T$ with $n$ conditional attributes $f_1, \ldots, f_n$, and the graph $\Delta(T)$.

Output: The set $\text{Par}(t_{U,H,\psi}(T))$ of Pareto optimal points for the set of pairs $t_{U,H,\psi}(T) = \{(U^H(T, \Gamma), \psi(T, \Gamma)) : \Gamma \in \text{Tree}^*(\Delta(T), T)\}$.

1. If all nodes in $\Delta(T)$ are processed, then return the set $B(T)$. Otherwise, choose a node $\Theta$ in the graph $\Delta(T)$ which is not processed yet and which is either a terminal node of $\Delta(T)$ or a nonterminal node of $\Delta(T)$ 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 $\Theta$ is a terminal node, then set $B(\Theta) = \{(U(\Theta), \psi^0(\Theta))\}$. Mark the node $\Theta$ as processed and proceed to step 1.

3. If $\Theta$ is a nonterminal node then, for each $f_i \in E(\Theta)$, apply the algorithm $\mathcal{A}_3$ to the functions $H, F$ 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). \text{ Set } C(\Theta, f_i) \text{ the output of the algorithm } \mathcal{A}_3 \text{ and }

\begin{align*}
B(\Theta, f_i) &= C(\Theta, f_i) \langle ++ \rangle \{0, w(\Theta)\} \\
&= \{(a, b + w(\Theta)) : (a, b) \in C(\Theta, f_i)\}.
\end{align*}

4. Construct the multiset \(A(\Theta) = \{(U(\Theta), \psi^0(\Theta))\} \cup \bigcup_{f_i \in E_G(\Theta)} B(\Theta, f_i)\) 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 \(\text{Par}(A(\Theta))\). Set \(B(\Theta) = \text{Par}(A(\Theta))\). Mark the node \(\Theta\) as processed and proceed to step 1.

**Proposition 33.** Let \(\psi\) be an increasing cost function for decision trees given by triple of functions \(\psi^0, F, w, U\) be an uncertainty measure, \(H \in \{\text{max}, \text{sum}\}\), and \(T\) be a decision table with \(n\) conditional attributes \(f_1, \ldots, f_n\). Then, for each node \(\Theta\) of the graph \(\Delta(T)\), the algorithm \(\mathcal{A}_7\) constructs the set \(B(\Theta) = \text{Par}(t_{U,H,\psi}(\Theta))\).

**Proof.** We prove the considered statement by induction on nodes of \(\Delta(T)\). Let \(\Theta\) be a terminal node of \(\Delta(T)\). Then \(\text{Tree}^*(\Delta(T), \Theta) = \{\text{tree}(\text{mcd}(\Theta))\}, t_{U,H,\psi}(\Theta) = \{(U(\Theta), \psi^0(\Theta))\}\), and \(B(\Theta) = \text{Par}(t_{U,H,\psi}(\Theta))\).

Let \(\Theta\) be a nonterminal node of \(\Delta(T)\) 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., \(B(\Theta(f_i, a_j)) = \text{Par}(t_{U,H,\psi}(\Theta(f_i, a_j)))\).

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

\[
P(f_i) = \{(H(b_1, \ldots, b_t), F(c_1, \ldots, c_t) + w(\Theta)) : (b_j, c_j) \in t_{U,H,\psi}(\Theta(f_i, a_j)), j = 1, \ldots, t\}
\]

and, for \(j = 1, \ldots, t\), we denote \(P_j = t_{U,H,\psi}(\Theta(f_i, a_j))\).
If we apply the algorithm $A_3$ to the functions $H, F$ and the sets $\operatorname{Par}(P_1), \ldots, \operatorname{Par}(P_t)$, we obtain the set $\operatorname{Par}(Q_t)$ where $Q_1 = P_1$, and, for $j = 2, \ldots, t$, $Q_j = Q_{j-1} \langle HF \rangle P_j$.

It is not difficult to show that $P(f_i) = Q_t \langle ++ \rangle \{(0, w(\Theta))\} = \{(a, b + w(\Theta)) : (a, b) \in Q_t\}$ and $\operatorname{Par}(P(f_i)) = \operatorname{Par}(Q_t) \langle ++ \rangle \{(0, w(\Theta))\}$.

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

One can show that $t_{U, H, \psi}(\Theta) = \{(U(\Theta), \psi^0(\Theta))\} \cup \bigcup_{f_i \in E(\Theta)} P(f_i)$. By Lemma 9

\[
\operatorname{Par}(t_{U, H, \psi}(\Theta)) \equiv \operatorname{Par} \left( \{(U(\Theta), \psi^0(\Theta))\} \cup \bigcup_{f_i \in E(\Theta)} P(f_i) \right)
\]

\[
\subseteq \{(U(\Theta), \psi^0(\Theta))\} \cup \bigcup_{f_i \in E(\Theta)} \operatorname{Par}(P(f_i)).
\]

Using Lemma 8 we obtain

\[
\operatorname{Par}(t_{U, H, \psi}(\Theta)) = \operatorname{Par} \left( \{(U(\Theta), \psi^0(\Theta))\} \cup \bigcup_{f_i \in E(\Theta)} \operatorname{Par}(P(f_i)) \right).
\]

Since $B(\Theta, f_i) = \operatorname{Par}(P(f_i))$ for any $f_i \in E(\Theta)$, $\operatorname{Par}(t_{U, H, \psi}(\Theta)) = \operatorname{Par}(A(\Theta)) = B(\Theta)$.

We now analyze the number of elementary operations made by the algorithm $A_7$ during the construction of the set $\operatorname{Par}(t_{U, H, \psi}(T))$ for integral cost functions, for the cases when $F(x, y) = \max(x, y)$ ($h$ is an example of such cost function) and when $F(x, y) = x + y$ (.tpl, L, Ln, and L_t are examples of such cost functions).
Let us recall that $\text{range}(T) = \max\{|E(T, f_i)| : i = 1, \ldots, n\}$,

$$ub(\psi, T) = \max\{\psi(\Theta, \Gamma) : \Theta \in SEP(T), \Gamma \in DT(\Theta)\},$$

and $\psi(\Theta, \Gamma) \in\{0, 1, \ldots, ub(\psi, T)\}$ for any separable subtable $\Theta$ of $T$ and for any decision tree $\Gamma$ for $\Theta$. Upper bounds on the value $ub(\psi, T)$ for $\psi \in \{h, \text{tpl}, L, L_n, L_t\}$ are given in Lemma 13: $ub(h, T) \leq n$, $ub(\text{tpl}, T) \leq nN(T)$, $ub(L, T) \leq 2N(T)$, $ub(L_n, T) \leq N(T)$, and $ub(L_t, T) \leq N(T)$.

**Proposition 34.** Let $\psi$ be an increasing integral cost function for decision trees given by triple of functions $\psi^0, F, w$, $F \in \{\max(x, y), x + y\}$, $U$ be an uncertainty measure, $H \in \{\max, \text{sum}\}$, and $T$ be a decision table with $n$ conditional attributes $f_1, \ldots, f_n$. Then, to construct the set $\text{Par}(t_{U,H,\psi}(T))$, the algorithm $A_7$ makes

$$O(L(\Delta(T))\text{range}(T)ub(\psi, T)^2n \log(ub(\psi, T)n))$$

elementary operations (computations of $\psi^0$, $F$, $w$, $H$, $U$, comparisons and additions) if $F = \max(x, y)$, and

$$O(L(\Delta(T))\text{range}(T)^2ub(\psi, T)^2n \log(\text{range}(T)ub(\psi, T)n))$$

elementary operations (computations of $\psi^0$, $F$, $w$, $H$, $U$, comparisons and additions) if $F = x + y$.

**Proof.** It is clear that, for any node $\Theta$ of the graph $\Delta(T)$,

$$t_{U,H,\psi}(\Theta)^{(2)} = \{b : (a, b) \in t_{U,H,\psi}(\Theta)\} \subseteq \{0, \ldots, ub(\psi, T)\}.$$ 

From Proposition 33 it follows that $B(\Theta) = \text{Par}(t_{U,H,\psi}(\Theta))$ for any node $\Theta$ of the graph $\Delta(T)$. Therefore, for any node $\Theta$ of the graph $\Delta(T)$, $B(\Theta)^{(2)} = \{b : (a, b) \in$
Let $F(x, y) = \max(x, y)$. To process a terminal node $\Theta$, the algorithm $A_7$ makes two elementary operations (computations of $\psi^0$ and $U$).

We now consider the processing of a nonterminal node $\Theta$ (see description of the algorithm $A_7$). We know that $B(\Theta(f_i, a_j))^{(2)} \subseteq \{0, \ldots, ub(\psi, T)\}$ for $j = 1, \ldots, t$, and $t \leq \text{range}(T)$. From Proposition [12] it follows that $|C(\Theta, f_i)| \leq ub(\psi, T) + 1$, and to construct the set $C(\Theta, f_i)$, the algorithm $A_3$ makes

$$O(\text{range}(T)ub(\psi, T)^2 \log ub(\psi, T))$$

elementary operations (computations of $F$, $H$ and comparisons). To construct the set $B(\Theta, f_i)$ from the set $C(\Theta, f_i)$, the algorithm makes a computation of $w$ and at most $ub(\psi, T) + 1$ additions. From here it follows that, to construct the set $B(\Theta, f_i)$, the algorithm makes

$$O(\text{range}(T)ub(\psi, T)^2 \log ub(\psi, T))$$

elementary operations. It is clear that $|E(\Theta)| \leq n$. Therefore, to construct the set $B(\Theta, f_i)$ for each $f_i \in E(\Theta)$, the algorithm makes

$$O(\text{range}(T)ub(\psi, T)^2 n \log ub(\psi, T))$$

elementary operations.

Since $|C(\Theta, f_i)| \leq ub(\psi, T) + 1$, we have $|B(\Theta, f_i)| \leq ub(\psi, T) + 1$. Since

$$|E(\Theta)| \leq n,$$

we have $|A(\Theta)| \leq n(ub(\psi, T) + 1) + 1$ where

$$A(\Theta) = \{(U(\Theta), \psi^0(\Theta))\} \cup \bigcup_{f_i \in E(\Theta)} B(\Theta, f_i).$$
From Proposition 10 it follows that, to construct the set \( B(\Theta) = \text{Par}(A(\Theta)) \), the algorithm \( A_2 \) makes \( O(\text{ub}(\psi, T)n \log(\text{ub}(\psi, T)n)) \) comparisons. So, to process a non-terminal node \( \Theta \) (to construct \( B(\Theta) = \text{Par}(t_{U,H,\psi}(\Theta)) \)) if \( B(\Theta(f_i, a_j)) = \text{Par}(t_{U,H,\psi}(\Theta(f_i, a_j))) \) is known for all \( f_i \in E(\Theta) \) and \( a_j \in E(\Theta, f_i) \), the algorithm \( A_7 \) makes

\[
O(\text{range}(T)\text{ub}(\psi, T)^2n \log(\text{ub}(\psi, T)n))
\]
elementary operations.

To construct the set \( \text{Par}(t_{U,H,\psi}(T)) \) for given decision table \( T \) with \( n \) conditional attributes and the graph \( \Delta(T) \), it is enough to make

\[
O(L(\Delta(T))\text{range}(T)\text{ub}(\psi, T)^2n \log(\text{ub}(\psi, T)n))
\]
elementary operations (computations of \( \psi^0, F, w, H, U \), comparisons and additions).

Let \( F(x, y) = x + y \). To process a terminal node \( \Theta \), the algorithm \( A_7 \) makes two elementary operations (computations of \( \psi^0 \) and \( U \)).

We now consider the processing of a nonterminal node \( \Theta \) (see description of the algorithm \( A_7 \)). We know that \( B(\Theta(f_i, a_j))^{(2)} \subseteq \{0, \ldots, \text{ub}(\psi, T)\} \) for \( j = 1, \ldots, t \), and \( t \leq \text{range}(T) \). From Proposition 12 it follows that \( |C(\Theta, f_i)| \leq t \times \text{ub}(\psi, T) + 1 \leq \text{range}(T)\text{ub}(\psi, T) + 1 \), and to construct the set \( C(\Theta, f_i) \), the algorithm \( A_3 \) makes

\[
O(\text{range}(T)^2\text{ub}(\psi, T)^2 \log(\text{range}(T)\text{ub}(\psi, T)))
\]
elementary operations (computations of \( F, H \) and comparisons). To construct the set \( B(\Theta, f_i) \) from the set \( C(\Theta, f_i) \), the algorithm makes a computation of \( w \) and at most \( \text{range}(T)\text{ub}(\psi, T) + 1 \) additions. From here it follows that, to construct the set
B(Θ, f_i) the algorithm makes

\[ O(range(T)^2 ub(\psi, T)^2 \log(range(T)ub(\psi, T))) \]

elementary operations. It is clear that |E(Θ)| ≤ n. Therefore, to construct the set
B(Θ, f_i) for each f_i ∈ E(Θ), the algorithm makes

\[ O(range(T)^2 ub(\psi, T)^2 n \log(range(T)ub(\psi, T))) \]

elementary operations.

Since |C(Θ, f_i)| ≤ range(T)ub(\psi, T) + 1, we have

\[ |B(Θ, f_i)| ≤ range(T)ub(\psi, T) + 1. \]

Since |E(Θ)| ≤ n, we have |A(Θ)| ≤ (range(T)ub(\psi, T) + 1)n + 1 where A(Θ) =
\{(U(Θ), \psi^0(Θ))}\∪\bigcup_{f_i \in E(Θ)} B(Θ, f_i). From Proposition 10 it follows that, to construct
the set B(Θ) = Par(A(Θ)), the algorithm A_2 makes

\[ O(range(T)ub(\psi, T)n \log(range(T)ub(\psi, T)n)) \]

comparisons. So, to process a nonterminal node Θ (to construct

\[ B(Θ) = Par(t_{U,H,\psi}(Θ)) \]

if B(Θ(f_i, a_j)) = Par(t_{U,H,\psi}(Θ(f_i, a_j))) is known for all f_i ∈ E(Θ) and a_j ∈ E(Θ, f_i),
the algorithm A_7 makes

\[ O(range(T)^2 ub(\psi, T)^2 n \log(range(T)ub(\psi, T)n)) \]
elementary operations.

To construct the set $Par(t_{U,H,\psi}(T))$ for given decision table $T$ with $n$ conditional attributes and the graph $\Delta(T)$, it is enough to make

$$O(L(\Delta(T))\text{range}(T)^2\text{ub}(\psi,T)^2n\log(\text{range}(T)\text{ub}(\psi,T)n))$$

elementary operations (computations of $\psi^0$, $F$, $w$, $H$, $U$, comparisons and additions).

\[ \square \]

**Proposition 35.** Let $\psi$ be a cost function for decision trees from the set $\{h,tpl,\newline L, L_n, L_t\}$ given by triple of functions $\psi^0, F, w, U$ be an uncertainty measure from the set $\{\text{me, rme, abs}\}$, $H \in \{\text{max, sum}\}$, and $U$ be a restricted information system. Then the algorithm $A_7$ has polynomial time complexity for decision tables from $T(U)$ depending on the number of conditional attributes in these tables.

**Proof.** Since $\psi \in \{h,tpl, L, L_n, L_t\}$, $\psi^0$ is a constant, the function $F$ is either $\max(x,y)$ or $x+y$, and the function $w$ is either a constant or $N(T)$. From Proposition 34 and Lemma 13, it follows that, for the algorithm $A_7$, the number of elementary operations (computations of $\psi^0$, $F$, $w$, $H$, $U$, comparisons and additions) is bounded from above by a polynomial depending on the size of input table $T$ and on the number of separable subtables of $T$. All operations with numbers are basic ones. The computations of numerical parameters of decision tables used by the algorithm $A_7$ (constants, $N(T)$, and $U(T)$) have polynomial time complexity depending on the size of decision tables.

According to Proposition 9, the algorithm $A_7$ has polynomial time complexity for decision tables from $T(U)$ depending on the number of conditional attributes in these tables.

\[ \square \]

We now consider the problem of construction of the set of Pareto optimal points for the sets $t^{\max}_{U,\psi}(T) = \{(U^{\max}(T,\Gamma),\psi(T,\Gamma)) : \Gamma \in DT^{\max}_{U}(T)\}$ and $t^{\text{sum}}_{U,\psi}(T) = \{(U^{\text{sum}}(T,\Gamma),\psi(T,\Gamma)) : \Gamma \in DT^{\text{sum}}_{U}(T)\}$. Let $\psi$ be bounded and increasing cost
function for decision trees (in particular, $h$, $tpl$, $L$, $L_n$, and $L_t$ are bounded and increasing cost functions). Then, by Lemma 6 and Proposition 24, $Par(t_{U,\psi}^{\max}(T)) = Par(t_{U,\max,\psi}(T))$. By Lemma 6 and Proposition 25,

$$Par(t_{U,\psi}^{\text{sum}}(T)) = Par(t_{U,\text{sum},\psi}(T)).$$

So to construct $Par(t_{U,\psi}^{\max}(T))$ and $Par(t_{U,\psi}^{\text{sum}}(T))$ we can use the algorithm $A_7$.

### 8.1.2 Relationships between Cost and Uncertainty

We now show how to construct graphs describing relationships between the cost and uncertainty of decision trees.

Let $\psi$ be bonded and increasing cost function for decision trees, $U$ be an uncertainty measure, $H \in \{\max(x,y), \text{sum}(x,y)\}$, and $T$ be a decision table.

To study relationships between cost function $\psi$ and uncertainty $U^H$ for decision trees on the set of decision trees $DT_U^H(T)$ we consider partial functions $T_{T,H}^{U,\psi} : \mathbb{R} \to \mathbb{R}$ and $T_{T,H}^{\psi,U} : \mathbb{R} \to \mathbb{R}$ defined as follows:

$$T_{T,H}^{U,\psi}(x) = \min\{\psi(T,\Gamma) : \Gamma \in DT_U^H(T), U^H(T,\Gamma) \leq x\},$$

$$T_{T,H}^{\psi,U}(x) = \min\{U^H(T,\Gamma) : \Gamma \in DT_U^H(T), \psi(T,\Gamma) \leq x\}.$$

Let $(a_1, b_1), \ldots, (a_k, b_k)$ be the normal representation of the set

$$Par(t_{U,\psi}^H(T)) = Par(t_{U,\psi}(T)).$$
where $a_1 < \ldots < a_k$ and $b_1 > \ldots > b_k$. By Lemma 11 and Remark 4 for any $x \in \mathbb{R}$,

$$
\mathcal{T}^{U,\psi}_{T,H}(x) = \begin{cases}
\text{undefined}, & x < a_1 \\
b_1, & a_1 \leq x < a_2 \\
\ldots & \ldots \\
b_{k-1}, & a_{k-1} \leq x < a_k \\
b_k, & a_k \leq x
\end{cases}
$$

$$
\mathcal{T}^{\psi,U}_{T,H}(x) = \begin{cases}
\text{undefined}, & x < b_k \\
a_k, & b_k \leq x < b_{k-1} \\
\ldots & \ldots \\
a_2, & b_2 \leq x < b_1 \\
a_1, & b_1 \leq x
\end{cases}
$$

8.2 Bi-criteria Optimization Problem for Inhibitory Trees: Cost vs. Completeness

In this section, we discuss bi-criteria optimization problem cost vs. completeness for inhibitory trees.

Let $T$ be a nondegenerate decision table with $n$ conditional attributes $f_1, \ldots, f_n$, $T^C$ be the decision table complementary to $T$, $U$ be an uncertainty measure, $W$ be a completeness measure, $U$ and $W$ are dual, $\psi \in \{h, tpl, L, L_n, L_t\}$, and $H \in \{\text{max}, \text{sum}\}$.

In Section 8.1 the algorithm $\mathcal{A}_7$ is described which constructs the set of Pareto optimal points $Par(t_{U,H,\psi}(T^C))$ for the set of points

$$
t_{U,H,\psi}(T^C) = \{(U^H(T^C, \Gamma), \psi(T^C, \Gamma)) : \Gamma \in DT(T^C)\}.
$$

We proved at the end of Section 8.1.1 that $Par(t_{U,H,\psi}(T^C)) = Par(t_{U,\psi}(T^C))$ where
\[ t^H_{U,\psi}(T_C) = \{(U^H(T_C, \Gamma), \psi(T_C, \Gamma)) : \Gamma \in DT^H_U(T_C)\}. \]

We denote \( it^{W,H,\psi}_{T} = \{(W^H(T, \Gamma), \psi(T, \Gamma)) : \Gamma \in IT(T)\} \) and \( it^{H,W,\psi}_{T} = \{(W^H(T, \Gamma), \psi(T, \Gamma)) : \Gamma \in IT^W_H(T)\} \). By Proposition 26, for any \( \Gamma \in DT(T_C) \), \((U^H(T_C, \Gamma), \psi(T_C, \Gamma)) = (W^H(T, \Gamma^C), \psi(T, \Gamma^C))\). From Corollary 10 it follows that \( DT(T_C) = IT(T) \) and \( DT^H_U(T_C) = IT^H_W(T) \). Therefore \( t_{U,H,\psi}(T_C) = it_{W,H,\psi}(T) \) and \( t^H_{U,\psi}(T_C) = it^{H,W,\psi}_{T}(T) \). Hence \( Par(it_{W,H,\psi}(T)) = Par(it^{H,W,\psi}_{T}(T)) = Par(t_{U,H,\psi}(T_C)) \).

To study relationships between cost function \( \psi \) and completeness \( W^H \) for inhibitory trees on the set of inhibitory trees \( IT^H_W(T) \) we consider partial functions \( IT^W_{T,H} : \mathbb{R} \to \mathbb{R} \) and \( IT^T_{W,H} : \mathbb{R} \to \mathbb{R} \) defined as follows:

\[
\begin{align*}
\mathcal{I}T^W_{T,H}(x) &= \min\{\psi(T, \Gamma) : \Gamma \in IT^H_W(T), W^H(T, \Gamma) \leq x\}, \\
\mathcal{I}T^T_{W,H}(x) &= \min\{W^H(T, \Gamma) : \Gamma \in IT^H_W(T), \psi(T, \Gamma) \leq x\}.
\end{align*}
\]

Let \((a_1, b_1), \ldots, (a_k, b_k)\) be the normal representation of the set

\[ Par(it^H_{W,\psi}(T)) = Par(t_{U,H,\psi}(T_C)) \]

where \( a_1 < \ldots < a_k \) and \( b_1 > \ldots > b_k \). By Lemma and Remark 4, for any \( x \in \mathbb{R} \),

\[
\mathcal{I}T^W_{T,H}(x) = \begin{cases} 
\text{undefined}, & x < a_1 \\
b_1, & a_1 \leq x < a_2 \\
\ldots & \ldots \\
b_{k-1}, & a_{k-1} \leq x < a_k \\
b_k, & a_k \leq x
\end{cases}
\]
8.3 Illustrative Examples

To illustrate the use of tools developed in this chapter, we consider some experimental results for the decision table with many-valued decisions cars-1 containing 432 rows and 5 conditional attributes. For this decision table, we describe how depth $h$, average depth $h_{avg}$, and the number of nodes $L$ of decision trees depend on the number of misclassifications $me^{sum}$ which is a kind of decision tree uncertainty. We also describe how depth $h$, average depth $h_{avg}$, and the number of nodes $L$ of inhibitory trees depend on the number of misclassifications $ime^{sum}$ which is a kind of inhibitory tree completeness. Corresponding relationships are shown in Figs. 8.1, 8.2, and 8.3. Filled circles in these figures are Pareto optimal points for the considered bi-criteria optimization problems cost vs. uncertainty for decision and cost vs. completeness for inhibitory trees.

Such relationships can be useful for the choice of the decision and inhibitory trees.
which represent knowledge contained in the decision table: a tree with enough small number of nodes and reasonable uncertainty (completeness) can be considered as a good model of data. A similar situation is with the choice of the decision and inhibitory trees that are considered as algorithms. In this case, we are interested in finding of a tree with small depth or average depth and reasonable uncertainty (completeness).

The similar approach can be useful in machine learning (see Chapter 9).
Chapter 9

Multi-pruning and Restricted Multi-pruning of Decision Trees

In this chapter, we develop an approach to constructing decision trees which can be used for knowledge representation and classification. This approach applies to decision tables with both categorical and numerical conditional attributes. It is based on the consideration of a large set of CART (Classification and Regression Trees)-like decision trees for the considered decision table. In particular, for the data set BREAST-CANCER containing 266 rows and 10 conditional attributes, the number of CART-like decision trees is equal to $1.42 \times 10^{193}$. Such trees use binary splits created on the base of conditional attributes. In contrast with standard CART [18] which use the best (from the point of view of impurity based on Gini index $gini$) splits among all attributes, CART-like trees use, additionally, the best splits for each attribute.

Created algorithms allow us to describe, for the considered set of CART-like decision trees given by a DAG, the set of Pareto optimal points for bi-criteria optimization problem relative to the number of nodes and the number of misclassifications. For each Pareto optimal point $(a, b)$, we can derive the decision trees for which the number of misclassifications is equal to $a$, and the number of nodes is equal to $b$.

Since the considered set of CART-like decision trees is closed under the operation of usual bottom-up pruning, we call the considered approach to the construction of decision trees for knowledge representation and classification multi-pruning (MP). The initial study of multi-pruning approach was done in [17].

We consider two applications of multi-pruning approach. The first application
is connected with knowledge representation. We use the initial decision table to
build the DAG and the set of Pareto optimal points. Then we choose a suitable
Pareto optimal point and derive the corresponding decision tree (we can choose, for
example, a point with the minimum Euclidean distance from the origin point (0, 0)).
If this tree has a relatively small number of nodes and relatively small number of
misclassifications, it can be considered as understandable and enough accurate model
for the knowledge contained in the table. Results of experiments with decision tables
from UCI ML Repository [59] show that we can construct such decision trees in many
cases.

Another application is connected with machine learning. We divide the initial
table into three subtables: training, validation, and testing. We use training subtable
to build the DAG and the set of Pareto optimal points. We derive randomly some
decision trees (five in our experiments) for each Pareto optimal point and find, based
on the validation subtable, a decision tree with the minimum number of misclassifi-
cations among all derived trees. We evaluate the accuracy of prediction for this tree
using testing subtable. We compare this process with the CART which continues to
be one of the best algorithms for the construction of classification trees. The classifiers
constructed by the process often have better accuracy than the classifiers constructed
by CART. The considered process is similar to the usual pruning of a decision tree
but it is applied here to many decision trees since the set of CART-like decision trees
is closed under the operation of usual bottom-up pruning.

The multi-pruning approach applies to medium-sized decision tables and can be
used as a research tool. To make this approach more scalable we consider restricted
multi-pruning approach (RMP) [17] where we use in each node of DAG only the
best splits for a small number of attributes, instead of using the best splits for all
attributes. The obtained DAGs with limited branching factor contain fewer nodes
and edges and require less time for the construction and processing. However, the
number of nodes and number of misclassifications of decision trees constructed by
the restricted multi-pruning approach are comparable to the case when we use the
best splits for all attributes. The prediction accuracy of decision trees constructed by
RMP approach is comparable to the case when we use MP approach. For example,
we did experiments with 15 decision tables from UCI ML Repository [59]. For 11
tables, the decision trees constructed by the multi-pruning approach which uses best
splits for all attributes outperform the trees built by CART. The same situation is
with decision trees constructed by the restricted multi-pruning approach which uses
only best splits for two attributes.

We extend multi-pruning and restricted multi-pruning approaches to the case of
decision tables with many-valued decisions. We cannot use gini uncertainty mea-
sure (Gini index) for the decision tables with many-valued decisions. Instead of gini
we use the uncertainty measure abs which works for both decision tables with single-
and many-valued decisions. The experimental results show that abs performs as good
as gini for decision tables with one-valued decisions. We also compared the multi-
pruning and restricted multi-pruning approaches in the case of decision tables with
many-valued decisions and found that restricted multi-pruning approach is compara-
ble with the multi-pruning approach.

We find the simplest variant of restricted multi-pruning approach which can work
successfully for both knowledge representation and classification and for both decision
tables with one-valued decisions and for decision tables with many-valued decisions.
In previous paper [17], we only tried to find such variant for the problem of clas-
sification and for the decision tables with one-valued decisions. In this chapter, we
found an appropriate variant of restricted multi-pruning approach which uses only
best splits for two attributes and uncertainty measure abs.

Note that this chapter is based on two papers: (1) “Multi-pruning of decision trees
for knowledge representation and classification” by Mohammad Azad, Igor Chikalov,
9.1 Decision Tables

A decision table consists of rows (objects) and columns (conditional attributes). Usually, each row is labeled with a single decision. We call such decision tables decision tables with one-valued decisions (see Fig. 9.1(a)). There is another type of decision tables where each row is labeled with a set of decisions instead of single decision. We call such decision tables decision tables with many-valued decisions (see Fig. 9.1(b)). Decision tables with one-valued decisions can be considered as a special case of decision tables with many-valued decisions.

In case of decision tables with one-valued decisions, the minimum decision which is attached to the maximum number of rows of the decision table $T$ is called the most common decision for $T$. For example, the most common decision for the decision table $T_{svd}$ is 1 (see Fig. 9.1(a)). On the other hand, in the case of decision tables with many-valued decisions, the minimum decision which belongs to the maximum number of sets of decisions attached to rows of the table $T$ is called the most common decision for $T$. For example, the most common decision for the decision table $T_{mvd}$
\[
\begin{array}{ccc|c}
T_{svd} & f_1 & f_2 & f_3 \\
0 & 0 & 0 & 1 \\
0 & 1 & 1 & 2 \\
1 & 0 & 1 & 1 \\
1 & 1 & 0 & 3 \\
0 & 0 & 1 & 2 \\
\end{array}
\]
\[
\begin{array}{ccc|c}
T_{mvd} & f_1 & f_2 & f_3 \\
0 & 0 & 0 & \{1\} \\
0 & 1 & 1 & \{1,2\} \\
1 & 0 & 1 & \{1,3\} \\
1 & 1 & 0 & \{2,3\} \\
0 & 0 & 1 & \{2\} \\
\end{array}
\]

(a) Decision table with one-valued decisions  
(b) Decision table with many-valued decisions

\[
\begin{array}{ccc|c}
T'_{svd} & f_1 & f_2 & f_3 \\
0 & 0 & 0 & 1 \\
1 & 0 & 1 & 1 \\
\end{array}
\]
\[
\begin{array}{ccc|c}
T'_{mvd} & f_1 & f_2 & f_3 \\
0 & 0 & 0 & \{1\} \\
0 & 1 & 1 & \{1,2\} \\
1 & 0 & 1 & \{1,3\} \\
\end{array}
\]

(a) Decision table with one-valued decisions  
(b) Decision table with many-valued decisions

Figure 9.1: Examples of decision tables

Figure 9.2: Examples of decision tables with common decisions

is 1 (see Fig. 9.1(b)).

In the case of decision tables with one-valued decisions, if there is a decision attached to each row of \( T \), then we call it the common decision for \( T \). On the other hand, in the case of decision tables with many-valued decisions, if there is a decision which, for each row of \( T \), belongs to the set of decisions attached to this row, then we call it a common decision for \( T \). For example, 1 is the common decision for both tables \( T'_{svd} \) and \( T'_{mvd} \) in Fig. 9.2.

An uncertainty measure \( U \) is a function from a set of decision tables to the set of real numbers such that \( U(T) \geq 0 \), and \( U(T) = 0 \) if and only if \( T \) has a common decision. We use two uncertainty measures in this chapter: \( gini \) and \( abs \). These uncertainty measures are equal to 0 for each empty decision table.

Let \( T \) be a nonempty table having \( n \) conditional attributes, \( N = N(T) \) rows, and its rows be labeled with \( m \) different decisions \( d_1, \ldots, d_m \) (in the case of decision tables with one-valued decisions) or sets containing \( m \) different decisions \( d_1, \ldots, d_m \) (in the
case of decision tables with many-valued decisions). For \( i = 1, \ldots, m \), let \( N_i \) be the number of rows in \( T \) that are labeled with decision \( d_i \) (in the case of decision tables with one-valued decisions) or with sets of decisions containing the decision \( d_i \) (in the case of decision tables with many-valued decisions), and \( p_i = \frac{N_i}{N} \). Then

\[
\hat{\text{gini}}(T) = 1 - \sum_{i=1}^{m} p_i^2 \quad \text{(this uncertainty measure, Gini index, is applicable to decision tables with one-valued decisions)},
\]

\[
\hat{\text{abs}}(T) = \prod_{i=1}^{m} q_i \quad \text{where } q_i = 1 - p_i \quad \text{(this uncertainty measure, absence, is applicable to all decision tables)}.
\]

9.2 \((m_1, m_2, U)\)-Decision Trees

In this section, we discuss the notion of \((m_1, m_2, U)\)-decision trees. Such trees use the same type of binary splits as considered in CART \[18\]. We call such trees \(\text{CART-like decision trees}$. Let \( T \) be a decision table with \( n \) conditional attributes \( f_1, \ldots, f_n \) which are categorical or numerical, and with a categorical decision attribute \( d \) that has either a single decision for each row in the case of decision table with one-valued decisions or a set of decisions for each row in the case of decision table with many-valued decisions.

Let \( \Theta \) be a subtable of the table \( T \) obtained from \( T \) by removal of some rows. Instead of conditional attributes we use (as in CART) binary splits (binary attributes) each of which is based on a conditional attribute \( f_i \). If \( f_i \) is a categorical attribute with the set of values \( B \), then we consider a partitioning of \( B \) into two nonempty subsets \( B_0 \) and \( B_1 \). The value of the corresponding binary split \( s \) is equal to 0 if the value of \( f_i \) belongs to \( B_0 \), and 1 otherwise. We consider splits for all possible partitioning. If \( f_i \) is a numerical attribute then we consider a real threshold \( \alpha \). The value of corresponding binary split \( s \) is equal to 0 if the value of \( f_i \) is less than \( \alpha \), and 1 otherwise. Each binary split \( s \) divides the table \( \Theta \) into two subtables \( \Theta_{s=0} \) and
\( \Theta_{s=1} \) according to the values of \( s \) on rows of \( \Theta \). We consider the splits for all possible thresholds (middle point between two adjacent values).

Let \( U \) be an uncertainty measure: if \( T \) is a decision table with one-valued decisions then \( U \in \{ \text{gini}, \text{abs} \} \), and if \( T \) is a decision table with many-valued decisions, then \( U = \text{abs} \). The impurity \( I_U(\Theta, s) \) of the split \( s \) is equal to the weighted sum of uncertainties of subtables \( \Theta_{s=0} \) and \( \Theta_{s=1} \), where the weights are proportional to the number of rows in subtables \( \Theta_{s=0} \) and \( \Theta_{s=1} \), respectively. The impurity of splits is considered as a quality measure for splits where small impurity is better. A split \( s \) based on attribute \( f_i \) with minimum impurity \( I_U(\Theta, s) \) is called the best split for \( U, \Theta \) and \( f_i \).

Let \( m_1 \) and \( m_2 \) be nonnegative integers such that \( 0 < m_1 + m_2 \leq n \). We now describe a way for construction of a set \( S_{m_1,m_2,U}(\Theta) \) of admissible splits for the subtable \( \Theta \). Let \( E(\Theta) \) be the set of all conditional attributes which are not constant on \( \Theta \), and \( |E(\Theta)| = p \). For each attribute \( f_i \in E(\Theta) \), we find a best split for \( U, \Theta \) and the attribute \( f_i \). Let \( s_1, \ldots, s_p \) be the obtained splits in order from the best to the worst. If \( m_1 \geq p \) then \( S_{m_1,m_2,U}(\Theta) = \{ s_1, \ldots, s_p \} \). Let \( m_1 < p \). Then \( S_{m_1,m_2,U}(\Theta) \) contains splits \( s_1, \ldots, s_{m_1} \) and \( \min(p - m_1, m_2) \) splits randomly chosen from the set \( \{ s_{m_1+1}, \ldots, s_p \} \).

We consider \((m_1, m_2, U)\)-decision trees for \( T \) in which each leaf is labeled with a decision (value of the decision attribute \( d \)), each internal node is labeled with a binary split corresponding to one of the conditional attributes, and two outgoing edges from this node are labeled with 0 and 1, respectively. We correspond to each node \( v \) of a decision tree \( \Gamma \) a subtable \( T(\Gamma, v) \) of \( T \) that contains all rows of \( T \) for which the computation of \( \Gamma \) passes through the node \( v \). We assume that, for each internal node \( v \), the subtable \( T(\Gamma, v) \) has no any common decisions, and the node \( v \) is labeled with a split from \( S_{m_1,m_2,U}(T(\Gamma, v)) \). We also assume that, for each leaf \( v \), the node \( v \) is labeled with the most common decision for \( T(\Gamma, v) \).
9.3 DAG $G_{m_1,m_2,U}(T)$

In this section, we study a directed acyclic graph $G_{m_1,m_2,U}(T)$ which is used to describe the set of $(m_1, m_2, U)$-decision trees for $T$. Nodes of the graph $G_{m_1,m_2,U}(T)$ are some subtables of the table $T$. We now describe an algorithm for the construction of the directed acyclic graph $G_{m_1,m_2,U}(T)$ which can be considered as a definition of this DAG.

Algorithm $A_{DAG}$ (Construction of the DAG $G_{m_1,m_2,U}(T)$).

Input: A decision table $T$ with $n$ conditional attributes, an uncertainty measure $U$ (if $T$ is a decision table with one-valued decisions, then $U \in \{gini, abs\}$, and if $T$ is a decision table with many-valued decisions then $U = abs$), and nonnegative integers $m_1$ and $m_2$ such that $0 < m_1 + m_2 \leq n$.

Output: The DAG $G_{m_1,m_2,U}(T)$.

1. Construct a graph which contains only one node $T$ which is marked as not processed.

2. If all nodes of the graph are processed, then return it as $G_{m_1,m_2,U}(T)$ and finish. Otherwise, choose a node (subtable) $\Theta$ which is not processed yet.

3. If $\Theta$ has a common decision, mark $\Theta$ as processed and proceed to step 2.

4. Otherwise, construct set $S_{m_1,m_2,U}(\Theta)$ of admissible splits for $\Theta$ and, for each split $s$ from $S_{m_1,m_2,U}(\Theta)$, draw two edges from $\Theta$ to subtables $\Theta_{s=0}$ and $\Theta_{s=1}$, and label these edges with $s = 0$ and $s = 1$, respectively (this pair of edges is called an $s$-pair). If some of the subtables $\Theta_{s=0}$ and $\Theta_{s=1}$ are not in the graph, add them to the graph. Mark $\Theta$ as processed and proceed to step 2.

One can show that the time complexity of this algorithm is bounded from above by a polynomial in the size of decision table $T$ and the number of nodes in the graph $G_{m_1,m_2,U}(T)$. 

178
We correspond to each node Θ of the graph \( G_{m_1,m_2,U}(T) \) a set of decision trees \( DT_{m_1,m_2,U}(Θ) \). We denote by \( \text{tree}(Θ) \) the decision tree with exactly one node labeled with the most common decision for Θ. If Θ has a common decision then \( DT_{m_1,m_2,U}(Θ) \) contains only one tree \( \text{tree}(Θ) \). Otherwise, \( DT_{m_1,m_2,U}(Θ) \) contains \( \text{tree}(Θ) \) and all trees of the following kind: the root of tree is labeled with a split \( s \) such that an \( s \)-pair of edges starts in \( Θ \), two edges start in the root which are labeled with 0 and 1 and enter to the roots of decision trees from \( DT_{m_1,m_2,U}(Θ_s=0) \) and \( DT_{m_1,m_2,U}(Θ_s=1) \), respectively.

Note that the set \( DT_{m_1,m_2,U}(T) \) is closed under the operation of usual bottom-up pruning of decision trees. One can prove that the set of decision trees \( DT_{m_1,m_2,U}(T) \) coincides with the set of all \( (m_1, m_2, U) \)-decision trees for \( T \) if, for any subtable Θ of \( T \), we consider the same set \( S_{m_1,m_2,U}(Θ) \) in the definition of the set \( DT_{m_1,m_2,U}(T) \), and in the description of DAG \( G_{m_1,m_2,U}(T) \).

### 9.4 Set of Pareto Optimal Points \( POP_{m_1,m_2,U}(T) \)

Let \( A \) be a finite set of points in two-dimensional Euclidean space. A point \((a, b) \in A\) is called a Pareto optimal point (POP) for \( A \) if there is no point \((c, d) \in A\) such that \((a, b) \neq (c, d), c \leq a, \text{ and } d \leq b\). We denote by \( \text{Par}(A) \) the set of Pareto optimal points for \( A \). It is easy to construct the set \( \text{Par}(A) \) using \( O(|A| \log |A|) \) comparisons (see the algorithm \( A_2 \)).

Let \( Γ \) be a decision tree from \( DT_{m_1,m_2,U}(T) \). We denote by \( L(Γ) \) the number of nodes in \( Γ \) and by \( mc(T, Γ) \) the number of misclassifications of \( Γ \) on rows of \( T \), i.e., the number of rows of \( T \) for which the work of \( Γ \) ends in a leaf that is labeled with either a decision different from the decision attached to the row in the case of decision tables with one-valued decisions, or a decision which is not contained in the set of decisions attached to the row in the case of decision tables with many-valued decisions. We correspond to each decision tree \( Γ \in DT_{m_1,m_2,U}(T) \) the point \((mc(T, Γ), L(Γ))\). As a
result, we obtain the set of points \( \{(mc(T, \Gamma), L(\Gamma)) : \Gamma \in DT_{m_1, m_2, U}(T)\} \). Our aim
is to construct, for this set, the set of all Pareto optimal points \( POP_{m_1, m_2, U}(T) = Par(\{(mc(T, \Gamma), L(\Gamma)) : \Gamma \in DT_{m_1, m_2, U}(T)\}) \).

We describe now an algorithm which attaches to each node \( \Theta \) of the DAG
\( G_{m_1, m_2, U}(T) \) the set

\[
POP_{m_1, m_2, U}(\Theta) = Par(\{(mc(\Theta, \Gamma), L(\Gamma)) : \Gamma \in DT_{m_1, m_2, U}(\Theta)\}).
\]

This algorithm works in a bottom-up fashion beginning with subtables which have
common decisions.

**Algorithm** \( A_{POPs} \) (Construction of the set \( POP_{m_1, m_2, U}(T) \)).

**Input**: The DAG \( G_{m_1, m_2, U}(T) \) for a decision table \( T \).

**Output**: The set \( POP_{m_1, m_2, U}(T) \).

1. If all nodes of \( G_{m_1, m_2, U}(T) \) are processed then return the set \( POP_{m_1, m_2, U}(T) \)
atached to the node \( T \) and finish. Otherwise, choose a node \( \Theta \) of \( G_{m_1, m_2, U}(T) \)
which is not processed yet and such that either \( \Theta \) has a common decision or all
children of \( \Theta \) are already processed.

2. If \( \Theta \) has a common decision then attach to \( \Theta \) the set
\( POP_{m_1, m_2, U}(\Theta) = \{(0, 1)\} \),
mark \( \Theta \) as processed, and proceed to step 1.

3. If all children of \( \Theta \) are already processed, and \( S(\Theta) \) is the set of splits \( s \) such
that an \( s \)-pair of edges starts in \( \Theta \) then attach to \( \Theta \) the set

\[
POP_{m_1, m_2, U}(\Theta) = Par(\{(mc(\Theta, tree(\Theta)), 1)\}) \cup \bigcup_{s \in S(\Theta)} \{(a + c, b + d + 1) : (a, b) \in POP_{m_1, m_2, U}(\Theta_{s=0}), (c, d) \in POP_{m_1, m_2, U}(\Theta_{s=1})\},
\]

mark \( \Theta \) as processed and proceed to step 1.
Let $\Theta$ be a node of the graph $G_{m_1,m_2,U}(T)$ and $N(\Theta)$ be the number of rows in $\Theta$. It is clear that all points from $POP_{m_1,m_2,U}(\Theta)$ have pairwise different first coordinates which belong to the set $\{0,1,\ldots,N(\Theta)\}$. Therefore the cardinality of the set $POP_{m_1,m_2,U}(\Theta)$ is bounded from above by the number $N(\Theta)+1$. From here it follows that the time complexity of the considered algorithm is bounded from above by a polynomial in the size of the decision table $T$ and the number of nodes in the graph $G_{m_1,m_2,U}(T)$.

For each Pareto optimal point from the set $POP_{m_1,m_2,U}(\Theta)$, we keep information about its construction: either it corresponds to the tree $\text{tree}(\Theta)$ or is obtained as a combination $(a+c,b+d+1)$ of points $(a,b) \in POP_{m_1,m_2,U}(\Theta_{s=0})$ and $(c,d) \in POP_{m_1,m_2,U}(\Theta_{s=1})$ for some $s \in S(\Theta)$. In the last case, we keep all such combinations. This information allows us to derive, for each point $(x,y)$ from $POP_{m_1,m_2,U}(T)$, decision trees from $DT_{m_1,m_2,U}(T)$, such that $(mc(T,\Gamma),L(\Gamma)) = (x,y)$ for each derived tree $\Gamma$.

### 9.5 Multi-pruning (MP) and Restricted Multi-pruning (RMP) Approaches

We know that the set $DT_{m_1,m_2,U}(T)$ is closed under the operation of usual bottom-up pruning of decision trees. In contrast with pruning of one decision tree, we consider here the pruning of many decision trees. When $m_1 = n$, where $n$ is the number of conditional attributes in $T$, and $m_2 = 0$, we will call our approach to the study of decision trees, the multi-pruning (MP). Furthermore, in general case, when $0 < m_1 + m_2 \leq n$, we will call our approach, the restricted multi-pruning (RMP). In the case of MP, we can use optimal splits for all non-constant attributes, whereas, in the case of RMP, we can use optimal splits for at most $m_1 + m_2$ non-constant attributes. We decided to consider not only MP but also RMP due to time constraints.
<table>
<thead>
<tr>
<th>Decision table</th>
<th>Rows</th>
<th>Attributes</th>
<th>Type of attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>BALANCE-SCALE</td>
<td>625</td>
<td>5</td>
<td>Categorical</td>
</tr>
<tr>
<td>BANKNOTE</td>
<td>1372</td>
<td>5</td>
<td>Numerical</td>
</tr>
<tr>
<td>BREAST-CANCER</td>
<td>266</td>
<td>10</td>
<td>Categorical</td>
</tr>
<tr>
<td>CARS</td>
<td>1728</td>
<td>7</td>
<td>Categorical</td>
</tr>
<tr>
<td>GLASS</td>
<td>214</td>
<td>10</td>
<td>Numerical</td>
</tr>
<tr>
<td>HAYES-ROTH-DATA</td>
<td>69</td>
<td>5</td>
<td>Categorical</td>
</tr>
<tr>
<td>HOUSE-VOTES-84</td>
<td>279</td>
<td>17</td>
<td>Categorical</td>
</tr>
<tr>
<td>IRIS</td>
<td>150</td>
<td>5</td>
<td>Numerical</td>
</tr>
<tr>
<td>LYMPHOGRAPHY</td>
<td>148</td>
<td>19</td>
<td>Categorical</td>
</tr>
<tr>
<td>NURSERY</td>
<td>12960</td>
<td>9</td>
<td>Categorical</td>
</tr>
<tr>
<td>SOYBEAN-SMALL</td>
<td>47</td>
<td>36</td>
<td>Categorical</td>
</tr>
<tr>
<td>SPECT-TEST</td>
<td>169</td>
<td>23</td>
<td>Categorical</td>
</tr>
<tr>
<td>TIC-TAC-TOE</td>
<td>958</td>
<td>10</td>
<td>Categorical</td>
</tr>
<tr>
<td>WINE</td>
<td>178</td>
<td>13</td>
<td>Numerical</td>
</tr>
<tr>
<td>ZOO-DATA</td>
<td>59</td>
<td>17</td>
<td>Categorical</td>
</tr>
</tbody>
</table>

Table 9.1: Decision tables with one-valued decisions from UCI ML Repository

9.6 Data Sets Used in Experiments

We did experiments for two purposes: to represent important information from the decision table, and to classify unknown instances using decision trees. We have used data sets (decision tables) from UCI ML Repository [59] containing categorical or numerical attributes.

The first group of data sets (Table 9.1) refers to the usual decision tables with one-valued decisions. Before the experiment work, some preprocessing procedures were performed. An attribute is removed if it has a unique value for each row. The missing value for an attribute is filled up with most common value for that attribute. If there are duplicate rows with equal values of conditional attributes, then only one row is kept with the most common decision for that group of equal rows and others are removed. In Table 9.1, the first column ‘Decision table’ refers to the name of the decision table from UCI ML Repository, the column ‘Rows’ refers to the number of rows, the column ‘Attributes’ refers to the number of attributes, and the column ‘Types of Attributes’ refers to either categorical or numerical type of conditional attributes in the table.
<table>
<thead>
<tr>
<th>Decision table</th>
<th>Rows</th>
<th>Attributes</th>
<th>Labels</th>
<th>Spectrum #1, #2, #3, ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>CARS-1</td>
<td>432</td>
<td>5</td>
<td>4</td>
<td>258, 161, 13</td>
</tr>
<tr>
<td>LYMPHO-1-13-14-15-18</td>
<td>122</td>
<td>13</td>
<td>4</td>
<td>113, 9</td>
</tr>
<tr>
<td>LYMPHO-13-14-15-18</td>
<td>136</td>
<td>14</td>
<td>4</td>
<td>132, 4</td>
</tr>
<tr>
<td>LYMPHO-1-5-6-7</td>
<td>240</td>
<td>4</td>
<td>5</td>
<td>97, 96, 47</td>
</tr>
<tr>
<td>NURSERY-1-5-6-7</td>
<td>4320</td>
<td>7</td>
<td>5</td>
<td>2858, 1460, 2</td>
</tr>
<tr>
<td>POKER-HAND-2-3-4-6-8</td>
<td>3323</td>
<td>5</td>
<td>10</td>
<td>130, 1850, 1137, 199, 6, 1</td>
</tr>
<tr>
<td>POKER-HAND-2-4-6-8-10</td>
<td>1024</td>
<td>5</td>
<td>10</td>
<td>0, 246, 444, 286, 44, 4</td>
</tr>
<tr>
<td>ZOO-DATA-2-6-8-9-13</td>
<td>43</td>
<td>11</td>
<td>7</td>
<td>40, 1, 2</td>
</tr>
<tr>
<td>ZOO-DATA-2-9-13-14</td>
<td>44</td>
<td>12</td>
<td>7</td>
<td>40, 4</td>
</tr>
<tr>
<td>ZOO-DATA-6-13</td>
<td>46</td>
<td>14</td>
<td>7</td>
<td>44, 2</td>
</tr>
</tbody>
</table>

Table 9.2: Decision tables with many-valued decisions based on tables from UCI ML Repository

The second group of data sets (Table 9.2) refers to the decision tables with many-valued decisions containing a set of one or more decisions for each row (object). We took some decision tables from UCI ML Repository and removed one or more conditional attributes from them. As a result, for each table, there are multiple rows that have equal values of conditional attributes but have different decisions which are then merged into a single row containing the set of decisions from the group of equal rows. Similar preprocessing steps as described above for the decision tables with one-valued decisions are performed as well. In Table 9.2, the first column ‘Decision table’ refers to the name of the decision table from UCI ML Repository along with the numbers of attributes which were removed from the original data set, the column ‘Rows’ refers to the number of rows, the column ‘Attributes’ refers to the number of attributes, the column ‘Labels’ refers to the total number of decisions, and the column ‘Spectrum’ refers to a sequence #1, #2, #3, ... where #i means the number of rows that are labeled with sets of decisions containing i decisions.

9.7 Experimental Results: Knowledge Representation

In this section, we consider the problem of knowledge representation. For a given decision table T, in the framework of MP or RMP approaches, we construct the set
of Pareto optimal points \( POP_{m_1,m_2,U}(T) \) for two parameters of decision trees: number of nodes and number of misclassifications. We choose a POP with minimum Euclidean distance from the origin (see an arrow sign in Fig. 9.3 where POPs are shown such that \( x \) axis corresponds to \( L \), i.e., number of nodes and \( y \) axis corresponds to \( mc \), i.e., number of misclassifications). In many cases, the chosen point corresponds to decision trees with the reasonable number of nodes and number of misclassifications. For example, for ‘hayes-roth-data’ decision table, the number of nodes in the chosen tree is equal to 13, and the number of misclassifications is equal to 8 (the misclassification error rate is equal to 12%). Such trees can be used for knowledge representation. Also, note that the user can choose other point and derive a decision tree with parameters corresponding to this point. For example, if the user chooses the number of nodes equal to 9, then the corresponding number of misclassifications will be 13 (the misclassification error rate is equal to 19%).

9.7.1 Decision Tables with One-valued Decisions

For decision tables with one-valued decisions, we have used 13 data sets described in Table 9.1 (excluding ‘glass’ and ‘wine’). The results are shown for two uncertainty measures: \( gini \) and \( abs \). For each data set, we created the sets of POPs using both MP and RMP approaches. The aim of this experiment is to compare \( gini \) and \( abs \) and to choose \((m_1,m_2)\) RMP algorithm with small sum \( m_1 + m_2 \) and enough good results.

To compare the results among 35 RMP algorithms (we have 35 combinations of \( m_1 \) and \( m_2 \)), we calculate the minimum distance of POPs from the origin. The parameter \( m_2 \) corresponds to randomly chosen attributes. To make the result stable, we repeated the experiment 10 times and took the average. Therefore we get the average minimum distance of POPs from the origin. We name it by ‘\( \text{avg\_min\_dist} \)’. Furthermore, we have results for 13 data sets and for each algorithm. We can compare
Figure 9.3: Pareto optimal points for decision tables with one-valued decisions and uncertainty measure \textit{gini}

algorithms statistically over multiple data sets by taking the average of ranks. For each data set, we rank algorithms as Rank 1, Rank 2, and so on according to the value of \textit{avg.min.dist}. If there is a tie, we can break it by the average of ranks (i.e., Rank 1.5). After that, we take the average of ranks over all data sets \cite{71}. The average of ranks is shown in Table 9.3(a) for \textit{gini}. We highlighted the top three algorithms that have the lowest rank. Similar rankings are shown in Table 9.3(b) for \textit{abs}.

We found (5, 4) RMP algorithm (i.e., when $m_1 = 5$ and $m_2 = 4$), (4, 5) RMP algorithm and (5, 5) RMP algorithm are the three top algorithms in terms of minimum average of rank for both \textit{gini} and \textit{abs}.

We compare the minimum distance of POPs from the origin for MP approach with the best results for RMP approach in Table 9.4. For the case of RMP, we include the algorithm that produces the minimum distance (for the case of multiple algorithms, we took one with the minimum of the sum of $m_1$ and $m_2$). It is clear that the results
for RMP are very close to the results for MP, in fact, they are the same for \textit{gini} but slightly higher for \textit{abs}. The last row shows the average value of the minimum distance.

The \((2, 0)\) RMP algorithm is closer to the MP algorithm for the maximum number of the data sets. Therefore we compared the results for MP and \((2, 0)\) RMP algorithms in Table 9.5. One can see that, on average, the distance from the origin to the set of POPs for \((2, 0)\) RMP algorithm is very close to the distance for MP algorithm.

The obtained results show that, for knowledge representation for decision tables with one-valued decisions, instead of \textit{gini} we can use \textit{abs} and instead of MP algorithm we can use \((2, 0)\) RMP algorithm.

### 9.7.2 Decision Tables with Many-Valued Decisions

The experiments are also performed with the decision tables with many-valued decisions. We took eight data sets from Table 9.2 (excluding the ‘POKER-HAND’-
Table 9.4: Comparison of minimum distances from the origin to the Pareto optimal points for decision tables with one-valued decisions

<table>
<thead>
<tr>
<th>Decision table</th>
<th>MP gini distance</th>
<th>Best RMP gini distance $(m_1, m_2)$</th>
<th>MP abs distance</th>
<th>Best RMP abs distance $(m_1, m_2)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>BALANCE-SCALE</td>
<td>97.0464</td>
<td>97.0464 (1, 1)</td>
<td>97.0464</td>
<td>97.0464 (3, 0)</td>
</tr>
<tr>
<td>BANKNOTE</td>
<td>19.2094</td>
<td>19.2094 (1, 1)</td>
<td>19.2094</td>
<td>19.2094 (1, 1)</td>
</tr>
<tr>
<td>BREAST-CANCER</td>
<td>44.6542</td>
<td>44.6542 (2, 4)</td>
<td>44.6542</td>
<td>44.6542 (2, 4)</td>
</tr>
<tr>
<td>CARS</td>
<td>76.0592</td>
<td>76.0592 (2, 1)</td>
<td>76.0592</td>
<td>76.0592 (4, 0)</td>
</tr>
<tr>
<td>HAYES-ROTH-DATA</td>
<td>15.2643</td>
<td>15.2643 (2, 0)</td>
<td>15.2643</td>
<td>15.2643 (2, 0)</td>
</tr>
<tr>
<td>HOUSE-VOTES-84</td>
<td>14.2127</td>
<td>14.2127 (2, 0)</td>
<td>14.2127</td>
<td>14.2127 (2, 0)</td>
</tr>
<tr>
<td>IRIS</td>
<td>7.81025</td>
<td>7.81025 (2, 0)</td>
<td>7.81025</td>
<td>7.81025 (2, 0)</td>
</tr>
<tr>
<td>LYMPHOGRAPHY</td>
<td>20.2485</td>
<td>20.2485 (4, 5)</td>
<td>20.2485</td>
<td>20.5183 (3, 3)</td>
</tr>
<tr>
<td>NURSERY</td>
<td>248.244</td>
<td>248.244 (2, 0)</td>
<td>248.244</td>
<td>248.244 (2, 0)</td>
</tr>
<tr>
<td>SOYBEAN-SMALL</td>
<td>7</td>
<td>7 (2, 0)</td>
<td>7</td>
<td>7 (2, 0)</td>
</tr>
<tr>
<td>SPECT-TEST</td>
<td>8.06226</td>
<td>8.06226 (2, 0)</td>
<td>8.06226</td>
<td>8.06226 (2, 0)</td>
</tr>
<tr>
<td>TIC-TAC-TOE</td>
<td>61.8547</td>
<td>61.8547 (1, 1)</td>
<td>61.8547</td>
<td>61.8547 (1, 1)</td>
</tr>
<tr>
<td>ZOO-DATA</td>
<td>12.083</td>
<td>12.083 (2, 0)</td>
<td>12.083</td>
<td>12.083 (2, 0)</td>
</tr>
<tr>
<td><strong>Average:</strong></td>
<td><strong>48.59607</strong></td>
<td><strong>48.59607</strong></td>
<td><strong>48.59607</strong></td>
<td><strong>48.61682</strong></td>
</tr>
</tbody>
</table>

Table 9.5: Comparison of minimum distance from the origin to the Pareto optimal points for MP and (2, 0) RMP algorithms for decision tables with one-valued decisions

<table>
<thead>
<tr>
<th>Decision table</th>
<th>MP gini distance</th>
<th>(2, 0) RMP gini distance</th>
<th>MP abs distance</th>
<th>(2, 0) RMP abs distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>BALANCE-SCALE</td>
<td>97.0464</td>
<td>97.0618</td>
<td>97.0464</td>
<td>98.8382</td>
</tr>
<tr>
<td>BREAST-CANCER</td>
<td>44.6542</td>
<td>47.8539</td>
<td>44.6542</td>
<td>47.8539</td>
</tr>
<tr>
<td>CARS</td>
<td>76.0592</td>
<td>79.2023</td>
<td>76.0592</td>
<td>79.2023</td>
</tr>
<tr>
<td>HAYES-ROTH-DATA</td>
<td>15.2643</td>
<td>15.2643</td>
<td>15.2643</td>
<td>15.2643</td>
</tr>
<tr>
<td>IRIS</td>
<td>7.81025</td>
<td>7.81025</td>
<td>7.81025</td>
<td>7.81025</td>
</tr>
<tr>
<td>NURSERY</td>
<td>248.244</td>
<td>248.244</td>
<td>248.244</td>
<td>248.244</td>
</tr>
<tr>
<td>SOYBEAN-SMALL</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>SPECT-TEST</td>
<td>8.06226</td>
<td>8.06226</td>
<td>8.06226</td>
<td>8.06226</td>
</tr>
<tr>
<td>TIC-TAC-TOE</td>
<td>61.8547</td>
<td>61.8547</td>
<td>61.8547</td>
<td>61.8547</td>
</tr>
<tr>
<td><strong>Average:</strong></td>
<td><strong>48.59607</strong></td>
<td><strong>49.61094</strong></td>
<td><strong>48.59607</strong></td>
<td><strong>49.74759</strong></td>
</tr>
</tbody>
</table>

In these experiments, we only used the uncertainty measure abs. The aim of the experiments is to choose a $(m_1, m_2)$ RMP algorithm with the small sum of $m_1$ and $m_2$ which can be used instead of MP algorithm.

For each dataset, we created the set of POPs using both MP and RMP approaches. We compare the results in the same way as described in the previous section. The average values of ranks of 35 RMP algorithms are shown in Table 9.6 in which the top three algorithms are highlighted.

For the case of decision tables with many-valued decisions, $(5, 5)$ RMP algorithm, $(5, 4)$ RMP algorithm and $(5, 3)$ RMP algorithm are the top algorithms in terms of the
Table 9.6: Average values of ranks for the average minimum distance from the origin to the Pareto optimal points for decision tables with many-valued decisions

<table>
<thead>
<tr>
<th>$m_1 \setminus m_2$</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>N/A</td>
<td>35</td>
<td>34</td>
<td>33</td>
<td>28.12</td>
<td>26.75</td>
</tr>
<tr>
<td>1</td>
<td>21.56</td>
<td>20.56</td>
<td>20.19</td>
<td>19.31</td>
<td>18.44</td>
<td>16.62</td>
</tr>
<tr>
<td>2</td>
<td>19.88</td>
<td>17.94</td>
<td>17.31</td>
<td>16.88</td>
<td>14.5</td>
<td>13.38</td>
</tr>
<tr>
<td>3</td>
<td>19.88</td>
<td>18.31</td>
<td>16.88</td>
<td>14.88</td>
<td>14.31</td>
<td>13.81</td>
</tr>
<tr>
<td>4</td>
<td>16.38</td>
<td>15.5</td>
<td>13.69</td>
<td>13.19</td>
<td>13.06</td>
<td>12.69</td>
</tr>
<tr>
<td>5</td>
<td>12.81</td>
<td>12.56</td>
<td>12.44</td>
<td>12.12</td>
<td>12.12</td>
<td>11.94</td>
</tr>
</tbody>
</table>

Table 9.7: Comparison of minimum distances from the origin to the Pareto optimal points for decision tables with many-valued decisions

<table>
<thead>
<tr>
<th>Decision table</th>
<th>MP abs distance</th>
<th>Best RMP abs distance ($m_1, m_2$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CARS-1</td>
<td>9.21954</td>
<td>9.21954 (2, 0)</td>
</tr>
<tr>
<td>LYMPHO-1-13-14-15-18</td>
<td>19.2354</td>
<td>19.2354 (4, 0)</td>
</tr>
<tr>
<td>LYMPHO-13-14-15-18</td>
<td>22.6716</td>
<td>22.6716 (2, 2)</td>
</tr>
<tr>
<td>NURSERY-1-5-6-7</td>
<td>5</td>
<td>5 (2, 0)</td>
</tr>
<tr>
<td>NURSERY-1</td>
<td>30.8058</td>
<td>30.8058 (1, 1)</td>
</tr>
<tr>
<td>ZOO-DATA-2-6-8-9-13</td>
<td>10.2956</td>
<td>10.2956 (2, 0)</td>
</tr>
<tr>
<td>ZOO-DATA-2-9-13-14</td>
<td>12.0416</td>
<td>12.0416 (2, 0)</td>
</tr>
<tr>
<td>ZOO-DATA-6-13</td>
<td>11.4018</td>
<td>11.4018 (2, 0)</td>
</tr>
<tr>
<td><strong>Average:</strong></td>
<td><strong>15.08392</strong></td>
<td><strong>15.08392</strong></td>
</tr>
</tbody>
</table>

average of ranks. We compared the minimum distance for MP and RMP algorithms in Table 9.7. For the case of RMP, we include the algorithm that gives the minimum distance (for the case of multiple algorithms, we took one with the minimum sum of $m_1$ and $m_2$). One can see that the minimum distance from the origin is the same for MP and RMP approaches.

Since (2, 0) RMP algorithm is closer to the MP algorithm for the maximum number of data sets, we compared the results for MP and (2, 0) RMP algorithms in Table 9.8. On average, the distance from the origin to the set of POPs for (2, 0) RMP algorithm is very close to the distance for MP algorithm.

The obtained results show that, for knowledge representation for decision tables with many-valued decisions, instead of MP algorithm we can use (2, 0) RMP algorithm.

188
<table>
<thead>
<tr>
<th>Decision table</th>
<th>MP abs</th>
<th>(2, 0) RMP abs</th>
</tr>
</thead>
<tbody>
<tr>
<td>CARS-1</td>
<td>9.21954</td>
<td>9.21954</td>
</tr>
<tr>
<td>LYMPHO-13-14-15-18</td>
<td>22.6716</td>
<td>23.8537</td>
</tr>
<tr>
<td>NURSERY-1-5-6-7</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>NURSERY-1</td>
<td>30.8058</td>
<td>32.2025</td>
</tr>
<tr>
<td>ZOO-DATA-2-6-8-9-13</td>
<td>10.2956</td>
<td>10.2956</td>
</tr>
<tr>
<td>ZOO-DATA-2-9-13-14</td>
<td>12.0416</td>
<td>12.0416</td>
</tr>
<tr>
<td>ZOO-DATA-6-13</td>
<td>11.4018</td>
<td>11.4018</td>
</tr>
<tr>
<td><strong>Average:</strong></td>
<td>15.08392</td>
<td>15.51742</td>
</tr>
</tbody>
</table>

Table 9.8: Comparison of minimum distance from the origin to the Pareto optimal points for MP and (2, 0) RMP algorithms for decision tables with many-valued decisions

9.8 Experimental Results: Classification

The second group of experiments refers to the problem of classification when we should predict decisions for unknown instances. We use the misclassification error rate (the number of misclassifications divided by the whole number of rows) as a performance measure for classifiers. Let $T$ be a decision table. We divide $T$ into three parts: $T_{\text{train}}$, $T_{\text{val}}$, and $T_{\text{test}}$. The first part, $T_{\text{train}}$, is used to train the model, i.e., decision tree for the classification task. We construct the DAG $G_{m_1,m_2,U}(T_{\text{train}})$ and based on this DAG we construct the set of Pareto optimal points $POP_{m_1,m_2,U}(T_{\text{train}})$. For each point $(a,b) \in POP_{m_1,m_2,U}(T_{\text{train}})$, we derive randomly $k$ decision trees (in our experiments, $k = 5$) $\Gamma_1, \ldots, \Gamma_k$ from $DT_{m_1,m_2,U}(T_{\text{train}})$ such that $(a,b) = (mc(T_{\text{train}},\Gamma_i), L(\Gamma_i))$ for $i = 1, \ldots, k$. Among such decision trees, we choose a decision tree $\Gamma$ which has the minimum number of misclassifications on $T_{\text{val}}$. The third part of table, $T_{\text{test}}$ is used to measure the performance of the final classifier $\Gamma$. We have used 2-fold cross-validation and, for each fold, we repeated the experiment 5 times and took the average misclassification error rate. For the 2-fold cross-validation, 50 percent of data is used for training and the rest is for testing. Within 50 percent of data for training, 70 percent is for actual training and the rest 30 percent is for validation.
9.8.1 Decision Tables with One-valued Decisions

We have used 15 data sets from UCI ML Repository described in Table 9.1 as decision tables with one-valued decisions. The results are obtained using both \( gini \) and \( abs \). For each of the uncertainty measures and each data set, we have classifier based on MP approach and 35 classifiers based on RMP approach. The aim of experiments is to compare \( gini \) and \( abs \), and to choose \( (m_1, m_2) \) RMP algorithms with small sum \( m_1 + m_2 \) comparable with MP algorithm from the point of view of classifier accuracy and having less time complexity.

A. Uncertainty Measure \( gini \)

First, for \( gini \) uncertainty measure, we have shown the average of ranks of the 35 classifiers in Table 9.9(a) based on misclassification error rate where we rank ‘Rank 1’ the classifiers which give the minimum misclassification error rate. We highlighted the top three algorithms. We have also shown the average of ranks of the 35 classifiers in Table 9.9(b) based on the time of construction and usage of the classifiers where lowest time-consuming algorithm gets the ‘Rank 1’, and we also highlighted the top three algorithms.

From results in Table 9.9 it follows that the classifiers based on \( (2, 0) \) RMP algorithm and \( (3, 0) \) RMP algorithm are the top ranked classifiers (note that classifiers based on \( (5, 2) \) RMP algorithm are too time consuming and classifiers based on \( (1, 0) \) RMP algorithm have bad misclassification error rate). Now, we compare the top ranked classifiers based on MP and RMP approaches with CART in Table 9.10 for misclassification error rate, and in Table 9.11 for the time of construction and usage.

From the point of view of the accuracy, the classifiers based on MP approach outperform CART classifiers for 11 out of 15 decision tables. Classifiers based on \( (2, 0) \) RMP algorithm outperform CART classifiers in ten cases and have ties in one case, and classifiers based on \( (3, 0) \) RMP algorithm outperform CART classifiers in
Table 9.9: Average values of ranks of classifiers based on \textit{gini} for decision tables with one-valued decisions

<table>
<thead>
<tr>
<th>( m_1 ) ( m_2 )</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>N/A</td>
<td>31.87</td>
<td>24.80</td>
<td>23.80</td>
<td>17.47</td>
<td>17.33</td>
</tr>
<tr>
<td>1</td>
<td>21.63</td>
<td>21.57</td>
<td>22.23</td>
<td>17.40</td>
<td>15.30</td>
<td>16.27</td>
</tr>
<tr>
<td>2</td>
<td>15.23</td>
<td>15.90</td>
<td>16.83</td>
<td>18.23</td>
<td>19.43</td>
<td>15.93</td>
</tr>
<tr>
<td>3</td>
<td>15.13</td>
<td>15.40</td>
<td>17.43</td>
<td>18.10</td>
<td>15.57</td>
<td>16.43</td>
</tr>
<tr>
<td>4</td>
<td>17.00</td>
<td>17.07</td>
<td>19.00</td>
<td>17.70</td>
<td>16.87</td>
<td>17.03</td>
</tr>
<tr>
<td>5</td>
<td>15.90</td>
<td>15.83</td>
<td>14.33</td>
<td>16.77</td>
<td>16.13</td>
<td>17.07</td>
</tr>
</tbody>
</table>

(a) Relative to misclassification error rate

<table>
<thead>
<tr>
<th>( m_1 ) ( m_2 )</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>N/A</td>
<td>15.47</td>
<td>08.87</td>
<td>07.13</td>
<td>09.67</td>
<td>11.40</td>
</tr>
<tr>
<td>1</td>
<td>04.53</td>
<td>05.33</td>
<td>05.33</td>
<td>07.93</td>
<td>10.93</td>
<td>11.67</td>
</tr>
<tr>
<td>2</td>
<td>04.33</td>
<td>18.67</td>
<td>22.53</td>
<td>10.07</td>
<td>28.27</td>
<td>29.40</td>
</tr>
<tr>
<td>3</td>
<td>03.40</td>
<td>21.73</td>
<td>22.87</td>
<td>27.73</td>
<td>28.00</td>
<td>25.93</td>
</tr>
<tr>
<td>4</td>
<td>20.00</td>
<td>22.20</td>
<td>23.60</td>
<td>26.33</td>
<td>29.67</td>
<td>25.87</td>
</tr>
<tr>
<td>5</td>
<td>22.00</td>
<td>22.00</td>
<td>19.93</td>
<td>27.60</td>
<td>24.13</td>
<td>25.47</td>
</tr>
</tbody>
</table>

(b) Relative to time for construction and usage

From the obtained results it follows that, in the case of classification problem, decision tables with one-valued decisions and \textit{gini} uncertainty measure, we can use \((2, 0)\) RMP and \((3, 0)\) RMP algorithms instead of MP algorithm.

B. Uncertainty Measure \textit{abs}

We did similar experiments with \textit{abs}. In Table 9.12(a), the average of ranks among 35 types of classifiers are shown, and the top three types of classifiers are highlighted. The similar ranking is shown in Table 9.12(b) when we compare the time of construction and usage among 35 types of classifiers, and the top three types of classifiers are also highlighted.

We can see that only classifiers based on \((3, 0)\) RMP algorithm are among the top
classifiers for both classification and time perspective. We also find that classifiers based on \((2, 0)\) RMP algorithm are top based on the time and their rank based on classification accuracy is not far from the top-ranked classifiers. Therefore, we include classifiers based on \((3, 0)\) RMP and \((2, 0)\) RMP algorithms for the comparison with CART as well as classifiers based on MP approach (see Tables 9.13 and 9.14).

Table 9.13 shows that classifiers based on MP approach outperform CART classifiers for eight out of 15 decision tables. However, classifiers based on \((3, 0)\) RMP algorithm outperform CART classifiers in 11 cases and have ties in one case. Also, classifiers based on \((2, 0)\) RMP algorithm outperform CART classifiers in ten cases. We have shown the average value at the end of the table. The average time for the construction and usage of classifiers based on RMP approach is almost five times or eight times less than for MP classifiers (see Table 9.14).

The obtained results show that, in the case of classification problem, decision tables with one-valued decisions and \(\text{abs}\) uncertainty measure, we can use \((2, 0)\) RMP and \((3, 0)\) RMP algorithms instead of MP algorithm. The comparison of Tables 9.10

<table>
<thead>
<tr>
<th>Decision table</th>
<th>CART</th>
<th>MP</th>
<th>RMP ((3, 0))</th>
<th>RMP ((2, 0))</th>
</tr>
</thead>
<tbody>
<tr>
<td>BALANCE-SCALE</td>
<td>23.81</td>
<td>23.68</td>
<td>24.70</td>
<td>25.83</td>
</tr>
<tr>
<td>BANKNOTE</td>
<td>3.20</td>
<td>1.91</td>
<td>1.96</td>
<td>2.14</td>
</tr>
<tr>
<td>BREAST-CANCER</td>
<td>29.17</td>
<td>29.32</td>
<td>29.77</td>
<td>28.12</td>
</tr>
<tr>
<td>CARS</td>
<td>5.16</td>
<td>5.08</td>
<td>5.03</td>
<td>4.86</td>
</tr>
<tr>
<td>GLASS</td>
<td>39.34</td>
<td>38.49</td>
<td>37.93</td>
<td>40.38</td>
</tr>
<tr>
<td>HAYES-ROTH-DATA</td>
<td>43.26</td>
<td>30.99</td>
<td>26.97</td>
<td>25.28</td>
</tr>
<tr>
<td>HOUSE-VOTES-84</td>
<td>6.81</td>
<td>6.67</td>
<td>6.60</td>
<td>7.03</td>
</tr>
<tr>
<td>IRIS</td>
<td>5.03</td>
<td>5.16</td>
<td>5.43</td>
<td>5.03</td>
</tr>
<tr>
<td>LYMPHOGRAPHY</td>
<td>27.97</td>
<td>24.86</td>
<td>28.92</td>
<td>24.86</td>
</tr>
<tr>
<td>NURSERY</td>
<td>1.40</td>
<td>1.44</td>
<td>1.39</td>
<td>1.28</td>
</tr>
<tr>
<td>SOYBEAN-SMALL</td>
<td>21.36</td>
<td>6.70</td>
<td>9.35</td>
<td>12.32</td>
</tr>
<tr>
<td>SPECT-TEST</td>
<td>5.21</td>
<td>4.74</td>
<td>4.74</td>
<td>4.74</td>
</tr>
<tr>
<td>TIC-TAC-TOE</td>
<td>10.21</td>
<td>7.81</td>
<td>5.97</td>
<td>7.52</td>
</tr>
<tr>
<td>WINE</td>
<td>12.47</td>
<td>11.80</td>
<td>11.35</td>
<td>11.57</td>
</tr>
<tr>
<td>ZOO-DATA</td>
<td>23.74</td>
<td>25.82</td>
<td>25.14</td>
<td>25.48</td>
</tr>
<tr>
<td><strong>Average:</strong></td>
<td><strong>17.21</strong></td>
<td><strong>14.97</strong></td>
<td><strong>15.02</strong></td>
<td><strong>15.10</strong></td>
</tr>
</tbody>
</table>

Table 9.10: Comparison of average misclassification error rates of classifiers based on \(gini\) for decision tables with one-valued decisions
<table>
<thead>
<tr>
<th>Decision table</th>
<th>MP</th>
<th>RMP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(3, 0)</td>
<td>(2, 0)</td>
</tr>
<tr>
<td>BALANCE-SCALE</td>
<td>149.19</td>
<td>60.94</td>
</tr>
<tr>
<td>BANKNOTE</td>
<td>66.30</td>
<td>13.31</td>
</tr>
<tr>
<td>BREAST-CANCER</td>
<td>114.20</td>
<td>24.46</td>
</tr>
<tr>
<td>CARS</td>
<td>194.69</td>
<td>71.83</td>
</tr>
<tr>
<td>GLASS</td>
<td>1387.40</td>
<td>24.41</td>
</tr>
<tr>
<td>HAYES-ROTH-DATA</td>
<td>23.54</td>
<td>9.49</td>
</tr>
<tr>
<td>HOUSE-VOTES-84</td>
<td>24.92</td>
<td>9.70</td>
</tr>
<tr>
<td>IRIS</td>
<td>12.48</td>
<td>7.41</td>
</tr>
<tr>
<td>LYMPHOGRAPHY</td>
<td>125.06</td>
<td>16.28</td>
</tr>
<tr>
<td>NURSERY</td>
<td>1399.62</td>
<td>419.75</td>
</tr>
<tr>
<td>SOYBEAN-SMALL</td>
<td>13.44</td>
<td>4.21</td>
</tr>
<tr>
<td>SPECT-TEST</td>
<td>30.04</td>
<td>7.34</td>
</tr>
<tr>
<td>TIC-TAC-TOE</td>
<td>324.67</td>
<td>68.67</td>
</tr>
<tr>
<td>WINE</td>
<td>204.31</td>
<td>6.47</td>
</tr>
<tr>
<td>ZOO-DATA</td>
<td>19.06</td>
<td>8.28</td>
</tr>
<tr>
<td><strong>Average:</strong></td>
<td><strong>272.59</strong></td>
<td><strong>50.17</strong></td>
</tr>
</tbody>
</table>

Table 9.11: Comparison of average time in seconds to construct and use classifiers based on \( gini \) for decision tables with one-valued decisions

and 9.11 with Tables 9.13 and 9.14 shows also that instead of \( gini \) we can use \( abs \) uncertainty measure for classification problem and decision tables with one-valued decisions.

### 9.8.2 Decision Tables with Many-Valued Decisions

We also work with classification problem for the decision tables with many-valued decisions. We took ten data sets described in Table 9.2.

The experiment settings are similar to the case of decision tables with one-valued decisions, and we have results for both MP and RMP approaches. The average of ranks based on misclassification error rate among classifiers constructed by 35 RMP algorithms are shown in Table 9.15(a) where the top three classifiers are highlighted. We also have the average of ranks based on the time for construction and usage of the classifiers in Table 9.15(b).

We do not have the equivalent of CART classifiers in the case of decision tables with many-valued decisions (CART is applicable only for decision tables with one-
Table 9.12: Average values of ranks of classifiers based on \( \text{abs} \) for decision tables with one-valued decisions.

We only compare between classifiers based on MP and RMP approaches in Table 9.16. We took three types of classifiers according to the results from Table 9.15: classifiers based on \((2, 0)\) RMP algorithm are top classifiers relative to both accuracy and time, classifiers based on \((3, 0)\) RMP algorithm are top relative to the time, and classifiers based on \((4, 0)\) RMP algorithm are top relative to the misclassification error rate. The classifiers based on \((2, 0)\) RMP algorithm outperform the classifiers based on MP approach in six cases and have ties in two cases out of ten. We took average to compare the results. We compared the time of classifier construction and usage in Table 9.17 and found that the classifiers based on \((2, 0)\) RMP algorithm require least time (on average, 37 times less than MP classifiers).

From the obtained results it follows that \((2, 0)\) RMP algorithm can be used instead of MP algorithm for the construction of classifiers for decision tables with many-valued decisions.
<table>
<thead>
<tr>
<th>Decision table</th>
<th>CART</th>
<th>MP</th>
<th>RMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>balance-scale</td>
<td>23.81</td>
<td>24.26</td>
<td>24.35</td>
</tr>
<tr>
<td>banknote</td>
<td>3.2</td>
<td>1.91</td>
<td>1.96</td>
</tr>
<tr>
<td>breast-cancer</td>
<td>29.17</td>
<td>29.85</td>
<td>30.6</td>
</tr>
<tr>
<td>cars</td>
<td>5.16</td>
<td>5.2</td>
<td>5.02</td>
</tr>
<tr>
<td>glass</td>
<td>39.34</td>
<td>40.56</td>
<td>37.37</td>
</tr>
<tr>
<td>Hayes-Roth-data</td>
<td>43.26</td>
<td>32.75</td>
<td>26.7</td>
</tr>
<tr>
<td>House-votes-84</td>
<td>6.81</td>
<td>6.67</td>
<td>6.6</td>
</tr>
<tr>
<td>iris</td>
<td>5.03</td>
<td>5.16</td>
<td>5.57</td>
</tr>
<tr>
<td>lymphography</td>
<td>27.97</td>
<td>25.68</td>
<td>27.7</td>
</tr>
<tr>
<td>nursery</td>
<td>1.4</td>
<td>1.43</td>
<td>1.4</td>
</tr>
<tr>
<td>soybean-small</td>
<td>21.36</td>
<td>6.7</td>
<td>9.35</td>
</tr>
<tr>
<td>spect-test</td>
<td>5.21</td>
<td>4.74</td>
<td>4.74</td>
</tr>
<tr>
<td>tic-tac-toe</td>
<td>10.21</td>
<td>7.72</td>
<td>5.95</td>
</tr>
<tr>
<td>wine</td>
<td>12.47</td>
<td>11.46</td>
<td>11.24</td>
</tr>
<tr>
<td>zoo-data</td>
<td>23.74</td>
<td>26.48</td>
<td>23.46</td>
</tr>
</tbody>
</table>

Average: 17.21 15.37 14.8 14.83

Table 9.13: Comparison of average misclassification error rates of classifiers based on \( \abs \) for decision tables with one-valued decisions

<table>
<thead>
<tr>
<th>Decision table</th>
<th>MP</th>
<th>RMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>balance-scale</td>
<td>12.95</td>
<td>78.01</td>
</tr>
<tr>
<td>banknote</td>
<td>17.09</td>
<td>18.66</td>
</tr>
<tr>
<td>breast-cancer</td>
<td>32.21</td>
<td>30.82</td>
</tr>
<tr>
<td>cars</td>
<td>20.53</td>
<td>95.36</td>
</tr>
<tr>
<td>glass</td>
<td>5627.34</td>
<td>33.25</td>
</tr>
<tr>
<td>Hayes-Roth-data</td>
<td>1.56</td>
<td>15.14</td>
</tr>
<tr>
<td>House-votes-84</td>
<td>4.35</td>
<td>15.24</td>
</tr>
<tr>
<td>iris</td>
<td>0.84</td>
<td>14</td>
</tr>
<tr>
<td>lymphography</td>
<td>51.97</td>
<td>31</td>
</tr>
<tr>
<td>nursery</td>
<td>483.39</td>
<td>959.63</td>
</tr>
<tr>
<td>soybean-small</td>
<td>1.63</td>
<td>10.54</td>
</tr>
<tr>
<td>spect-test</td>
<td>7.32</td>
<td>18.51</td>
</tr>
<tr>
<td>tic-tac-toe</td>
<td>117.89</td>
<td>141.16</td>
</tr>
<tr>
<td>wine</td>
<td>179.55</td>
<td>14.51</td>
</tr>
<tr>
<td>zoo-data</td>
<td>1.71</td>
<td>20.5</td>
</tr>
</tbody>
</table>

Average: 437.36 99.76 53.23

Table 9.14: Comparison of average time in seconds to construct and use classifiers based on \( \abs \) for decision tables with one-valued decisions

195
<table>
<thead>
<tr>
<th>$m_1 \setminus m_2$</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>N/A</td>
<td>29.9</td>
<td>28.85</td>
<td>21.45</td>
<td>15.85</td>
<td>17.85</td>
</tr>
<tr>
<td>1</td>
<td>25.25</td>
<td>25.5</td>
<td>22.75</td>
<td>21.5</td>
<td>20.85</td>
<td>19.55</td>
</tr>
<tr>
<td>2</td>
<td><strong>11.85</strong></td>
<td><strong>11.35</strong></td>
<td>13.4</td>
<td>17.3</td>
<td>20.1</td>
<td>17.9</td>
</tr>
<tr>
<td>3</td>
<td>15.7</td>
<td>13.6</td>
<td>15.1</td>
<td>14.65</td>
<td>17.15</td>
<td>17.35</td>
</tr>
<tr>
<td>4</td>
<td><strong>12.2</strong></td>
<td>18</td>
<td>15.3</td>
<td>15.5</td>
<td>17.05</td>
<td>16.2</td>
</tr>
<tr>
<td>5</td>
<td>13.5</td>
<td>17.25</td>
<td>19.2</td>
<td>18.1</td>
<td>16.65</td>
<td>16.3</td>
</tr>
</tbody>
</table>

(a) Relative to the misclassification error rate

<table>
<thead>
<tr>
<th>$m_1 \setminus m_2$</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>N/A</td>
<td>18.4</td>
<td>6.5</td>
<td>24.1</td>
<td>25.2</td>
<td>31.1</td>
</tr>
<tr>
<td>1</td>
<td>2.9</td>
<td>8.7</td>
<td>6</td>
<td>24.1</td>
<td>26.1</td>
<td>27.2</td>
</tr>
<tr>
<td>2</td>
<td><strong>1.8</strong></td>
<td>14.8</td>
<td>8.8</td>
<td>24.7</td>
<td>25</td>
<td>25.6</td>
</tr>
<tr>
<td>3</td>
<td>5.1</td>
<td>14.4</td>
<td>14.5</td>
<td>27.2</td>
<td>26.7</td>
<td>24.9</td>
</tr>
<tr>
<td>4</td>
<td>8.9</td>
<td>11.9</td>
<td>10.1</td>
<td>23.1</td>
<td>28.1</td>
<td>22.6</td>
</tr>
<tr>
<td>5</td>
<td>15.2</td>
<td>10.9</td>
<td>8.7</td>
<td>26.9</td>
<td>26.6</td>
<td>23.2</td>
</tr>
</tbody>
</table>

(b) Relative to the time for construction and usage

Table 9.15: Average values of ranks of classifiers based on $abs$ for decision tables with many-valued decisions

<table>
<thead>
<tr>
<th>Decision table</th>
<th>MP</th>
<th>RMP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(4, 0)</td>
<td>(3, 0)</td>
</tr>
<tr>
<td>CARS-1</td>
<td>1.53</td>
<td>1.57</td>
</tr>
<tr>
<td>NURSERY-1-5-6-7</td>
<td>1.17</td>
<td>1.17</td>
</tr>
<tr>
<td>NURSERY-1</td>
<td>0.69</td>
<td>0.63</td>
</tr>
<tr>
<td>POKER-HAND-2-3-4-6-8</td>
<td>2.42</td>
<td>2.34</td>
</tr>
<tr>
<td>POKER-HAND-2-4-6-8-10</td>
<td>0.27</td>
<td>0.27</td>
</tr>
<tr>
<td>ZOO-DATA-2-6-8-9-13</td>
<td>31.26</td>
<td>29.39</td>
</tr>
<tr>
<td>ZOO-DATA-2-9-13-14</td>
<td>38.64</td>
<td>39.55</td>
</tr>
<tr>
<td>ZOO-DATA-6-13</td>
<td>29.57</td>
<td>27.83</td>
</tr>
<tr>
<td><strong>Average:</strong></td>
<td><strong>15.90</strong></td>
<td><strong>15.64</strong></td>
</tr>
</tbody>
</table>

Table 9.16: Average misclassification error rates of classifiers based on $abs$ for decision tables with many-valued decisions
<table>
<thead>
<tr>
<th>Decision table</th>
<th>MP</th>
<th>RMP</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(4,0)</td>
<td>(3,0)</td>
<td>(2,0)</td>
<td></td>
</tr>
<tr>
<td>CARS-1</td>
<td>1.23</td>
<td>1.54</td>
<td>1.32</td>
<td>1.28</td>
</tr>
<tr>
<td>LYMPHO-1-13-14-15-18</td>
<td>2.83</td>
<td>3.24</td>
<td>2.89</td>
<td>2.32</td>
</tr>
<tr>
<td>LYMPHO-13-14-15-18</td>
<td>4.58</td>
<td>3.46</td>
<td>2.91</td>
<td>2.55</td>
</tr>
<tr>
<td>NURSERY-1-5-6-7</td>
<td>0.79</td>
<td>0.8</td>
<td>0.9</td>
<td>0.73</td>
</tr>
<tr>
<td>NURSERY-1</td>
<td>26.13</td>
<td>17.9</td>
<td>9.02</td>
<td>5.31</td>
</tr>
<tr>
<td>POKER-HAND-2-3-4-6-8</td>
<td>1072.2</td>
<td>1018.15</td>
<td>138.81</td>
<td>13.26</td>
</tr>
<tr>
<td>POKER-HAND-2-4-6-8-10</td>
<td>1.1</td>
<td>1.14</td>
<td>0.99</td>
<td>0.84</td>
</tr>
<tr>
<td>ZOO-DATA-2-6-8-9-13</td>
<td>1.13</td>
<td>1.48</td>
<td>1.33</td>
<td>1.14</td>
</tr>
<tr>
<td>ZOO-DATA-2-9-13-14</td>
<td>1.08</td>
<td>1.61</td>
<td>1.29</td>
<td>1.1</td>
</tr>
<tr>
<td>ZOO-DATA-6-13</td>
<td>1.12</td>
<td>1.8</td>
<td>1.37</td>
<td>1.17</td>
</tr>
<tr>
<td>Average:</td>
<td>111.22</td>
<td>105.11</td>
<td>16.08</td>
<td>2.97</td>
</tr>
</tbody>
</table>

Table 9.17: Average time in seconds to construct and use of classifiers based on *abs* for decision tables with many-valued decisions
Chapter 10

Three Approaches to Handle Inconsistency in Decision Tables

We devoted this chapter to the study of inconsistent decision tables. We consider three approaches to handle inconsistency in decision tables: decision tables with many-valued decisions, decision tables with generalized decisions, and decision tables with most common decisions. After that, we compare three approaches based on the complexity and classification accuracy of decision trees constructed in the framework of these approaches. This chapter contains the experimental results from [35, 37].

10.1 Inconsistent Decision Tables and Three Approaches

It is pretty common in real life problems to have inconsistent decision tables where there are groups of examples (rows) with equal values of conditional attributes but different decisions (values of the decision attribute). One of the reasons is that we do not have enough number of attributes of the domain to separate examples (rows). Furthermore, it is natural to have such data sets in optimization problems such as finding a Hamiltonian circuit with the minimum length in traveling salesman problem, finding nearest post office [2], etc. They also arise when we study, e.g., problem of semantic annotation of images [3], music categorization into emotions [5], functional genomics [6], text categorization [7], etc. As an example in Figure 10.1, we can see that $T^0$ is an inconsistent decision table since the rows 2 and 3 have equal values of conditional attributes but different decisions (1 and 3). In this section, we discuss
three ways to handle such inconsistent decision tables.

In the thesis, we study decision tables with many-valued decisions. In this approach (MVD), a single row is considered from the group of equal rows, and a set of decisions has been formed consisting of different decisions from the group. As a result, the inconsistent decision table $T$ is transformed into the decision table $T_{MVD}$ with many-valued decisions (see table $T_{MVD}^0$ in Figure [10.1]).

In the rough set theory [31], generalized decisions have been used to handle inconsistency. In this approach ($GD$), each set of decisions in the table $T_{MVD}$ has been encoded by a number (decision) such that equal sets are encoded by equal numbers and different sets by different numbers. As a result, we obtain a consistent decision table $T_{GD}$ (see table $T_{GD}^0$ in Figure [10.1]). We call such decision tables as decision tables with generalized decisions.

We have also used another approach named the most common decisions ($MCD$) which is derived from the concept of using the most common value to fill a missing value [72]. Instead of a group of equal rows with (probably) different decisions, we consider one row given by values of conditional attributes, and we attach to this row the most common decision for rows from the group. If there are more than one most common decisions, we chose the minimum one. As a result, the initial table $T$ is transformed into a consistent table $T_{MCD}$ (see table $T_{MCD}^0$ in Figure [10.1]). We call such decision tables as decision tables with most common decisions.

For the decision tables with many-valued decisions, our goal is to find a single decision from the attached set of decisions for each row. This approach can be used for the study of classical optimization problems (finding a Hamiltonian circuit with the minimum length or finding the nearest post office) where we have multiple optimal solutions, but we have to give only one output. The considered approach is applicable in all cases when decisions from the sets attached to rows are equally good, and it is enough to find only one decision.
Note that, we will interpret decision tables with one-valued decisions, i.e., \( T_{GD} \) and \( T_{MCD} \), as a special case of decision tables with many-valued decisions where each row is labeled with a set of decisions that have only a single decision. Hence, we can apply the same algorithms for all three cases.

The aim of this chapter is to compare complexity and classification accuracy of decision trees constructed by greedy heuristics in the framework of the three approaches, i.e., compare decision trees for tables \( T_{MVD}, T_{GD}, \) and \( T_{MCD} \).

### 10.2 Comparison of Complexity of Decision Trees

In this section, we compare depth, average depth, and number of nodes of decision trees constructed by a greedy heuristic for decision tables \( T_{MVD}, T_{GD}, \) and \( T_{MCD} \). We use \( ws/me \) greedy heuristic (see Section 7.3) which is based on weighted sum type of impurity function \( ws \) and misclassification error uncertainty measure \( me \).
10.2.1 Decision Tables Used in Experiments

Some data sets from KEEL-Dataset Repository [73] as well as from UCI ML Repository [59] have been considered. Data sets from KEEL are already in the format of decision tables with many-valued decisions, i.e., \( T = T_{MVD} \). These tables are further converted into formats \( T_{MCD} \) (in this case, the first decision is selected from the set of decisions attached to a row as the most common decision) and \( T_{GD} \) by the procedure described in Section 1.1.1. Some conditional attributes are removed from the UCI ML Repository data sets to convert these data sets into inconsistent decision tables. We also performed other preprocessing steps, i.e., we removed a conditional attribute if it had a unique value for each row and imputed the missing value for a conditional attribute with the corresponding most common value. These inconsistent tables \( T \) were further converted into \( T_{MVD}, T_{GD}, \) and \( T_{MCD} \) by the procedures described in Section 1.1.1. These modified tables have been renamed as the name of the initial table with an index equal to the number of removed conditional attributes. The information about the considered decision tables is shown in Tables 10.1 and 10.2.

These tables contain name of decision table \( T \), number of rows (column “Rows”), number of conditional attributes (column “Attr”), and spectrum of the table \( T_{MVD} \) (column “Spectrum”). Spectrum of a decision table with many-valued decisions is a sequence \( \#1, \#2, \ldots \), where \( \#i, i = 1, 2, \ldots \), is the number of rows labeled with sets of decisions with the cardinality equal to \( i \). For some tables (marked with * in Table 10.2), the spectrum is too long to fit in the page width. Hence, it has been shown up to the element \( \#10 \).

10.2.2 Experimental Results

Tables 10.3 and 10.4 contain depth, average depth, and number of nodes for decision trees constructed by the greedy heuristic \( ws_{me} \) for decision tables \( T_{MVD}, T_{MCD}, \) and \( T_{GD} \).
<table>
<thead>
<tr>
<th>Decision table</th>
<th>Rows</th>
<th>Attr</th>
<th>Spectrum</th>
<th>#1</th>
<th>#2</th>
<th>#3</th>
<th>#4</th>
<th>#5</th>
<th>#6</th>
</tr>
</thead>
<tbody>
<tr>
<td>BALANCE-SCALE-1</td>
<td>125</td>
<td>3</td>
<td></td>
<td>45</td>
<td>50</td>
<td>30</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BREAST-CANCER-1</td>
<td>193</td>
<td>8</td>
<td></td>
<td>169</td>
<td>24</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BREAST-CANCER-5</td>
<td>98</td>
<td>4</td>
<td></td>
<td>58</td>
<td>40</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CARS-1</td>
<td>432</td>
<td>5</td>
<td></td>
<td>258</td>
<td>161</td>
<td>13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLAGS-5</td>
<td>171</td>
<td>21</td>
<td></td>
<td>159</td>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HAYES-ROTH-DATA-1</td>
<td>39</td>
<td>3</td>
<td></td>
<td>22</td>
<td>13</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KR-VS-KP-5</td>
<td>1987</td>
<td>31</td>
<td></td>
<td>1564</td>
<td>423</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KR-VS-KP-4</td>
<td>2061</td>
<td>32</td>
<td></td>
<td>1652</td>
<td>409</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LYMPHOGRAPHY-5</td>
<td>122</td>
<td>13</td>
<td></td>
<td>113</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MUSHROOM-5</td>
<td>4078</td>
<td>17</td>
<td></td>
<td>4048</td>
<td>30</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NURSERY-1</td>
<td>4320</td>
<td>7</td>
<td></td>
<td>2858</td>
<td>1460</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NURSERY-4</td>
<td>240</td>
<td>4</td>
<td></td>
<td>97</td>
<td>96</td>
<td>47</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPECT-TEST-1</td>
<td>164</td>
<td>21</td>
<td></td>
<td>161</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TEETH-1</td>
<td>22</td>
<td>7</td>
<td></td>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TEETH-5</td>
<td>14</td>
<td>3</td>
<td></td>
<td>6</td>
<td>3</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>TIC-TAC-TOE-1</td>
<td>231</td>
<td>5</td>
<td></td>
<td>102</td>
<td>129</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIC-TAC-TOE-3</td>
<td>449</td>
<td>6</td>
<td></td>
<td>300</td>
<td>149</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZOO-DATA-5</td>
<td>42</td>
<td>11</td>
<td></td>
<td>36</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 10.1: Characteristics of decision tables obtained from UCI ML Repository data sets

<table>
<thead>
<tr>
<th>Decision table</th>
<th>Rows</th>
<th>Attr</th>
<th>Spectrum</th>
<th>#1</th>
<th>#2</th>
<th>#3</th>
<th>#4</th>
<th>#5</th>
<th>#6</th>
<th>#7</th>
<th>#8</th>
<th>#9</th>
<th>#10</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIBTEX*</td>
<td>7355</td>
<td>1836</td>
<td></td>
<td>2791</td>
<td>1825</td>
<td>1302</td>
<td>399</td>
<td>179</td>
<td>87</td>
<td>46</td>
<td>18</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>COREL5K</td>
<td>4998</td>
<td>499</td>
<td></td>
<td>3</td>
<td>376</td>
<td>1559</td>
<td>3013</td>
<td>17</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>DELICIOUS*15862</td>
<td>944</td>
<td>95</td>
<td></td>
<td>207</td>
<td>292</td>
<td>340</td>
<td>422</td>
<td>536</td>
<td>714</td>
<td>930</td>
<td>1108</td>
<td>1460</td>
<td></td>
</tr>
<tr>
<td>ENRON*</td>
<td>1561</td>
<td>1001</td>
<td></td>
<td>179</td>
<td>238</td>
<td>441</td>
<td>337</td>
<td>200</td>
<td>91</td>
<td>51</td>
<td>15</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>GENBASE-1</td>
<td>662</td>
<td>1186</td>
<td></td>
<td>27</td>
<td>360</td>
<td>31</td>
<td>31</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>MEDICAL</td>
<td>967</td>
<td>1449</td>
<td></td>
<td>741</td>
<td>212</td>
<td>14</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Table 10.2: Characteristics of decision tables from KEEL-Dataset Repository

For each of the cost functions and for each of the decision tables (with the exception of the depth for ‘ENRON’), the cost of decision trees constructed in the framework of $MVD$ is at most the cost of decision trees constructed in the framework of $MCD$, and the cost of latter trees is at most the cost of trees constructed in the framework of $GD$ approach.

One can show that, the decision trees constructed in the framework of $MVD$
approach are usually simpler than the decision trees constructed in the framework of \(MCD\) approach, and the decision trees constructed in the framework of \(MCD\) approach are usually simpler than the decision trees constructed in the framework of \(GD\) approach.

<table>
<thead>
<tr>
<th>Decision table</th>
<th>Depth</th>
<th>Average Depth</th>
<th>Number of Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(MVD)</td>
<td>(MCD)</td>
<td>(GD)</td>
</tr>
<tr>
<td>balance-scale-1</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>breast-cancer-1</td>
<td>6</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>breast-cancer-5</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>cars-1</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>flags-5</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>hayes-roth-data-1</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>kr-vs-kp-5</td>
<td>13</td>
<td>14</td>
<td>15</td>
</tr>
<tr>
<td>kr-vs-kp-4</td>
<td>14</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>lymphography-5</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>mushroom-5</td>
<td>7</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>nursery-1</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>nursery-4</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>spect-test-1</td>
<td>6</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>teeth-1</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>teeth-5</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>tic-tac-toe-4</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>tic-tac-toe-3</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>zoo-data-5</td>
<td>4</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td>5.67</td>
<td>6.39</td>
<td>6.56</td>
</tr>
</tbody>
</table>

Table 10.3: Depth, average depth and number of nodes for decision trees constructed for tables \(T_{MVD}\), \(T_{MCD}\) and \(T_{GD}\), UCI ML Repository

<table>
<thead>
<tr>
<th>Decision table</th>
<th>Depth</th>
<th>Average Depth</th>
<th>Number of Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(MVD)</td>
<td>(MCD)</td>
<td>(GD)</td>
</tr>
<tr>
<td>bibtex</td>
<td>39</td>
<td>42</td>
<td>43</td>
</tr>
<tr>
<td>corel5k</td>
<td>156</td>
<td>156</td>
<td>157</td>
</tr>
<tr>
<td>delicious</td>
<td>79</td>
<td>92</td>
<td>92</td>
</tr>
<tr>
<td>enron</td>
<td>28</td>
<td>26</td>
<td>41</td>
</tr>
<tr>
<td>medical</td>
<td>16</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td>63.6</td>
<td>66.4</td>
<td>69.8</td>
</tr>
</tbody>
</table>

Table 10.4: Depth, average depth and number of nodes for decision trees constructed for tables \(T_{MVD}\), \(T_{MCD}\) and \(T_{GD}\), KEEL-Dataset Repository
10.3 Comparison of Accuracy of Classifiers

In this section, we study 12 greedy heuristics for construction of decision trees. We consider the decision trees constructed by these heuristics as classifiers and compare their accuracy in the frameworks of MVD, MCD, and GD approaches.

10.3.1 Greedy Heuristics and Decision Tables Used in Experiments

We considered four uncertainty measures (\(me, \text{ abs}, \text{ entS}, \text{ and } \text{entM}\)) and three types of impurity function (\(ws, \text{ wm}, \text{ and } \text{Mult}_w\)). Each uncertainty measure and each type of impurity function define a greedy heuristic for decision tree construction (see details in Section 7.3). As a result, we have 12 greedy heuristics (see Table 10.5).

<table>
<thead>
<tr>
<th>Name of heuristic</th>
<th>Uncertainty measure</th>
<th>Type of impurity</th>
</tr>
</thead>
<tbody>
<tr>
<td>(wm_{\text{entM}})</td>
<td>(\text{entML})</td>
<td>(\text{wm})</td>
</tr>
<tr>
<td>(wm_{\text{entS}})</td>
<td>(\text{entSort})</td>
<td>(\text{wm})</td>
</tr>
<tr>
<td>(wm_{\text{abs}})</td>
<td>(\text{abs})</td>
<td>(\text{wm})</td>
</tr>
<tr>
<td>(wm_{\text{me}})</td>
<td>(\text{me})</td>
<td>(\text{wm})</td>
</tr>
<tr>
<td>(ws_{\text{entM}})</td>
<td>(\text{entML})</td>
<td>(\text{ws})</td>
</tr>
<tr>
<td>(ws_{\text{entS}})</td>
<td>(\text{entSort})</td>
<td>(\text{ws})</td>
</tr>
<tr>
<td>(ws_{\text{abs}})</td>
<td>(\text{abs})</td>
<td>(\text{ws})</td>
</tr>
<tr>
<td>(ws_{\text{me}})</td>
<td>(\text{me})</td>
<td>(\text{ws})</td>
</tr>
<tr>
<td>(M_{\text{ws}}_{\text{entM}})</td>
<td>(\text{entML})</td>
<td>(\text{Mult}_w)</td>
</tr>
<tr>
<td>(M_{\text{ws}}_{\text{entS}})</td>
<td>(\text{entSort})</td>
<td>(\text{Mult}_w)</td>
</tr>
<tr>
<td>(M_{\text{ws}}_{\text{abs}})</td>
<td>(\text{abs})</td>
<td>(\text{Mult}_w)</td>
</tr>
<tr>
<td>(M_{\text{ws}}_{\text{me}})</td>
<td>(\text{me})</td>
<td>(\text{Mult}_w)</td>
</tr>
</tbody>
</table>

Table 10.5: Greedy heuristics

For the experiments, we use five decision tables (LYMPHOGRAPHY-5, CARS-1, FLAGS-5, NURSERY-1, and ZOO-DATA-5) described in Table 10.1 and five decision tables (BIBTEX, COREL5K, ENRON, GENBASE-1, and MEDICAL) described in Table 10.2.

204
10.3.2 Experimental Results

Let $T$ be one of the considered decision tables and $H$ be one of the considered greedy heuristics. We use 3-fold cross-validation in which the decision table $T$ is divided into 3 folds. At $i$-th ($i = 1, 2, 3$) iteration, $i$-th fold is used as the test subtable, and the rest of table is partitioned randomly into train (70%) and validation (30%) subtables. We apply the heuristic $H$ to the train subtable, and as a result, we obtain the trained decision tree $\Gamma_1$. We successively prune the nodes of the trained decision tree, calculate the error rates of obtained classifiers on the validation subtable, and return the pruned decision tree $\Gamma_2$ with minimum error rate. After pruning, we calculate the error rate of the decision tree $\Gamma_2$ on the test subtable. We repeat the experiment 5 times and take the average of the obtained 15 error rates.

Let us consider one of the approaches MVD, MCD, or GD. We have 12 greedy heuristics $H_1, \ldots, H_{12}$ for decision tree construction and 10 decision tables $T_1, \ldots, T_{10}$. For each decision table $T_i$, $i = 1, \ldots, 10$, we rank the heuristics $H_1, \ldots, H_{12}$ on $T_i$ based on their performance scores of classification error rates, where we assign the best performing algorithm the rank of 1, the second best rank 2, and so on. We break ties by computing the average of ranks. Let $r_{ij}$ be the rank of the $j$-th of 12 algorithms on the decision table $T_i$. For $j = 1, \ldots, 12$, we correspond to the heuristic $H_j$ the average rank $R_j = \frac{1}{10} \cdot \sum_{i=1}^{10} r_{ij}$.

Table 10.6 shows the three best ranked heuristics (with average ranks) for the approaches MVD, MCD, and GD. It is clear that $M_{\text{ws.abs}}$ is the best ranked heuristic for MVD and MCD approaches, and $M_{\text{ws.entM}}$ is the best ranked heuristic for GD approach.

We show the classification error rates for each decision table (based on the three best-ranked heuristics), and for the three approaches in Table 10.7.

On average, MVD approach gives better classification results compared to others.
<table>
<thead>
<tr>
<th>Name of heuristic</th>
<th>Average rank</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MVD</td>
</tr>
<tr>
<td>M_ws_abs</td>
<td>3.45</td>
</tr>
<tr>
<td>M_ws_entM</td>
<td>3.60</td>
</tr>
<tr>
<td>M_ws_entS</td>
<td>3.90</td>
</tr>
</tbody>
</table>

Table 10.6: Average ranks of top three greedy heuristics

<table>
<thead>
<tr>
<th>Decision table</th>
<th>MVD</th>
<th>MCD</th>
<th>GD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>M_ws</td>
<td>M_ws</td>
<td>M_ws</td>
</tr>
<tr>
<td></td>
<td>_abs</td>
<td>_entS</td>
<td>_entM</td>
</tr>
<tr>
<td>BIBTEX</td>
<td>57.38</td>
<td>59.96</td>
<td>57.72</td>
</tr>
<tr>
<td>CARS-1</td>
<td>3.66</td>
<td>3.80</td>
<td>4.44</td>
</tr>
<tr>
<td>COREL5K</td>
<td>74.39</td>
<td>76.52</td>
<td>77.55</td>
</tr>
<tr>
<td>ENRON</td>
<td>35.93</td>
<td>27.05</td>
<td>30.29</td>
</tr>
<tr>
<td>FLAGS-5</td>
<td>61.75</td>
<td>63.86</td>
<td>63.51</td>
</tr>
<tr>
<td>GENBASE-1</td>
<td>4.46</td>
<td>5.34</td>
<td>4.85</td>
</tr>
<tr>
<td>LYMPHOGRAPHY-5</td>
<td>25.20</td>
<td>24.87</td>
<td>24.54</td>
</tr>
<tr>
<td>MEDICAL</td>
<td>20.00</td>
<td>23.43</td>
<td>21.72</td>
</tr>
<tr>
<td>NURSERY-1</td>
<td>2.05</td>
<td>2.70</td>
<td>2.71</td>
</tr>
<tr>
<td>ZOO-DATA-5</td>
<td>34.76</td>
<td>32.38</td>
<td>30.48</td>
</tr>
<tr>
<td>AVERAGE</td>
<td>31.96</td>
<td>31.99</td>
<td>31.78</td>
</tr>
</tbody>
</table>

Table 10.7: Classification error rate (in %)
Chapter 11

Local and Global Approaches to Study of Decision and Inhibitory Trees

This chapter is devoted to the study of time complexity of decision and inhibitory trees over arbitrary sets of attributes represented by information systems.

An information system \( \mathcal{U} = (A, B, F) \) consists of a set \( A \) of elements and a set of attributes \( F \) which are functions that are defined on \( A \) and have values from a finite set \( B \). We will assume that \( F \) does not contain constant attributes. This information system is called finite if \( F \) is a finite set, and infinite, otherwise. The notion of information system considered here has much in common with the notion of information system proposed by Zdzislaw Pawlak [54].

The notion of a problem with many-valued decisions over the information system \( \mathcal{U} \) is defined as follows. Take a finite number of attributes \( f_1, \ldots, f_n \) from \( F \). These attributes create a partition of the set \( A \) into classes (domains). For each class, values of the attributes \( f_1, \ldots, f_n \) are constant on elements from the class. A nonempty finite set of decisions is attached to each class. The number \( n \) is called the dimension of the considered problem.

We also study problems with one-valued decisions for which a decision is attached to each class. A problem with one-valued decisions can be considered as a special case of problems with many-valued decisions where each class is labeled with a set of decisions containing only one decision.

There are two interpretations of each problem with many-valued decisions: decision and inhibitory. For the decision interpretation, for a given element \( a \) from \( A \), it
is required to recognize a decision from the set attached to the class which contains $a$. For the inhibitory interpretation, for a given element $a$ from $A$, it is required to recognize a decision for this problem which does not belong to the set attached to the class which contains $a$. For problems with one-valued decisions, we consider only decision interpretation.

As algorithms for solving problems with many-valued decisions, decision trees are considered for decision interpretation, and inhibitory trees are considered for inhibitory interpretation. The depth of a tree (the maximum length of a path from the root to a terminal node) is considered as time complexity measure. For problems with one-valued decisions, we study only decision trees.

We consider two approaches to the study of the decision and inhibitory trees: local and global. The local approach assumes that trees can use only attributes $f_1, \ldots, f_n$ from the problem description. The global approach allows us to use arbitrary attributes from the set $F$ in trees. We use indexes $l$ and $g$ to distinguish local and global approaches, respectively.

For problems with one-valued decisions, for each information system and for each approach (local and global), we define a Shannon function which characterizes the growth in the worst case of minimum depth of decision trees with the growth of problem dimension.

As an example, consider the notion of Shannon function $H^g_{\mathcal{U}}$ for problems with one-valued decisions over an information system $\mathcal{U} = (A, B, F)$, decision trees, and the global approach. Let $z$ be a problem with one-valued decisions over the information system $\mathcal{U}$ and $h^g_{\mathcal{U}}(z)$ be the minimum depth of a decision tree over $\mathcal{U}$ which solves the problem $z$ in the decision interpretation and uses arbitrary attributes from the set $F$. Then, for any natural $n$, Shannon function $H^g_{\mathcal{U}}(n)$ is equal to the maximum value of $h^g_{\mathcal{U}}(z)$ among all problems $z$ with one-valued decisions over the information system $\mathcal{U}$ which dimension (the number of attributes in the description of $z$) is at most $n$. 

208
For problems with many-valued decisions, for each information system and for each approach (local and global), we define two Shannon functions which characterize the growth in the worst case of (i) minimum depth of decision trees and (ii) minimum depth of inhibitory trees with the growth of problem dimension.

In Sect. 11.1, we define all these Shannon functions, and study relationships between Shannon functions for problems with one-valued and many-valued decisions (they are equal), and relationships between Shannon functions for decision and inhibitory approaches (they are also equal).

We aim to describe the behavior of Shannon functions for problems with many-valued decisions. To this end, we should either study Shannon functions for problems with many-valued decisions directly or extend results for Shannon functions for problems with one-valued decisions to problems with many-valued decisions using statements from Sect. 11.1.

For the study of the most of cases, it is enough to consider results published years ago in [74, 75, 76, 77, 78, 2]. The only exception is the case of the global approach and finite information systems which was considered in the paper Time complexity of decision trees and decision rule systems over finite information systems by Fawaz Alsolami, Mohammad Azad, Mikhail Moshkov, and Beata Zielosko submitted to Discrete Applied Mathematics. In Sect. 11.2, we study this case for problems with one-valued decisions.

In Sect. 11.3, we describe the behavior of Shannon functions for local and global approaches, for decision and inhibitory trees for problems with many-valued decisions over finite and infinite information systems.

11.1 Various Types of Shannon Functions

In this section, we define different kinds of Shannon functions and study relationships between Shannon functions for problems with one-valued and many-valued decisions,
and relationships between Shannon functions for decision and inhibitory approaches.

Let $A$ be a nonempty set of elements, $B$ be a finite set with at least two elements, and $F$ be a nonempty set of functions from $A$ to $B$. Functions from $F$ are called attributes and the triple $\mathcal{U} = (A, B, F)$ is called an information system. This information system is called finite if $F$ is a finite set, and infinite otherwise. We assume that $f \not\equiv \text{const}$ for any $f \in F$.

An equation system over $\mathcal{U}$ is an arbitrary system of the kind

$$\{f_1(x) = \delta_1, \ldots, f_m(x) = \delta_m\}$$

where $f_1, \ldots, f_m \in F$ and $\delta_1, \ldots, \delta_m \in B$. It is possible that the considered system does not have equations. Such system is called empty. The set of solutions of the empty system coincides with the set $A$. There is a one-to-one correspondence between equation systems over $\mathcal{U}$ and words from the set $\Omega_{F,B} = \{(f, \delta) : f \in F, \delta \in B\}^*$: the word $(f_1, \delta_1) \ldots (f_m, \delta_m)$ corresponds to the considered equation system, the empty word $\lambda$ corresponds to the empty equation system. For any $\alpha \in \Omega_{F,B}$, denote by $\text{Sol}_\mathcal{U}(\alpha)$ the set of solutions over $A$ of the equation system corresponding to the word $\alpha$.

### 11.1.1 Shannon Functions for Decision Trees. Problems with One-valued Decisions

Let $\omega = \{0, 1, 2, \ldots\}$ be the set of nonnegative integers including all natural numbers and the number 0. A problem over $\mathcal{U}$ with one-valued decisions is an arbitrary $(n+1)$-tuple $z = (\nu, f_1, \ldots, f_n)$ where $\nu : B^n \to \omega$ and $f_1, \ldots, f_n \in F$. The number $\dim z = n$ is called the dimension of the problem $z$. Denote $\text{At}(z) = \{f_1, \ldots, f_n\}$. The problem $z$ may be interpreted as a problem of searching for the value $z(a) = \nu(f_1(a), \ldots, f_n(a))$ for an arbitrary $a \in A$. We consider this interpretation as the decision one. Note that
one can interpret $f_1, \ldots, f_n$ as conditional attributes and $z$ as a decision attribute. Denote $\text{Probl}_U$ the set of problems with one-valued decisions over $U$.

Decision trees are considered as algorithms for problem-solving when we study its decision interpretation.

A decision tree over $U$ is a labeled finite rooted directed tree in which each terminal node is labeled with a number from $\omega$; each node which is not terminal (such nodes are called \textit{working}) is labeled with an attribute from $F$; each edge is labeled with an element from $B$, and edges starting in a working node are labeled with distinct elements. Denote $\text{Tree}_U$ the set of decision trees over $U$.

Let $\Gamma$ be a decision tree over $U$. Denote $\text{At}(\Gamma)$ the set of attributes assigned to working nodes of $\Gamma$. A complete path in $\Gamma$ is an arbitrary path from the root to a terminal node. Denote $\text{Path}(\Gamma)$ the set of all complete paths in $\Gamma$. Let $\xi \in \text{Path}(\Gamma)$. Define a word $\pi(\xi)$ from the set $\Omega_{F,B}(\Gamma) = \{(f, \delta) : f \in \text{At}(\Gamma), \delta \in B\}$* associated with $\xi$. If there are no working nodes in $\xi$ then $\pi(\xi) = \lambda$. Note that in this case the set $\text{Sol}_U(\pi(\xi))$ coincides with the set $A$. Let $\xi = v_1, d_1, \ldots, v_m, d_m, v_{m+1}$ where $m > 0$, $v_1$ is the root, $v_{m+1}$ is a terminal node, and $v_i$ is the initial and $v_{i+1}$ is the terminal node of the edge $d_i$ for $i = 1, \ldots, m$. Let the node $v_i$ be labeled with the attribute $f_i$, and the edge $d_i$ be labeled with the element $\delta_i$ from $B$, $i = 1, \ldots, m$. Then $\pi(\xi) = (f_1, \delta_1) \ldots (f_m, \delta_m)$. Note that in this case the set $\text{Sol}_U(\pi(\xi))$ coincides with the set of solutions over $A$ of the equation system \{\(f_1(x) = \delta_1, \ldots, f_m(x) = \delta_m\)}.

A decision tree $\Gamma$ over $U$ solves a problem $z$ over $U$ if, for each $a \in A$, there exists a complete path $\xi$ in $\Gamma$ such that $a \in \text{Sol}_U(\pi(\xi))$, and the terminal node of the path $\xi$ is labeled with the number $z(a)$.

For decision trees, as time complexity measure the depth of a decision tree is considered which is the maximum number of working nodes in a complete path in the tree. Denote $h(\Gamma)$ the depth of a decision tree $\Gamma$. For a problem $z$ over $U$, denote $h^o_U(z)$ the minimum depth of a decision tree over $U$ which solves $z$. For a problem
over \( U \), denote \( h_U^1(z) \) the minimum depth of a decision tree \( \Gamma \) over \( U \) which solves \( z \) and for which \( \text{At}(\Gamma) \subseteq \text{At}(z) \). The considered parameters correspond to global and local approaches to the study of decision trees, respectively. One can show that

\[
h_U^2(z) \leq h_U^1(z) \leq \dim z \quad \text{for each problem } z \text{ over } U.
\]

Define two Shannon functions \( H_U^2(n) \) and \( H_U^1(n) \). Let \( n \in \omega \setminus \{0\} \). Then

\[
H_U^2(n) = \max \{ h_U^2(z) : z \in \text{Probl}_U, \dim z \leq n \}, \\
H_U^1(n) = \max \{ h_U^1(z) : z \in \text{Probl}_U, \dim z \leq n \}.
\]

It is clear that \( H_U^2(n) \leq H_U^1(n) \leq n \) for any \( n \in \omega \setminus \{0\} \).

Let \( U \) be a finite information system. A problem \( z \) over \( U \) is tree-stable if \( h_U^2(z) = \dim z \). Denote \( \text{ts}(U) \) the maximum dimension of a tree-stable problem from \( \text{Probl}_U \).

### 11.1.2 Shannon Functions for Decision Trees. Problems with Many-valued Decisions

Let us consider an information system \( U = (A, B, F) \). A problem with many-valued decisions over the information system \( U \) is an arbitrary \((n+1)\)-tuple \( z = (\nu, f_1, \ldots, f_n) \) where \( \nu : B^n \rightarrow \text{Fin}(\omega) \), \( \text{Fin}(\omega) \) is the set of all nonempty finite subsets of \( \omega \), and \( f_1, \ldots, f_n \in F \). The number \( \dim z = n \) is called the dimension of the problem \( z \). Denote \( \text{At}(z) = \{f_1, \ldots, f_n\} \). The problem \( z \) may be interpreted as a problem of searching for a value from the set \( z(a) = \nu(f_1(a), \ldots, f_n(a)) \) for an arbitrary \( a \in A \).

We consider this interpretation as the decision one. Denote \( \text{Probl}_U^\infty \) the set of problems with many-valued decisions over \( U \).

Decision trees are considered as algorithms for problem-solving when we study its decision interpretation.

A decision tree \( \Gamma \) over \( U \) solves a problem \( z \in \text{Probl}_U^\infty \) if, for each \( a \in A \), there exists a complete path \( \xi \) in \( \Gamma \) such that \( a \in \text{Sol}_U(\pi(\xi)) \), and the terminal node of the path \( \xi \) is labeled with a number belonging to \( z(a) \). Denote \( h_U^2(z) \) the minimum
Let $\omega$ show that two parameters correspond to global and local approaches, respectively. One can show that $h_l^g(z) \leq h_l^l(z) \leq \dim z$.

Define two Shannon functions $H^g_{U,\omega}(n)$ and $H^l_{U,\omega}(n)$. Let $n \in \omega \setminus \{0\}$. Then

$$H^g_{U,\omega}(n) = \max\{h^g_l(z) : z \in \text{Probl}^\omega_U, \dim z \leq n\},$$
$$H^l_{U,\omega}(n) = \max\{h^l_l(z) : z \in \text{Probl}^\omega_U, \dim z \leq n\}.$$

It is clear that $H^g_{U,\omega}(n) \leq H^l_{U,\omega}(n) \leq n$ for any $n \in \omega \setminus \{0\}$.

Let $U$ be a finite information system. A problem $z \in \text{Probl}^\omega_U$ is **tree-stable** if $h^g_l(z) = \dim z$. Denote $\text{ts}^\omega(U)$ the maximum dimension of a tree-stable problem from $\text{Probl}^\omega_U$.

**Proposition 36.** Let $U = (A, B, F)$ be an information system. Then, for any $n \in \omega \setminus \{0\}$, $H^g_{U,\omega}(n) = H^g_l(n)$ and $H^l_{U,\omega}(n) = H^l_l(n)$. If $U$ is a finite information system then $\text{ts}^\omega(U) = \text{ts}(U)$.

*Proof. Let $f_1, \ldots, f_n \in F$. Denote by $\nu_n$ a mapping of the set $B^n$ into the set $\omega$ such that $\nu_n(\tilde{\delta}_1) \neq \nu_n(\tilde{\delta}_2)$ for any $\tilde{\delta}_1, \tilde{\delta}_2 \in B^n$, $\tilde{\delta}_1 \neq \tilde{\delta}_2$. Let $z_n = (\nu_n, f_1, \ldots, f_n)$ and $z$ be an arbitrary problem of the kind $(\nu, f_1, \ldots, f_n)$ where $\nu : B^n \to \omega$. One can show that $h^g_l(z) \leq h^g_l(z_n)$ and $h^l_l(z) \leq h^l_l(z_n)$.

Denote by $\nu^\omega_n$ a mapping of the set $B^n$ into the set $\text{Fin}(\omega)$ such that $\nu^\omega_n(\tilde{\delta}_1) \cap \nu^\omega_n(\tilde{\delta}_2) = \emptyset$ for any $\tilde{\delta}_1, \tilde{\delta}_2 \in B^n$, $\tilde{\delta}_1 \neq \tilde{\delta}_2$. Let $z^\omega_n = (\nu^\omega_n, f_1, \ldots, f_n)$ and $z^\omega$ be an arbitrary problem of the kind $(\nu^\omega, f_1, \ldots, f_n)$ where $\nu^\omega : B^n \to \text{Fin}(\omega)$. One can show that $h^g_l(z^\omega) \leq h^g_l(z^\omega_n)$ and $h^l_l(z^\omega) \leq h^l_l(z^\omega_n)$.

It is easy to see that $h^g_l(z_n) = h^g_l(z^\omega_n)$ and $h^l_l(z_n) = h^l_l(z^\omega_n)$. Using these facts it is not difficult to show that, for any $n \in \omega \setminus \{0\}$, $H^g_{U,\omega}(n) = H^g_l(n)$ and $H^l_{U,\omega}(n) = H^l_l(n)$, and if $U$ is a finite information system, then $\text{ts}^\omega(U) = \text{ts}(U)$ and $\text{rs}^\omega(U) = \text{rs}(U)$.
11.1.3 Shannon Functions for Inhibitory Trees. Problems with Many-valued Decisions

Let \( \mathcal{U} = (A, B, F) \) be an information system and \( z = (\nu, f_1, \ldots, f_n) \in \text{Probl}_\infty^\mathcal{U} \).
Denote \( \text{Row}_\mathcal{U}(z) \) the set of \( n \)-tuples \((\delta_1, \ldots, \delta_n) \in B^n \) such that the equation system
\[
\{ f_1(x) = \delta_1, \ldots, f_n(x) = \delta_n \}
\]
has a solution over \( A \), and
\[
D_\mathcal{U}(z) = \bigcup_{(\delta_1, \ldots, \delta_n) \in \text{Row}_\mathcal{U}(z)} \nu(\delta_1, \ldots, \delta_n).
\]
We will say that the problem \( z \) is inhibitory-correct if \( D_\mathcal{U}(z) \setminus \nu(\delta_1, \ldots, \delta_n) \neq \emptyset \) for any tuple \((\delta_1, \ldots, \delta_n) \in \text{Row}_\mathcal{U}(z)\). Denote \( \text{IProbl}_\mathcal{U}^\infty \) the set of inhibitory-correct problems from \( \text{Probl}_\mathcal{U}^\infty \).

Let \( z = (\nu, f_1, \ldots, f_n) \in \text{IProbl}_\mathcal{U}^\infty \). The problem \( z \) may be interpreted as a problem of searching for a value from the set \( D_\mathcal{U}(z) \setminus z(a) \) for an arbitrary \( a \in A \) where
\[
z(a) = \nu(f_1(a), \ldots, f_n(a)).
\]
We consider this interpretation as the inhibitory one. Inhibitory trees are considered as algorithms for problem solving when we study its inhibitory interpretation.

An inhibitory tree over \( \mathcal{U} \) is a labeled finite rooted directed tree in which each terminal node is labeled with an expression \( \neq t \) where \( t \in \omega \); each node which is not terminal (such nodes are called working) is labeled with an attribute from \( F \); each edge is labeled with an element from \( B \), and edges starting in a working node are labeled with pairwise different elements.

Let \( \Gamma \) be an inhibitory tree over \( \mathcal{U} \). Denote \( \text{At}(\Gamma) \) the set of attributes assigned to working nodes of \( \Gamma \). A complete path in \( \Gamma \) is an arbitrary path from the root to a terminal node. Denote \( \text{Path}(\Gamma) \) the set of all complete paths in \( \Gamma \). Let \( \xi \in \text{Path}(\Gamma) \).
Define a word \( \pi(\xi) \) from the set \( \Omega_{F,B}(\Gamma) = \{ (f, \delta) : f \in \text{At}(\Gamma), \delta \in B \}^* \) associated
with $\xi$. If there are no working nodes in $\xi$ then $\pi(\xi) = \lambda$. Note that in this case the set $\text{Sol}_U(\pi(\xi))$ coincides with the set $A$. Let $\xi = v_1, d_1, \ldots, v_m, d_m, v_{m+1}$ where $m > 0$, $v_1$ is the root, $v_{m+1}$ is a terminal node, and $v_i$ is the initial and $v_{i+1}$ is the terminal node of the edge $d_i$ for $i = 1, \ldots, m$. Let the node $v_i$ be labeled with the attribute $f_i$, and the edge $d_i$ be labeled with the element $\delta_i$ from $B$, $i = 1, \ldots, m$. Then $\pi(\xi) = (f_1, \delta_1) \ldots (f_m, \delta_m)$. Note that in this case the set $\text{Sol}_U(\pi(\xi))$ coincides with the set of solutions over $A$ of the equation system \{\(f_1(x) = \delta_1, \ldots, f_m(x) = \delta_m\)\}.

An inhibitory tree $\Gamma$ over $U$ solves a problem $z$ from $\text{IProbl}_U^\infty$ if, for each $a \in A$, there exists a complete path $\xi$ in $\Gamma$ such that $a \in \text{Sol}_U(\pi(\xi))$, and the terminal node of the path $\xi$ is labeled with an expression $\neq t$ where $t \in D_U(z)$ and $t \notin z(a)$.

For inhibitory trees, as time complexity measure the depth of an inhibitory tree is considered which is the maximum number of working nodes in a complete path in the tree. Denote $h(\Gamma)$ the depth of an inhibitory tree $\Gamma$. For a problem $z$ from $\text{IProbl}_U^\infty$, denote $ih^g_U(z)$ the minimum depth of an inhibitory tree over $U$ which solves $z$. For a problem $z$ from $\text{IProbl}_U^\infty$, denote $ih^l_U(z)$ the minimum depth of an inhibitory tree $\Gamma$ over $U$ which solves $z$ and for which $\text{At}(\Gamma) \subseteq \text{At}(z)$. The considered two parameters correspond to global and local approaches, respectively. One can show that $ih^g_U(z) \leq ih^l_U(z) \leq \dim z$ for each problem $z$ from $\text{IProbl}_U^\infty$.

Define two Shannon functions $IH^g_{U, \infty}(n)$ and $IH^l_{U, \infty}(n)$. Let $n \in \omega \setminus \{0\}$. Then

\[
IH^g_{U, \infty}(n) = \max\{ih^g_U(z) : z \in \text{IProbl}_U^\infty, \dim z \leq n\},
\]

\[
IH^l_{U, \infty}(n) = \max\{ih^l_U(z) : z \in \text{IProbl}_U^\infty, \dim z \leq n\}.
\]

It is clear that $IH^g_{U, \infty}(n) \leq IH^l_{U, \infty}(n) \leq n$ for any $n \in \omega \setminus \{0\}$.

**Proposition 37.** Let $U = (A, B, F)$ be an information system. Then, for any $n \in \omega \setminus \{0\}$, $IH^g_{U, \infty}(n) = H^g_{U, \infty}(n)$ and $IH^l_{U, \infty}(n) = H^l_{U, \infty}(n)$.

**Proof.** Let $z = (\nu, f_1, \ldots, f_n) \in \text{IProbl}_U^\infty$. The problem $z_C = (\nu^C, f_1, \ldots, f_n)$ where,
for any \((\delta_1, \ldots, \delta_n) \in B^n, \nu^C(\delta_1, \ldots, \delta_n) = D_U(z) \setminus \nu(\delta_1, \ldots, \delta_n)\) is called the complementary problem for \(z\).

Let \(\Gamma\) be a decision tree over \(U\). We denote by \(\Gamma^-\) an inhibitory tree over \(U\) obtained from \(\Gamma\) by changing labels of terminal nodes: if a terminal node of \(\Gamma\) is labeled with \(t\) then the corresponding terminal node of \(\Gamma^-\) is labeled with \(\neq t\).

One can show that the decision tree \(\Gamma\) solves the problem \(z^C\) if and only if the inhibitory tree \(\Gamma^-\) solves the problem \(z\). From here it follows that \(ih^l_{\Gamma^\infty}(z) = h^l_{\Gamma^\infty}(z^C)\). Let \(n \in \omega \setminus \{0\}\). We now show that \(IH^l_{\Gamma^\infty}(n) = H^l_{\Gamma^\infty}(n)\). Let \(z \in \operatorname{Probl}_{\Gamma^\infty}^c\) and \(\dim z \leq n\). We know that \(\dim z = \dim z^C\) and \(ih^g_{\Gamma^\infty}(z) = h^g_{\Gamma^\infty}(z^C)\). Therefore \(IH^g_{\Gamma^\infty}(n) \leq H^g_{\Gamma^\infty}(n)\).

Let \(z \in \operatorname{Probl}_{\Gamma^\infty}^\infty\), \(\dim z \leq n\), and \(h^g_{\Gamma^\infty}(z) = H^g_{\Gamma^\infty}(n)\). From the proof of Proposition 36 it follows that the problem \(z\) can be chosen such that \(z = (\nu, f_1, \ldots, f_m)\) where \(m \leq n\) and \(\nu(\bar{\delta}_1) \cap \nu(\bar{\delta}_2) = \emptyset\) for any \(\bar{\delta}_1, \bar{\delta}_2 \in \text{Row}_U(z)\), \(\bar{\delta}_1 \neq \bar{\delta}_2\). Since \(F\) does not contain constant attributes, \(|\text{Row}_U(z)| \geq 2\). From here it follows that \(z^C \in \operatorname{Probl}_{\Gamma^\infty}^\infty\) and \(h^g_{\Gamma^\infty}(z) = h^g_{\Gamma^\infty}(z^C)\). Denote \(q = z^C\). We have \(ih^g_{\Gamma^\infty}(q) = h^g_{\Gamma^\infty}(q^C) = h^g_{\Gamma^\infty}(z) = H^g_{\Gamma^\infty}(n)\). Therefore \(IH^g_{\Gamma^\infty}(n) \geq H^g_{\Gamma^\infty}(n)\) and \(IH^l_{\Gamma^\infty}(n) = H^l_{\Gamma^\infty}(n)\).

We can prove in similar way that, for any \(n \in \omega \setminus \{0\}\), \(IH^l_{\Gamma^\infty}(n) = H^l_{\Gamma^\infty}(n)\).  

### 11.2 Problems with One-valued Decisions over Finite Information Systems. Global Approach

In this section, we study the behavior of Shannon function for decision trees for problems with one-valued decisions over finite information systems in the frameworks of the global approach.

Let \(U = (A, B, F)\) be a finite information system. Define the notion of independence dimension (or, in short, \(I\)-dimension) of information system \(U\). A subset \(\{f_1, \ldots, f_p\}\) of the set \(F\) is called an independent set if there exist two-element
subsets $B_1,\ldots,B_p$ of the set $B$ such that, for any $\delta_1 \in B_1,\ldots,\delta_p \in B_p$, the set $\text{Sol}_U((f_1,\delta_1)\ldots(f_p,\delta_p))$ is nonempty, i.e., the system of equations

$$\{f_1(x) = \delta_1,\ldots,f_p(x) = \delta_p\}$$

has a solution over $A$. If the subset $\{f_1,\ldots,f_p\}$ is not independent then it is called 
\textit{dependent}. The I-dimension of $U$ is the maximum cardinality of a subset of $F$ which is an independent set. Denote it by $I(U)$.

A subset $\{f_1,\ldots,f_p\}$ of the set $F$ is called \textit{redundant} if $p \geq 2$ and there exist $i \in \{1,\ldots,p\}$ and mapping $\mu : B^{p-1} \rightarrow B$ such that

$$f_i(a) = \mu(f_1(a),\ldots,f_{i-1}(a),f_{i+1}(a),\ldots,f_p(a))$$

for any $a \in A$. If $\{f_1,\ldots,f_p\}$ is not redundant it is called \textit{irredundant}. Denote $\text{ir}(U)$ the maximum cardinality of an irredundant subset of $F$. It is clear that each redundant subset of $F$ is a dependent set. Therefore

$$I(U) \leq \text{ir}(U) .$$

Let us remind the notions of tree-stable problem. A problem $z$ over $U$ is \textit{tree-stable} if $h_{ts}^0(z) = \dim z$. We denoted by $\text{ts}(U)$ the maximum dimension of a tree-stable problem over $U$. It is clear that, if a problem $z = (\nu,f_1,\ldots,f_m)$ is a tree-stable problem, then the set $\{f_1,\ldots,f_m\}$ is irredundant. Therefore

$$\text{ts}(U) \leq \text{ir}(U) .$$

Consider now three examples which show that $I(U)$ can be less than $\text{ts}(T)$, can be equal to $\text{ts}(T)$, and can be greater than $\text{ts}(T)$. Note that, for the case when $B$
contains exactly two elements, $I(T)$ is at most $ts(T)$ (see proof of Theorem 5.3 in [78]).

**Example 1.** Let $n$ be a natural number and $n \geq 3$. Consider a finite information system $U_1 = (A_1, \{0, 1\}, F_1)$ such that $A_1$ is the two-dimensional Euclidean plane and $F_1$ is the set of attributes corresponding to $n$ straight lines in the plane that form a convex polygon $P$ with $n$ sides. Each straight line divides the plane into open and closed half-planes on which the corresponding attribute has values 0 and 1, respectively. Two nonparallel straight lines divide the plane into four parts. There are no three straight lines that divide the plane into eight parts. Therefore $I(U_1) = 2$.

The membership problem for the polygon $P$ can be stated as a problem over $U_1$ with dimension $n$. One can show that this problem is tree-stable. Therefore $ts(U_1) \geq n$. It is clear that $ir(U_1) \leq |F_1| = n$. Thus $ts(U_1) = ir(U_1) = n$. As a result, $I(U_1) = 2$ and $ts(U_1) = ir(U_1) = n$.

**Example 2.** Let $n$ be a natural number. Consider a finite information system $U_2 = (A_2, \{0, 1\}, F_2)$ where the set $A_2$ contains $2^n$ elements and $F_2$ contains all possible nonconstant attributes defined on $A_2$ and with values from $\{0, 1\}$. One can show that there exist attributes $f_1, \ldots, f_n \in F_2$ such that, for any $b_1, \ldots, b_n \in \{0, 1\}$, the system of equations $\{f_1(x) = b_1, \ldots, f_n(x) = b_n\}$ has a solution over $A_2$. Therefore $I(U_2) \geq n$. Since $|A_2| = 2^n$, $I(U_2) \leq n$ and $I(U_2) = n$. Let $\nu' : \{0, 1\}^n \to \omega$ and, for any $\bar{a}, \bar{b} \in \{0, 1\}^n$, $\nu'(\bar{a}) \neq \nu'(\bar{b})$. It is not difficult to show that the problem $z' = (\nu', f_1, \ldots, f_n)$ is a tree-stable problem. Therefore $ts(U_2) \geq n$. One can show that, for any problem $z$ over $U_2$, $h_{ts}^0(z) \leq n$. Therefore $ts(U_2) = n$. Let $A_2 = \{a_1, \ldots, a_{2^n}\}$ and, for $i = 1, \ldots, 2^n - 1$, let $g_i$ be an attribute from $F_2$ which is equal to 1 only on the element $a_i$. One can show that $\{g_1, \ldots, g_{2^n-1}\}$ is an irredundant set. Therefore $ir(U_2) \geq 2^n - 1$. As a result, $I(U_2) = ts(U_2) = n$, and $ir(U_2) \geq 2^n - 1$.

**Example 3.** Let $n$ be an even natural number. Consider a finite information system $U_3 = (A_3, \{0, 1, 2, 3\}, F_3)$ where the set $A_3$ contains $2^n$ elements and $F_3$ contains all...
possible nonconstant attributes defined on $A_3$ and with values from $\{0, 1, 2, 3\}$. As for the case of information system $U_2$, one can show that $I(U_3) = n$ and $\text{ir}(U_3) \geq 2^n - 1$. One can show that there exist attributes $f_1, \ldots, f_{n/2} \in F_3$ such that, for any $b_1, \ldots, b_{n/2} \in \{0, 1, 2, 3\}$, the system of equations $\{f_1(x) = b_1, \ldots, f_{n/2}(x) = b_{n/2}\}$ has a solution over $A_3$. Let $\nu' : \{0, 1, 2, 3\}^{n/2} \rightarrow \omega$ and, for any $\bar{a}, \bar{b} \in \{0, 1, 2, 3\}^{n/2}$, $\nu'(\bar{a}) \neq \nu'(\bar{b})$. It is not difficult to show that the problem $z' = (\nu', f_1, \ldots, f_{n/2})$ is a tree-stable problem. Therefore $\text{ts}(U_2) \geq n/2$. One can show that, for any problem $z$ over $U_3$, $h_{U_3}(z) \leq n/2$. Therefore $\text{ts}(U_3) = n/2$. As a result, $\text{ts}(U_3) = n/2$, and $I(U_3) = n$.

Let us remind the definition of Shannon function $H^0_{U}(n)$. Let $n \in \omega \setminus \{0\}$. Then

$$H^0_{U}(n) = \max\{h^0_{U}(z) : z \in \text{Probl}_U, \dim z \leq n\}.$$ 

Our aim is to study the behavior of function $H^0_{U}(n)$ for an arbitrary finite information system $U$. We will prove the following theorem.

**Theorem 27.** Let $U = (A, B, F)$ be a finite information system, $k = |B|$, and $m = \max(\text{ts}(U), I(U))$. Then, for any $n \in \omega \setminus \{0\}$, the following statements hold:

a) if $n \leq m$ then $\frac{n}{\log_2 k} \leq H^0_{U}(n) \leq n$;

b) if $m < n \leq \text{ir}(U)$ then

$$\max\left\{\text{ts}(U), \frac{I(U)}{\log_2 k}, \log_k(n + 1)\right\} \leq H^0_{U}(n) \leq \min\left\{n - 1, 4(m + 1)^4(\log_2 n)^2 + 4(m + 1)^5(\log_2 n)(\log_2 k)\right\};$$

c) if $n \geq \text{ir}(U)$ then $H^0_{U}(n) = H^0_{U}(\text{ir}(U))$.

The behavior of the function $H^0_{U}(n)$ was studied in [78] (see Theorem 5.3) for the case of finite two-valued information systems where $|B| = 2$. In Theorem 27, the considered result is adapted for the case of the arbitrary finite information system.
The proof from [78] cannot be extended directly to this case since the behavior of independence dimension for many-valued information systems can be different in comparison with two-valued ones.

11.2.1 Proof of Theorem 27

First, we consider some auxiliary statements. For a problem \( z = (\nu, f_1, \ldots, f_n) \) over \( U \) with single-valued decisions, denote \( \text{Row}_U(z) \) the set of \( n \)-tuples \( (\delta_1, \ldots, \delta_n) \in B^n \) such that the equation system \( \{ f_1(x) = \delta_1, \ldots, f_n(x) = \delta_n \} \) has a solution over \( A \), \( N_U(z) \) the cardinality of the set \( \text{Row}_U(z) \), and \( S_U(z) \) the cardinality of the set \( \{ \nu(\delta_1, \ldots, \delta_n) : (\delta_1, \ldots, \delta_n) \in \text{Row}_U(z) \} \). The next statement follows from Theorem 4.6 from [78].

Lemma 15. Let \( z \) be a problem over finite information system \( U = (A, B, F) \) and \( k = |B| \). Then

\[
N_U(z) \leq (k^2 \dim z) I(U) .
\]

The next statement follows from Theorems 3.2 and 4.2 from [78].

Lemma 16. Let \( z \) be a problem over information system \( U = (A, B, F) \) and \( k = |B| \). Then

\[
h^\varphi_U(z) \geq \lceil \log_k S_U(z) \rceil .
\]

For any word \( \alpha \in \Omega_{F,B} \), we denote by \( \text{Alph}(\alpha) \) the set of letters from the alphabet \( \{(f, \delta) : f \in F, \delta \in B\} \) contained in \( \alpha \). Define the parameter \( M_U(z) \) for the problem \( z = (\nu, f_1, \ldots, f_n) \). For \( \bar{\delta} = (\delta_1, \ldots, \delta_n) \in B^n \), denote \( M_U(z, \bar{\delta}) \) the minimum length of a word \( \gamma \) such that \( \text{Alph}(\gamma) \subseteq \text{Alph}((f_1, \delta_1) \ldots (f_n, \delta_n)) \) and either \( \text{Sol}_U(\gamma) = \emptyset \) or the function \( \nu(f_1, \ldots, f_n(x)) \) is constant on the set \( \text{Sol}_U(\gamma) \). Then \( M_U(z) = \max \{ M_U(z, \bar{\delta}) : \bar{\delta} \in B^n \} \). The next statement follows from Theorems 3.5 and 4.1 from [78].
Lemma 17. Let $z$ be a problem over information system $\mathcal{U} = (A, B, F)$. Then
\[
h_{\mathcal{U}}^g(z) \leq M_{\mathcal{U}}(z) \log_2 N_{\mathcal{U}}(z) .
\]

For $\Gamma \in \text{Tree}_{\mathcal{U}}$, denote $L_w(\Gamma)$ the number of working nodes in $\Gamma$. The next statement follows from Lemma 3.4 from [78].

Lemma 18. Let $\Gamma \in \text{Tree}_{\mathcal{U}}$ and $k = |B|$. Then $L_w(\Gamma) \leq (k^{h(\Gamma)} - 1)/(k - 1)$.

We now adapt Lemma 5.24 from [78] for the case of the arbitrary finite information system.

Lemma 19. Let $\mathcal{U} = (A, B, F)$ be a finite information system, $k = |B|$, and $m = \max(\text{ts}(\mathcal{U}), I(\mathcal{U}))$. Then, for any natural $n \geq 2$, the following inequality holds:
\[
H_{\mathcal{U}}^g(n) \leq 4(m + 1)^4(\log_2 n)^2 + 4(m + 1)^5(\log_2 n)(\log_2 k).
\]

Proof. We begin with the overview of the proof. Let $z = (\nu, f_1, \ldots, f_r)$ be an arbitrary problem over $\mathcal{U}$ with dim $z = r \leq n$. We define a subset $J(z)$ of the set $F$ that is an extension of the set $\{f_1, \ldots, f_r\}$. Let $J(z) = \{f_1, \ldots, f_p\}$. We consider a problem $z' = (\nu', f_1, \ldots, f_p)$ such that $z'(a) = z(a)$ for any $a \in A$. The choice of $J(z)$ allows us to prove that
\[
N_{\mathcal{U}}(z') \leq 2^{2(m+1)^3(\log_2 n)^2 + 2(m+1)^4(\log_2 n)(\log_2 k)}
\]
and
\[
M_{\mathcal{U}}(z') \leq 2(m + 1).
\]

By Lemma [17],
\[
h_{\mathcal{U}}(z') \leq M_{\mathcal{U}}(z') \log_2 N_{\mathcal{U}}(z') \leq 4(m + 1)^4(\log_2 n)^2 + 4(m + 1)^5(\log_2 n)(\log_2 k).
\]

It is clear that $h_{\mathcal{U}}(z) = h_{\mathcal{U}}(z')$. Since $z$ is an arbitrary problem over $\mathcal{U}$ with dim $z \leq n$,
\[ H(n) \leq 4(m+1)^4(\log_2 n)^2 + 4(m+1)^5(\log_2 n)(\log_2 k). \] Note that both the description and the study of the set \( J(z) \) as well as the study of the problem \( z' \) require some effort.

It is clear that \( H(n) \leq n \). For an arbitrary \( p \in \omega \setminus \{0\} \), denote \( \nu_p \) the mapping of the set \( B^p \) into the set \( \omega \) such that \( \nu_p(\bar{\delta}_1) \neq \nu_p(\bar{\delta}_2) \) for any \( \bar{\delta}_1, \bar{\delta}_2 \in B^p \), \( \bar{\delta}_1 \neq \bar{\delta}_2 \).

By Lemma 15, for any problem \( z \) over \( \mathcal{U} \), the following inequality holds:

\[ N(z) \leq k^{2m}(\dim z)^m. \] (11.3)

Let \( f_1, \ldots, f_{m+1} \) be pairwise distinct attributes from \( F \) (if in the set \( F \) there are no \( m+1 \) pairwise distinct attributes, then \( H(n) \leq m \) for any natural \( n \geq 2 \), and the statement of the lemma holds). Denote

\[ z(f_1, \ldots, f_{m+1}) = (\nu_{m+1}, f_1, \ldots, f_{m+1}). \]

From \( H(n) \leq m \) it follows the existence of a decision tree \( \Gamma(f_1, \ldots, f_{m+1}) \) over \( \mathcal{U} \) such that \( h(\Gamma(f_1, \ldots, f_{m+1})) \leq m \) and the decision tree \( \Gamma(f_1, \ldots, f_{m+1}) \) solves the problem \( z(f_1, \ldots, f_{m+1}) \). Evidently, for any \( \delta_1, \ldots, \delta_{m+1} \in B \) and for any complete path \( \xi \) in the tree \( \Gamma(f_1, \ldots, f_{m+1}) \), either

\[ \text{Sol}_U(\pi(\xi)) \cap \text{Sol}_U((f_1, \delta_1) \ldots (f_{m+1}, \delta_{m+1})) = \emptyset \]

or

\[ \text{Sol}_U(\pi(\xi)) \subseteq \text{Sol}_U((f_1, \delta_1) \ldots (f_{m+1}, \delta_{m+1})). \]

By Lemma 18,

\[ L_w(\Gamma(f_1, \ldots, f_{m+1})) \leq k^m. \] (11.4)

Let \( f_1, \ldots, f_q \) be pairwise distinct attributes from \( F \) and let \( q \leq m \). Define a decision
tree \( \Gamma(f_1, \ldots, f_q) \) over \( \mathcal{U} \) in the following way. Let every working node of the tree \( \Gamma(f_1, \ldots, f_q) \) have exactly \( k \) edges issuing from it, and let every complete path in the tree \( \Gamma(f_1, \ldots, f_q) \) contain exactly \( q \) working nodes. Let \( \xi = v_1, d_1, \ldots, v_q, d_q, v_{q+1} \) be an arbitrary complete path in the tree \( \Gamma(f_1, \ldots, f_q) \). Then, for \( i = 1, \ldots, q \), the node \( v_i \) is labeled with the attribute \( f_i \), and the node \( v_{q+1} \) is labeled with the number 0.

For every problem \( z \) over \( \mathcal{U} \), we define by induction a subset \( J(z) \) of the set \( \mathcal{F} \). If \( \dim z \leq m \) then \( J(z) = \text{At}(z) \). Assume that for some \( n, n \geq m + 1 \), for any problem \( z' \) over \( \mathcal{U} \) with \( \dim z' < n \) the set \( J(z') \) has already been defined. Define the set \( J(z) \) for a problem \( z = (\nu, f_1, \ldots, f_n) \) over \( \mathcal{U} \). Let \( n = t(m+1) + q \), where \( t \in \omega \setminus \{0\} \) and \( 0 \leq q \leq m \). For \( i = 1, \ldots, t \), denote \( \Gamma_i = \Gamma(f_{(m+1)(i-1)+1}, \ldots, f_{(m+1)(i-1)+m+1}) \).

Define a decision tree \( \Gamma_{t+1} \) over \( \mathcal{U} \). If \( q = 0 \) then the tree \( \Gamma_{t+1} \) contains only the node labeled with the number 0. If \( q > 0 \) then \( \Gamma_{t+1} = \Gamma(f_{(m+1)t+1}, \ldots, f_{(m+1)t+q}) \).

Define decision trees \( G_1, \ldots, G_{t+1} \) from \( \text{Tree}_\mathcal{U} \) in the following way: \( G_1 = \Gamma_1 \) and, for \( i = 1, \ldots, t \), the tree \( G_{i+1} \) is obtained from the tree \( G_i \) by replacing of every terminal node \( v \) in the tree \( G_i \) with the tree \( \Gamma_{i+1} \) (the edge which had entered the node \( v \) will be entered the root of the tree \( \Gamma_{i+1} \)). Denote by \( \Gamma(z) \) the decision tree that consists of all nodes and edges of the tree \( G_{t+1} \) for each of which there exists a complete path \( \xi \) containing it and satisfying the condition \( \text{Sol}_\mathcal{U}(\pi(\xi)) \neq \emptyset \). One can show that \( \bigcup_{\xi \in \text{Path}(\Gamma(z))} \text{Sol}_\mathcal{U}(\pi(\xi)) = \mathcal{A} \). Denote \( c = 2m/(2m+1) \). One can easily show \( h(G_{t+1}) \leq mt + q \leq cn \). Therefore

\[
h(\Gamma(z)) \leq cn . \tag{11.5}
\]

From (11.4) and from the description of the tree \( \Gamma_{t+1} \) it follows that \( |\text{At}(G_{t+1})| \leq tk^m + q \leq nk^m \). Using these inequalities and the inequality (11.3) we conclude that the tree \( G_{t+1} \) contains at most \( k^{2m}(nk^m)^m = n^m k^{m^2 + 2m} \) complete paths \( \xi \) such that
Sol_{\mathcal{U}}(\pi(\xi)) \neq \emptyset. Therefore

\[ |\text{Path}(\Gamma(z))| \leq n^m k^{m^2+2m}. \tag{11.6} \]

We correspond to every complete path \( \xi \) in the tree \( \Gamma(z) \) a problem \( z_\xi \) over \( \mathcal{U} \). Let \( \{f_{i_1}, \ldots, f_{i_p}\} \) be the set of attributes from \( F \) attached to working nodes of the path \( \xi \). Then \( z_\xi = (\nu_p, f_{i_1}, \ldots, f_{i_p}) \). From (11.5) it follows that, for any \( \xi \in \text{Path}(\Gamma(z)) \), the inequality

\[ \dim z_\xi \leq cn \tag{11.7} \]

holds. Hence, by assumption, the set \( J(z_\xi) \) has already been determined for any \( \xi \in \text{Path}(\Gamma(z)) \). Set

\[ J(z) = \text{At}(z) \cup \left( \bigcup_{\xi \in \text{Path}(\Gamma(z))} J(z_\xi) \right). \tag{11.8} \]

For \( n \in \omega \setminus \{0\} \), denote

\[ J_{\mathcal{U}}(n) = \max\{|J(z)| : z \in \text{Probl}_{\mathcal{U}}, \dim z \leq n\}. \]

The inequality

\[ J_{\mathcal{U}}(n) \leq n^{2(m+1)^2 \ln n} k^{2(m+1)^3 \ln n} \tag{11.9} \]

will be proven by induction on \( n \geq 1 \). It is clear that if \( n \leq m \) then \( J_{\mathcal{U}}(n) \leq n \). Hence for \( n \leq m \) the inequality (11.9) holds. Let for some \( n, n \geq m + 1 \), for any \( n', 1 \leq n' < n \), the inequality (11.9) hold. Let us show that it holds also for \( n \). Let \( z \in \text{Probl}_{\mathcal{U}} \) and \( \dim z \leq n \). If \( \dim z < n \), then using inductive hypothesis one can show \( |J(z)| \leq n^{2(m+1)^2 \ln n} k^{2(m+1)^3 \ln n} \). Let \( \dim z = n \). Evidently, \( 1 \leq \lfloor cn \rfloor < n \) and \( J_{\mathcal{U}}(\lfloor cn \rfloor) \geq 1 \). Using (11.6) - (11.8) obtain \( |J(z)| \leq n + n^m k^{m^2+2m} J_{\mathcal{U}}(\lfloor cn \rfloor) \leq n^{2(m+1)^2 \ln n} k^{2(m+1)^3 \ln n} \). Therefore

\[ J_{\mathcal{U}}(n) \leq n^{2(m+1)^2 \ln n} k^{2(m+1)^3 \ln n}. \]
\[ n^m k^{m^2 + 2m + 1} J_U([cn]). \] Using the inductive hypothesis obtain

\[
|J(z)| \leq n^m k^{(m+1)^2 ([cn])^2 (m+1)^2 \ln(cn) k^{2(m+1)^3 \ln(cn)}} \leq n^{m+2(m+1)^2 \ln(cn)} k^{(m+1)^2 + 2(m+1)^3 \ln(cn)}.
\]

From the inequality \( \ln(1 + 1/r) > 1/(r + 1) \) which is true for any natural \( r \) it follows that \( \ln c < -1/(2m + 1) < -1/2(m + 1) \). Hence

\[
|J(z)| \leq n^{2(m+1)^2 \ln n} k^{2(m+1)^3 \ln n}.
\]

Since \( z \) is an arbitrary problem over \( U \) such that \( \dim z \leq n \), the inequality (11.9) holds.

Prove the following statement by induction on \( n \). Let \( z = (\nu, f_1, \ldots, f_n) \in \text{Probl}_U \),
\[
J(z) = \{ f_1, \ldots, f_p \}, \bar{\delta} = (\delta_1, \ldots, \delta_p) \in B^p,
\]
let
\[
\alpha(z, \bar{\delta}) = (f_1, \delta_1) \ldots (f_p, \delta_p)
\]
and \( \beta(z, \bar{\delta}) = (f_1, \delta_1) \ldots (f_n, \delta_n) \). Then there exists a word \( \gamma(z, \bar{\delta}) \) from the set \( \Omega_{F,B} \) such that \( \text{Alph}(\gamma(z, \bar{\delta})) \subseteq \text{Alph}(\alpha(z, \bar{\delta})) \), \( \text{Sol}_U(\gamma(z, \bar{\delta})) \subseteq \text{Sol}_U(\beta(z, \bar{\delta})) \), and the length of the word \( \gamma(z, \bar{\delta}) \) is at most \( 2(m + 1) \).

For \( n \leq 2(m + 1) \), this statement is true since we can take the word \( \beta(z, \bar{\delta}) \) as the word \( \gamma(z, \bar{\delta}) \). Suppose that for certain \( n, n \geq 2(m + 1) + 1 \), the statement is true for any problem \( z \) over \( U \) with \( \dim z < n \). Let us show that the considered statement holds for an arbitrary problem \( z = (\nu, f_1, \ldots, f_n) \) over \( U \). Let \( J(z) = \{ f_1, \ldots, f_p \} \) and \( \bar{\delta} = (\delta_1, \ldots, \delta_p) \in B^p \). One can show that \( \text{At}(\Gamma(z)) \subseteq J(z) \). Consider a directed path \( \kappa = v_1, d_1, \ldots, v_r, d_r, v_{r+1} \) in the tree \( \Gamma(z) \) starting in the root and possessing the following properties:

1) if the node \( v_i, i \in \{1, \ldots, r\} \), is labeled with an attribute \( f_l \), then the edge \( d_i \) is labeled with \( \delta_l \);
2) if \( v_{r+1} \) is a working node in the tree \( \Gamma(z) \) which is labeled with the attribute \( f_i \) then an edge labeled with \( \delta_l \) does not issue from \( v_{r+1} \).

First, assume that \( \kappa \) is a complete path in the tree \( \Gamma(z) \). Let \( n = t(m + 1) + q \) where \( t \geq 1 \) and \( 0 \leq q \leq m \). For \( i = 1, \ldots, t \), denote

\[
\Gamma_i = \Gamma(f_{(m+1)(i-1)+1}, \ldots, f_{(m+1)(i-1)+m+1})
\]

Define a decision tree \( \Gamma_{t+1} \) over \( \mathcal{U} \). If \( q = 0 \), then \( \Gamma_{t+1} \) consists of the root labeled with 0. If \( q > 0 \), then \( \Gamma_{t+1} = \Gamma(f_{(m+1)t+1}, \ldots, f_{(m+1)t+q}) \). Define words \( \beta_1, \ldots, \beta_{t+1} \).

For \( i = 1, \ldots, t \), let

\[
\beta_i = (f_{(m+1)(i-1)+1}, \delta_{(m+1)(i-1)+1}) \cdots (f_{(m+1)(i-1)+m+1}, \delta_{(m+1)(i-1)+m+1})
\]

If \( q = 0 \), then \( \beta_{t+1} = \lambda \). If \( q > 0 \), then

\[
\beta_{t+1} = (f_{(m+1)t+1}, \delta_{(m+1)t+1}) \cdots (f_{(m+1)t+q}, \delta_{(m+1)t+q})
\]

Evidently, \( \beta(z, \overline{\delta}) = \beta_1 \cdots \beta_{t+1} \). One can show that the word \( \pi(\kappa) \) can be represented in the form \( \pi(\kappa) = \pi(\xi_1) \cdots \pi(\xi_{t+1}) \) where \( \xi_i \) is a complete path in the tree \( \Gamma_i \), \( i = 1, \ldots, t + 1 \).

Let there exist \( i \in \{1, \ldots, t\} \) such that \( \text{Sol}_\mathcal{U}(\beta_i) \cap \text{Sol}_\mathcal{U}(\pi(\xi_i)) = \emptyset \). Denote \( \gamma = \beta_i \pi(\xi_i) \). It is clear that \( \text{Alph}(\gamma) \subseteq \text{Alph}(\alpha(z, \overline{\delta})) \) and \( \text{Sol}_\mathcal{U}(\gamma) = \emptyset \). Hence \( \text{Sol}_\mathcal{U}(\gamma) \subseteq \text{Sol}_\mathcal{U}(\beta(z, \overline{\delta})) \) and the length of the word \( \gamma \) is at most \( m + 1 + m < 2(m + 1) \). Thus, in the considered case the word \( \gamma \) can be taken as the word \( \gamma(z, \overline{\delta}) \).

Let \( \text{Sol}_\mathcal{U}(\beta_i) \cap \text{Sol}_\mathcal{U}(\pi(\xi_i)) \neq \emptyset \) for \( i = 1, \ldots, t \). Then, as mentioned above, \( \text{Sol}_\mathcal{U}(\pi(\xi_i)) \subseteq \text{Sol}_\mathcal{U}(\beta_i) \) for \( i = 1, \ldots, t \). Evidently, \( \text{Sol}_\mathcal{U}(\pi(\xi_{t+1})) = \text{Sol}_\mathcal{U}(\beta_{t+1}) \) and hence

\[
\text{Sol}_\mathcal{U}(\pi(\kappa)) \subseteq \text{Sol}_\mathcal{U}(\beta(z, \overline{\delta})) \cdot \tag{11.10}
\]
Consider the problem $z_k$. Let $z_k = (\nu_1, f_{j_1}, \ldots, f_{j_k})$ and $J(z_k) = \{f_{j_1}, \ldots, f_{j_k}\}$. From (11.8) it follows that $J(z_k) \subseteq J(z)$. Denote $\bar{\delta}' = (\delta_{j_1}, \ldots, \delta_{j_k})$. Using (11.7) obtain $\dim z_k < n$. From this inequality and from the inductive hypothesis it follows that there exists a word $\gamma(z_k, \bar{\delta}') \in \Omega_{F,B}$ such that $\text{Alph}(\gamma(z_k, \bar{\delta}')) \subseteq \text{Alph}(\alpha(z_k, \bar{\delta}'))$, $	ext{Sol}_U(\gamma(z_k, \bar{\delta}')) \subseteq \text{Sol}_U(\beta(z_k, \bar{\delta}'))$, and the length of the word $\gamma(z_k, \bar{\delta}')$ is at most $2(m + 1)$. It is clear that $\text{Alph}(\alpha(z_k, \bar{\delta}')) \subseteq \text{Alph}(\alpha(z, \bar{\delta}))$ and $\text{Alph}(\gamma(z_k, \bar{\delta}')) \subseteq \text{Alph}(\alpha(z, \bar{\delta}))$. One can easily show

$$\text{Sol}_U(\pi(\kappa)) = \text{Sol}_U(\beta(z_k, \bar{\delta}')).$$

Using (11.10) obtain $\text{Sol}_U(\gamma(z_k, \bar{\delta}')) \subseteq \text{Sol}_U(\beta(z, \bar{\delta}))$. Hence in this case the word $\gamma(z_k, \bar{\delta}')$ can be taken as the word $\gamma(z, \bar{\delta})$.

Suppose now that the path $\kappa$ is not a complete path in the tree $\Gamma(z)$. Evidently, there exists a complete path $\xi$ in the tree $\Gamma(z)$ containing the node $v_{r+1}$. Consider the problem $z_\xi$. Let $z_\xi = (\nu_1, f_{j_1}, \ldots, f_{j_k})$ and $J(z_\xi) = \{f_{j_1}, \ldots, f_{j_k}\}$. From (11.8) it follows that $J(z_\xi) \subseteq J(z)$. Denote $\bar{\delta}' = (\delta_{j_1}, \ldots, \delta_{j_k})$. Recalling that the path $\kappa$ is not a complete path in the tree $\Gamma(z)$ one can show $\text{Sol}_U(\beta(z_\xi, \bar{\delta}')) = \emptyset$. Using (11.7) obtain $\dim z_\xi < n$. From this inequality and from the inductive hypothesis it follows that there exists a word $\gamma(z_\xi, \bar{\delta}') \in \Omega_{F,B}$ such that $\text{Alph}(\gamma(z_\xi, \bar{\delta}')) \subseteq \text{Alph}(\alpha(z_\xi, \bar{\delta}'))$, $\text{Sol}_U(\gamma(z_\xi, \bar{\delta}')) \subseteq \text{Sol}_U(\beta(z_\xi, \bar{\delta}'))$, and the length of the word $\gamma(z_\xi, \bar{\delta}')$ is at most $2(m + 1)$. It is clear that $\text{Alph}(\alpha(z_\xi, \bar{\delta}')) \subseteq \text{Alph}(\alpha(z, \bar{\delta}))$. Therefore $\text{Alph}(\gamma(z_\xi, \bar{\delta}')) \subseteq \text{Alph}(\alpha(z, \bar{\delta}))$. From the relations $\text{Sol}_U(\gamma(z_\xi, \bar{\delta}')) \subseteq \text{Sol}_U(\beta(z_\xi, \bar{\delta}')) = \emptyset$ it follows that

$$\text{Sol}_U(\gamma(z_\xi, \bar{\delta}')) \subseteq \text{Sol}_U(\beta(z, \bar{\delta})).$$

Thus, in the considered case the word $\gamma(z_\xi, \bar{\delta}')$ can be taken as the word $\gamma(z, \bar{\delta})$.

Let $n \geq 2$. Consider an arbitrary problem $z$ over $\mathcal{U}$ with $\dim z \leq n$. Let $z = (\nu, f_1, \ldots, f_r)$ and $J(z) = \{f_1, \ldots, f_p\}$. Consider also the problem $z' = (\nu', f_1, \ldots, f_p)$
where \( \nu' : B^p \to \omega \) and the equality \( \nu'(\bar{\delta}) = \nu(\delta_1, \ldots, \delta_r) \) holds for any tuple \( \bar{\delta} = (\delta_1, \ldots, \delta_p) \in B^p \). Using (11.3) and (11.9) obtain

\[
N_U(z') \leq k^{2m} (n^{2(m+1)^2} \ln n)_k^{2(m+1)^3} \ln n)^m \\
= n^{2m(m+1)^2} \ln n)_k^{2m+2m(m+1)^3} \ln n} \\
\leq 2^{2(m+1)^3(\log_2 n)^2 + 2(m+1)^4(\log_2 n)(\log_2 k)} .
\]

(11.11)

Let us show that

\[ M_U(z') \leq 2(m + 1) . \]

(11.12)

Let \( \bar{\delta} = (\delta_1, \ldots, \delta_p) \in B^p \). Then, by proved above, there exists a word \( \gamma(z, \bar{\delta}) \) from the set \( \Omega_{F,B} \) such that \( \text{Alph}(\gamma(z, \bar{\delta})) \subseteq \text{Alph}(\alpha(z, \bar{\delta})) \),

\[ \text{Sol}_U(\gamma(z, \bar{\delta})) \subseteq \text{Sol}_U(\beta(z, \bar{\delta})) , \]

and the length of the word \( \gamma(z, \bar{\delta}) \) is at most \( 2(m + 1) \). It is clear that \( \text{Alph}(\gamma(z, \bar{\delta})) \subseteq \{ (f_1, \delta_1), \ldots, (f_p, \delta_p) \} \). Taking into account \( \text{Sol}_U(\gamma(z, \bar{\delta})) \subseteq \text{Sol}_U(\beta(z, \bar{\delta})) \) we can easily show that either \( \text{Sol}_U(\gamma(z, \bar{\delta})) = \emptyset \) or the function \( \nu'(f_1(x), \ldots, f_p(x)) \) is constant on the set \( \text{Sol}_U(\gamma(z, \bar{\delta})) \). Therefore \( M_U(T, \bar{\delta}) \leq 2(m + 1) \). Recalling that \( \bar{\delta} \) is an arbitrary tuple from \( B^p \) we conclude that the inequality (11.12) holds. From Lemma 17 and from inequalities (11.11) and (11.12) it follows that \( h^0_U(z') \leq M_U(z') \log_2 N_U(z') \leq 4(m + 1)^4(\log_2 n)^2 + 4(m + 1)^5(\log_2 n)(\log_2 k) \). Evidently, for any element \( a \in A \), the equality \( z(a) = z'(a) \) holds. Therefore \( h^0_U(z) = h^0_U(z') \). Since \( z \) is an arbitrary problem over \( U \) with \( \dim z \leq n \), \( H^0_U(n) \leq 4(m + 1)^4(\log_2 n)^2 + 4(m + 1)^5(\log_2 n)(\log_2 k) \).

Proof of Theorem 27. a). Let \( n \leq \text{ts}(U) \), and \( z = (\nu, f_1, \ldots, f_{\text{ts}(U)}) \) be a problem over \( U \) such that \( h^0_U(z) = \text{ts}(U) \). Consider the problem \( z' = (\nu_n, f_1, \ldots, f_n) \) over \( U \) (the mapping \( \nu_p \) for any natural \( p \) is defined at the beginning of the proof of Lemma 19). Assume that \( h^0_U(z') < n \). One can show that in this case there exists a decision
tree $\Gamma'$ over $\mathcal{U}$ such that $\Gamma'$ solves the problem $z'$, $h(\Gamma') < n$, and $\text{Sol}_{\mathcal{U}}(\pi(\xi)) \neq \emptyset$ for any complete path $\xi$ of $\Gamma'$. It is clear that, for any terminal node of $\Gamma'$, the solution of $z'$ attached to this node allows one to restore the values of the attributes $f_1, \ldots, f_n$. Therefore, by computing values of additional attributes $f_{n+1}, \ldots, f_{\text{ts}(\mathcal{U})}$, one can transform $\Gamma'$ into a decision tree $\Gamma$ over $\mathcal{U}$ which solves $z$ and for which $h(\Gamma) < n + \text{ts}(\mathcal{U}) - n = \text{ts}(\mathcal{U})$. In this case $h^0_{\mathcal{U}}(z) < \text{ts}(\mathcal{U})$ which is impossible. Therefore $h^0_{\mathcal{U}}(z') = n$ and $H^0_{\mathcal{U}}(n) \geq n$. It is clear that $H^0_{\mathcal{U}}(n) \leq n$. Thus, $H^0_{\mathcal{U}}(n) = n$.

Let $n \leq I(\mathcal{U})$. Then there exist attributes $f_1, \ldots, f_n \in F$ and two-element subsets $B_1, \ldots, B_n$ of the set $B$ such that, for any $\delta_1 \in B_1, \ldots, \delta_n \in B_n$, the set

$$\text{Sol}_{\mathcal{U}}((f_1, \delta_1) \ldots (f_n, \delta_n))$$

is nonempty. Consider the problem $z = (\nu_n, f_1, \ldots, f_n)$ over $\mathcal{U}$. It is clear that $S_{\mathcal{U}}(z) \geq 2^n$. By Lemma 16 $h^0_{\mathcal{U}}(z) \geq \log_k 2^n = \frac{n}{\log_2 k}$ where $k = |B|$. Using the equality $\dim z_n \leq H^0_{\mathcal{U}}(n)$. It is clear that $H^0_{\mathcal{U}}(n) \leq n$.

b). Let $m < n \leq \text{ir}(\mathcal{U})$. From the part a) of the proof it follows that $\text{ts}(\mathcal{U}) \leq H^0_{\mathcal{U}}(n)$ and $\frac{I(\mathcal{U})}{\log_2 k} \leq H^0_{\mathcal{U}}(n)$. Let us show that $\log_k (n + 1) \leq H^0_{\mathcal{U}}(n)$. From the inequality $n \leq \text{ir}(\mathcal{U})$ it follows that there exists an independent subset $\{f_1, \ldots, f_n\}$ of the set $F$. It is clear that $n \geq 2$. For $i = 1, \ldots, n$, denote $z_i = (\nu_i, f_1, \ldots, f_i)$. Since $f_1 \not\equiv \text{const}$, $N_{\mathcal{U}}(z_1) \geq 2$. Let us show that $N_{\mathcal{U}}(z_i) < N_{\mathcal{U}}(z_{i+1})$ for $i = 1, \ldots, n - 1$. Assume the contrary: let $N_{\mathcal{U}}(z_i) = N_{\mathcal{U}}(z_{i+1})$ for some $i \in \{1, \ldots, n - 1\}$. One can show that in this case there exists a mapping $\mu : B^i \to B$ such that $f_{i+1}(a) = \mu(f_1(a), \ldots, f_i(a))$ for any $a \in A$ which is impossible. Thus, $N_{\mathcal{U}}(z_1) \geq 2$ and $N_{\mathcal{U}}(z_i) < N_{\mathcal{U}}(z_{i+1})$ for $i = 1, \ldots, n - 1$. Therefore $N_{\mathcal{U}}(z_n) \geq n + 1$ and $S_{\mathcal{U}}(z_n) \geq n + 1$. By Lemma 16 $h^0_{\mathcal{U}}(z_n) \geq \log_k (n + 1)$ where $k = |B|$. Using the equality $\dim z_n = n$ obtain

$$\log_k (n + 1) \leq H^0_{\mathcal{U}}(n).$$

It is clear that $H^0_{\mathcal{U}}(n) \leq n - 1$. The inequality $H^0_{\mathcal{U}}(n) \leq 4(m+1)^4(\log_2 n)^2 + 4(m+}$
Let \( n \geq \text{ir}(\mathcal{U}) \). Consider an arbitrary problem \( z \) over \( \mathcal{U} \) such that \( \dim z \leq n \). Let \( \{f_1, \ldots, f_t\} \) be an independent subset of the set \( \text{At}(z) \) with maximum cardinality. It is clear that \( t \leq \text{ir}(\mathcal{U}) \). One can show that there exists a mapping \( \nu : B^t \to \omega \) such that, for the problem \( z' = (\nu, f_1, \ldots, f_t) \), the equality \( z(a) = z'(a) \) holds for any \( a \in A \). It is clear that \( h_{\mathcal{U}}^g(z) = h_{\mathcal{U}}^g(z') \). Therefore \( h_{\mathcal{U}}^g(z) \leq H_{\mathcal{U}}^3(t) \leq H_{\mathcal{U}}^3(\text{ir}(\mathcal{U})) \). Since \( z \) is an arbitrary problem over \( \mathcal{U} \) such that \( \dim z \leq n \), \( H_{\mathcal{U}}^3(n) \leq H_{\mathcal{U}}^3(\text{ir}(\mathcal{U})) \). It is clear that \( H_{\mathcal{U}}^3(n) \geq H_{\mathcal{U}}^3(\text{ir}(\mathcal{U})) \). Thus, \( H_{\mathcal{U}}^3(n) = H_{\mathcal{U}}^3(\text{ir}(\mathcal{U})) \). □

11.3 Behavior of Shannon Functions for Problems with Many-valued Decisions

In this section, we consider the behavior of Shannon functions for local and global approaches, for decision and inhibitory trees for problems with many-valued decisions over finite and infinite information systems.

11.3.1 Local Approach. Infinite Information Systems

We will say that an information system \( \mathcal{U} = (A, B, F) \) is restricted if there exists a number \( r \in \omega \setminus \{0\} \) such that, for each consistent (having a solution over the set \( A \)) system of equations of the kind

\[
\{ f_1(x) = \delta_1, \ldots, f_m(x) = \delta_m \}
\]

where \( m \in \omega \setminus \{0\} \), \( f_1, \ldots, f_m \in F \) and \( \delta_1, \ldots, \delta_m \in B \), there exists a subsystem of this system which has the same set of solutions over the set \( A \) and contains at most \( r \) equations.

In [73] (see also [76, 78]), the behavior of Shannon function \( H_{\mathcal{U}}^1(n) \) was studied for infinite information systems. The next theorem follows from the obtained results.
Theorem 28. Let $\mathcal{U}$ be an infinite information system. Then the following statements hold:

a) if $\mathcal{U}$ is restricted, then $IH_{\mathcal{U},\infty}^l(n) = H_{\mathcal{U},\infty}^l(n) = \Theta(\log n)$;

b) if $\mathcal{U}$ is not restricted, then $IH_{\mathcal{U},\infty}^l(n) = H_{\mathcal{U},\infty}^l(n) = n$ for each $n \in \omega \setminus \{0\}$.

Now we extend Example 8.3 from [2] to the case of problems with many-valued decisions.

Example 4. Let $m, t \in \omega \setminus \{0\}$. We denote by $Pol(m)$ the set of all polynomials which have integer coefficients and depend on variables $x_1, \ldots, x_m$. We denote by $Pol(m, t)$ the set of all polynomials from $Pol(m)$ such that the degree of each polynomial is at most $t$. We define information systems $\mathcal{U}(m)$ and $\mathcal{U}(m, t)$ as follows: $\mathcal{U}(m) = (\mathbb{R}^m, E, F(m))$ and $\mathcal{U}(m, t) = (\mathbb{R}^m, E, F(m, t))$ where $E = \{-1, 0, +1\}$, $F(m) = \{\text{sign}(p) : p \in Pol(m), \text{sign}(p) \not\equiv \text{const}\}$ and $F(m, t) = \{\text{sign}(p) : p \in Pol(m, t), \text{sign}(p) \not\equiv \text{const}\}$. Here $\text{sign}(x) = -1$ if $x < 0$, $\text{sign}(x) = 0$ if $x = 0$, and $\text{sign}(x) = +1$ if $x > 0$. One can prove that $IH_{\mathcal{U}(m),\infty}^l(n) = H_{\mathcal{U}(m),\infty}^l(n) = n$ for each $n \in \omega \setminus \{0\}$, $IH_{\mathcal{U}(m,1),\infty}^l(n) = H_{\mathcal{U}(m,1),\infty}^l(n) = \Theta(\log n)$, and if $m > 1$ or $t > 1$ then

$$IH_{\mathcal{U}(m,t),\infty}^l(n) = H_{\mathcal{U}(m,t),\infty}^l(n) = n$$

for each $n \in \omega \setminus \{0\}$.

11.3.2 Local Approach. Finite Information Systems

Let $\mathcal{U} = (A, B, F)$ be a finite information system.

Let us remind that a set of attributes $\{f_1, \ldots, f_n\} \subseteq F$ is called redundant if $n \geq 2$ and there exist $i \in \{1, \ldots, n\}$ and $\mu : B^{n-1} \to B$ such that

$$f_i(a) = \mu(f_1(a), \ldots, f_{i-1}(a), f_{i+1}(a), \ldots, f_n(a))$$
for each \( a \in A \). If the set \( \{f_1, \ldots, f_n\} \) is not redundant then it is called \textit{irredundant}. We denoted by \( \text{ir}(U) \) the maximum number of attributes in an irredundant subset of the set \( F \).

A systems of equations over \( U \)

\[
\{ f_1(x) = \delta_1, \ldots, f_n(x) = \delta_n \}
\]  

is called \textit{cancelable} if \( n \geq 2 \) and there exists a number \( i \in \{1, \ldots, n\} \) such that the system of equations

\[
\{ f_1(x) = \delta_1, \ldots, f_{i-1}(x) = \delta_{i-1}, f_{i+1}(x) = \delta_{i+1}, \ldots, f_n(x) = \delta_n \}
\]

has the same set of solutions over \( A \) just as the system \( \text{(11.13)} \). If the system \( \text{(11.13)} \) is not cancelable then it is called \textit{uncancelable}. We denote by \( \text{un}(U) \) the maximum number of equations in an uncancelable consistent system of equations over \( U \).

Let \( S = \{ f_1(x) = \delta_1, \ldots, f_n(x) = \delta_n \} \) be an uncancelable system of equations over \( U \). It is clear that the set of attributes \( \{f_1, \ldots, f_n\} \) is irredundant. Therefore \( \text{un}(U) \leq \text{ir}(U) \). Evidently, \( 1 \leq \text{un}(U) \). Thus

\[
1 \leq \text{un}(U) \leq \text{ir}(U) .
\]

In \cite{75}, the behavior of Shannon function \( H_{U,\infty}^l(n) \) for finite information systems was studied. The next theorem follows from the obtained results (see Theorem 4 from \cite{75}) and Proposition 37.

**Theorem 29.** Let \( U = (A, B, F) \) be a finite information system, and \( n \in \omega \setminus \{0\} \). Then the following statements hold:

a) if \( n \leq \text{un}(U) \) then \( \text{IH}_{U,\infty}^l(n) = H_{U,\infty}^l(n) = n \);
b) if \( \text{un}(\mathcal{U}) \leq n \leq \text{ir}(\mathcal{U}) \) then
\[
\max\{\text{un}(\mathcal{U}), \log_k(n+1)\} \leq IH_{\mathcal{U},\infty}^I(n) \leq \min\{n, 2^{\text{un}(\mathcal{U})^2 \log_2 2(kn + 1)}\}
\]

where \( k = |B|; \)

c) if \( n \geq \text{ir}(\mathcal{U}) \) then \( IH_{\mathcal{U},\infty}^I(n) = H_{\mathcal{U},\infty}^I(n) = IH_{\mathcal{U},\infty}^I(\text{ir}(\mathcal{U})) = H_{\mathcal{U},\infty}^I(\text{ir}(\mathcal{U})). \)

We now consider Example 8.7 from [2].

Example 5. Denote by \( P \) the set of all points in the two-dimensional Euclidean plane. Consider an arbitrary straight line \( l \), which divides the plane into positive and negative open half-planes, and the line \( l \) itself. Assign a function \( f : P \to \{0, 1\} \) to the line \( l \). The function \( f \) takes the value 1 if a point is situated on the positive half-plane, and \( f \) takes the value 0 if a point is situated on the negative half-plane or on the line \( l \). Denote by \( F \) the set of functions which correspond to certain \( r \) mutually disjoint finite classes of parallel straight lines. Consider a finite information system \( \mathcal{U} = (P, \{0, 1\}, F) \). One can show that \( \text{ir}(\mathcal{U}) = |F| \) and \( \text{un}(\mathcal{U}) \leq 2r \).

11.3.3 Global Approach. Infinite Information Systems

Extend the notion of independence dimension (or, in short, I-dimension) to the case of infinite information system \( \mathcal{U} = (A, B, F) \). A finite subset \( \{f_1, \ldots, f_p\} \) of the set \( F \) is called an independent set if there exist two-element subsets \( B_1, \ldots, B_p \) of the set \( B \) such that, for any \( \delta_1 \in B_1, \ldots, \delta_p \in B_p \), the system of equations
\[
\{f_1(x) = \delta_1, \ldots, f_p(x) = \delta_p\}
\]
(11.14)
is consistent (has a solution over the set \( A \)). If for any natural \( p \) there exists a subset of the set \( F \), which cardinality is equal to \( p \) and which is an independent set, then
we will say that the information system \( U \) has infinite I-dimension. Otherwise, I-dimension of \( U \) is the maximum cardinality of a subset of \( F \), which is an independent set.

The notion of I-dimension is closely connected with the well-known notion of Vapnik-Chervonenkis dimension \([79]\). In particular, an information system

\[
(A, \{0, 1\}, F)
\]

has finite I-dimension if and only if it has finite VC-dimension \([80]\).

Now we consider the condition of decomposition for the information system \( U \). Let \( p \in \omega \setminus \{0\} \). A nonempty subset \( D \) of the set \( A \) will be called a \((p, U)\)-set if \( D \) coincides with the set of solutions over \( A \) of a system of the kind \([11.14]\) where \( f_1, \ldots, f_p \in F \) and \( \delta_1, \ldots, \delta_p \in B \) (we admit that among the attributes \( f_1, \ldots, f_p \) there are identical ones).

We will say that the information system \( U \) satisfies the condition of decomposition if there exist numbers \( m, t \in \omega \setminus \{0\} \) such that every \((m+1, U)\)-set is a union of \( t \) sets each of which is an \((m, U)\)-set (we admit that among the considered \( t \) sets there are identical ones).

We now consider Example 8.8 from \([2]\).

**Example 6.** Let \( P \) be the set of all points in the two-dimensional Euclidean plane and \( l \) be a straight line in the plane. This line divides the plane into two open half-planes \( H_1 \) and \( H_2 \), and the line \( l \). We correspond one attribute to the line \( l \). This attribute takes value 0 on points from \( H_1 \), and value 1 on points from \( H_2 \) and \( l \). Denote by \( F \) the set of all attributes corresponding to lines in the plane. Let us consider the information system \( U = (P, \{0, 1\}, F) \). The information system \( U \) has finite I-dimension: there are no three lines which divide the plane into eight domains.

The information system \( U \) satisfies the condition of decomposition: each \((4, U)\)-set is
a union of two \((3, \mathcal{U})\)-sets.

In [77], the behavior of Shannon functions \(H_{\mathcal{U}}^g(n)\) and \(L_{\mathcal{U}}^g(n)\) for infinite information systems was studied. The next theorem follows from the obtained results (see Theorem 2.1 from [77]) and Propositions 36 and 37.

**Theorem 30.** Let \(\mathcal{U} = (A, B, F)\) be an infinite information system. Then the following statements hold:

a) if \(\mathcal{U}\) has finite I-dimension and satisfies the condition of decomposition then, for any \(\varepsilon > 0\), \(0 < \varepsilon < 1\), \(IH_{\mathcal{U}, \infty}^g(n) = H_{\mathcal{U}, \infty}^g(n) = \Omega(\log n)\) and \(IH_{\mathcal{U}, \infty}^g(n) = H_{\mathcal{U}, \infty}^g(n) = O((\log n)^{1+\varepsilon})\).

b) if \(\mathcal{U}\) has infinite I-dimension or does not satisfy the condition of decomposition then, for any \(n \in \omega \setminus \{0\}\), \(IH_{\mathcal{U}, \infty}^g(n) = H_{\mathcal{U}, \infty}^g(n) = n\).

Let us consider Example 8.12 from [2].

**Example 7.** Let \(m, t \in \omega \setminus \{0\}\). We consider the same information systems \(\mathcal{U}(m)\) and \(\mathcal{U}(m, t)\) as in Example 4. One can prove that \(\mathcal{U}(m, t)\) has finite I-dimension and satisfies the condition of decomposition, and \(\mathcal{U}(m)\) has infinite I-dimension.

**11.3.4 Global Approach. Finite Information Systems**

Let \(\mathcal{U} = (A, B, F)\) be a finite information system. In this section, we consider the following previously defined parameters of the information system \(\mathcal{U}\):

- \(I(\mathcal{U})\) (I-dimension of \(\mathcal{U}\)) – the maximum cardinality of a subset of \(F\) which is an independent set (page 217);
- \(ir(\mathcal{U})\) – the maximum cardinality of an irredundant subset of \(F\) (page 217);
- \(ts(\mathcal{U})\) – the maximum dimension of a tree-stable problem from Probl\(_{\mathcal{U}}\) (page 212);
• $\text{ts}^\infty(U)$ – the maximum dimension of a tree-stable problem from Probl$^\infty_U$ (page 213).

From (11.1) it follows that

$$I(U) \leq \text{ir}(U).$$

According to Proposition 36, $\text{ts}^\infty(U) = \text{ts}(U)$. From here and from (11.2) it follows that

$$\text{ts}^\infty(U) \leq \text{ir}(U).$$

Examples 1-3 show different behavior of parameters $I(U)$, $\text{ir}(U)$, and $\text{ts}^\infty(U) = \text{ts}(U)$.

The next theorem follows from Theorem 27 and Propositions 36 and 37.

**Theorem 31.** Let $U = (A, B, F)$ be a finite information system, $k = |B|$, and $m = \max(\text{ts}^\infty(U), I(U))$. Then, for any $n \in \omega \setminus \{0\}$, the following statements hold:

a) if $n \leq m$ then

$$\frac{n}{\log_2 k} \leq \text{IH}^g_{U,\infty}(n) = H^g_{U,\infty}(n) \leq n;$$

b) if $m < n \leq \text{ir}(U)$ then

$$\max \left\{ \text{ts}^\infty(U), \frac{I(U)}{\log_2 k}, \log_k (n + 1) \right\} \leq \text{IH}^g_{U,\infty}(n) = H^g_{U,\infty}(n) \leq \min \{ n - 1, 4(m + 1)^4(\log_2 n)^2 + 4(m + 1)^5(\log_2 n)(\log_2 k) \};$$

c) if $n \geq \text{ir}(U)$ then

$$\text{IH}^g_{U,\infty}(n) = H^g_{U,\infty}(n) = \text{IH}^g_{U,\infty}(\text{ir}(U)) = H^g_{U,\infty}(\text{ir}(U)).$$
Chapter 12

Conclusions

We conclude this thesis with the summary of the work and directions of future research.

12.1 Summary

Decision trees are one of the most common tools to represent knowledge and predict unknown instances from decision tables with the one-valued decisions as well as many-valued decisions. Nevertheless, we also extend our study to inhibitory trees. Our results are mainly involved to optimize decision and inhibitory tree parameters such as depth or average depth which can be used to reduce searching time and number of nodes or terminal nodes which can be used to reduce the storage requirements.

We construct a DAG for the given decision table with many-valued decisions, and after that, we process the DAG depending on our goals. Based on this, we

- describe the set of all decision and inhibitory trees;
- develop multi-stage optimization of the decision and inhibitory trees for a sequence of various criteria;
- develop techniques for bi-criteria optimization (cost vs. cost and cost vs. uncertainty (completeness)) of the decision (inhibitory) trees;
- develop methods to compare greedy heuristics as algorithms for single- and bi-criteria optimization (cost vs. cost) of the decision and inhibitory trees;
• develop applications to machine learning problems using multi-pruning and restricted multi-pruning approaches.

We also develop methods to compare the three approaches to handle inconsistency of decision tables. Furthermore, we also develop local and global approaches to study of the decision and inhibitory trees over arbitrary information systems.

The limitation of our methods is the low scalability. We can work with medium-sized decision tables only. However, with respect to the research, this work is useful in knowledge representation and structure optimization of the decision and inhibitory trees.

12.2 Future Research

The work has been done already is mostly theoretical, along with, some experimental analysis. However, there are many possibilities to work further.

One obvious extension is to restrict the number of branches in the DAG since to explore all branches causes higher order of running time.

Another possible extension is to use specific greedy heuristics for decision tree construction based on the specific domain.

Recently, multi-label learning has been explored in the machine learning community. It has been used in semantic tagging of images and videos (news clips, movies) [3], text categorizations (news articles, web pages, patents, e-mails, bookmarks, etc.) [7], functional genomics (gene, and protein functions) [6], music categorization into emotions [5], and others. Our methods work based on the premise that the terminal node in decision tree contains only one decision. We can extend it to contain more than one decision. In this way, our methodology can apply to the above problems. It is also interesting to consider not only the decision but also inhibitory trees for such applications.
Another extension can be made to create different approaches to handle inconsistency. It is possible to consider not only our approach, or generalized decisions as in the rough set theory, or most common decisions but also approaches based on additional information such as the importance of decisions, etc.
REFERENCES


[36] ———, “Classification and optimization of decision trees for inconsistent decision tables represented as MVD tables,” in 2015 Federated Conference on Computer


[76] ——, “On time complexity of decision trees,” in Rough Sets in Knowledge Discovery 1: Methodology and Applications, ser. Studies in Fuzziness and Soft Com-


APPENDICES
Published Papers


c7 Fawaz Alsolami, Mohammad Azad, Igor Chikalov, and Mikhail Moshkov. Decision rule classifiers for multi-label decision tables. In Second International Confer-


Submitted and Accepted Papers


