Diagnosis of Constant Faults in Read-Once Contact Networks over Finite Bases using Decision Trees

Thesis by

Monther Ibrahim Busbait

In Partial Fulfillment of the Requirements

For the Degree of

Masters of Science

King Abdullah University of Science and Technology, Thuwal,

Kingdom of Saudi Arabia

May, 2014

The thesis of Monther Ibrahim Busbait is approved by the examination committee

Committee Chairperson: Mikhail Moshkov

Committee Member: Victor Soloviev

Committee Member: Antoine Vigneron

ABSTRACT

Diagnosis of Constant Faults in Read-Once Contact Networks over Finite Bases using Decision Trees

Monther Ibrahim Busbait

We study the depth of decision trees for diagnosis of constant faults in read-once contact networks over finite bases. This includes diagnosis of 0-1 faults, 0 faults and 1 faults. For any finite basis, we prove a linear upper bound on the minimum depth of decision tree for diagnosis of constant faults depending on the number of edges in a contact network over that basis. Also, we obtain asymptotic bounds on the depth of decision trees for diagnosis of each type of constant faults depending on the number of edges in contact networks in the worst case per basis. We study the set of indecomposable contact networks with up to 10 edges and obtain sharp coefficients for the linear upper bound for diagnosis of constant faults in contact networks over bases of these indecomposable contact networks. We use a set of algorithms, including one that we create, to obtain the sharp coefficients.

ACKNOWLEDGEMENTS

I would like to express my gratitude to my advisor Prof. Mikhail Moshkov for exposing me to the problem in this thesis and for providing me with reference to sources related to the field of study. I also would like to thank him for following up with my work and for the enriching discussions we had. My gratitude also goes to the committee members Prof. Victor Soloviev and Prof. Antoine Vigneron for spending their time on reviewing and evaluating the work of this thesis.

I would like to thank Dr. Igor Chicalov for providing me access and support on using DAGGER (a tool for design and analysis of decision trees an rules). Also, I would like to thank Mr. Shahid Hussain for his support on using LaTeX. My gratitude is extended to many colleagues and friends for their valuable advice and critical comments.

I would like to thank my parents for their support and encouragement to continue my work. Special thanks go to my wife and my little son for giving me free time to work on this thesis. Their support is highly appreciated. Finally, I would like to thank King Abdullah University of Science and Technology for providing me with the proper environment and the good resources for doing research.

TABLE OF CONTENTS

LIST OF SYMBOLS

Symbol	**Meaning**
C-faults	C is a set of constant values that can be assigned to edges as faults
$\mathrm{Net}(B)$	Set of all networks over basis B
$L(S)$	Number of edges in a network S
$h_C(S)$	Minimum depth of decision tree for diagnosis of C-faults in a network S
$\tau_C(S)$	A coefficient equal to $\frac{h_C(S)}{L(S)-1}$
$F_C(S)$	Set of all possible functions corresponding to C-faults in a network S

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

In this thesis, we consider the representation of Boolean functions by contact networks (networks), each of which is an undirected connected multigraph with two vertices fixed as poles and Boolean variables assigned to edges. For a given tuple of variable values, the value of the function corresponding to the network is equal to 1 if and only if there exists a path between poles such that the value of each variable assigned to the edges in this path is equal to 1. Contact networks are also known as contact schemes, contact circuits, and switching circuits – see papers by Shannon [1], Chegis and Yablonskii [2], and Karpova [3]. We consider only read-once contact networks (also known as iteration-free, non-repeating, and without repetition) in which pairwise different variables are assigned to the edges. For clarity, we draw poles of a network as open circles while its usual nodes are drawn as filled circles.

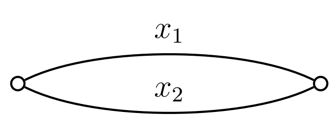


Figure 1.1: Network of 2 Parallel Edges

Example 1.1. Consider the network in Figure 1.1. It is clear that the network shown is read-once because pairwise different variables x_1 and x_2 are assigned to the edges. Let us build truth tables for the function implemented by the network and for the Boolean function $x_1 \vee x_2$ as follows. It is clear that both functions are equivalent.

x_1	x_2	Output
0	0	0
0	1	1
1	0	1
1	1	1

Table 1.1: Truth Table for the Network in Figure 1.1

x_1	x_2	$x_1 \vee x_2$
0	0	0
0	1	1
1	0	1
1	1	1

Table 1.2: Truth Table for $x_1 \vee x_2$

We study read-once networks over a finite set B of networks called basis. These networks can be obtained by replacing some edges in a network from B with recursively constructed networks (including networks from B). We assume that all networks in B are indecomposable, i.e. cannot be obtained from two nontrivial (containing more than one edge) networks by replacing an edge in the first network with the second one. The notion of read-once network over finite basis B of networks is very close to the notion of read-once formula over the set of Boolean functions corresponding to the networks from B.

Example 1.2. Let us consider a basis that contains the network in Figure 1.1. The network in Figure 1.2 can be obtained by replacing edge x_2 from the network in Figure 1.1 with the network itself. The network in Figure 1.3 can be obtained by replacing an edge from the network in Figure 1.1 with the network itself three times recursively. We say that Figure 1.2 and Figure 1.3 are networks over basis of the network in Figure 1.1.

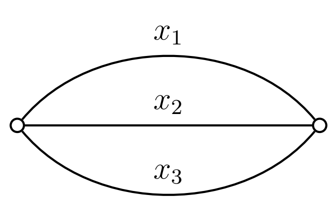


Figure 1.2: Network of 3 Parallel Edges

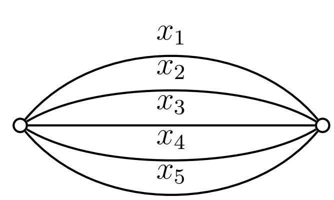


Figure 1.3: Network of 5 Parallel Edges

We consider constant faults (C-faults) of read-once networks each of which consists in assigning constant values from the set C to some of the network variables. We consider three types of C-faults: 0-1 faults where $C = \{0, 1\}$, 0 faults where $C = \{0\}$ and 1 faults where $C = \{1\}$. We study the problem of diagnosis of C-faults: for a given faulty network we should recognize the function implemented by this network. To solve this problem we use decision trees with membership queries: we can ask about value of the function implemented by a given faulty network on an arbitrary tuple of values of the network variables. The depth of a decision tree is the maximum number of nonterminal nodes (queries) in a path from the root to a terminal node.

Example 1.3. Consider Table 1.3 that characterizes the network in Figure 1.1 with 0-1 faults. The first four columns of the table correspond to application of membership queries on the network. The corresponding entries in rows are outputs of these queries.

00	01	10	11	Function	Comment
0	0	0	0	0	0 faults to x_1 and x_2
1	1	1	1	1	1 fault to x_1 or x_2
0	0	1	1	x_1	0 fault to x_2
0	1	0	1	x_2	0 fault to x_1
0	1	1	1	$x_1 \vee x_2$	No faults assigned

Table 1.3: Decision Table for the Network in Figure 1.1

The decision tree in Figure 1.4 solves the problem of diagnosis of 0-1 faults in the network in Figure 1.1. The depth of this decision tree is 3.

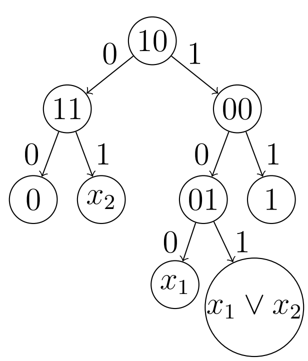


Figure 1.4: Decision Tree for Diagnosis of 0-1 Faults in the Network in Figure 1.1

For any read-once network S over basis B, we denote by $L(S)$ the number of edges in S, by $h_C(S)$ the minimum depth of a decision tree for diagnosis of C-faults in S and by $F_C(S)$ the set of possible functions corresponding to C-faults in S.

1.1 Problem Statement

We study decision trees for diagnosis of C-faults in read-once networks with objective to *minimize depth*. For networks over some finite basis B, we prove a linear upper bound on depth of decision tree for diagnosis of C-faults depending on the number of edges in a network and we are interested to *minimize coefficients* for that upper bound.

For networks over some finite basis, we are interested to characterize the relation between depth of decision tree for diagnosis of C-faults and number of edges in the network in the worst case in asymptotic notation. We consider each type of C-faults separately.

1.2 Contributions

We work to find sharp coefficients for the linear upper bound on depth of decision tree for diagnosis of C-faults for bases of indecomposable networks. It is enough to study indecomposable networks since any network can be represented as a network over bases of such networks. However, the set of indecomposable networks is infinite.

We consider the problem of minimizing the depth of decision tree for diagnosis of C-faults in all indecomposable networks with up to 10 edges and, thus, finding sharp coefficients for any networks over basis of indecomposable networks with at most 10 edges. We describe a new algorithm that finds a tree with minimum depth. Using the new algorithm, we obtain exact values for minimum depth of decision trees for all the considered networks which cannot be obtained using common greedy-based methods.

For networks over some finite basis of indecomposable networks, we prove asymptotic bounds on depth of decision tree for diagnosis of all possible C-faults depending on number of edges in the worst case.

1.3 Structure of the Thesis

The remaining part of thesis is as follows. Chapter 2 covers problems related to our work and their published results. Chapter 3 defines notions necessary to understand the problem. Chapter 4 proves bounds on depth of decision trees for diagnosis of C-faults. Chapter 5 shows our work to estimate minimum depth of decision trees for diagnosis of C-faults in all indecomposable networks with at most 10 edges. This includes description for algorithms we use in our work. Chapter 6 states some concluding remarks.

Chapter 2

Related Work

In this chapter, we describe published works related to our problem of diagnosis. There are different directions of study that intersect with the problem defined in this thesis.

2.1 Networks as Boolean Functions

Shannon in [1] proposed that contact networks, resembling electrical circuits, can be used to represent Boolean formula. Assuming a battery is connected in parallel at one pole and a lamp is connected at the other pole and edges represent switches, we can either turn the lamp on or off depending on the state of switches. If there is a path between the battery and the lamp with all switches are turned on, the lamp is turned on. Otherwise, the lamp is turned off.

Boolean variables are assigned to edges of the network with values correspond to the impedance of the corresponding switch. If a switch has a zero impedance, the switch is connected and a value of 0 is assigned to the variable of the corresponding edge. If a switch has an infinite impedance, the switch is disconnected and a value of 1 is assigned to the variable of the corresponding edge. A lamp will turn on with value 0 if and only if there is a path with all edges' variables are assigned to 0. This notation is opposite to the one used in this thesis.

Using the previous notation, Shannon studied logical properties of these networks including converting networks into Boolean functions, simplification of electrical circuits and identification of equivalent configurations.

2.2 Study of Indecomposable Networks

The intuition behind studying indecomposable networks is that any decomposable network can be obtained over bases of indecomposable networks only. Kuznetsov in [4] showed that there is an infinite number of indecomposable networks. He showed that any network in the form of Figure 2.1 is indecomposable.

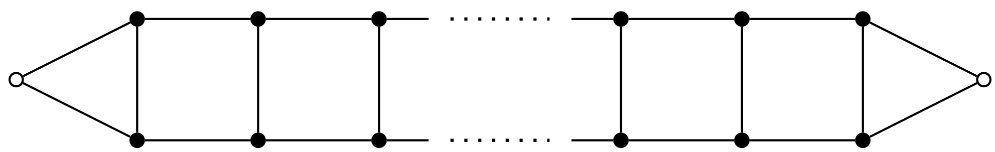


Figure 2.1: Infinite Set of Indecomposable Networks

Kuznetsov also proposed a list of 56 networks containing all indecomposable networks with at most 10 edges. In this thesis, we obtain the minimum depth of tree for diagnosis of C-faults in these networks.

2.3 Learning Read-Once Networks and Formulas

There are two problems related to learning:

(i) For a given read-once contact network with n edges, using only membership queries we should find a read-once network with n edges implementing the same function as the initial network.

(ii) For a given read-once formula with n variables over a basis P of Boolean functions using only membership queries, we should find a read-once formula over P with n variables that is logically equivalent to the initial formula.

These problems are more complex than the problems of diagnosis of C-Faults for read-once contact networks or for read-once formulas. The algorithm for solving the problem (i) proposed by Raghavan and Schach in [5] and the algorithm for solving the problem (ii) with $P = \{x \vee y, x \wedge y\}$ proposed by Angluin, Hellerstein, and Karpinski in [6], both use $O(n^2)$ membership queries, in contrast to the linear algorithm presented here.

2.4 Diagnostic Tests vs. Decision Trees

A *diagnostic test* is a fixed set of membership queries which allows us to solve the problem of diagnosis of C-faults. For functions corresponding to faults that are considered in Table 1.3, we can distinguish functions using only a subset of membership queries as in Table 2.1. The table shows that the set: $\{00, 01, 10\}$ is a diagnostic test.

00	01	10	Function
0	0	0	0
1	1	1	1
0	0	1	x_1
0	1	0	x_2
0	1	1	$x_1 \vee x_2$

Table 2.1: A Reduction of Table 1.3 with only Queries from Diagnostic Test

For diagnostic tests, the most interesting results were obtained by Madatyan in [7]. He proved that for each read-once network S over the basis of indecomposable networks with 2 edges, the minimum cardinality of a diagnostic test for diagnosis of 0-1 faults is at most $\frac{3}{2}L(S)$. It is well known that the depth of decision tree for any data analysis is at most the minimum cardinality of its diagnostic test (see, for example, [8]). From here it follows that the upper bound $h_{\{0,1\}}(S) \leq \frac{3}{2}L(S)$ is true for any read-once network S over the basis of indecomposable networks with 2 edges. This result is slightly better than the result obtained in this thesis using decision trees.

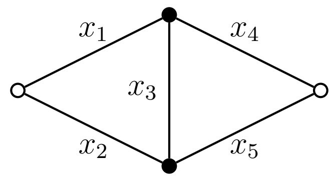


Figure 2.2: Indecomposable Network with 5 Edges

Madatyan also showed that, for $k = 1, 2, \ldots$, there exists a read-once network over the basis of the network in Figure 2.2 with $4k + 1$ edges for which the minimum cardinality of a diagnostic test for diagnosis of 0-1 faults is equal to $3 \cdot 2^{k+1} - 4$. From results obtained in this thesis, it follows that the minimum depth of a decision tree for diagnosis of the considered network is at most $7k$. We see here a huge adaptivity gap between complexity of diagnostic decision trees which can be interpreted as *adaptive algorithms* (the choice of the next query depends on the results of the previous queries) and diagnostic tests which can be interpreted as *non-adaptive algorithms* (the sequence of queries is fixed).

For decision trees, interesting results were obtained in the study of iteration-free combinatorial circuits (read-once formulas) over finite bases P of Boolean functions. From results obtained by Moshkov in [9], which generalize essentially the results presented by Goldman and Chipulis [10] and by Karavai [11], it follows that if P contains only linear or only unate Boolean functions, then the minimum depth of decision trees in the worst case has at most linear growth depending on the number of gates in the circuits, and it grows exponentially if P contains a non-linear function and a function which is not unate. Interestingly, read-once contact networks are unate functions.

Interesting results were also obtained for the study of read-once contact networks. Moshkov in [12] proposed the following result about minimum depth of decision trees without proof. He showed for $C = \{0, 1\}$ that $h_C(S) \leq t_C(B)(L(S) - 1)$ for any read-once network S over a basis B with C-faults, where $t_C(B) = \max\{\tau_C(Q) : Q \in B\}$ and $\tau_C(Q) = \frac{h_C(Q)}{L(Q)-1}$ for any $Q \in B$. This inequality is proven in Chapter 4 for arbitrary $C \in \{\{0, 1\}, \{0\}, \{1\}\}$.

For diagnosis of 0-1 faults in read-once contact networks, Goduhina in [13] obtained lower and upper bounds estimations on the parameters $h_{\{0,1\}}(Q)$ and $\tau_{\{0,1\}}(Q)$ for each indecomposable network Q with at most eight edges. In chapter 5, We compare results of Goduhina's paper with our results.

Chapter 3

Main Notions

In this chapter, we formally define all concepts related to our problem. This includes contact networks, Boolean functions, constant faults, problem of diagnosis and decision trees.

3.1 Contact Networks

A read-once contact network (*network*) S is an undirected multigraph, i.e. with possible multiple edges between two vertices, and without loops in which two different nodes called *poles* are fixed. For each edge e in S, there is a *simple* path, i.e. without repeating nodes, between poles which contains e. We will assume that the set of edges of S is ordered and edges labeled with pairwise different variables. We denote by $L(S)$ the number of edges in the network S.

A network is *nontrivial* if it contains more than one edge. A network S is said to be *decomposable* if there exists two nontrivial networks N_1 and N_2 such that S is formed by replacing an edge e in N_1 with the network N_2 with poles of N_2 match on the end points of the edge to be replaced. The edges of S are ordered as follows: we start with all edges of N_1 with the exception of e in the same order as in N_1 and after that all edges of N_2 in the same order as in N_2. If a network is not decomposable it is called *indecomposable*.

A *basis* B is an arbitrary finite set of networks. For a nonempty basis B, we consider the *set of networks over* B denoted by $\mathrm{Net}(B)$. $\mathrm{Net}(B)$ is defined recursively as follows: Let $\mathrm{Net}_1(B) = B$. For any natural $r \geq 2$, we denote by $\mathrm{Net}_r(B)$ the set of all networks $S \notin \mathrm{Net}_1(B) \cup \ldots \cup \mathrm{Net}_{r-1}(B)$ such that S can be obtained by replacing some edges in a network from B with networks from $\mathrm{Net}_1(B) \cup \ldots \cup \mathrm{Net}_{r-1}(B)$. Then $\mathrm{Net}(B) = \mathrm{Net}_1(B) \cup \mathrm{Net}_2(B) \cup \ldots$.

Networks represent Boolean functions. For a natural number n, a *Boolean* function is a function $f(x_1, \ldots, x_n)$ such that $f : \{0,1\}^n \to \{0,1\}$. This function is called *monotone* if, for any n-tuples $a = (a_1, \ldots, a_n)$ and $b = (b_1, \ldots, b_n)$ from $\{0,1\}^n$, if $a_1 \leq b_1, \ldots, a_n \leq b_n$ then $f(a) \leq f(b)$. A variable x_i of the function f is called *essential* if there exist two tuples $a, b \in \{0,1\}^n$, which are different only in the i-th digit and for which $f(a) \neq f(b)$.

Let S be a network with $L(S) = n$ and edges $e_1, \ldots, e_n$. We assign to these edges, variables $x_1, \ldots, x_n$ respectively and define a Boolean function $f_S(x_1, \ldots, x_n)$ *implemented* by S. Let $a_1, \ldots, a_n \in \{0,1\}$ be values of variables $x_1, \ldots, x_n$ respectively. Then $f_S(a_1, \ldots, a_n) = 1$ if and only if there is a path between poles of S in which all variables corresponding to edges in the path have value 1. One can show that f_S is a monotone function, and all variables of this function are essential.

3.2 The Problem of Diagnosis

We consider three types of *constant faults*: $\{0,1\}$-faults (0-1 faults), $\{0\}$-faults (0 faults), and $\{1\}$-faults (1 faults). Let $C \in \{\{0,1\}, \{0\}, \{1\}\}$. A C-fault of S consists in assigning of constants from C to some variables of f_S. More formally, a *C-fault* of S is an n-tuple $\rho = (\rho_1, \ldots, \rho_n) \in \{C \cup \{2\}\}^n$. For $i = 1, \ldots, n$, the value ρ_i is called the value of the C-fault ρ for the variable x_i. The network S with the C-fault ρ implements the function

$$f_{S,\rho}(x_1,\ldots,x_n) = f_S(x_1 \circ \rho_1,\ldots,x_n \circ \rho_n)$$

where, for $i = 1,\ldots,n$,

$$x_i \circ \rho_i = \begin{cases} 0, & \rho_i = 0; \\ 1, & \rho_i = 1; \\ x_i, & \rho_i = 2. \end{cases}$$

Note that $f_{S,\rho}$ is a monotone function. A C-fault ρ is called *normal* if, for $i = 1,\ldots,n$, $\rho_i = 2$ if and only if the variable x_i is an essential variable of the function $f_{S,\rho}$. One can show that, for any C-fault δ of S, there exists a normal C-fault ρ of S such that $f_{S,\rho} = f_{S,\delta}$.

We denote $F_C(S) = \{f_{S,\rho} : \rho \in \{C \cup \{2\}\}^n\}$. The set $F_C(S)$ contains all functions defined on $\{0,1\}^n$ that can be obtained from the function f_S implemented by S by assigning of constants from C to some variables of f_S.

We consider the *problem of diagnosis of C-faults*. For a network S with n edges and an unknown C-fault δ of S, we should find a normal C-fault ρ of S such that $f_{S,\rho}$=$f_{S,\delta}$. In order to find such a fault, we are allowed to ask about value of $f_{S,\delta}$ on arbitrary tuple from $\{0,1\}^n$. We call this procedure a *membership query* [6].

We consider using adaptive algorithms for solving the problem of diagnosis. These can be represented in the form of *decision trees*, each of which is a directed tree with root. Terminal nodes of the tree are labeled with normal C-faults of S. Each nonterminal node is labeled with a membership query – an n-tuple from $\{0,1\}^n$. Two edges start in this node that are labeled with 0 and 1 respectively. The *depth* $h(\Gamma)$ of a decision tree Γ is the maximal *length* of a path, i.e. number of edges in the path, from the root to a terminal node. We denote by $h_C(S)$ the minimum depth of a decision tree which solves the problem of diagnosis of C-faults for S.

Let Γ be a decision tree. For the network S with n edges and a C-fault δ, this tree works in the following way. If the root of Γ is a terminal node then the result of Γ work is the normal C-fault attached to the root. Otherwise, we find the value of function $f_{S,\delta}$ on the n-tuple attached to the root and pass along the edge which starts in the root and is labeled with this value, etc. We will say that Γ *solves* the problem of diagnosis of C-faults in S if, for any C-fault δ of S, the result of Γ work is a normal C-fault ρ of S such that $f_{S,\rho} = f_{S,\delta}$.

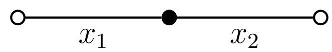


Figure 3.1: Network of 2 Edges in Series (S_{Series})

Example 3.1. Let us consider the network S_{Series} in Figure 3.1. This network implements the function $f_{S_{\text{Series}}}(x_1, x_2) = x_1 \wedge x_2$. Now we describe in details the work of decision tree similar to the work of Example 1.3. Since larger networks implement complex functions, we prefer to attach to terminal nodes normal C-faults rather than functions themselves.

Considering 0-1 faults, we have nine faults in network S_{Series} which implement functions from the set $F_{\{0,1\}}(S_{\text{Series}})$=$\{x_1 \wedge x_2,\ x_1,\ x_2,\ 0,\ 1\}$: $f_{S_{\text{Series}},\delta}(x_1, x_2) = 0$ for any $\delta \in \{(0,0),\ (0,1),\ (0,2),\ (1,0),\ (2,0)\}$, $f_{S_{\text{Series}},(1,1)}(x_1, x_2) = 1$, $f_{S_{\text{Series}},(2,1)}(x_1, x_2) = x_1$, $f_{S_{\text{Series}},(1,2)}(x_1, x_2) = x_2$, and $f_{S_{\text{Series}},(2,2)}(x_1, x_2) = x_1 \wedge x_2$. The faults $(0,0)$, $(1,1)$, $(2,1)$, $(1,2)$, and $(2,2)$ are normal 0-1 faults for S_{Series} corresponding to the functions 0, 1, x_1, x_2 and $x_1 \wedge x_2$ respectively. Note that for normal C-faults corresponding to equivalent functions, it is enough to nominate one as the respective fault, e.g. $(0,0)$, $(0,1)$ and $(1,0)$. Table 3.1 summarizes the previous description. The decision tree in Figure 3.2 solves the problem of diagnosis of 0-1 faults in S_{Series}. The depth of this tree is equal to 3. Therefore $h_{\{0,1\}}(S_{\text{Series}}) \leq 3$.

00	01	10	11	Fault Tuple
0	0	0	0	00
1	1	1	1	11
0	0	1	1	21
0	1	0	1	12
0	0	0	1	22

Table 3.1: Characteristic Table for 0-1 Faults in Network S_{Series}

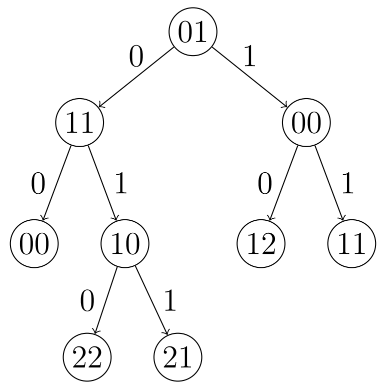


Figure 3.2: Decision Tree for Diagnosis of 0-1 Faults in Network S_{Series}

Assume that we want to diagnose the same network for 0 faults. Then, network S_{Series} contains four 0 faults which implement functions from the set $F_{\{0\}}(S_{\text{Series}}) = \{x_1 \wedge x_2, 0\}$: $f_{S_{\text{Series}},\delta}(x_1, x_2) = 0$ for any $\delta \in \{(0,0), (0,2), (2,0)\}$, $f_{S_{\text{Series}},(2,2)}(x_1, x_2) = x_1 \wedge x_2$. The faults $(0,0)$ and $(2,2)$ are normal 0 faults for S_{Series} corresponding to the functions 0 and $x_1 \wedge x_2$ respectively. The decision tree in Figure 3.3 solves the problem of diagnosis of 0 faults in S_{Series}. Thus, $h_{\{0\}}(S_{\text{Series}}) \leq 1$.

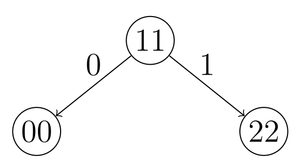


Figure 3.3: Decision Tree for Diagnosis of 0 Faults in Network S_{Series}

Chapter 4

Analysis of Decision Trees for Problem of Diagnosis

In this chapter, we study lower and upper bounds on minimum depth of decision trees $h_C(S)$ for diagnosis of C-faults in an arbitrary network S and characterize the behavior of minimum depth of decision trees for diagnosis in the worst case, per basis, depending on number of edges in networks over bases of indecomposable networks.

4.1 Bounds on Minimum Depth of Decision Trees

Proposition 4.1. *For any network S, the following inequality holds:*

$$h_C(S) \geq \lceil \log_2 |F_C(S)| \rceil .$$

Proof. Let Γ be a decision tree which solves the problem of diagnosis of C-faults for S and for which $h(\Gamma) = h_C(S)$. For each function $f \in F_C(S)$, in the tree Γ there is a terminal node which is labeled with a C-fault ρ for S such that $f = f_{S,\rho}$. Therefore, the number of terminal nodes in Γ is at least $|F_C(S)|$. One can show that the number of terminal nodes in Γ is at most $2^{h(\Gamma)}$. Thus $|F_C(S)| \leq 2^{h_C(S)}$ and $h_C(S) \geq \log_2 |F_C(S)|$. Since $h_C(S)$ is an integer, we have $h_C(S) \geq \lceil \log_2 |F_C(S)| \rceil$. $\square$

Proposition 4.1 gives us a general lower bound for minimum depth of decision tree for diagnosis of C-faults. We now consider an upper bound on the minimum depth of decision trees for diagnosis of C-faults in networks over B depending on the number of edges in the network. Recall that $t_C(B) = \max\{\tau_C(Q) : Q \in B\}$ where $\tau_C(Q) = \frac{h_C(Q)}{L(Q)-1}$ for any $Q \in B$ and $L(S)$ is the number of edges in a network S.

Theorem 4.2. *For any network $S \in \mathrm{Net}(B)$, the following inequality*[1] *holds:*

$$h_C(S) \leq t_C(B)(L(S) - 1).$$

Proof. We will prove the considered inequality by induction on index r such that $S \in \mathrm{Net}_r(B)$. Let $S \in \mathrm{Net}_1(B)$. Then $h_C(S) = \frac{h_C(S)}{L(S)-1}(L(S)-1) = \tau_C(S)(L(S)-1) \leq t_C(B)(L(S) - 1)$.

Let now $r \geq 2$ and, for any network from $\mathrm{Net}_1(B) \cup \cdots \cup \mathrm{Net}_{r-1}(B)$, the considered inequality hold. Let us prove that the considered inequality holds for any $S \in \mathrm{Net}_r(B)$. Since $r \geq 2$, there exist networks $P_0 \in B$ and $P_1, \ldots, P_k \in \mathrm{Net}_1(B) \cup \cdots \cup \mathrm{Net}_{r-1}(B)$, $k \leq L(S)$, such that S can be obtained by replacing k edges in the network P_0 with the networks $P_1, \ldots, P_k$. It is clear that $L(S) = L(P_0) + (L(P_1) - 1) + \ldots + (L(P_k) - 1)$.

We proved that $h_C(P_0) \leq t_C(B)(L(P_0) - 1)$. By induction hypothesis, $h_C(P_i) \leq t_C(B)(L(P_i) - 1)$ for $i = 1, \ldots, k$. For $i = 0, 1, \ldots, k$, let Γ_i be a decision tree which solves the problem of diagnosis for P_i and for which $h(\Gamma_i) \leq t_C(B)(L(P_i) - 1)$. We consider now a decision tree Γ which solves the problem of diagnosis for S. We describe the work of Γ for the network S with a C-fault δ. As a result we obtain a normal C-fault ρ for S such that $f_{S,\rho} = f_{S,\delta}$.

First, the decision tree Γ simulates in some way the work of the decision tree Γ_0. The network P_0 (as a skeleton of S) is analyzed at that. Let P_0 contain $m = L(P_0)$

[1] This inequality was published in [12] without proof for $C = \{0, 1\}$.

edges, $e_1, \dots, e_m$. Let a nonterminal node of the decision tree Γ_0 be labeled with an m-tuple $(a_1, \dots, a_m) \in \{0,1\}^m$. For $i = 1, \dots, m$, we assign the value a_i to some variables of S (we denote the set of these variables X_i). If the edge e_i was not replaced with a network then we assign the value a_i to the variable corresponding to e_i in S (X_i contains only this variable). If the edge e_i was replaced with a network P_j then we assign the value a_i to all variables corresponding to edges of the network P_j in S (X_i contains all these variables). The value of the function implemented by P_j with the C-fault δ will be equal a_j provided that this function is not constant.

Let the outcome of the decision tree Γ_0 be a normal C-fault $\gamma = (\gamma_1, \dots, \gamma_m)$ for P_0. For $i = 1, \dots, m$, we fix values of ρ for some variables of f_S. To this end, the decision tree Γ will simulate the work of some decision trees Γ_j, $j \in \{1, \dots, k\}$.

If $\gamma_i \neq 2$ then the value of ρ, for each variable from X_i, is equal to γ_i. Let $\gamma_i = 2$. If X_i contains only one variable then the value of ρ for this variable is equal to 2. Let X_i contain two or more variables. Then the edge e_i was replaced in S with a network P_j, $j \in \{1, \dots, k\}$. Since $\gamma_i = 2$, there are values $b_1, \dots, b_{i-1}, b_{i+1}, \dots, b_m \in \{0,1\}$ such that $f_{P_0,\gamma}(b_1, \dots, b_{i-1}, 0, b_{i+1}, \dots, b_m) \neq f_{P_0,\gamma}(b_1, \dots, b_{i-1}, 1, b_{i+1}, \dots, b_m)$. Now, the decision tree Γ simulates the work of the decision tree Γ_j. Let P_j contain $l = L(P_j)$ edges $\varepsilon_1, \dots, \varepsilon_l$. Let a nonterminal node of the decision tree Γ_j be labeled with an l-tuple $(a_1, \dots, a_l) \in \{0,1\}^l$. We assign values $a_1, \dots, a_l$ to variables corresponding in S to the edges $\varepsilon_1, \dots, \varepsilon_l$, respectively. For $s = 1, \dots, i-1, i+1, \dots, m$, we assign the value b_s to all variables from X_s. Let the outcome of the decision tree Γ_j be a normal C-fault $\alpha = (\alpha_1, \dots, \alpha_l)$ for P_j. Then $\alpha_1, \dots, \alpha_l$ are the values of ρ for the variables corresponding to the edges $\varepsilon_1, \dots, \varepsilon_l$, respectively.

The outcome of Γ is the tuple ρ which will be completely defined after the finishing of the work with γ_m. One can show that $f_{S,\delta} = f_{S,\rho}$. Therefore Γ solves the problem of diagnosis for S. It is clear that $h(\Gamma) \leq h(\Gamma_0) + \ldots + h(\Gamma_k) \leq t_C(B)(L(P_0) - 1) + \ldots + t_C(B)(L(P_k) - 1) = t_C(B)(L(S) - 1)$. $\square$

Remark 4.1. The bound from Theorem 4.2 is in some sense sharp: the coefficient $t_C(B)$ is the minimum number k such that $h_C(S) \leq k(L(S)-1)$ for any $S \in \text{Net}(B)$. It is clear that the equality $h_C(S) = t_C(B)(L(S)-1)$ holds for at least one network S from B, i.e. there exists a network $S \in B$ for which $\tau_C(S) = t_C(B)$.

Remark 4.2. It is not necessary to design and memorize the decision tree Γ for diagnosis of a network $S \in \text{Net}(B)$ whose depth satisfies the inequality from Theorem 4.2. The proof of Theorem 4.2 shows that we can simulate work of this tree efficiently if we know decision trees with minimum depth for the diagnosis of networks from B and if we know how the network S was built from the networks contained in B.

Remark 4.3. If, for each network $Q \in B$, we have a decision tree Γ_Q for diagnosis of Q (not necessarily with minimum depth) we can almost repeat the proof of Theorem 4.2 and prove that $h_C(S) \leq d_C(B)(L(S)-1)$ for any $S \in \text{Net}(B)$, where $d_C(B) = \max\{\delta_C(Q) : Q \in B\}$ and $\delta_C(Q) = \frac{h(\Gamma_Q)}{L(Q)-1}$ for any $Q \in B$. We can simulate work of the decision tree for diagnosis of S whose depth satisfies the considered inequality if we know how the network S was built from the networks contained in B.

4.2 Behavior in Worst Case per Basis of Indecomposable Networks

Theorem 4.2 gives us an upper bound on $h_C(S)$ which depends linearly on $L(S)$ and is true for any $S \in \text{Net}(B)$ where B is finite. We have no lower bound on $h_C(S)$ which is linear depending on $L(S)$ and holds for any $S \in \text{Net}(B)$. However, we can obtain such a bound in the worst case with exception of two cases where the upper and the lower bound on $h_C(S)$ are equal to 1. To this end, we consider a function H_B^C which characterizes the dependence in the worst case, per basis, of the minimum depth of a decision tree for diagnosis of C-faults in a network from $\text{Net}(B)$ on the number of edges in the network. For each natural n,

$$H_B^C(n) = \max\{h_C(S) : S \in \mathrm{Net}(B), L(S) \leq n\}.$$

Theorem 4.3. *Let B be a finite nonempty set of indecomposable nontrivial networks. Then $H_B^{\{0,1\}}(n) = \Theta(n)$.*

Proof. From Theorem 4.2 it follows that $H_B^{\{0,1\}}(n) = O(n)$. Now we also show that $H_B^{\{0,1\}}(n) = \Omega(n)$.

Let B contain the network S_1^2 with two nodes and two parallel edges connecting these nodes. Then, for any natural $n \geq 2$, the set $\mathrm{Net}(B)$ contains a network P_n with n edges such that $f_{P_n} = x_1 \vee \ldots \vee x_n$. Let $I \subseteq \{1, \ldots, n\}$. We consider a {0,1}-fault δ of P_n whose value for the variable x_i is equal to 2 if $i \in I$ and is equal to 0 if $i \in \{1, \ldots, n\} \setminus I$. It is clear that $f_{P_n,\delta} = \bigvee_{i \in I} x_i$ ($f_{P_n,\delta} = 0$ if $I = \emptyset$). Since $F_{\{0,1\}}(P_n)$ contains the function $\bigvee_{i \in I} x_i$ for each $I \subseteq \{x_1, \ldots, x_n\}$, $\left|F_{\{0,1\}}(P_n)\right| \geq 2^n$ and, by Proposition 4.1, $h_{\{0,1\}}(P_n) \geq n$. From here it follows that $H_B^{\{0,1\}}(n) \geq n$ and $H_B^{\{0,1\}}(n) = \Omega(n)$.

Assume S_1^2 does not belong to B, then B contains a network S_2 with a simple path between poles whose length is at least 2 (it is easy to show that S_2^2 is the only indecomposable nontrivial network in which the length of each simple path between poles is equal to 1). Let $L(S_2) = t$. One can show that, for any natural $m \geq 2$, there exists a network S_m from $\mathrm{Net}(B)$ such that $L(S_m) \leq (m-1)t$ and there is a simple path τ between poles of S_m with length $p \geq m$. Let $x_{j_1}, \ldots, x_{j_p}$ be variables corresponded to edges in τ. Let $I \subseteq \{j_1, \ldots, j_p\}$. We consider a {0,1}-fault δ of S_m whose value for each variable x_i of S_m is equal to 2 if $i \in I$, is equal to 1 if $i \in \{j_1, \ldots, j_p\} \setminus I$, and is equal to 0 if $i \notin \{j_1, \ldots, j_p\}$. It is clear that $f_{S_m,\delta} = \bigwedge_{i \in I} x_i$ ($f_{S_m,\delta} = 1$ if $I = \emptyset$). Since $F_{\{0,1\}}(S_m)$ contains the function $\bigwedge_{i \in I} x_i$ for each $I \subseteq \{j_1, \ldots, j_p\}$, $\left|F_(S_m)\right| \geq 2^p \geq 2^m$ and, by Proposition 4.1, $h_{\{0,1\}}(S_m) \geq m$.

Let n be a natural number such that $n \geq t$ and $m = \lfloor \frac{n}{t} \rfloor + 1$. It is clear that $L(S_m) \leq (m-1)t \leq n$. Since $h_{\{0,1\}}(S_m) \geq m \geq \frac{n}{t}$, we have $H_B^{\{0,1\}}(n) \geq \frac{n}{t}$ for any natural $n \geq t$. Therefore $H_B^{\{0,1\}}(n) = \Omega(n)$. □

Theorem 4.4. *Let B be a finite nonempty set of indecomposable nontrivial networks. If $B = \{S_2^2\}$ (see Figure 5.1), then $H_B^{\{0\}}(n) = 1$ for $n \geq 2$. Otherwise, $H_B^{\{0\}}(n) = \Theta(n)$.*

Proof. Let $B = \{S_2^2\}$ where S_2^2 is a network consisting of simple path of length two connecting two poles. An arbitrary network S from the set $\mathrm{Net}(B)$ consists of simple path of length $L(S) \geq 2$ connecting two poles. Therefore $F_{\{0\}}(S) = \{x_1 \wedge \ldots \wedge x_{L(S)}, 0\}$ and $h_{\{0\}}(S) = 1$. Hence $H_B^{\{0\}}(n) = 1$ for $n \geq 2$.

Let $B \neq \{S_2^2\}$. From Theorem 4.2 it follows that $H_B^{\{0\}}(n) = O(n)$. Let us show that $H_B^{\{0\}}(n) = \Omega(n)$. Since $B \neq \{S_2^2\}$, the set B contains a network S_2 with at least 2 edges connected to a pole (it is easy to show that S_2^2 is the only indecomposable nontrivial network in which exactly one edge is connected to each pole). Let $L(S_2) = t$. One can show that, for any natural $m \geq 2$, there exists a network S_m from $\mathrm{Net}(B)$ such that $L(S_m) \leq (m-1)t$ and there are $p \geq m$ edges connected to a pole. Let $x_{j_1}, \ldots, x_{j_p}$ be variables corresponded to these edges. Let $I \subseteq \{j_1, \ldots, j_p\}$. We consider a $\{0\}$-fault $\delta(I)$ of S_m whose value for each variable x_i of S_m is equal to 2 if $i \in \{j_1, \ldots, j_p\} \setminus I$ or $i \notin \{j_1, \ldots, j_p\}$ and is equal to 0 if $i \in I$. Let $I_1, I_2 \subseteq \{j_1, \ldots, j_p\}$ and $I_1 \neq I_2$. Then there is $k \in I_1 \cup I_2$ such that $k \notin I_1 \cap I_2$. Let us consider a tuple of values of variables of S_m in which $x_k = 1$, $x_i = 0$ for any $i \in \{j_1, \ldots, j_p\} \setminus \{k\}$ and $x_i = 1$ for any $i \notin \{j_1, \ldots, j_p\}$. It is easy to show that $f_{S_m,\delta(I_1)}(\alpha) \neq f_{S_m,\delta(I_2)}(\alpha)$. Therefore $\left|F_{\{0\}}(S_m)\right| \geq 2^p \geq 2^m$ and, by Proposition 4.1, $h_{\{0\}}(S_m) \geq m$. Let n be a natural number such that $n \geq t$ and $m = \lfloor \frac{n}{t} \rfloor + 1$. It is clear that $L(S_m) \leq (m-1)t \leq n$. Since $h_{\{0\}}(S_m) \geq m \geq \frac{n}{t}$, we have $H_B^{\{0\}}(n) \geq \frac{n}{t}$ for any natural $n \geq t$. Therefore $H_B^{\{0\}}(n) = \Omega(n)$. □

Theorem 4.5. *Let B be a finite nonempty set of indecomposable nontrivial networks. If $B = \{S_1^2\}$ (see Figure 5.1), then $H_B^{\{1\}}(n) = 1$ for $n \geq 2$. Otherwise, $H_B^{\{1\}}(n) = \Theta(n)$.*

Proof. Let $B = \{S_1^2\}$ where S_1^2 is a network with two poles and two parallel edges connecting the poles. An arbitrary network S from the set $\mathrm{Net}(B)$ consists of two poles and $L(S) \geq 2$ edges connecting the poles. Therefore $F_{\{1\}}(S) = \{x_1 \vee \ldots \vee x_{L(S)}, 1\}$ and $h_{\{1\}}(S) = 1$. Hence $H_B^{\{1\}}(n) = 1$ for $n \geq 2$.

Let $B \neq \{S_1^2\}$. From Theorem Theorem 4.2 it follows that $H_B^{\{1\}}(n) = O(n)$. Let us show that $H_B^{\{1\}}(n) = \Omega(n)$. Since $B \neq \{S_1^2\}$, the set B contains a network S_2 with a simple path between poles whose length is at least 2 (it is easy to show that S_1^2 is the only indecomposable nontrivial network in which the length of each simple path between poles is equal to 1). Let $L(S_2) = t$. One can show that, for any natural $m \geq 2$, there exists a network S_m from $\mathrm{Net}(B)$ such that $L(S_m) \leq (m-1)t$ and there is a simple path τ between poles of S_m with length $p \geq m$. Let $x_{j_1}, \ldots, x_{j_p}$ be variables corresponded to edges in τ. Let $I \subseteq \{j_1, \ldots, j_p\}$. We consider a $\{1\}$-fault $\delta(I)$ of S_m whose value for each variable x_i of S_m is equal to 2 if $i \in \{j_1, \ldots, j_p\} \setminus I$ or $i \notin \{j_1, \ldots, j_p\}$ and is equal to 1 if $i \in I$. Let $I_1, I_2 \subseteq \{j_1, \ldots, j_p\}$ and $I_1 \neq I_2$. Then there is $k \in I_1 \cup I_2$ such that $k \notin I_1 \cap I_2$. Let us consider a tuple of values of variables of S_m in which $x_k = 0$, $x_i = 1$ for any $i \in \{j_1, \ldots, j_p\} \setminus \{k\}$ and $x_i = 0$ for any $i \notin \{j_1, \ldots, j_p\}$. It is easy to show that $f_{S_m,\delta(I_1)}(\alpha) \neq f_{S_m,\delta(I_2)}(\alpha)$. Therefore $\left|F_{\{1\}}(S_m)\right| \geq 2^p \geq 2^m$ and, by Proposition 4.1, $h_{\{1\}}(S_m) \geq m$. Let n be a natural number such that $n \geq t$ and $m = \left\lfloor \frac{n}{t} \right\rfloor + 1$. It is clear that $L(S_m) \leq (m-1)t \leq n$. Since $h_{\{1\}}(S_m) \geq m \geq \frac{n}{t}$, we have $H_B^{\{1\}}(n) \geq \frac{n}{t}$ for any natural $n \geq t$. Therefore $H^{\{1\}}(n)_B = \Omega(n)$. □

Note. It is easy to show that Theorems 4.3, 4.4 and 4.5 can be generalized for bases of decomposable networks. However, it is enough to study bases of indecomposable networks over which we can obtain any network.

Chapter 5

Results for Diagnosis of Indecomposable Networks

In this chapter, we study decision trees for diagnosis of C-faults in all indecomposable networks with up to 10 edges. Specifically, we study the list of 56 indecomposable networks published by Kuznetsov in [4] and, for each network S in the list, we consider lower and upper bounds on the parameters $h_C(S)$ and $\tau_C(S) = \frac{h_C(S)}{L(S)-1}$ for 0-1 faults, 0 faults and 1 faults. For 0-1 faults, we compare our results with results obtained by Goduhina in [13].

We also describe algorithms that we use to obtain an upper bound estimation on the parameters $h_C(S)$ and $\tau_C(S)$. Lower bounds are initially obtained from Proposition 4.1. Based on values of $\tau_C(S)$, we obtain values for $t_C(B)$.

5.1 List of Kuznetsov's Networks

The following figures are graphical representations of all non-trivial indecomposable contact networks with up to 10 edges in which two empty circles are the poles and the filled circles are usual nodes of the network. Each network is indexed in the form of S^i_j where i is the number of edges in the network and j is the index of the network in the list of networks with i edges.

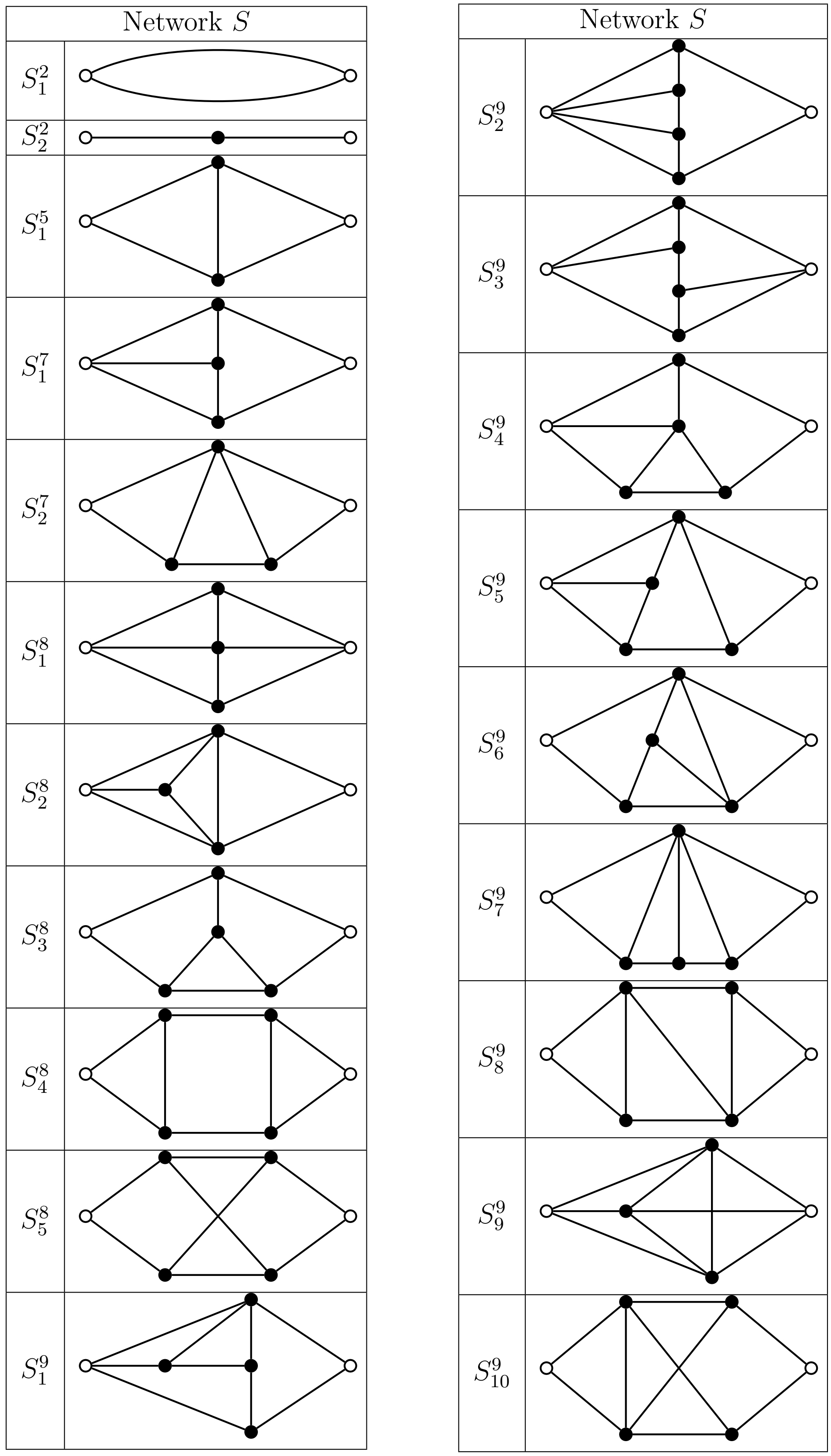


Figure 5.1: Indecomposable Networks with up to 9 Edges

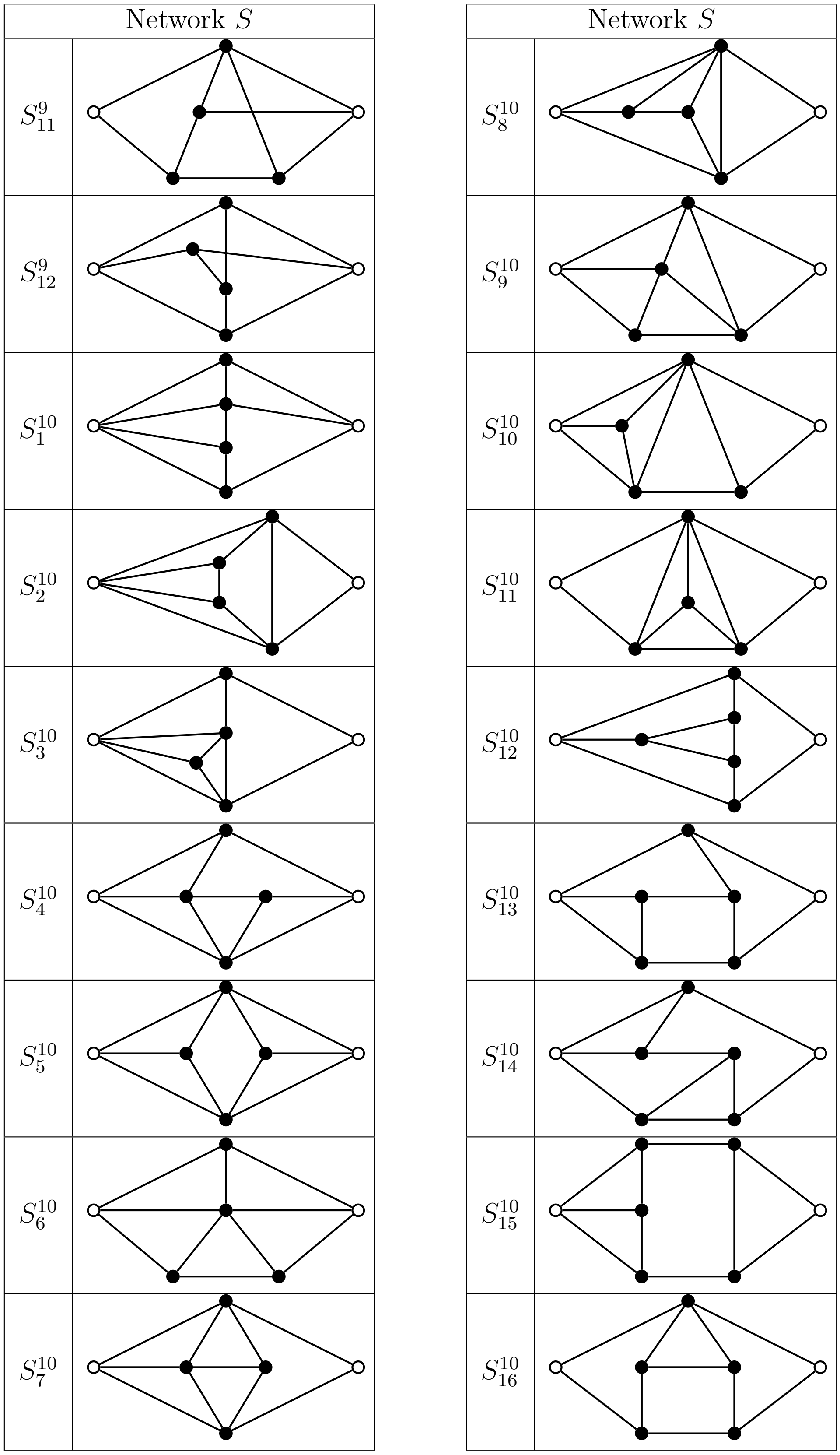


Figure 5.2: Indecomposable Networks with 9 or 10 Edges

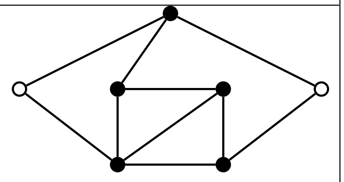
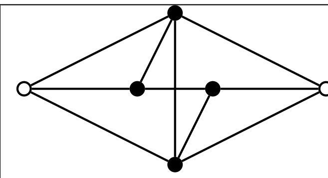
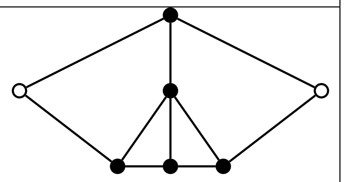
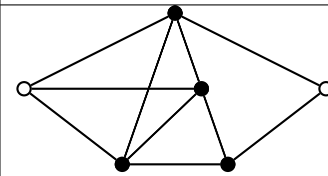
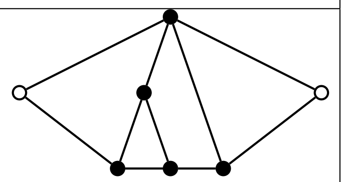
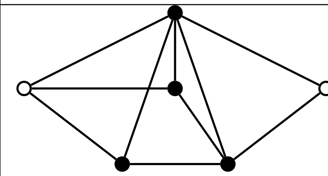
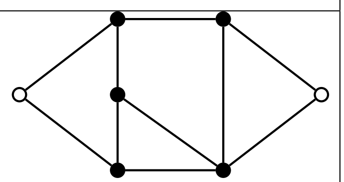
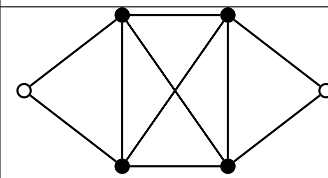
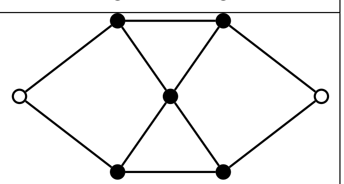
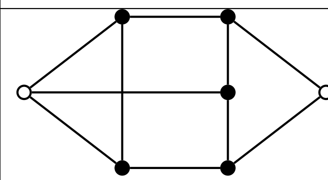
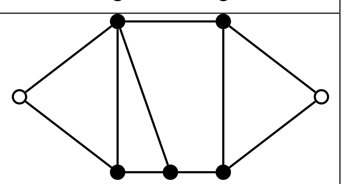
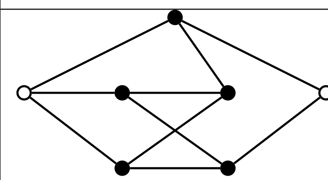
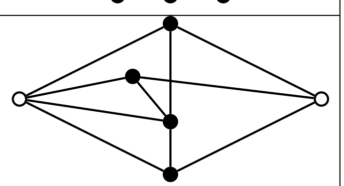
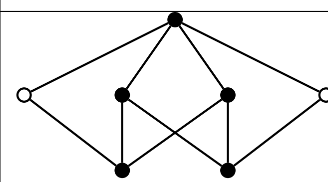
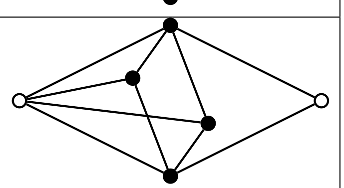
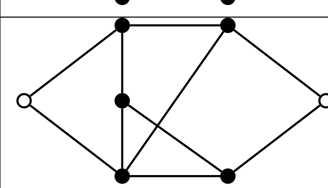
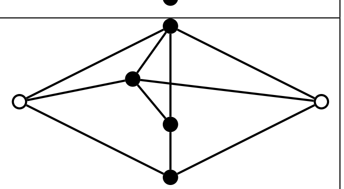
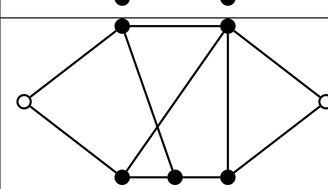
Network S		Network S	
S^{10}_{17}		S^{10}_{26}	
S^{10}_{18}		S^{10}_{27}	
S^{10}_{19}		S^{10}_{28}	
S^{10}_{20}		S^{10}_{29}	
S^{10}_{21}		S^{10}_{30}	
S^{10}_{22}		S^{10}_{31}	
S^{10}_{23}		S^{10}_{32}	
S^{10}_{24}		S^{10}_{33}	
S^{10}_{25}		S^{10}_{34}	

Figure 5.3: Indecomposable Networks with 10 Edges

5.2 Algorithms for Constructing Decision Trees

We describe algorithms with which we obtain an upper bound on values of $h_C(S)$ and $\tau_C(S)$ for each network S in the set of networks considered in this chapter. This includes a dynamic programming algorithm as in [14, 15, 16], a list of greedy algorithms as in [17] and an algorithm (called Bounded Depth) that we create for this thesis. We are interested to find the best method to minimize the gap between the lower and the upper bound for $h_C(S)$ and, consequently, $t_C(S)$.

5.2.1 Dynamic Programming

We describe a dynamic programming algorithm which, for diagnosis of C-faults in a network S, allows us to find exact value of $h_C(S)$. Description is adapted to the problem of diagnosis.

Let $n = L(S)$. For membership queries $\alpha_1, \ldots, \alpha_m \in \{0,1\}^n$ and for $b_1, \ldots, b_m \in \{0,1\}$, we denote by $F_C(S)(\alpha_1, b_1)\ldots(\alpha_m, b_m)$ the set of all functions f from $F_C(S)$ for which $f(\alpha_1) = b_1, \ldots, f(\alpha_m) = b_m$. All such nonempty subsets of $F_C(S)$ including the set $F_C(S)$ are called *separable* subsets of the set $F(S)$. Let $G \subseteq F_C(S)$. Then $G(\alpha_1, b_1)\ldots(\alpha_m, b_m) = F(S)(\alpha_1, b_1)\ldots(\alpha_m, b_m) \cap G$.

First, the dynamic programming algorithm constructs the set $F_C(S)$. Next, this algorithm constructs the set $\Delta_C(S)$ of all separable subsets of the set $F_C(S)$. Each subset $G \in \Delta_C(S)$ is considered as a subproblem of the initial problem $F_C(S)$: for a given function $f \in G$ we should find a normal C-fault ρ for S such that $f = f_{S,\rho}$. We denote by $h(G)$ the minimum depth of a decision tree solving the problem G. For a problem G, we denote by $T(G)$ the set of all membership queries α_i from $\{0,1\}^n$ such that there are two functions $f, g \in G$ for which $f(\alpha_i) \neq g(\alpha_i)$. The considered algorithm is based on the following equalities: $h(G) = 0$ if $|G| = 1$, and $h(G) = \min\{1+\max\{h(G(\alpha_i, 0)), h(G(\alpha_i, 1))\} : \alpha_i \in T(G)\}$ if $|G| \geq 2$. The algorithm begins the computation of the value $h(G)$ from smallest separable subsets and finishes when the value $h(F_C(S))$ is computed.

Note. The dynamic programming approach is limited for the considered problem. It was impossible to use dynamic programming for diagnosis of networks with more than 7 edges. This limitation comes from the need of large memory to store all separable subsets, which grows quickly for networks with more edges.

5.2.2 Greedy Algorithms

We describe greedy algorithms which, for diagnosis of C-faults in a network S, allows us to find an upper bound close to the lower bound on value of $h_C(S)$. Description is adapted to the problem of diagnosis. To describe the work of greedy algorithm, we define *Impurity Function* $I(G, \alpha)$.

We now consider the first step of greedy algorithm – the choice of membership query α which will be attached to the root of the decision tree. We start with $G = F_C(S)$. Each greedy algorithm chooses α which minimizes the value of $I(G, \alpha)$.

Later, the greedy algorithm works in the same way with separable subsets $G(\alpha, 0)$ and $G(\alpha, 1)$ corresponding to the children of the root, etc. The construction of the decision tree is finished when separable subsets corresponding to terminal nodes are singletons.

Let $n = L(S)$. At each step of each greedy algorithm, we have $G \subseteq F_C(S)$ and we denote $N = |G|$. For a given membership query α from $\{0, 1\}^n$, we denote $N_0(\alpha) = |G(\alpha, 0)|$ and $N_1(\alpha) = |G(\alpha, 1)|$.

In total, we experiment with 16 different greedy algorithms given by 16 impurity functions and find that 2 algorithms give the least values for depth of decision tree for all the considered networks. The first greedy algorithm (similar to the one considered in [18]) is based on the following impurity function: $I(G, \alpha) = \max\{N_0(\alpha), N_1(\alpha)\}$. The second greedy algorithm (similar to the one considered in [19]) is based on the following impurity function: $I(G,\alpha) = N_0(\alpha)\log_2 N_0(\alpha) + N_1(\alpha)\log_2 N_1(\alpha)$.

Note. Greedy algorithms do not guarantee finding a tree with minimum depth. In Section 5.3, we obtain decision trees from greedy algorithms whose depth is not minimum and, using other algorithms, it was possible to find trees with lower depth.

5.2.3 Bounded Depth Algorithm

We describe an algorithm which recognizes for a nonnegative integer t if there exists a tree Γ for diagnosis of C-faults in a network S such that $h(\Gamma) \leq t$, and if such tree exists, it constructs one.

Let d be a nonnegative integer and G be a subset of $F_C(S)$ with $|G| \geq 2$. We describe a sequence $T(G,d)$ of membership queries – tuples from $\{0,1\}^n$ where $n = L(S)$. We denote by $A(G)$ the set of membership queries such that $|G(\alpha,0)| \geq 1$ and $|G(\alpha,1)| \geq 1$. We say that two membership queries $\alpha, \beta \in A(G)$ are equivalent if, for any $f \in G$, $f(\alpha) = f(\beta)$. This equivalence relation divides $A(G)$ into classes of equivalence $A_1,\ldots,A_p$. We choose one representative from each class and order these representatives $\alpha_1,\ldots,\alpha_p$ such that, for $i = 1,\ldots,p-1$, $I(G,\alpha_i) \leq I(G,\alpha_{i+1})$ where for $j = 1,\ldots,p$, $I(G,\alpha_j) = \max\{|G(\alpha_j,0)|, |G(\alpha_j,1)|\}$. If $\lceil \log_2 I(G,\alpha_1)\rceil > d-1$, the set $T(G,d)$ is empty. Let $\lceil \log_2 I(G,\alpha_1)\rceil \leq d-1$ and k be the maximum number from $\{1,\ldots,p\}$ such that $\lceil \log_2 I(G,\alpha_k)\rceil \leq d-1$. Then $T(G,d) = (\alpha_1,\ldots,\alpha_k)$.

During each step, the algorithm works with a tree D which is a prefix of some decision tree for diagnosis of C-faults for S. Each node v of the tree D is labeled by a subset $G(v)$ of $F_C(S)$, an integer $d(v)$ and a sequence $T(v)$ of membership queries.

We assume that in each nonterminal node of D, the edge starting in this node and labeled with 0 is going to the left and the edge labeled with 1 is going to the right. In this case, we have a natural ordering of terminal nodes of D from the left to the right.

At the first step, we form the root node r of D for which $G(r) = F_C(S)$, $d(r) = t$ and $T(r) = T(F_C(S), t)$. Let us assume that we already made $m \geq 1$ steps and constructed a tree D. We now describe the step number $m+1$. If each terminal node of D is labeled with a normal C-fault for S, then D is a decision tree for diagnosis of C-faults for the network S with $h(D) \leq t$. If not, we choose the leftmost terminal node v of D which is not labeled with a normal C-fault.

Let $T(v)$ be empty. If v is the root of D, then there is no decision tree Γ for diagnosis of C-faults for S such that $h(\Gamma) \leq t$ and the algorithm finishes its work. If v is not a root of D, then we remove v and the edge entering v. Also, we remove the subtree with root at the sibling v' of v and remove the edge entering v'. Moreover, we remove the membership query α attached to the parent of v. Then, we proceed to step number $m+2$.

Let $T(v)$ be nonempty and α is the first membership query in $T(v)$. We remove α from $T(v)$ and we attach it to the node v. We add to D two nodes v_0, v_1 and we draw edges from v to v_0 and v_1 labeled with 0 and 1 respectively. We attach to v_0 the triple $d(v_0) = d(v) - 1$, $G(v_0) = G(V)(\alpha, 0)$ and $T(v) = T(G(v_0), d(v_0))$. We also attach to v_1 the triple $d(v_1) = d(v) - 1$, $G(v_1) = G(V)(\alpha, 1)$ and $T(v) = T(G(v_1), d(v_1))$. If $|G(v_0)| = 1$, we attach to node v_0 the corresponding normal C-fault for function from $G(v_0)$. Similarly, If $|G(v_1)| = 1$, we attach to node v_1 the corresponding normal C-fault for function from $G(v_1)$. Then, we proceed to step number $m+2$.

Note. We use this algorithm after obtaining results of greedy algorithms *to find exact values of* $h_C(S)$. Given an indecomposable network S with C-faults and $L(S) \leq 10$, we denote by d_{Greedy}, the best estimation for value of $h_C(S)$ using greedy algorithms, and by d_{LB}, the lower bound for $h_C(S)$ obtained from Proposition 4.1. For each network, we obtain a difference between d_{Greedy} and d_{LB} of at most 1. If $d_{\text{Greedy}} = d_{\text{LB}}$, then we have a tree of minimum depth with $h_C(S) = d_{\text{LB}}$. Let $d_{\text{Greedy}} = d_{\text{LB}} + 1$. We now apply the Bounded Depth algorithm with $t = d_{\text{Greedy}} - 1 = d_{\text{LB}}$. If the algorithm constructs a tree, then we have a tree of minimum depth with $h_C(S) = d_{\text{LB}}$. Otherwise, the minimum depth $h_C(S)$ is equal to d_{Greedy}.

Remark 5.1. For diagnosis of C-faults for network S with $L(S) = n$, the algorithm in the worst case tries to attach all possible membership queries from $\{0,1\}^n$ to each node during the construction of decision tree. This means that in the worst case, the running time is exponential. However, it is very unlikely for the algorithm to run in exponential time for the following three facts:

(i) We are using the lower bound $\lceil \log_2 I(G,\alpha) \rceil \leq d-1$ to reduce number of membership queries in $T(G,d)$ for each node v.

(ii) We assign only one representative for each equivalent class. As we go deeper in the tree, $|G|$ becomes smaller and, thus, we have less number of equivalent classes and more equivalent (redundant) membership queries.

(iii) We use ordering in $T(G,d)$ for each node v based on criteria of a greedy algorithm. Thus, it is highly probable that we can find an optimal tree without the need to explore most of the membership queries in $T(G,d)$. It is also clear that if the value of t is chosen to be at least equal to some value obtained by greedy algorithm, then the algorithm runs as fast as the greedy algorithm.

5.3 Experimental Work

In our work we find minimum values for $h_C(S)$ and, consequently, $t_C(S)$ for indecomposable networks S with at most 10 edges for all $C \in \{\{0,1\},\{0\},\{1\}\}$. For each network, we show upper bounds and lower bounds on value of $h_C(S)$. We use two approaches to obtain these bounds with goal to minimize the difference between them.

The first approach (Dynamic-Greedy) is to apply dynamic programming and greedy algorithms (algorithms are described in sections 5.2.1 and 5.2.2 respectively) as follows. We use dynamic programming algorithm for indecomposable networks with $2 \leq i < t$ edges, where for $t < 10$ there exists networks with $n \geq t$ edges in which dynamic programming algorithm cannot work (for 0-1 faults, i is 7 and for 0 faults and 1 faults, i is 8). In this case we use the set of greedy algorithms and nominate the best result for indecomposable networks with $t \leq i \leq 10$. For decision trees Γ obtained using dynamic programming, we say that lower and upper bounds for $h_C(S)$ equal to $h(\Gamma)$. For decision trees Γ obtained using greedy algorithm, we say that the upper bound for $h_C(S)$ equal to $h(\Gamma)$. The lower bound, however, is calculated from Proposition 4.1.

The second approach (Bounded Depth) is to use Bounded Depth algorithm to try to improve decision trees obtained by greedy algorithms whose depth is not minimum, i.e. depth of the obtained tree is not equal to the value obtained from Proposition 4.1.

For each C-fault, we compare between results of each approach and state the number of networks that has improved values for $h_C(S)$ and $t_C(S)$. For diagnosis of 0-1 faults only, we also show results published by Goduhina in [13] and include it in our comparison. In the next pages, we show results for diagnosis of all C-faults. We start with 0-1 faults, then 0 faults, and, finally, 1 faults.

S	$F_C(S)$	$h_C(S)$ lower	$t_C(S)$ lower	Goduhina		Dynamic-Greedy		Bounded Depth	
				$h_C(S)$	$t_C(S)$	$h_C(S)$	$t_C(S)$	$h_C(S)$	$t_C(S)$
S_1^2	5	3	3	3	3	3	3	3	3
S_2^2	5	3	3	3	3	3	3	3	3
S_1^5	54	7[1]	7/4	7	7/4	7	7/4	7	7/4
S_1^7	304	9	3/2	9	3/2	9	3/2	9	3/2
S_2^7	304	9	3/2	10	5/3	10	5/3	9	3/2
S_1^8	759	10	10/7	11	11/7	11	11/7	10	10/7
S_2^8	778	10	10/7	11	11/7	11	11/7	10	10/7
S_3^8	778	10	10/7	11	11/7	11	11/7	10	10/7
S_4^8	759	10	10/7	11	11/7	11	11/7	10	10/7
S_5^8	959	10	10/7	12	12/7	11	11/7	10	10/7
S_1^9	1899	11	11/8	N/A	N/A	12	3/2	11	11/8
S_2^9	1567	11	11/8	N/A	N/A	12	3/2	11	11/8
S_3^9	1847	11	11/8	N/A	N/A	12	3/2	11	11/8
S_4^9	1858	11	11/8	N/A	N/A	12	3/2	11	11/8
S_5^9	1858	11	11/8	N/A	N/A	12	3/2	11	11/8
S_6^9	1899	11	11/8	N/A	N/A	12	3/2	11	11/8
S_7^9	1567	11	11/8	N/A	N/A	12	3/2	11	11/8
S_8^9	1847	11	11/8	N/A	N/A	12	3/2	11	11/8
S_9^9	2164	12	3/2	N/A	N/A	12	3/2	12	3/2
S_{10}^9	2229	12	3/2	N/A	N/A	12	3/2	12	3/2
S_{11}^9	2210	12	3/2	N/A	N/A	13	13/8	12	3/2
S_{12}^9	2188	12	3/2	N/A	N/A	12	3/2	12	3/2
S_1^{10}	4270	13	13/9	N/A	N/A	13	13/9	13	13/9
S_2^{10}	4376	13	13/9	N/A	N/A	13	13/9	13	13/9

Table 5.1: Results for Diagnosis of 0-1 Faults in Networks with up to 10 Edges

[1]Theoretically, the lower bound on depth is 6. However, using dynamic programming, we show that the lower bound is actually 7.

S	$F_C(S)$	$h_C(S)$ lower	$t_C(S)$ lower	Goduhina		Dynamic-Greedy		Bounded Depth	
				$h_C(S)$	$t_C(S)$	$h_C(S)$	$t_C(S)$	$h_C(S)$	$t_C(S)$
S_3^{10}	4369	13	13/9	N/A	N/A	14	14/9	13	13/9
S_4^{10}	4911	13	13/9	N/A	N/A	14	14/9	13	13/9
S_5^{10}	4891	13	13/9	N/A	N/A	14	14/9	13	13/9
S_6^{10}	4286	13	13/9	N/A	N/A	13	13/9	13	13/9
S_7^{10}	5098	13	13/9	N/A	N/A	14	14/9	13	13/9
S_8^{10}	4464	13	13/9	N/A	N/A	14	14/9	13	13/9
S_9^{10}	4960	13	13/9	N/A	N/A	14	14/9	13	13/9
S_{10}^{10}	4329	13	13/9	N/A	N/A	13	13/9	13	13/9
S_{11}^{10}	4444	13	13/9	N/A	N/A	13	13/9	13	13/9
S_{12}^{10}	4444	13	13/9	N/A	N/A	13	13/9	13	13/9
S_{13}^{10}	4960	13	13/9	N/A	N/A	14	14/9	13	13/9
S_{14}^{10}	4329	13	13/9	N/A	N/A	13	13/9	13	13/9
S_{15}^{10}	4286	13	13/9	N/A	N/A	14	14/9	13	13/9
S_{16}^{10}	5098	13	13/9	N/A	N/A	14	14/9	13	13/9
S_{17}^{10}	4464	13	13/9	N/A	N/A	13	13/9	13	13/9
S_{18}^{10}	4376	13	13/9	N/A	N/A	14	14/9	13	13/9
S_{19}^{10}	4369	13	13/9	N/A	N/A	13	13/9	13	13/9
S_{20}^{10}	4911	13	13/9	N/A	N/A	14	14/9	13	13/9
S_{21}^{10}	4891	13	13/9	N/A	N/A	13	13/9	13	13/9
S_{22}^{10}	4270	13	13/9	N/A	N/A	13	13/9	13	13/9
S_{23}^{10}	4912	13	13/9	N/A	N/A	13	13/9	13	13/9
S_{24}^{10}	4980	13	13/9	N/A	N/A	14	14/9	13	13/9
S_{25}^{10}	5660	13	13/9	N/A	N/A	14	14/9	13	13/9
S_{26}^{10}	5561	13	13/9	N/A	N/A	14	14/9	13	13/9
S_{27}^{10}	5641	13	13/9	N/A	N/A	14	14/9	13	13/9
S_{28}^{10}	5010	13	13/9	N/A	N/A	14	14/9	13	13/9

Table 5.2: Results for Diagnosis of 0-1 Faults in Networks with 10 Edges

S	$F_C(S)$	$h_C(S)$ lower	$t_C(S)$ lower	Goduhina		Dynamic-Greedy		Bounded Depth	
				$h_C(S)$	$t_C(S)$	$h_C(S)$	$t_C(S)$	$h_C(S)$	$t_C(S)$
S_{29}^{10}	5683	13	13/9	N/A	N/A	14	14/9	13	13/9
S_{30}^{10}	4458	13	13/9	N/A	N/A	14	14/9	13	13/9
S_{31}^{10}	5706	13	13/9	N/A	N/A	14	14/9	13	13/9
S_{32}^{10}	5768	13	13/9	N/A	N/A	14	14/9	13	13/9
S_{33}^{10}	5691	13	13/9	N/A	N/A	14	14/9	13	13/9
S_{34}^{10}	5623	13	13/9	N/A	N/A	14	14/9	13	13/9

Table 5.3: Results for Diagnosis of 0-1 Faults in Networks with 10 Edges

For $i = 2, 5, 7, 8, 9, 10$, Figure 5.4 summarizes results of tables 5.1-5.3 as follows: for every i, we evaluate the average depth for networks in the set of indecomposable networks with i edges. Also, Figure 5.5 shows for each basis of all indecomposable networks with i edges, $B(i)$, the value of $t_{\{0,1\}}$ as follows: $t_{\{0,1\}}(B(i)) = \max\{\tau_{\{0,1\}}(Q) : Q \in B(i)\}$.

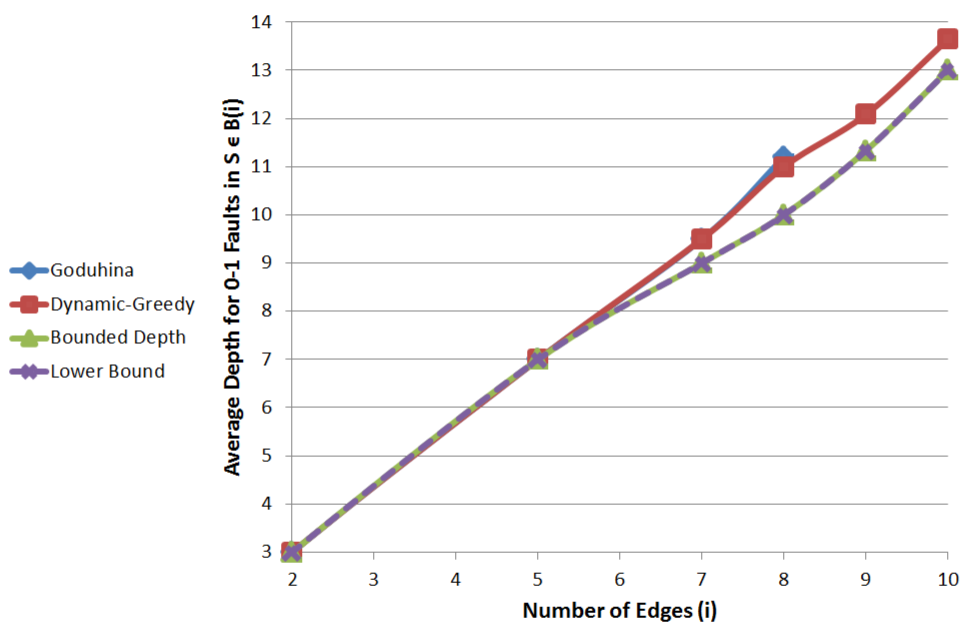


Figure 5.4: Average Depth of Decision Trees per $B(i)$ for Networks with 0-1 faults

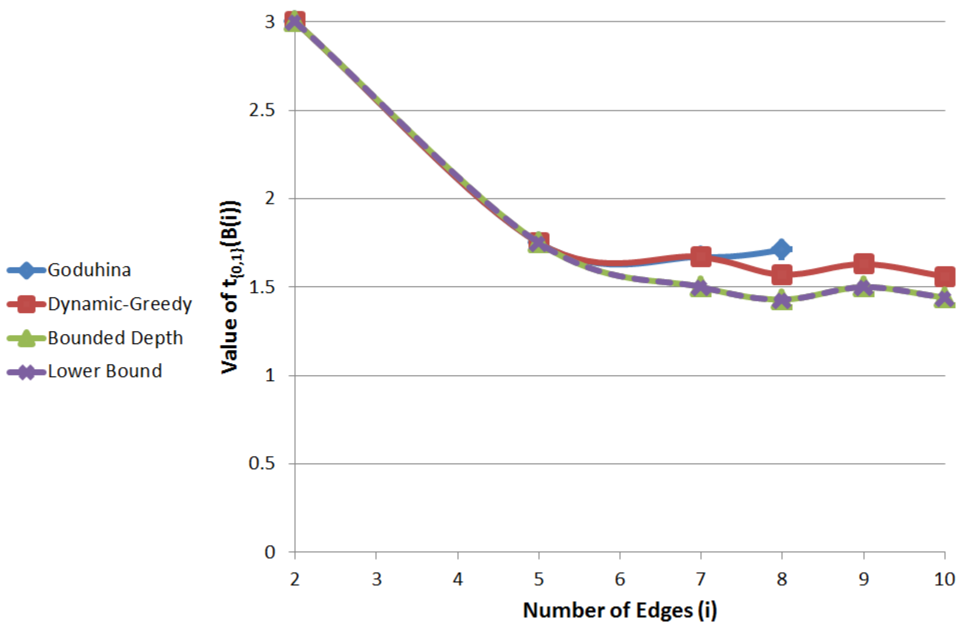


Figure 5.5: Values of $t_{\{0,1\}}(B(i))$ vs. Number of Edges for Networks with 0-1 faults

Using Dynamic-Greedy approach, we improved the result of Goduhina in two networks: S_1^5 (in which we showed, using dynamic programming, that lower bound is 7 rather than 6) and S_5^8. Using Bounded Depth approach, we improved the results of Dynamic-Greedy in 37 networks out of 56. All values out of Bounded Depth approach are minimum.

S	$F_{\{0\}}(S)$	$h_{\{0\}}(S)$ lower	$t_{\{0\}}(S)$ lower	Dynamic-Greedy		Bounded Depth	
				$h_{\{0\}}(S)$	$t_{\{0\}}(S)$	$h_{\{0\}}(S)$	$t_{\{0\}}(S)$
S_1^2	4	2	2	2	2	2	2
S_2^2	2	1	1	1	1	1	1
S_1^5	11	4	1	4	1	4	1
S_1^7	27	5	5/6	5	5/6	5	5/6
S_2^7	30	5	5/6	5	5/6	5	5/6
S_1^8	68	7	1	7	1	7	1
S_2^8	73	7	1	7	1	7	1
S_3^8	32	5	5/7	6	6/7	5	5/7
S_4^8	35	6	6/7	6	6/7	6	6/7
S_5^8	36	6	6/7	6	6/7	6	6/7
S_1^9	77	7	7/8	7	7/8	7	7/8
S_2^9	63	6	3/4	6	3/4	6	3/4
S_3^9	72	7	7/8	7	7/8	7	7/8
S_4^9	77	7	7/8	7	7/8	7	7/8
S_5^9	82	7	7/8	7	7/8	7	7/8
S_6^9	82	7	7/8	7	7/8	7	7/8
S_7^9	77	7	7/8	7	7/8	7	7/8
S_8^9	91	7	7/8	7	7/8	7	7/8
S_9^9	192	8	1	9	9/8	8	1
S_{10}^9	89	7	7/8	7	7/8	7	7/8
S_{11}^9	84	7	7/8	7	7/8	7	7/8
S_{12}^9	75	7	7/8	7	7/8	7	7/8
S_1^{10}	168	8	8/9	8	8/9	8	8/9
S_2^{10}	185	8	8/9	8	8/9	8	8/9
S_3^{10}	177	8	8/9	8	8/9	8	8/9

Table 5.4: Results for Diagnosis of 0 Faults in Networks with up to 10 Edges

S	$F_{\{0\}}(S)$	$h_{\{0\}}(S)$ lower	$t_{\{0\}}(S)$ lower	Dynamic-Greedy		Bounded Depth	
				$h_{\{0\}}(S)$	$t_{\{0\}}(S)$	$h_{\{0\}}(S)$	$t_{\{0\}}(S)$
S_4^{10}	201	8	8/9	8	8/9	8	8/9
S_5^{10}	209	8	8/9	9	1	8	8/9
S_6^{10}	182	8	8/9	8	8/9	8	8/9
S_7^{10}	210	8	8/9	9	1	8	8/9
S_8^{10}	191	8	8/9	9	1	8	8/9
S_9^{10}	218	8	8/9	9	1	8	8/9
S_{10}^{10}	206	8	8/9	9	1	8	8/9
S_{11}^{10}	200	8	8/9	9	1	8	8/9
S_{12}^{10}	81	7	7/9	7	7/9	7	7/9
S_{13}^{10}	89	7	7/9	7	7/9	7	7/9
S_{14}^{10}	81	7	7/9	7	7/9	7	7/9
S_{15}^{10}	91	7	7/9	7	7/9	7	7/9
S_{16}^{10}	94	7	7/9	7	7/9	7	7/9
S_{17}^{10}	86	7	7/9	7	7/9	7	7/9
S_{18}^{10}	86	7	7/9	7	7/9	7	7/9
S_{19}^{10}	91	7	7/9	7	7/9	7	7/9
S_{20}^{10}	99	7	7/9	7	7/9	7	7/9
S_{21}^{10}	94	7	7/9	7	7/9	7	7/9
S_{22}^{10}	100	7	7/9	7	7/9	7	7/9
S_{23}^{10}	173	8	8/9	8	8/9	8	8/9
S_{24}^{10}	189	8	8/9	8	8/9	8	8/9
S_{25}^{10}	206	8	8/9	9	1	8	8/9
S_{26}^{10}	213	8	8/9	9	1	8	8/9
S_{27}^{10}	222	8	8/9	9	1	8	8/9
S_{28}^{10}	203	8	8/9	8	8/9	8	8/9

Table 5.5: Results for Diagnosis of 0 Faults in Networks with 10 Edges

S	$F_{\{0\}}(S)$	$h_{\{0\}}(S)$ lower	$t_{\{0\}}(S)$ lower	Dynamic-Greedy		Bounded Depth	
				$h_{\{0\}}(S)$	$t_{\{0\}}(S)$	$h_{\{0\}}(S)$	$t_{\{0\}}(S)$
S_{29}^{10}	245	8	8/9	9	1	8	8/9
S_{30}^{10}	84	7	7/9	7	7/9	7	7/9
S_{31}^{10}	91	7	7/9	8	8/9	7	7/9
S_{32}^{10}	100	7	7/9	8	8/9	7	7/9
S_{33}^{10}	100	7	7/9	7	7/9	7	7/9
S_{34}^{10}	106	7	7/9	7	7/9	7	7/9

Table 5.6: Results for Diagnosis of 0 Faults in Networks with 10 Edges

For $i = 2, 5, 7, 8, 9, 10$, Figure 5.6 summarizes results of tables 5.4-5.6 as follows: for every i, we evaluate the average depth for networks in the set of indecomposable networks with i edges. Also, Figure 5.7 shows for each basis of all indecomposable networks with i edges, $B(i)$, the value of $t_{\{0\}}$ as follows: $t_{\{0\}}(B(i)) = \max\{\tau_{\{0\}}(Q) : Q \in B(i)\}$.

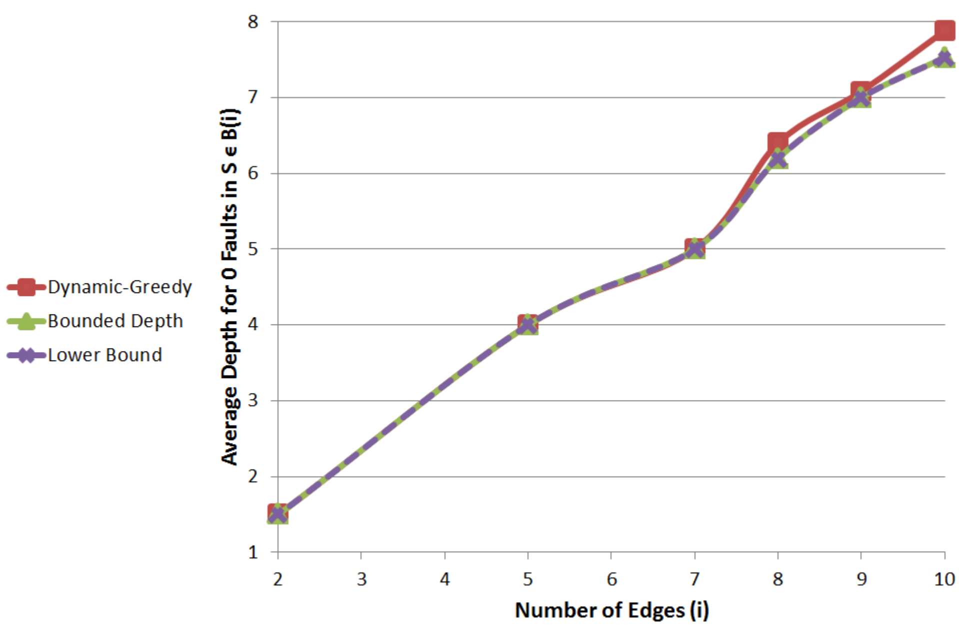


Figure 5.6: Average Depth of Decision Trees per $B(i)$ for Networks with 0 faults

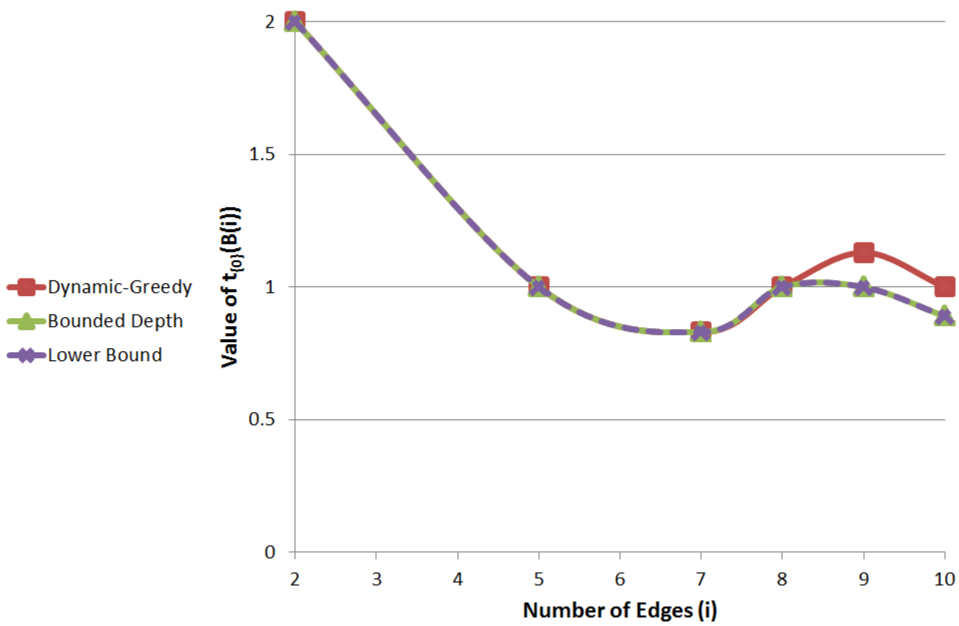


Figure 5.7: Values of $t_{\{0\}}(B(i))$ vs. Number of Edges for Networks with 0 faults

Using Bounded Depth approach, we improve the results of Dynamic-Greedy approach in 14 out of 56 networks. All values out of Bounded Depth approach are minimum.

S	$F_{\{1\}}(S)$	$h_{\{1\}}(S)$ lower	$t_{\{1\}}(S)$ lower	Dynamic-Greedy		Bounded Depth	
				$h_{\{1\}}(S)$	$t_{\{1\}}(S)$	$h_{\{1\}}(S)$	$t_{\{1\}}(S)$
S_1^2	2	1	1	1	1	1	1
S_2^2	4	2	2	2	2	2	2
S_1^5	11	4	1	4	1	4	1
S_1^7	30	5	5/6	5	5/6	5	5/6
S_2^7	27	5	5/6	5	5/6	5	5/6
S_1^8	35	6	6/7	6	6/7	6	6/7
S_2^8	32	5	5/7	5	5/7	5	5/7
S_3^8	73	7	1	7	1	7	1
S_4^8	68	7	1	7	1	7	1
S_5^8	82	7	1	7	1	7	1
S_1^9	82	7	7/8	7	7/8	7	7/8
S_2^9	77	7	7/8	7	7/8	7	7/8
S_3^9	91	7	7/8	7	7/8	7	7/8
S_4^9	82	7	7/8	7	7/8	7	7/8
S_5^9	77	7	7/8	7	7/8	7	7/8
S_6^9	77	7	7/8	7	7/8	7	7/8
S_7^9	63	6	3/4	6	3/4	6	3/4
S_8^9	72	7	7/8	7	7/8	7	7/8
S_9^9	38	6	3/4	6	3/4	6	3/4
S_{10}^9	86	7	7/8	7	7/8	7	7/8
S_{11}^9	91	7	7/8	7	7/8	7	7/8
S_{12}^9	101	7	7/8	8	1	7	7/8
S_1^{10}	100	7	7/9	7	7/9	7	7/9
S_2^{10}	86	7	7/9	7	7/9	7	7/9
S_3^{10}	91	7	7/9	7	7/9	7	7/9

Table 5.7: Results for Diagnosis of 1 Faults in Networks with up to 10 Edges

S	$F_{\{1\}}(S)$	$h_{\{1\}}(S)$ lower	$t_{\{1\}}(S)$ lower	Dynamic-Greedy		Bounded Depth	
				$h_{\{1\}}(S)$	$t_{\{1\}}(S)$	$h_{\{1\}}(S)$	$t_{\{1\}}(S)$
S_4^{10}	99	7	7/9	7	7/9	7	7/9
S_5^{10}	94	7	7/9	7	7/9	7	7/9
S_6^{10}	91	7	7/9	7	7/9	7	7/9
S_7^{10}	94	7	7/9	7	7/9	7	7/9
S_8^{10}	86	7	7/9	7	7/9	7	7/9
S_9^{10}	89	7	7/9	7	7/9	7	7/9
S_{10}^{10}	81	7	7/9	7	7/9	7	7/9
S_{11}^{10}	81	7	7/9	7	7/9	7	7/9
S_{12}^{10}	200	8	8/9	9	1	8	8/9
S_{13}^{10}	218	8	8/9	9	1	8	8/9
S_{14}^{10}	206	8	8/9	9	1	8	8/9
S_{15}^{10}	182	8	8/9	8	8/9	8	8/9
S_{16}^{10}	210	8	8/9	8	8/9	8	8/9
S_{17}^{10}	191	8	8/9	8	8/9	8	8/9
S_{18}^{10}	185	8	8/9	8	8/9	8	8/9
S_{19}^{10}	177	8	8/9	8	8/9	8	8/9
S_{20}^{10}	201	8	8/9	8	8/9	8	8/9
S_{21}^{10}	209	8	8/9	8	8/9	8	8/9
S_{22}^{10}	168	8	8/9	8	8/9	8	8/9
S_{23}^{10}	110	7	7/9	8	8/9	7	7/9
S_{24}^{10}	100	7	7/9	7	7/9	7	7/9
S_{25}^{10}	107	7	7/9	8	8/9	7	7/9
S_{26}^{10}	102	7	7/9	7	7/9	7	7/9
S_{27}^{10}	97	7	7/9	7	7/9	7	7/9
S_{28}^{10}	95	7	7/9	7	7/9	7	7/9

Table 5.8: Results for Diagnosis of 1 Faults in Networks with 10 Edges

S	$F_{\{1\}}(S)$	$h_{\{1\}}(S)$ lower	$t_{\{1\}}(S)$ lower	Dynamic-Greedy		Bounded Depth	
				$h_{\{1\}}(S)$	$t_{\{1\}}(S)$	$h_{\{1\}}(S)$	$t_{\{1\}}(S)$
S_{29}^{10}	91	7	7/9	7	7/9	7	7/9
S_{30}^{10}	233	8	8/9	9	1	8	8/9
S_{31}^{10}	241	8	8/9	9	1	8	8/9
S_{32}^{10}	219	8	8/9	9	1	8	8/9
S_{33}^{10}	224	8	8/9	8	8/9	8	8/9
S_{34}^{10}	210	8	8/9	8	8/9	8	8/9

Table 5.9: Results for Diagnosis of 1 Faults in Networks with 10 Edges

For $i = 2, 5, 7, 8, 9, 10$, Figure 5.8 summarizes results of tables 5.7-5.9 as follows: for every i, we evaluate the average depth for networks in the set of indecomposable networks with i edges. Also, Figure 5.9 shows for each basis of all indecomposable networks with i edges, $B(i)$, the value of $t_{\{1\}}$ as follows: $t_{\{1\}}(B(i)) = \max\{\tau_{\{1\}}(Q) : Q \in B(i)\}$.

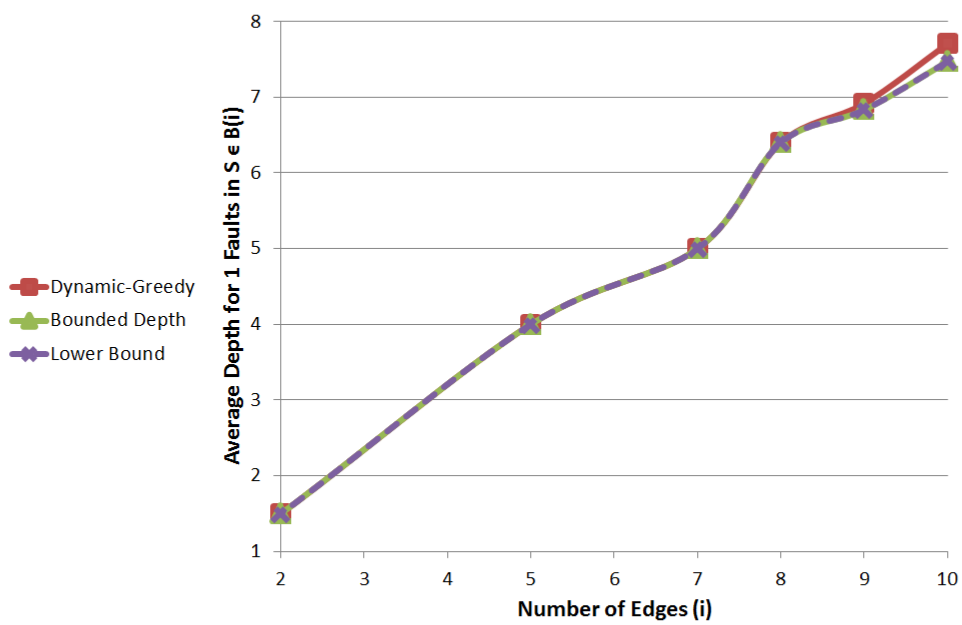


Figure 5.8: Average Depth of Decision Trees per $B(i)$ for Networks with 1 faults

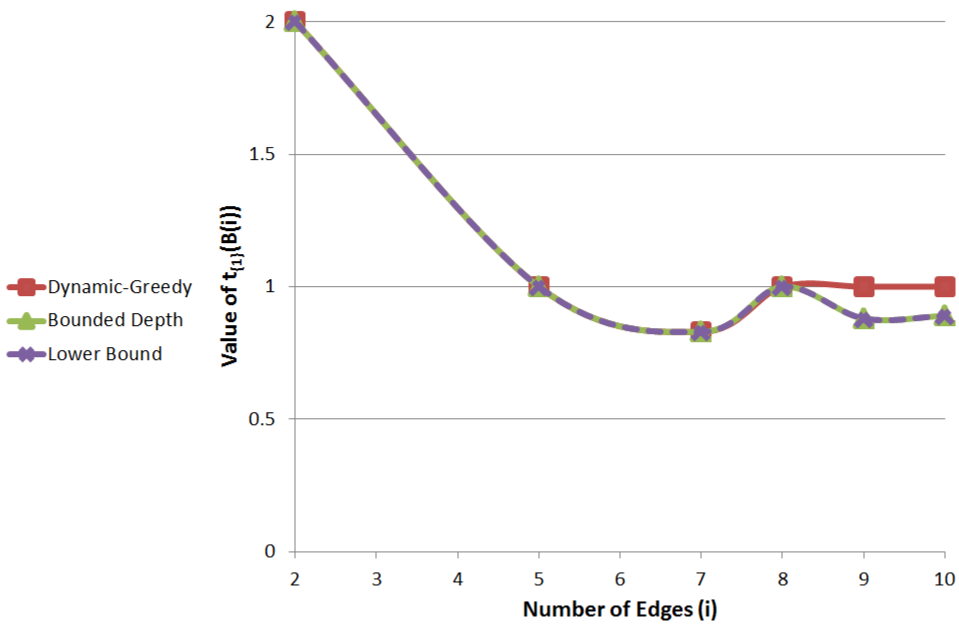


Figure 5.9: Values of $t_{\{1\}}(B(i))$ vs. Number of Edges for Networks with 1 faults

Using Bounded Depth approach, we improved the results of Dynamic-Greedy approach in 9 out of 56 networks. All values out of Bounded Depth approach are minimum.

5.4 Analysis of the Results

From tables of the previous section we can obtain some results. Using Bounded Depth algorithm, we are able to find trees with minimum depth for all the considered networks. The runtime for the algorithm is very reasonable. Using a Python script to write the algorithm, the run time for the algorithm does not take more than 30 seconds for any of the considered networks.

From the previous tables, we obtain a number of corollaries about values of $t_C(B)$. For $i = 2, 5, 7, 8, 9, 10$, we denote by $B(i)$ the set of all indecomposable networks with i edges. Table 5.10 shows our best obtained upper bounds for $t_C(B(i))$ for all types of C-faults. All of the obtained values are sharp.

i	0-1 Faults	0 Faults	1 Faults
2	3	2	2
5	7/4	1	1
7	3/2	5/6	5/6
8	10/7	1	1
9	3/2	1	7/8
10	13/9	8/9	8/9

Table 5.10: Values of $t_C(B(i))$ for Indecomposable Networks with up to 10 Edges

Chapter 6

Concluding Remarks

In this thesis, we study depth of decision trees for diagnosis of constant faults, i.e. 0-1 faults, 0 faults and 1 faults, in read-once contact networks over finite bases. For any finite basis, we prove Theorem 4.2 which gives us a linear upper bound on the minimum depth of decision tree for diagnosis of constant faults depending on the number of edges in a network.

Theorems 4.3, 4.4 and 4.5 study bounds on depth of decision trees for diagnosis of each type of constant faults depending on number of edges in networks in the worst case per basis. For all constant faults, we obtain linear bounds with exception of two cases: diagnosis of 0 faults with $B = \{S_2^2\}$ and diagnosis of 1 faults with $B = \{S_2^2\}$. In these special cases, we obtain constant bounds.

We study the set of indecomposable networks with up to 10 edges with objective to minimize depth of decision tree for diagnosis of constant faults and, thus, minimize coefficients for upper bounds obtained in Theorem 4.2 for basis of indecomposable networks with up to 10 edges. Using the Bounded Depth algorithm described in Section 5.2.3, we obtain exact values for minimum depth of decision trees for all the considered networks which cannot be obtained using common greedy-based methods described in Section 5.2.2. It is interesting to consider in the future minimizing average depth of decision trees. Knuth in [20] gives a lower bound on average depth of decision tree which we can use for estimating minimum average depth.

REFERENCES

[1] C.E. Shannon, A symbolic analysis of relay and switching circuits, Trans. AIEE, 57 (1938) 713–723.

[2] I.A. Chegis, S.V. Yablonskii, Logical methods for testing the operation of electrical circuits, Trudy Mat. Inst. Steklov., 51 (1958) 270–360 (in Russian).

[3] N.A. Karpova, Contact scheme, Encyclopedia of Mathematics, `http://www.encyclopediaofmath.org/index.php/Contact_scheme`

[4] A.V. Kuznetsov, Non-repeating contact schemes and non-repeating superpositions of functions of algebra of logic, Trudy Mat. Inst. Steklov., 51 (1958) 186–225 (in Russian).

[5] V. Raghavan, S. Schach, Learning switch configurations, in: Proc. 3rd Annual Workshop on Computational Learning Theory, Morgan Kaufmann Publishers, 1990, pp. 38–51.

[6] D. Angluin, L. Hellerstein, M. Karpinski, Learning read-once formulas with queries, Journal of the ACM, 40 (1993) 185–210.

[7] Kh.A. Madatyan, On complete tests for contact circuits without repetitions, Problemy Kibernet., 23 (1970) 103–118 (in Russian).

[8] M. Moshkov, B. Zielosko, Sets of tests, decision rules and trees, in: Combinatorial Machine Learnaing, A Rough Set Approach, Springer, 2011, pp. 23–36.

[9] M. Moshkov, Diagnosis of constant faults in circuits, in: O.B. Lupanov (Ed.), Mathematical Problems of Cybernetics, vol. 9, Nauka Publishers, Moscow, 2000, pp. 79–100 (in Russian).

[10] R.S. Goldman, V.P. Chipulis, Diagnosis of iteration-free combinatorial circuits, Diskret. Analiz 14 (1969) 3–15 (in Russian).

[11] M.F. Karavai, Diagnosis of treelike circuits of arbitrary basis, Avtomat. i Telemekhan., 34 (1973) 173–181 (in Russian).

[12] M. Moshkov, Diagnosis of 0-1 faults in iteration-free contact networks, in: Proc. 6th Scientific Conference of Young Scientists of Volga-Vyatka Region, Gorky, 1986, pp. 103–104 (in Russian).

[13] G.B. Goduhina, Experimental study of conditional tests for diagnosis of constant faults in iteration-free contact networks, in: Al.A. Markov (Ed.), Combinatorial-Algebraic and Probabilistic Methods in Applied Mathematics, Gorky State University, Gorky, 1988, pp. 16–21 (in Russian).

[14] A. Alkhalid, I. Chikalov, S. Hussain, M. Moshkov, Extensions of dynamic programming as a new tool for decision tree optimization, in: S. Ramanna, R.J. Howlett, L.C. Jain (Eds.), Emerging Paradigms in Machine Learning, Smart Innovation, Systems and Technologies, vol. 13, Springer, Heidelberg, 2013, pp. 11–29.

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

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

[17] A. Alkhalid, I. Chikalov, M. Moshkov, Comparison of greedy algorithms for decision tree optimization, in: A. Skowron, Z. Suraj (Eds.), Rough Sets and Intelligent Systems – Professor Zdzislaw Pawlak in Memoriam, Intelligent Systems Reference Library, vol. 43, Springer, Heidelberg, 2013, pp. 21–40.

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

[19] J.R. Quinlan, Induction of decision trees, Machine Learning, 1 (1986) 81–106.

[20] D.E. Knuth, The art of computer programming, vol. 3: Sorting and searching. Addison-Wesley series in computer science, 1973.

APPENDICES

A Papers Submitted and Under Preparation

• Monther Busbait, Igor Chikalov, Shahid Hussain, Mikhail Moshkov, "Diagnosis of Constant Faults in Read-Once Contact Networks over Finite Bases", *Submitted to Discrete Applied Mathematics*, June 2013 – "Can be accepted with minor changes."

• Monther Busbait, Mikhail Moshkov, "Extended Diagnosis of Constant Faults in Read-Once Contact Networks over Finite Bases", *In Progress.*