On the Test Complexity of VLSI Systems

Hongzhong Wu
University of Saarland

Dissertation

1994

Contents

Zusammenfassung ... 1

Preface ... 5

1 VLSI Systems and Tests ... **1**
- 1.1 X-Category and VLSI Systems ... 1
- 1.2 Fault Models and Tests ... 5
- 1.3 Functional Test ... 7
- 1.4 Structural Test ... 11

2 Assignment Complexity of Uniform Trees ... **23**
- 2.1 Assignment Complexity of Uniform Trees ... 23
- 2.2 $\Theta(1)$ Assignable Trees ... 27
- 2.3 Jump from $\Theta(1)$ to $\Omega((\lg n)^{\frac{1}{m-1}})$ ... 32
- 2.4 Problem Conversion ... 33
- 2.5 Commutative Trees ... 37
- 2.6 Decidability ... 42

3 Test Complexity of Uniform Trees ... **47**
- 3.1 Faults and Diagnosis Signals ... 47
- 3.2 $\Theta(1)$ Testable and TLP_f ... 53
- 3.3 Jump from $\Theta(1)$ to $\Omega(\lg n)$ ... 57
- 3.4 Jump from $O(\lg n)$ to $\Omega(n^\alpha)$ ... 62
- 3.5 Arrangement Complexity ... 67

4 Test Complexity of Uniform Tree Circuits ... **74**
- 4.1 Uniform Tree Circuits ... 74
- 4.2 Commutative Tree Circuits ... 77
- 4.3 Unate Tree Circuits ... 83
- 4.4 Summary ... 93

5 Synthesis of $O(\lg n)$ Testable Trees ... **95**
- 5.1 Kernel Sensitive Functions ... 96
- 5.2 Synthesis of Functions ... 99
- 5.3 Synthesis of Trees ... 102

6 An Approach to Pseudoexhaustive Testing **105**
6.1 Introduction 105
6.2 Divide and Conquer 106
6.3 Partition Algorithm 108
6.4 Basic Problems 113
6.5 Applications and Computation Complexity 114

7 Monomial Oriented Pseudorandom Test **120**
7.1 Introduction 120
7.2 k-Monomial and its Probability 121
7.3 Expected Test Length 122
7.4 Experiments 125

Concluding Remarks **130**

Bibliography **133**

Zusammenfassung

Mit dem zunehmenden Einsatz von VLSI-Systemen sind die Anforderungen an ihre Zuverlässigkeit immer mehr gestiegen. Die Zuverlässigkeit eines VLSI-Systems hängt von seinen Komponenten – hochintegrierten Schaltkreisen – ab. Leider ist der Fertigungsprozeß hochintergrierter Schaltkreise extrem fehleranfällig. Nach inoffiziellen Angaben beträgt die Defektrate für große Schaltkreise bei einem neuen Fertigungsprozeß über 60%. Daher ist ein Test der Schaltkreise unbedingt notwendig. Allerdings beträgt der Aufwand für solche Tests mehr als 25% der Gesamtkosten.

Normalerweise enthält ein VLSI-System sowohl kombinatorische als auch sequentielle Schaltkreise. Mit Hilfe von Prüfbussen kann das Testproblem für die sequentiellen Komponenten auf den kombinatorischen Fall zurückgeführt werden. Deshalb spielt der Test von kombinatorischen Schaltkreisen eine große Rolle. Diese Arbeit betrachtet das Testproblem kombinatorischer Schaltkreise.

Ein vollständiger Test eines Schaltkreises durch Anlegen aller Eingaben ist in der Praxis fast immer unmöglich. Deswegen müssen Annahmen über die Art der am häufigsten vorkommenden Fehler gemacht werden, die dann in einem Fehlermodell zusammengefaßt werden. Das am häufigsten in der Praxis verwendete Fehlermodell ist das Single-Stuck-at-Fehlermodell. Hier wird angenommen, daß innerhalb des ganzen Schaltkreises höchstens eine Leitung ständig auf einem festen logischen Wert (d.h. 0 oder 1) liegt. Dieses populäre Fehlermodell kann jedoch nicht alle auftretenden Fehler überdecken. In dieser Arbeit betrachten wir daher zusätzlich das mächtigere Einzel-Zellenfehlermodell.

Die Testkosten werden bestimmt durch die Kosten der Testerzeugung und der Testdurchführung. Wir definieren die Testkomplexität eines Schaltkreises S als die minimale Anzahl von Testmustern, die man benötigt, um S nach dem gegebenen Fehlermodell zu prüfen.

Das Schwergewicht der vorliegenden Arbeit liegt auf der Untersuchung der Testprobleme bezüglich baumartiger Schaltkreise, pseudoerschöpfend und pseudozufällig testbarer Schaltkreise, sowie auf der Entwicklung von Verfahren zur Erzeugung optimaler Testmustermengen.

Die Arbeit gliedert sich in sieben Kapitel. Das erste Kapitel enthält eine formale Beschreibung von VLSI-Schaltkreisen und des Testproblems mit Hilfe der X-Kategorie [Hotz65, Hotz74]. Diese Formulierung führt zu einer Vereinfachung der Diskussion und zu einer Verallgemeinerung der Resultate.

Wir bezeichnen im folgenden baumartige Schaltkreise als Bäume. Ein Baum ist

uniform, falls alle Knoten des Baums die selbe Funktion realisieren. Bäume sind Basiskomponenten von vielen VLSI-Systemen, insbesondere auch von parallelen Architekturen. Im folgenden werden nur uniforme Bäume betrachtet. Bekannt ist, daß bestimmte baumartige Schaltkreise bzgl. ihrer Testbarkeit nach Klassen partitioniert werden können [BeSp91]. Dies hat uns motiviert, die Testkomplexität von allgemeinen baumartigen Schaltkreisen zu untersuchen. Unsere Untersuchungen zielen dabei ab auf die Generierung einer minimalen Testmustermenge und eine eventuelle Modifikation des Schaltkreises.

Vom zweiten bis zum fünften Kapitel konzentrieren wir uns auf die Untersuchung der Testkomplexität baumartiger Schaltkreise nach dem Einzel-Zellenfehlermodell.

Wir benutzen $T_f^{(n)}$ als Bezeichnung für einen über der Funktion f definierten, balancierten baumartigen Schaltkreis mit n Eingängen. Eine Zelle von $T_f^{(n)}$ ist fehlerhaft, falls für eine bestimmte Belegung dieser Zelle der Ausgangswert nicht richtig ist. Um solche Fehler zu testen, müssen wir an dem Schaltkreis ein n-stelliges Muster anlegen. Um einen konkreten Fehler zu entdecken, muß dieses Testmuster sowohl eine den Fehler produzierende Belegung der Eingänge der fehlerhaften Zelle erzeugen, als auch dafür sorgen, daß das entsprechende fehlerhafte Signal zum primären Ausgang propagiert wird. Eine vollständige Testmustermenge für $T_f^{(n)}$ besteht aus Testmustern, die alle Fehler des zugrundeliegenden Fehlermodells testen können. Die unmittelbaren Fragen sind: Wie groß ist die Testkomplexität von $T_f^{(n)}$ und wie kann eine optimale Testmustermenge erzeugt werden?

Eine Mustermenge heißt eine vollständige Belegung für $T_f^{(n)}$, falls sie an jeder Zelle von $T_f^{(n)}$ alle Belegungen erzeugen kann. Die $IDDQ$-Testtechnik testet Fehler durch Messung von Leckstrom und betrachtet nicht zusätzlich die Propagierung der fehlerhaften Signale [MaSu82, HSFH87]. Für diese Testtechnik ist eine Testmustermenge vollständig, falls diese Testmustermenge eine vollständige Belegung ist. Wir definieren die Belegungskomplexität von $T_f^{(n)}$ als die Größe seiner kleinsten vollständigen Belegung.

Im zweiten Kapitel untersuchen wir die Belegungskomplexität von $T_f^{(n)}$. Hier ist f eine Funktion von $\{0, 1, ..., m-1\}^k$ nach $\{0, 1, ..., m-1\}$ und $0, 1, m-1$ werden als Symbole betrachtet. Dabei ist es uns gelungen, die Belegungskomplexität von uniformen Bäumen vollständig zu charakterisieren. Es wird gezeigt, daß die Belegungskomplexität eines balancierten uniformen Baums entweder $\Theta(1)$ oder $\Theta((\lg n)^\alpha)$ ($\alpha \in (0, 1]$) ist. Falls der uniforme Baum über einer kommutativen Funktion definiert ist, kann α nur 1 sein.

Ist eine kommutative Funktion f gegeben, so kann man nach der Definition von f ein Integer Programming IP_f und einen gerichteten Graph G_f definieren. Im zweiten

Kapitel wird bewiesen, daß die folgenden Behauptungen äquivalent sind:

1. Die Belegungskomplexität von $T_f^{(n)}$ ist $\Theta(1)$.
2. IP_f hat eine zulässige Lösung.
3. G_f ist stark zusamenhängend.

Außerdem wird auch gezeigt, daß die Belegungkomplexität von $T_f^{(n)}$ in Zeit $\Theta(m^2)$ entscheidbar ist. Hierin ist f eine kommutative Funktion von $\{0, 1, ..., m-1\}^k$ nach $\{0, 1, ..., m-1\}$.

Für eine andere übliche Testtechnik muß die Propagierung der fehlerhaften Signale mit größter Sorgfalt behandelt werden. Im dritten Kapitel betrachten wir die Testkomplexität von $T_f^{(n)}$ unter der Annahme, daß f eine Funktion von $\{0, 1, ..., m-1\}^2$ nach $\{0, 1, ..., m-1\}$ ist. Wir zeigen, daß $T_f^{(n)}$ entweder $\Theta(1)$ oder $\Omega((\lg n)^\beta)(\beta > 0)$ testbar ist. Als Testkomplexität eines balancierten uniformen Baums, der über einer kommutativen Funktion definiert ist, ist genau einer der folgenden drei Fälle möglich: $\Theta(1)$, $\Theta(\lg n)$ und $\Omega(n^\gamma)$ ($\gamma \in (0, 1]$).

Falls die Basisfunktion f von $\{0, 1\}^k$ nach $\{0, 1\}$ geht, kann die Testkomplexität von $T_f^{(n)}$ exakt bestimmt werden. Das vierte Kapitel beinhaltet die folgenden Resultate:

- $T_f^{(n)}$ ist entweder $\Theta(1)$ oder $\Omega(\lg n)$ testbar.
- Die Testkomplexität von $T_f^{(n)}$ für eine kommutative Funktion ist entweder $\Theta(1)$ oder $\Theta(\lg n)$ oder $\Omega(n^\gamma)$ ($0 < \gamma \leq 1$).
- $T_f^{(n)}$ über monotonen Funktionen ist immer $\Omega(n^\gamma)$ ($0 < \gamma \leq 1$) testbar.

Des weiteren geben wir im vierten Kapitel Kriterien für die Zugehörigkeit zu den oben genannten Klassen an.

Im fünften Kapitel zeigen wir, daß jeder Baum durch eine Modifikation seiner Basiszellen so umgewandelt werden kann, so daß er in Zeit $O(\lg n)$ testbar ist. Der alte Baum wird durch den neuen Baum simuliert. Außerdem stellen wir ein Verfahren zur Synthese der $O(\lg n)$ testbaren Bäume vor.

Falls ein Schaltkreis mehrere primäre Ausgänge hat und jeder Ausgang nur von einigen der primären Eingänge abhängt, kann der Schaltkreis durch erschöpfendes Testen aller Teilschaltungen getestet werden. Ein solches Verfahren nennt man pseudoerschöpfender Test. Das sechste Kapitel präsentiert einen effizienten Algorithmus zur Erzeugung pseudoerschöpfender Testmustermengen. Dieser Algorithmus findet auch Anwendungen auf den Gebieten des Systementwurfs und der Fehlertoleranz.

Für große Schaltkreise ist die optimale pseudoerschöpfende Testmustermenge sehr schwierig zu berechnen. Probabilistische Verfahren können hier zur Senkung der Kosten beitragen [Wund87, Hart93]. Im siebten Kapitel stellen wir ein neues Konzept vor, sogenannte "Monomial Oriented Pseudorandom Tests". Die Grundidee besteht im Entwurf eines Testmustergenerators, mit dem man eine kleine Testmustermenge erzeugt, die alle kleinen Monome überdeckt. Ein solcher pseudozufälliger Testgenerator hängt nicht von einem konkreten Schaltkreis ab, daher ist seine Anwendung nicht auf konkrete Schaltkreise begrenzt.

Preface

Motivation

The wide use of computers in various fields of society makes it clear — computers must be more and more reliable. The reliability of a computer depends strongly upon testing its basic components — VLSI systems. Through test one knows whether the VLSI systems have been manufactured properly and behave correctly.

Generally speaking, A VLSI system is made up of the sequential circuit part and combinational circuit part. The test generation for sequential circuits is usually much more difficult than that for combinational ones, since the controllability and observability of sequential circuits are poor. In order to overcome the difficulty, some design technique have been developed. By using those techniques a sequential logic can be so designed that its test can be reduced to that for some combinational logics. Hence the key to the test of a VLSI system lies in the test of its combinational logic part.

This thesis focuses on the test problem of the combinational part of VLSI systems.

The test of a VLSI system includes mainly the generation of a test set and the application of the test set to the system. The test complexity can be classified into the complexity of the test set generation and the complexity of the test set application. The former can be estimated by the computing complexity of generating the test set. The latter is measured by the cardinality of the test set.

The test generation approaches can be divided roughly into structural and functional methods. A structural method generates test patterns for a circuit with reference to the concrete logic structure of the circuit, while a functional method produces test patterns for a circuit without reference to the concrete logic structure of the circuit.

With the rapid development of VLSI technology the circuit density is increasing dramatically. The test of VLSI systems is becoming increasingly difficult and expensive. Although some techniques such as design for testability, new fault models and new test generation approaches have been proposed to moderate these problems, there is a great need to develop new design methodologies and test approaches.

The complexity of test generation and application of a VLSI system is related to the concrete structure of the system. Theory and practical experiences show that it seems to be impossible to find a universal method for treating various VLSI systems efficiently. One of the alternatives is to develop a suitable method for a kind of VLSI systems.

This thesis studies extensively the test problems related to tree systems, pseudoexhaustive and pseudorandom testable circuit systems. It develops several techniques for generating optimal test sets for different kinds of circuits.

The Structure of the Thesis

This thesis consists mainly of seven chapters. In chapter 1, we give a formal definition of the VLSI systems by using X-category for simplifying our discussion and generalizing the results easily, and make a brief view of the functional and structural test generation approaches so that we can have an impression on the advantages and disadvantages of the two approaches.

Tree systems are basic components for many VLSI systems, especially for systems performing parallel and fast computations. Many combinational circuits can be covered by a number of tree like circuits. Therefore, the study of the test complexity of tree structure systems is very useful to the design, optimization and test of VLSI systems.

Let $T_f^{(n)}$ denote a balanced uniform tree based on function f and having n primary input lines. The test complexity of $T_f^{(n)}$ is defined as the cardinality of the minimum complete test set of it and is measured as a function of the number of the primary input lines in the tree. The test complexity of uniform tree systems based on functions over monoids has been intensively studied and divided into $\Theta(1)$, $\Theta(\lg n)$ and $\Theta(n)$ testable classes [BeSp91]. It indicates that the test complexity of a tree system can jump from one class to another, when the definition of its basic cell is modified. It motivates us to analyze the test complexity of more general tree systems and explore the possibility of modifying them to change their test complexity from a high class to a low one.

The result in [BeSp91] is obtained under the assumption that the function implemented by the basic cell satisfies the associative law. After this condition is dropped, the scene really changes. For example, the boolean function NAND does not fulfill the associative low. Hayes [Haye71] shows that a tree system based on NAND gates is $\Theta(\sqrt{n})$ testable. We analyze the assignment and test complexity of more general tree systems and develop a method to synthesize tree systems for the low test complexity.

A complete assignment set to $T_f^{(n)}$ consists of a number of n-component patterns. By applying it to the primary input lines of $T_f^{(n)}$, every internal f cell in $T_f^{(n)}$ can be excited by all possible input combinations. The assignment complexity of $T_f^{(n)}$ is defined as the cardinal number of the minimum complete assignment set to it. In chapter 2 we deal with the assignment complexity of tree systems.

A test pattern for a fault in a faulty cell has to fulfill two conditions: 1) applying a right assignment to the faulty cell for sensitizing the fault, 2) making a channel to propagate the generated diagnosis signal(effect of the fault) to a primary output line for observing it. The $IDDQ$ testing method tests faults by measuring the leakage current. In $IDDQ$ testing, the site of fault has to be excited, and the propagation of the effect of the fault is automatic [MaSu82, HSFH87]. For $IDDQ$ testing, a complete assignment set to a tree is just a complete test set to it. It is appropriate to consider the assignment problem in the first stage since the assignment itself is a basic problem in the VLSI system design and test, and the construction of a complete assignment set is the first step towards the generation of a complete test set for other testing methods. We show that a tree system is either $\Theta(1)$ or $\Omega((\lg n)^\alpha)$ ($\alpha \in (0,1]$) assignable. When a uniform tree system is based on a commutative function, then it is either $\Theta(1)$ or $\Omega(\lg n)$ assignable.

Having explored the assignment complexity, we begin to analyze the test complexity of tree systems in chapter 3. We show that a balanced uniform tree system is either $O(1)$ or $\Omega((\lg n)^\alpha)$ ($\alpha \in (0,1]$) testable. Furthermore, we prove that the test complexity of balanced uniform tree systems based on the commutative functions can be exactly divided into $\Theta(1)$, $\Theta(\lg n)$ and $\Omega(n^\alpha)$ ($\alpha \in (0,1]$) classes.

Every balanced tree system is $O(\lg n)$ assignable. In other words, all faults can be sensitized through $O(\lg n)$ patterns simultaneously. Whether a balanced tree system is $O(\lg n)$ testable dependes on the diagnosis signal propagatability.

Chapter 2 and 3 are dedicated to tree systems based on symbolic functions. The results obtained there are more or less abstract. In the fourth chapter we investigate the test complexity of uniform tree circuits based on boolean functions, and show that a balanced uniform tree circuit is either $\Theta(1)$ or $\Omega(\lg n)$ testable. A balanced uniform tree circuit based on a boolean function $f : \{0,1\}^k \longrightarrow \{0,1\}$ is $\Theta(1)$ testable if and only if for every pair $X, Y \in \{0,1\}^k$ $f(X) \neq f(Y)$ if the Hamming distance between X and Y is 1. The test complexity of balanced uniform tree circuits based on commutative functions can be further divided into constant, logarithmic and polynomial classes, and balanced uniform tree circuits based on unate functions are all $\Omega(n^\alpha)$ ($\alpha \in (0,1]$) testable. The test complexity of uniform tree circuits based on general functions has more classes. These results are helpful for us to understand the test complexity structure and give us some hints for designing or modifying VLSI systems for testability.

Through the classification of the test complexity, we have found that if a uniform tree system is $O(\lg n)$ testable, there must be a constant κ so that one can simultaneously propagate a diagnosis signal from each of the lines in the same level to the primary output line by using κ patterns. In chapter 5 we propose a method of the function synthesis. Given a balanced tree T, we can always synthesize an $O(\lg n)$ testable tree $\mathcal{T}$ and embed T in $\mathcal{T}$ to trade the hardware overhead for the low test complexity. This idea is meaningful, since the cost of the hardware has been decreasing while the cost of the test has been increasing. In comparison with other methods of reducing the test complexity, this method requires more extra gates and less input and output pins. With the development of the VLSI technology, the gate density of VLSI system is increasing much more rapidly than the number of access terminals. Thus this method is attractive.

One of the test approaches independent of the functions implemented by the circuits is the exhaustive testing. Given a circuit with n primary input lines, the exhaustive testing generates all the 2^n patterns. Such a test set can detect all detectable combinational faults in the circuit. The advantage of this method is that no information about the circuit structure is required, and the test generator can be cheaply realized by using hardware. Its disadvantage is that the test sets for circuits with many primary input lines are so large that they can not be used in practice.

Assume that the given circuit has a number of primary output lines and n primary input lines, and none of these primary output lines depends on all the n primary input lines. We can imagine that the circuit can be covered through a number of subcircuits, and each of these subcircuits has at most $k (k < n)$ primary input lines. The approach that tests the circuit by testing these subcircuits exhaustively is called pseudoexhaustive test. The first method of pseudoexhaustive test generation was proposed in [BSMS81]. Thereafter, extensive works have been done to develop good mechanisms for generating practically acceptable pseudoexhaustive test sets.

In the sixth chapter we present an efficient algorithm for constructing pseudoexhaustive test sets. This algorithm has also applications to the design of threshold circuits and fault tolerant systems. By using this algorithm one can generate an acceptable test set for small k and practical n ($k \leq 10, n \leq 1024$).

Generally, the pseudoexhaustive test sets constructed by available algorithms are often too large to be used, and their cardinalities are much larger than the upper bounds of the corresponding optimal pseudoexhaustive test sets. One of the alternatives is the pseudorandom test. It has been shown that a fairly small pseudorandom test set can reach a high probability of the pseudoexhaustive test.

The main reason for using the pseudorandom test is that one can avoid the long and complex algorithmic test generation procedure. The pseudorandom techniques have two important applications. One is to generate a short random test preceding the long and laborious deterministic test to catch easy detect fault, another is to design built-in self test circuits. However, the pseudorandom test can not always guarantee the very high fault coverage. In order to improve the quality of pseudorandom test, a number of techniques have been proposed. The input signal biased random test and pattern biased random test are typical examples [Hart91, Hart93, SLC71, Wund87]. Their common idea is to design a special pseudorandom test generator for a given circuit by using the information about the given circuit fully. A so designed pseudorandom test generator is related to the structure of the given circuit, and its application is limited. Furthermore, the desired information for designing a properly weighted random test generator is not always available.

The seventh chapter proposes a new concept – Monomial Oriented Pseudorandom Test. Its key idea is to design a monomial oriented pseudorandom test generator that allows a fairly small test set to cover all small monomials. A monomial oriented pseudorandom test generator is not related to a concrete circuit structure, then its application is not limited to a concrete circuit. In chapter seven, we give a theoretical analysis of the soundness of such kind of pseudorandom test generators, and present some experimental results to demonstrate their advantages as well.

Acknowledgments

I wish to express my heartfelt thanks to my supervisor Prof. Günter Hotz. He gave me this interesting theme and a lot of concrete instructions. Without his patient help, this work would never have been finished. He provided me also the opportunity to learn German.

My thanks go to Uwe Sparmann who helped me a lot during several year's cooperation. Tanks also go to Armin Reichert and Björn Schieffer who read the manuscript carefully and gave detailed comments.

I am grateful to many German and Chinese colleagues for their advice and encouragement. I am indebted to Michael Biwersi, Thomas Busch, Wolfgang Collet, Thomas Fettig, Joachim Hartmann, Mattias Krallmann, Gisela Pitsch, Elmar Schömer, and Juergen Sellen for their help.

Here perhaps is an appropriate opportunity to express my deep gratitude to Prof. Wei Daozhen and Prof. Zhang Huangou who are the first two teachers in my *academic kindergarten.*

Chapter 1

VLSI Systems and Tests

This Chapter consists of four sections. In section 1.1 we give a formal description of the VLSI systems by using X-category theory developed in [Hotz65], so that we can simplify our discussion and generalize the results easily. Section 1.2 is about the fault model and test. In section 1.3 we discuss the problem of the functional test. Section 1.4 presents a brief view of the structural test generation for the regular VLSI systems.

1.1 X-Category and VLSI Systems

A *semigroup* $(S, \Box)$ is a set S and together with an associative binary operation $\Box : S^2 \longrightarrow S$.

A *monoid* $(M, \Box, u)$ is a semigroup $(M, \Box)$ with an element $u \in M$ such that $u \Box x = x \Box u = x$ for all $x \in M$. Such an element is called unit of M.

Given a set E, define

$$E^* = \{a_1...a_k \mid a_i \in E, \quad \text{for } i = 1, ..., k, \quad k \in \mathrm{N}_0\}.$$

E^* includes all of the finite strings over E. The length of the string $s = a_1...a_k$ $(a_i \in E)$ is k. In case $k = 0$, s is an empty string denoted by the symbol λ.

H is called a *free monoid*, when there is system of generators E of H such that the canonical homomorphism $\phi : E^* \longrightarrow H$ is isomorphism [Hotz90].

Let $T = \{t_1, t_2, ..., t_m\}$ be the set of the basic types of signals which can be transferred through a (symbolic) line or (symbolic) bus in VLSI systems. A signal type $t_i \in T$ consists of a number of signals called individuals(values). For instance, the 1-bit binary signal type includes logic 0 and logic 1 as its individuals, and a line of such a signal type can transfer both logic 0 and logic 1.

Let $\cdot$ be the concatenation operation of elements in T^*, and

$$t_1 \dots t_k \cdot t_{k+1} \dots t_n = t_1 \dots t_n$$

and

$$t_1 \dots t_k \cdot \lambda = \lambda \cdot t_1 \dots t_k = t_1 \dots t_k.$$

for $t_i \in T^*$. Then $(T^*, \cdot)$ is a free monoid having the empty element λ as its unit. In this Chapter, we use I_{t_i} to denote the set of all individuals of the signal type t_i and regard I_{uv} as $I_u \times I_v$ for $u, v \in T^*$.

Let $\mathcal{A}$ be the set of all building blocks of the VLSI systems. Every line in an element $F \in \mathcal{A}$ has a signal type. Two functions

$$Q, Z : \mathcal{A} \longrightarrow T^*$$

are used to determine the input and output types of building blocks in $\mathcal{A}$. For instance, the input type and output type of $F \in \mathcal{A}$ are $Q(F)$ and $Z(F)$, respectively.

Given two building blocks $F, G \in \mathcal{A}$, we can construct a new building block by using the parallel operation " $\times$ ", and the new building block is illustrated by Fig. 1.1. In case the output type $Z(G)$ of G is equal to the input type $Q(F)$ of F, we can construct a new building block by using the sequential operation " $\circ$ ". Fig. 1.2 shows the new building block which is constructed by linking the ith output line of G directly to the ith input line of F.

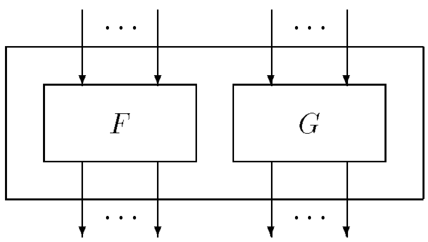


Fig. 1.1: $F \times G$

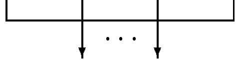


Fig. 1.2: $F \circ G$

For $F, G \in \mathcal{A}$

$$\begin{aligned} Q(F \times G) &= Q(F)Q(G), \qquad Z(F \times G) = Z(F)Z(G) \\ Q(F \circ G) &= Q(G), \qquad Z(F \circ G) = Z(F) \end{aligned}$$

Let $\mathbf{B}_T = \{B_1, B_2, ...\}$ and $\mathbf{D}_T = \{L_a, D_a, V_{ab} \mid a, b \in T\}$ be two classes of basic building blocks. Every basic building block B_i in $\mathbf{B}_T$ has only one output line and at least one line. The input and output types of B_i are $Q(B_i)$ and $Z(B_i)$, and the input and output values of B_i are limited to $I_{Q(B_i)}$ and $I_{Z(B_i)}$, respectively. The elements in $\mathbf{D}_T$ can be illustrated through the following figures.

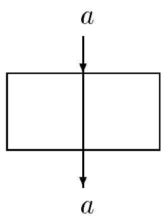


Fig. 1.3: L_a

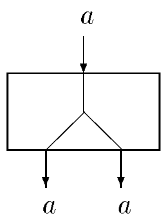


Fig. 1.4: D_a

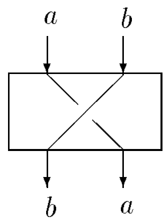


Fig. 1.5: V_{ab}

L_a denotes a line, D_a a fanout mechanism, and V_{ab} two cross lines. For the three building blocks,

$$\begin{aligned} Q(L_a) &= a, & Z(L_a) &= a \\ Q(D_a) &= a, & Z(D_a) &= aa \\ Q(V_{ab}) &= ab, & Z(V_{ab}) &= ba \end{aligned}$$

Let $\mathbf{A} = \mathbf{B}_T \cup \mathbf{D}_T$. $\mathcal{A}$. The set of all building blocks of VLSI systems can be formally defined as follows.

1. $\mathcal{A} \supset \mathbf{A}$;
2. $F, G \in \mathcal{A} \Longrightarrow F \times G \in \mathcal{A}$;
3. $F, G \in \mathcal{A}$ and $Z(G) = Q(F) \Longrightarrow F \circ G \in \mathcal{A}$.

It has been shown that

$$\mathbf{C} = (T^*, \mathcal{A}, Q, Z, \circ)$$

is a category, and

$$\mathcal{C} = (T^*, \mathcal{A}, Q, Z, \circ, \times) \tag{1.1}$$

is an X-category [Hotz65]. In the following we present the formal definitions of *category* and *X-category.*

Definition 1.1 (category) $\mathbf{C} = (\mathbf{O}(\mathbf{C}), \mathbf{M}(\mathbf{C}), Q, Z, \circ)$ *consisting of a set* $\mathbf{O}(\mathbf{C})$ *of objects, a set* $\mathbf{M}(\mathbf{C})$ *of morphisms, two mappings*

$$Q, Z : \mathbf{M}(\mathbf{C}) \rightarrow \mathbf{O}(\mathbf{C})$$

and a mapping

$$\circ : P_C \rightarrow \mathbf{M}(\mathbf{C}), \qquad P_C = \{(f,g) \mid Q(f) = Z(g) \ \wedge \ f, g, \in \mathbf{M}(\mathbf{C})\}$$

is called category, provided that

1. $\forall (f,g) \in P_C \, \{Q(f \circ g) = Q(g) \ \wedge \ Z(f \circ g) = Z(f)\}$;
2. *For all* $f, g, h \in \mathbf{M}(\mathbf{C})$, $f \circ (g \circ h) = (f \circ g) \circ h$ *if* $f \circ (g \circ h)$ *and* $(f \circ g) \circ h$ *are defined;*
3. *For every* $u \in \mathbf{O}(\mathbf{C})$ *there is an identity* $\mathbf{1}_u$ *such that* $f \circ \mathbf{1}_u = f$ *and* $\mathbf{1}_u \circ g = g$ *for all* $f, g \in \mathbf{M}(\mathbf{C})$ *with* $Q(f) = u$ *and* $Z(g) = u$.

Definition 1.2 (X-category) *$\mathcal{C} = (\mathbf{O}(\mathbf{C}), \mathbf{M}(\mathbf{C}), Q, Z, \circ, \times)$ is called X-category, provided that the following five conditions are satisfied.*

1. *$(\mathbf{O}(\mathbf{C}), \mathbf{M}(\mathbf{C}), Q, Z, \circ)$ is a category;*
2. *$(\mathbf{O}(\mathbf{C}), \times)$ and $(\mathbf{M}(\mathbf{C}), \times)$ are monoids;*
3. *$Q, Z : (\mathbf{M}(\mathbf{C}), \times) \to (\mathbf{O}(\mathbf{C}), \times)$ are two monoid homomorphisms;*
4. *$\forall u, v \in \mathbf{O}(\mathbf{C}) \{\mathbf{1}_{uv} = \mathbf{1}_u \times \mathbf{1}_v\}$;*
5. *$\forall (g_1, f_1), (g_2, f_2) \in P_C \{(g_1 \circ f_1) \times (g_2 \circ f_2) = (g_1 \times g_2) \circ (f_1 \times f_2)\}$.*

Suppose t is the 1-bit binary signal type, and $T = \{t\}$. If $\mathbf{D}_T = \{L_t, D_t, V_{tt}\}$ and $\mathbf{B}_T$ includes NOT, AND and OR gates, then the X-category $\mathcal{C}$ defined by (1.1) is corresponding to the whole combinational circuit system. It is corresponding to the whole tree system if $\mathbf{D}_T$ does not include the fanout mechanism D_t.

Assume that a building block $F \in \mathcal{A}$ implements a function f, and its domain and codomain are denoted by $Q'(f)$ and $Z'(f)$. $(Q'(f) = I_{Q(F)}, Z'(f) = I_{Z(F)})$. Let $\mathcal{T} = \{I_u | u \in T^*\}$. The set

$$\mathcal{F} = \{f : Q'(f) \to Z'(f) \mid Q'(f), Z'(f) \in \mathcal{T}^*\}$$

includes all the functions implemented by the elements of $\mathcal{A}$.

We define two operations $\odot$ and $\otimes$ over $\mathcal{F}$. Given two functions f and g, $f \otimes g$ is a function from $Q'(f)Q'(g)$ to $Z'(f)Z'(g)$. In case $Q'(f) = Z'(g)$, $f \odot g$ is defined as a function from $Q'(g)$ to $Z'(f)$. It can be shown that

$$\mathbf{K} = (\mathcal{T}^*, \mathcal{F}, Q', Z', \odot)$$

is a category, and

$$\mathcal{K} = (\mathcal{T}^*, \mathcal{F}, Q', Z', \odot, \otimes) \tag{1.2}$$

is an X-category. In the following we investigate the relationship between the two X-categories $\mathcal{C}$ and $\mathcal{K}$ by using *functor.*

Definition 1.3 (functor) *Given two categories*

$$\mathbf{C} = (\mathbf{O}(\mathbf{C}), \mathbf{M}(\mathbf{C}), Q, Z, \circ) \quad \textit{and} \quad \mathbf{K} = (\mathbf{O}(\mathbf{K}), \mathbf{M}(\mathbf{K}), Q', Z', \odot)$$

and two mappings

$$\phi_1 : \mathbf{O}(\mathbf{C}) \to \mathbf{O}(\mathbf{K}) \quad \textit{and} \quad \phi_2 : \mathbf{M}(\mathbf{C}) \to \mathbf{M}(\mathbf{K}),$$

$\phi = (\phi_1, \phi_2)$ is called a functor, provided that the following three conditions are satisfied.

1. *$\forall F \in \mathbf{M}(\mathbf{C}) \{Q'(\phi_2(F)) = \phi_1(Q(F)) \ \wedge \ Z'(\phi_2(F)) = \phi_1(Z(F))\}$;*
2. *$\forall F, G \in \mathbf{M}(\mathbf{C}) \{Q(F) = Z(G) \Longrightarrow \phi_2(F \circ G) = \phi_2(F) \odot \phi_2(G)\}$;*

3. $\forall u \in \mathbf{O}(\mathbf{C}) \left\{ \phi_2(\mathbf{1}_u) = \mathbf{1}_{\phi_1(u)} \right\}$.

We define two mappings

$$\phi_1: \quad T^* \to \mathcal{T}^*, \qquad \phi_1(t) = \ I_t, \qquad t \in T^*$$

and

$$\phi_2: \quad \mathcal{A} \to \mathcal{F}, \qquad \phi_2(F) = f \ \text{ if } \ f : I_{Q(F)} \to I_{Z(F)}, \ \ F \in \mathcal{A}.$$

It is easy to check that:

1. $\forall F \in \mathcal{A} \{ Q'(\phi_2(F)) = \phi_1(Q(F)) \ \wedge \ Z'(\phi_2(F)) = \phi_1(Z(F)) \}$;
2. $\forall F, G \in \mathcal{A} \{ Q(F) = Z(G) \Longrightarrow \phi_2(F \circ G) = \phi_2(F) \odot \phi_2(G) \}$;
3. $\forall u \in T^* \left\{ \phi_2(\mathbf{1}_u) = \mathbf{1}_{\phi_1(u)} \right\}$.

Thus $\phi = (\phi_1, \phi_2)$ is a functor from $\mathbf{C}$ to $\mathbf{K}$, and a functor from $\mathcal{C}$ to $\mathcal{K}$ as well since

$$\forall F, G \in \mathcal{A} \{ \phi_2(F \times G) = \phi_2(F) \otimes \phi_2(G) \}.$$

Thereafter, we use operators $\circ$ and $\times$ to replace $\odot$ and $\otimes$, and substitute mappings Q and Z for Q' and Z', provided that no confusion can be caused. For the sake of simplicity, we often call a basic building block *cell* and use a lower case letter to represent a function implemented by a building block represented by the corresponding upper case letter. For example, b is used to represent the function implemented by $B \in \mathcal{A}$. L_t is used to denote a line transferring signals of type t and the function $\phi_2(L_t)$ as well. Furthermore, we use u to represent $\phi_1(u)$ ($u \in T^*$), namely the set of values of type u. For instance, we use the form $f : t^k \longrightarrow t$ to represent a function from $\phi_1(t^k)$ to $\phi_1(t)$.

Every building block $F \in \mathcal{A}$ implements a function $\phi_2(F) : Q(F) \to Z(F)$. For example, L_a realizes an identical function for a, D_a a function from a to $a \times a$, and V_{ab} a function from $a \times b$ to $b \times a$.

Suppose x and y are two individuals of type a and b, respectively, then

$$\phi_2(L_a)(x) = x, \quad \phi_2(D_a)(x) = xx \quad \text{and} \quad \phi_2(V_{ab})(xy) = yx.$$

1.2 Fault Models and Test

Most of the literature on VLSI system tests uses the concepts defined by R. D. Eldred in [Eldr59], and formalized by J.P. Roth in [Roth66], in which only stuck-at-1 and stuck-at-0 faults are considered. Although the stuck-at fault model can model a lot of the faults occurring actually in a system, there still exist many faults that can not be modeled by it. In this thesis, we consider two fault models. The first is the *single line individual fault* model, and the second is the *single cell definition fault* model.

1.2.1 Single line individual fault model

For two distinct individuals $x, y \in t$, $L_{x \to y}$ denotes a cell realizing the following function.

$$\begin{aligned} \phi_2(L_{x\to y}) \; : \quad & t \longrightarrow t, && t \in T \\ \phi_2(L_{x\to y})(z) \quad = \; & \begin{cases} y & : \; z = x \\ z & : \; z \neq x \end{cases} && \text{for } x, y, z \in t \end{aligned} \tag{1.3}$$

In case t consists of 1, 2, 3 and 4, x and y are 1 and 2, then Table 1.1 shows just the definition of $\phi_2(L_{1\to 2})$.

z	1	2	3	4
$\phi_2(L_{1\to 2})(z)$	2	2	3	4

Table 1.1

A single line individual fault in a building block changes a line L_t in the building block into a cell $L_{x\to y}$ $(x, y \in t)$. The individual $x \in t$ is a test for this fault since

$$\phi_2(L_t)(x) = x \neq y = \phi_2(L_{x\to y})(x).$$

A line is called *fanout-stem* if it has fanout branches. Otherwise, it is called *input-line*. Under the line individual fault model, we are required to consider only the faults on the input-lines and fanout-stems.

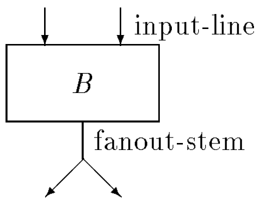


Fig. 1.6: Fanout-stem and Input-line

Like the popular stuck-at fault model, the line individual fault model has also some shortages and can not model all the faults in a cell. In some cases, one knows the function definition of a cell, but he has no further information about the internal structure of the cell. It has been shown that a given function can have different realizations which might have different minimal complete test sets. The cell definition fault model defined in next subsection can avoid this problem to some extent.

1.2.2 Single cell definition fault model

Assume $F \in \mathcal{A}$. The single cell definition fault model assumes that a basic cell B in F implements a function $\phi_2(B') : Q(B') \to Z(B')$ instead of the desired function $\phi_2(B) : Q(B) \to Z(B)$ due to a fault. However, $Q(B) = Q(B')$ and $Z(B) = Z(B')$, and there is an $x \in Q(B)$ such that $\phi_2(B)(x) \neq \phi_2(B')(x)$. The element $x \in Q(B)$ is a test for this cell definition fault. To test cell B completely, every element in $Q(B)$ has to be applied to it. A complete test set of F consists of a number of patterns from $Q(F)$, and by applying them to F every basic cell in F can be tested exhaustively.

We say that a fault u dominates another fault v, when every test for u is also a test for v.

Assume that cell B is desired to realize a function $b : \underbrace{t \times \cdots \times t}_{k} \to t$. Then we can consider that B should realize a function $L_t \circ b \circ (\underbrace{L_t \times L_t \times \cdots \times L_t}_{k})$, where L_t represents an input or output line linked to the cell B. Suppose there is a line individual fault at the first input line that changes the line L_t into a cell $L_{x \to y}$. Then it causes the cell B to implement a function $L_t \circ b \circ (L_{x \to y} \times \underbrace{L_t \times \cdots \times L_t}_{k-1})$. Thus a line individual fault on the lines of B can be considered as a cell definition fault in the cell B. A complete test set for the cell definition faults in B is certainly a complete test set for the line individual faults on the input and output lines of B. This indicates that every single line individual fault on the input and output lines of a cell is dominated by a single cell definition faults of the cell. All fanout-stem faults and input-line faults in $F \in \mathcal{A}$ are dominated by cell definition faults in F. This fault model is suitable to the VLSI systems with a regular structure.

The signal type of the input and output lines of NOT, AND and OR gates is $\{0, 1\}$. If only these gates are considered to be the basic cells of $\mathbf{B}_T$, the line individual faults are called stuck-at faults conventionally, and the single stuck-at fault model is often adopted. It is assumed that every cell definition fault in NOT, OR and AND gates can be dominated by stuck-at faults.

Given an $F \in \mathcal{A}$, we use F_f to denote the set of cells which are induced by a single fault of any basic cell in F.

Definition 1.4 (complete test set) *$D(F, F_f) \subset Q(F)$ is a complete test set of $F \in \mathcal{A}$ if and only if*

$$\forall F' \in F_f \exists x \in D(F, F_f) \{\phi_2(F)(x) \neq \phi_2(F')(x)\} \tag{1.4}$$

$Q(F)$ is the set of all input patterns of F, and it includes always a complete test set for the irredundant $F \in \mathcal{A}$. $Q(F)$ is a complete test set of F. However, it can not be used when $\#Q(F)$ is too large. One has to choose a subset of $Q(F)$ as the test set. The problem is how to generate an acceptable subset, which is a complete test set of F. We are also interested in the construction of the minimal complete test set for F.

1.3 Functional Test

Given a fault model and an $F \in \mathcal{A}$, a complete test set of F regarding the given fault model consists of a number of patterns from $Q(F)$. By applying them to F every concerned fault can be tested.

Assume that we know nothing about the concrete structure of F but the function expression of $\phi_2(F)$. Then the functional test has to be done. Among the approaches to functional test are the pseudoexhaustive, random, and universal tests. In this section we consider the generation of a universal test set for every irredundant realization of $\phi_2(F)$.

If $\phi_2(F)$ has a special property, a universal test set which is a complete test set for any of a variety of different irredundant realizations of $\phi_2(F)$ may be found. Akers [Aker73] examines the problem of finding the universal test set, and shows that, for AND/OR networks, universal test sets may be found, and the universal test sets detect not only all signal faults but also all multiple faults.

Assume $\phi_2(F)$ to be a more general function belonging to $\mathcal{F}$. In this section we explore the possibility of and the difficulty in the generation of a universal test set for $\phi_2(F)$.

Suppose F_0 and F_1 are two different realizations of $\phi_2(F)$, and F_1 can be transformed into F_0 by obeying some transformation regulations. We derive some fault transformation rules from the structure transformation regulations. According to these rules we can transform the faults associated with F_1 into some faults associated with F_0 so that a complete test set concerning the faults in F_0 is also a complete test set for F_1.

Before further discussion we construct two X-categories $\mathcal{B}$ and $\mathcal{D}$. Their morphism sets are $\mathcal{B}_T$ and $\mathcal{D}_T$ defined below.

1. The definition of $\mathcal{B}_T$:
   - $\mathcal{B}_T \supset \mathbf{B}_T$;
   - $F \times G \in \mathcal{B}_T$ if $F,\ G \in \mathcal{B}_T$;
   - $F \circ G \in \mathcal{B}_T$ if $F,\ G \in \mathcal{B}_T$ and $Z(G) = Q(F)$.
2. The definition of $\mathcal{D}_T$:
   - $\mathcal{D}_T \supset \mathbf{D}_T$;
   - $F \times G \in \mathcal{D}_T$ if $F,\ G \in \mathcal{D}_T$;
   - $F \circ G \in \mathcal{D}_T$ if $F,\ G \in \mathcal{D}_T$ and $Z(G) = Q(F)$.

It is easy to show that both

$$\mathcal{B} = (T^*, \mathcal{B}_T, Q, Z, \circ, \times) \quad \text{and} \quad \mathcal{D} = (T^*, \mathcal{D}_T, Q, Z, \circ, \times)$$

are X-categories. An element in $\mathcal{B}_T$ is called $\mathcal{B}$-tree, while an element in $\mathcal{D}_T$ is called $\mathcal{D}$-tree.

Assume that $G \in \mathcal{A}$, $Q(G) = \sigma = \underbrace{a \cdots a}_{k}$ and $Z(G) = a$. Then both building blocks $D_a \circ G$ and $(G \times G) \circ D_\sigma$ implement the same function. The former can be transformed into the latter. We call such a building block transformation a basic transformation. Fig. 1.7 illustrates the basic transformation.

Given a function $f \in \mathcal{F}$, there are various realizations of f. The following lemma, due to G. Hotz, states that every realization of f can be transformed into a standard realization which is made up of two trees.

Lemma 1.1 *For every $F \in \mathcal{A}$ there is an $F' = B' \circ D'$ with $B' \in \mathcal{B}_T$ and $D' \in \mathcal{D}_T$ such that $\phi_2(F) = \phi_2(F')$ holds for every functor $\phi = (\phi_1, \phi_2)$ from $\mathcal{C}$ to $\mathcal{K}$.*

The details of the proof of this lemma can be found in [Hotz74] .

Fig. 1.8 illustrates that for a given function f there are a number of realizations, and each of them can be transformed into a standard realization F_0 made up of a $\mathcal{D}$-tree and $\mathcal{B}$-tree.

The basic transformation is the replacement of the building block $H = D_a \circ G$ with $H' = (G \times G) \circ D_\sigma$ as illustrated by Fig. 1.7.

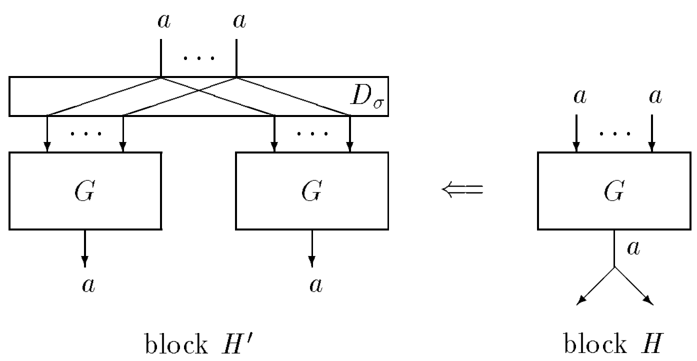


Fig. 1.7: Basic-Transformation

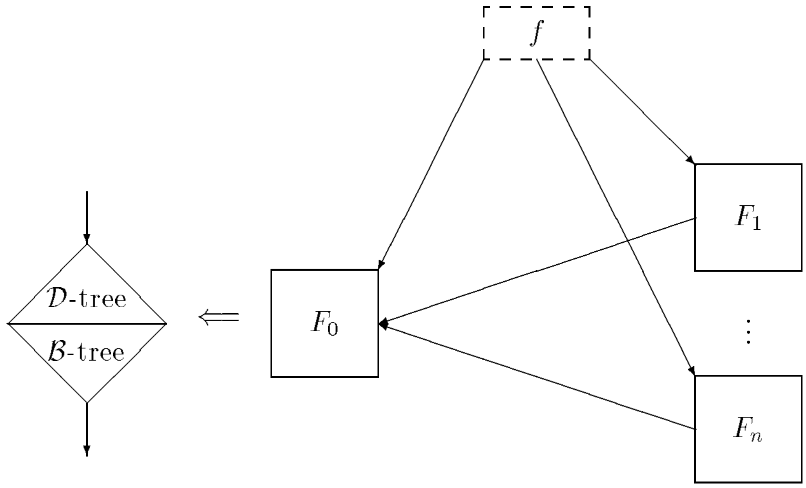


Fig. 1.8: Realization-Transformation

For cell definition fault model, cell faults dominate all line individual faults. Assume that an $F \in \mathcal{A}$ is transformed into F' through a basic transformation. Some single cell faults in F are transformed into multiple cell faults in F'. Furthermore, some single cell faults in F' have no equivalent fault in F. A complete test set of multiple cell faults for F_0 is a complete test set of single cell faults for each of F_1, ..., F_n.

Assume that F_1, ..., F_n can be transformed into F_0, which is a standard realization. Then we can state that a complete test set of multiple cell faults for $\mathcal{B}$-tree in F_0 is a complete test set of single cell faults for each of F_1, ..., F_n.

In case the line individual fault model is adopted, we assume that all concerned cell definition faults can be dominated by line individual faults. As mentioned in section 1.2.1, we are required to consider only faults on the fanout-stems and input-lines. Suppose F can be transformed into F' through a basic transformation demonstrated in Fig. 1.7, then a fanout-stem fault in F is transformed into a multiple input-line fault in F', and a single input-line fault in F is transformed into a fanout-stem fault in F'. Furthermore, some single input-line faults in F' have no equivalent fault in F. A complete test set of multiple line faults for F_0 is a complete test set of single line faults for each of F_1, ..., F_n.

Assume that F_1, ..., F_n can be transformed into a standard realization F_0. We can conclude that a complete test set of multiple line faults in F_0 is a complete test set of single line faults for each of F_1, ..., F_n. However, whether the complete test set of single line faults for all $F_1, ..., F_n$ is a complete test set of multiple line faults for F_0 is an open problem.

Observation 1.1 *The complete test set of the multiple variable faults for all of the boolean expressions of f is a complete test set of single-stuck-at faults for all of the realizations of the function f.*

Our argument for this observation is the following.

Assume that the basic cells used to realize f are NAND and NOR gates. Every realization $F_i \in \mathcal{A}$ implementing the given function f can be transformed into a standard realization F_0 consisting of a $\mathcal{D}$-tree and $\mathcal{B}$-tree. A complete test set for multiple line faults in F_0 is a complete test set for the single-stuck-at faults in F_i. Because every fault on the output line of a cell is dominated by some faults on some input lines, the multiple line faults in $\mathcal{D}$-tree of F_0 dominate the multiple line faults in the $\mathcal{B}$-tree. Then a complete test set for the multiple line faults in $\mathcal{D}$-tree of F_0 is a complete test set for the single-stuck-at faults in F_i. A $\mathcal{D}$-tree corresponds to a boolean expression of the function f. The complete test set for multiple line faults in $\mathcal{D}$-tree of F_0 corresponds to a complete test set for the multiple variable faults in the boolean expression. Thus, the complete test set of the multiple variable faults for all of the boolean expressions of the function f is a complete test set of the single-stuck-at faults for all of the realizations of the function f.

Assume that F_0 includes n cells, and $k-1$ possible faults can occur in every cell. The number of the distinct multiple cell faults in F_0 is equal to

$$\sum_{i=1}^{k} \binom{n}{i} (k-1)^i = k^n - 1.$$

Suppose a boolean expression of f includes n variables, then the number of the distinct multiple variable faults for the expression is equal to $3^n - 1$. The difficulty in generating a complete test set for multiple faults is very clear. Generally speaking, it is not realistic to construct the universal test set for a system with many primary input lines.

An interesting theoretical question is as follows: Given a boolean function $f(x_1, ..., x_n)$ in which both x_i and $\overline{x_i}$ appear ($i = 1, ..., n$), for any input combination $\vec{X} = (a_1, a_2, ..., a_n)$, $a_i = 0$ or 1, does there exist an irredundant circuit realization of f which requires the input $\vec{X}$ as a test pattern in order to detect all stuck-type faults. To date this conjecture has not been proven but no counterexample has been found[BrFr76]. If this conjecture is true, the universal test set for all the cells realizing such a function f has to include every combination $\vec{X} = (a_1, a_2, ..., a_n)$, $a_i = 0$ or 1.

1.4 Structural Test

The discussion in section 1.2 gives us an impression that it is quit difficult to generate a universal test set for a system with many primary input lines. If the system structure is regular and we know its structure information, the scene changes. In some cases, we can not only determine the test complexity of the system, but also generate the minimum complete test set for it. We show this through a brief discussion on the test problem of uniform trees.

Assume $T = \{t\}$, $\mathbf{B}_T = \{B\}$ and $\mathbf{D}_T = \{L_a, V_{ab} \mid a, b \in T\}$. Assume further that $Q(B) = \underbrace{t \cdots t}_{k}$ and $Z(B) = t$. Then $\mathcal{A}$ defined in section 1.1 is the set of all uniform trees based on the unique basic cell B. Before discussing the test problem deeply we study the assignment problem.

Definition 1.5 (assignment complexity) *Given an $S \subset Q(B)$ and an $F \in \mathcal{A}$. A complete assignment set $CA(F)$ for F regarding S is a subset of $Q(F)$. By applying all of the elements of $CA(F)$ to the primary input lines of F, every cell B in F can be excited by all of the elements in S. The assignment complexity of F regarding S is defined as the cardinal number of the minimum complete assignment set of F.*

The assignment complexity of F depends upon the property of $\phi_2(B)$, namely, b. In the following we will show that F is $O(1)$ assignable if b has certain property. In order to describe this property we require a new symbol D_a^i which is defined as follows:

1. $D_a^2 = D_a$, for $a \in T^*$;
2. $D_a^{i+1} = (D_a^i \times L_a) \circ D_a$, for $a \in T^*$.

The logical structure of D_a^{i+1} is illustrated by Fig. 1.9.

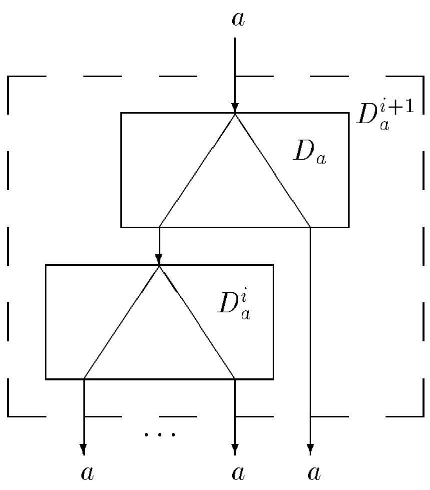


Fig. 1.9 D_a^{i+1}

In the rest of this Chapter we use σ to denote the string $\underbrace{t\cdots t}_{k}$, and T to stand for I_t.

Definition 1.6 (b-stable set) *Set $S \subset T^k$ is* b-stable *if and only if there are bijective mappings $\pi_1, ..., \pi_k : S \rightarrow S$ such that $(b \circ \pi_1 \times \cdots \times b \circ \pi_k) \circ D_\sigma^k(S) = S$.*

Lemma 1.2 *If $S \subset T^k$ is a b-stable set, then there are k bijective mappings $\pi_1, ..., \pi_k : S \rightarrow S$ such that*

$$(\underbrace{b \times \cdots \times b}_{k}) \circ (\pi_1 \times \cdots \times \pi_k) \circ D_\sigma^k \tag{1.5}$$

is the identical mapping of S.

Proof: Suppose $S \subset T^k$ is b-stable, and $(b \circ \pi_1 \times \cdots \times b \circ \pi_k) \circ D_\sigma^k$ is a bijective mapping from S to S. Define a bijective mapping $\pi_0 : S \rightarrow S$ as the inverse mapping of $(b \circ \pi_1 \times \cdots \times b \circ \pi_k) \circ D_\sigma^k$. Hence,

$$(b \circ \pi_1 \times \cdots \times b \circ \pi_k) \circ D_\sigma^k \circ \pi_0|_S \quad = \quad id|_S,$$

where $id|_S$ is the identical mapping of S.

Because of

$$D_\sigma^k \circ \pi_0 = (\underbrace{\pi_0 \times \cdots \times \pi_0}_{k}) \circ D_\sigma^k,$$

then

$$\begin{aligned}(b \circ \pi_1 \times \cdots \times b \circ \pi_k) \circ D_\sigma^k \circ \pi_0 \quad &= \quad (b \circ \pi_1 \times \cdots \times b \circ \pi_k) \circ (\underbrace{\pi_0 \times \cdots \times \pi_0}_{k}) \circ D_\sigma^k \\ &= \quad (b \times \cdots \times b) \circ (\pi_1 \circ \pi_0 \times \cdots \times \pi_k \circ \pi_0) \circ D_\sigma^k.\end{aligned}$$

Replacing $\pi_i \circ \pi_0$ by π_i, we have the lemma.

Q.E.D.

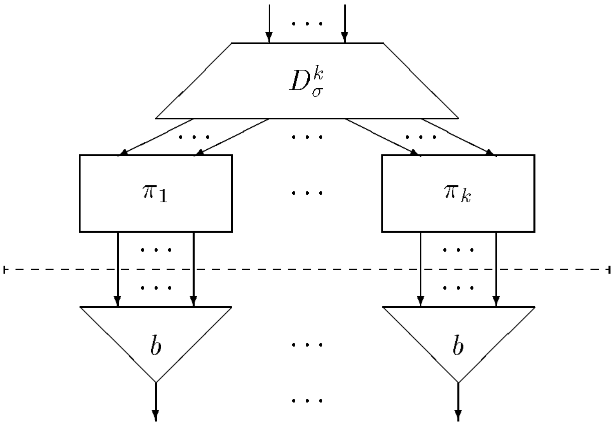


Fig. 1.10 f_b

We use f_b to denote the mapping defined by (1.5). Fig. 1.10 illustrates the structure of f_b. It is made up of two parts. The upper part $(\pi_1 \times \cdots \times \pi_k) \circ D_\sigma^k$ is a fanout mechanism, and it has one input line and k output lines of type σ (k input lines and k^2 output lines of type t) and implements a mapping from t^k to t^{k^2}. The lower part $(\underbrace{b \times \cdots \times b}_{k})$ consists of k b cells connected in parallel. A b cell can be considered as a uniform tree with k input lines of type t (an input line of type σ) and an output line of type t. Then $(\underbrace{b \times \cdots \times b}_{k})$ can be considered as a building block made up of k uniform trees. It has k input lines and one output line of type σ, and implements a mapping from t^{k^2} to t^k. The two parts realize together an identical mapping for t^k. The idea here is to construct a building block which includes uniform trees and realizes an identical mapping for a set S. By applying S to the building block, S is assigned to every uniform tree inside the building block. In the following we use this idea to construct a family of identical mappings $F^h (h \in \mathbf{N})$. F^h includes h-level balanced uniform trees based on b.

We use π to denote $(\pi_1 \times \cdots \times \pi_k) \circ D_\sigma^k$, and f_b to represent $(\underbrace{b \times \cdots \times b}_{k}) \circ \pi$. At first we give a recursive definitions of the balanced uniform tree F_h and fanout mechanism π^h.

Let $\mathbf{N}$ denote the set of all positive integers. The recursive definition for F_h is the following.

$$F_0 = L_t \tag{1.6}$$

$$F_{h+1} = B \circ (\underbrace{F_h \times \cdots \times F_h}_{k}), \quad h \in \mathbf{N} \tag{1.7}$$

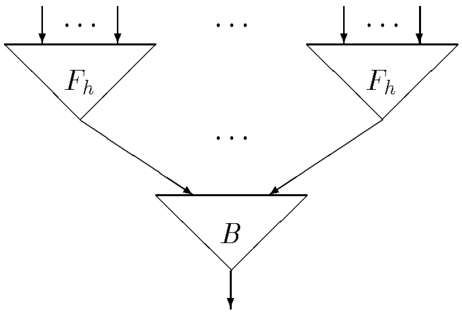


Fig. 1.11 $\quad F_{h+1}$

Fig. 1.11 is the diagram of F_{h+1}. It is easy to see that F_h ($h \in \mathbf{N}$) is a balanced uniform tree of h-level. The π^h-family is defined as follows:

$$\pi^0 = L_t \tag{1.8}$$
$$\pi^{h+1} = (\underbrace{\pi^h \times \cdots \times \pi^h}_{k}) \circ \pi, \quad h = 1, 2, ... \tag{1.9}$$

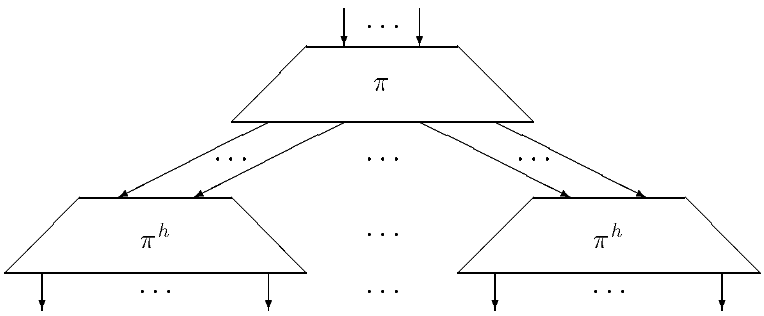


Fig. 1.12 $\quad \pi^{h+1}$

Fig. 1.12 is the diagram of π^{h+1}. Using F_h and π^h we can define the F^h-family.

$$F^h = (\underbrace{F_h \times \cdots \times F_h}_{k}) \circ \pi^h, \qquad h \in \mathbf{N} \tag{1.10}$$

Put it differently,

$$F^h = (\underbrace{F_h \circ \pi^{h-1} \times \cdots \times F_h \circ \pi^{h-1}}_{k}) \circ \pi \tag{1.11}$$

Fig. 1.13 is the diagram of F^h.

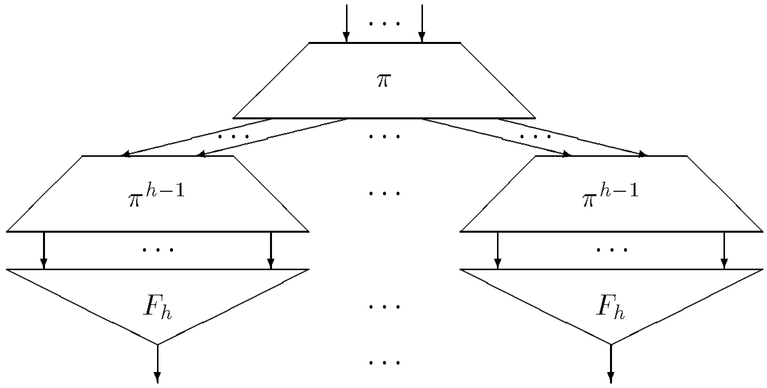


Fig. 1.13 F^h

According to the above definition

$$\begin{aligned}\phi_2(F^1) &= \phi_2((\underbrace{B \times \cdots \times B}_{k}) \circ \pi) \\ &= f_b.\end{aligned}$$

Let $f^h = \phi_2(F^h)$ for $h \in \mathbf{N}$. Then f^h is a mapping from S to S, and it is a generalization of f_b. The following lemma holds.

Lemma 1.3 *$f^h|_S = id|_S$ holds for all $h \in \mathbf{N}$.*

Proof: We prove this lemma by using induction on the parameter h. For $h = 1$, $f^1 = f_b$ and $f^1|_S = id|_S$. Suppose $f^{l-1}|_S = id|_S$ holds. According to the above definition

$$\begin{aligned}F_h \circ \pi^{h-1} &= B \circ (\underbrace{F_{h-1} \times \cdots \times F_{h-1}}_{k}) \circ \pi^{h-1} \\ &= B \circ F^{h-1}, \qquad\qquad h \in \mathbf{N}.\end{aligned}$$

Hence

$$\begin{aligned}F^l &= (\underbrace{F_l \times \cdots \times F_l}_{k}) \circ \pi^l \\ &= (\underbrace{F_l \times \cdots \times F_l}_{k}) \circ (\underbrace{\pi^{l-1} \times \cdots \times \pi^{l-1}}_{k}) \circ \pi \\ &= (\underbrace{F_l \circ \pi^{l-1} \times \cdots \times F_l \circ \pi^{l-1}}_{k}) \circ \pi \\ &= (\underbrace{B \circ F^{l-1} \times \cdots \times B \circ F^{l-1}}_{k}) \circ \pi \\ &= (\underbrace{B \times \cdots \times B}_{k}) \circ (\underbrace{F^{l-1} \times \cdots \times F^{l-1}}_{k}) \circ \pi.\end{aligned}$$

Thus

$$
\begin{aligned}
f^l|_S &= (\underbrace{b \times \cdots \times b}_{k}) \circ (\underbrace{f^{l-1} \times \cdots \times f^{l-1}}_{k}) \circ \pi|_S \\
&= (\underbrace{b \times \cdots \times b}_{k}) \circ \pi|_S \\
&= f^1|_S \\
&= id|_S
\end{aligned}
\tag{1.12}
$$

Q.E.D.

Theorem 1.1 *If $S \subset T^k$ is b-stable, then $\#S$ is the cardinality of the minimum complete assignment set for each of the trees in $\mathcal{A}$ regarding S.*

Proof: Given a tree $F' \in \mathcal{A}$, one can always find a balanced tree $F \in \mathcal{A}$ so that F' can be embedded into F. The assignment complexity of F is an upper boundary of that for F'. Thus trees in $\mathcal{A}$ are constant assignable if balanced trees in $\mathcal{A}$ are constant assignable.

A balanced tree $F_{h+1} \in \mathcal{A}$ has the structure $B \circ (\underbrace{F_h \times \cdots \times F_h}_{k})$. We know that

$$\phi_2((\underbrace{F_h \times \cdots \times F_h}_{k}) \circ \pi^h|_S) = id|_S$$

and

$$(\underbrace{F_h \times \cdots \times F_h}_{k}) \circ \pi^h(S) = S.$$

This indicates that when $\pi^h(S)$ is applied to the primary input lines of F_{h+1}, every cell B in F_{h+1} can be excited by all of the elements in S. Thus $\pi^h(S)$ is just a complete assignment set for F_{h+1} regarding S, and the cardinality of the minimum complete assignment set is $\#S$.

Q.E.D.

Definition 1.7 (sensitive) *Assume $M \subset T$. Function $b : T^k \rightarrow T$ is sensitive if and only if*

$$\forall i \in [1,k] \forall y, y' \in M \forall x_1 \in M^{i-1} \forall x_2 \in M^{k-i} \left\{ y \neq y' \Longleftrightarrow b(x_1 y x_2) \neq b(x_1 y' x_2) \right\}.$$

The sensitive property generalizes the group condition in [BeSp91].

Lemma 1.4 *Assume $M \subset T$. If b is sensitive, then M^k is b-stable.*

Proof: It suffices to prove that for every $i \in [1,k]$, there is a bijective mapping $\pi_i : M^k \rightarrow M^k$ such that $b \circ \pi_i(x_1, ..., x_i, ..., x_k) = x_i$. Without loss of generality, we assume $i = 1$. Given a $z \in M^{k-1}$, for every $x \in M$ there is a unique $y \in M$ such that $b(yz) = x$ since b is sensitive. Let $b^{-1}(x)$ denote the set $\{X \mid b(X) = x\}$. Then $\#b^{-1}(x) \leq \#M^{k-1}$. Because $\sum_{x \in M} \#b^{-1}(x) = \#M^k$, then $\#b^{-1}(x) = \#M^{k-1}$. Thus we can construct a bijective mapping $\pi_1 : M^k \to M^k$ so that $b \circ \pi_1(xz) = x$ for every $z \in M^{k-1}$. In a similar way we can construct a bijective mapping $\pi_i : M^k \to M^k$ so that $b \circ \pi_i(yxz) = x$ for all $y \in M^{i-1}$ and $z \in M^{k-i}$, and

$$(b \times \cdots \times b) \circ (\pi_1 \times \cdots \times \pi_k) \circ D_\sigma^k = id|_S.$$

Q.E.D.

Definition 1.8 (stable test set) *Assume $M \subset T$. M^k is called b-stable test set if*

1. *b is sensitive;*
2. *The complete test set $D(B, B_f)$ of B is a subset of M^k;*
3. *$\forall B' \in B_f \forall u \in M^k \{\phi_2(B')(u) \in M\}$.*

Based on the third condition in the above definition $(b \times \cdots \times b' \times \cdots \times b) \circ \pi(M^k)$ is a subset of M^k for all $B' \in B_f$, if M^k is a b-stable test set.

Theorem 1.2 *If M^k is a b-stable test set, then $\#M^k$ is the cardinality of the minimum complete test set for each of the trees in $\mathcal{A}$.*

Proof: According to the same argument for the proof of Theorem 1.1, it is enough to consider the balanced trees $F_{h+1} \in \mathcal{A}$.

Let $M \subset T$, and M^k be a b-stable test set. According to the above definition and Lemma 1.4, M^k is b-stable. Following Theorem 1.1, $\pi^h(M^k)$ is a complete assignment set for F_{h+1}.

Assume that there is a cell B in F_{h+1} that implements function b' ($B' \in B_f$) instead of b due to a fault. Then there is a $u \in M^k$, and when u is applied to the defected cell, it outputs $b'(u)$ instead of $b(u)$. In other words, u can sensitize the fault and drive a diagnosis signal $b(u)/b'(u)$ to the output line of the defected cell. The output line of the defected cell is either primary output line or a line linked directly to another cell B. According to the third condition of the definition of the *stable test set*, both $b(u)$ and $b'(u)$ belong to M. The diagnosis signal $b(u)/b'(u)$ can be further propagated towards the primary output line since b is sensitive.

Thus we can state that $\pi^h(M^k)$ is a complete assignment set as well as a complete test set for F_{h+1}. This implies that $\#M^k$ is the cardinal number of the minimum complete test set for F_{h+1}.

Q.E.D.

Corollary 1.1 *If $b : T^k \to T$ is sensitive, then $\#T^k$ is the cardinality of the minimum complete test set for $F \in \mathcal{A}$.*

The proofs of Theorem 1.1 and 1.2 are constructive. In fact, they correspond to the algorithms for constructing the minimal complete assignment set and the minimal complete test set for the balanced uniform tree based on function b.

As mentioned, if b is sensitive, then T^k is b-stable. However, b may not be sensitive, even though T^k is b-stable. The difference between the *stable* and *sensitive* can be shown by the following example.

Example 1.1: Let $M = \{1, 2\}$ and $S = M^2$. Function $b : S \to M$ defined below is not sensitive since $b(1,1) = b(2,1) = 1$.

b	1	2
1	1	2
2	1	2

However,

$$\pi_1 := \{(1,1) \to (1,1),\ (1,2) \to (2,1),\ (2,1) \to (1,2),\ (2,2) \to (2,2)\}$$

and

$$\pi_2 := \{(1,1) \to (1,1),\ (1,2) \to (1,2),\ (2,1) \to (2,1),\ (2,2) \to (2,2)\}$$

are two bijective mappings from S to S, and

$$(b \times b) \circ (\pi_1 \times \pi_2) \circ D_S^2|_S = id|_S.$$

It indicates that S is b-stable.

Definition 1.9 (related functions) *Functions $b_1, b_2 : T^k \to T$ are said to be* related to *each other (denoted by $b_1 \bowtie b_2$) if and only if there are bijective mappings $\mu_0 : T \to T$ and $\mu : T^k \to T^k$ such that $b_2 = \mu_0 \circ b_1 \circ \mu$.*

It is easy to show that $\bowtie$ is an equivalence relation. In other words,

1. $\forall b \in \mathcal{F}\,\{b \bowtie b\}$;
2. $\forall a, b \in \mathcal{F}\,\{a \bowtie b \Longrightarrow b \bowtie a\}$;
3. $\forall a, b, c \in \mathcal{F}\,\{a \bowtie b \wedge b \bowtie c \Longrightarrow a \bowtie c\}$.

Lemma 1.5 *Assume that $b_1 \bowtie b_2$ and S_1 is a b_1-stable set, then there is also a b_2-stable set S_2, and $\#S_1 = \#S_2$.*

Proof: Suppose there are bijective mappings $\mu_0 : T \to T$ and $\mu : T^k \to T^k$, such that $b_2 = \mu_0 \circ b_1 \circ \mu$. Assume S_1 to be b_1-stable and

$$(b_1 \circ \pi_1^{(1)} \times \cdots \times b_1 \circ \pi_k^{(1)}) \circ D_\sigma^k = id|_{S_1}$$

for k bijective mappings $\pi_1^{(1)}, ..., \pi_k^{(1)} : T^k \to T^k$, where $\sigma = \underbrace{t \cdots t}_{k}$.

Let $\pi_i^{(2)} = \mu^{-1} \circ \pi_i^{(1)} \circ (\underbrace{\mu_0^{-1} \times \cdots \times \mu_0^{-1}}_{k})$ $(i \in [1,k])$, and $S_2 = (\underbrace{\mu_0 \times \cdots \times \mu_0}_{k})(S_1)$, then

$$\begin{aligned}
& (b_2 \circ \pi_1^{(2)} \times \cdots \times b_2 \circ \pi_k^{(2)}) \circ D_\sigma^k(S_2) \\
= \ & (b_2 \circ \mu^{-1} \circ \pi_1^{(1)} \times \cdots \times b_2 \circ \mu^{-1} \circ \pi_k^{(1)}) \circ D_\sigma^k \circ (\underbrace{\mu_0^{-1} \times \cdots \times \mu_0^{-1}}_{k})(S_2) \\
= \ & (b_2 \circ \mu^{-1} \circ \pi_1^{(1)} \times \cdots \times b_2 \circ \mu^{-1} \circ \pi_k^{(1)}) \circ D_\sigma^k(S_1).
\end{aligned}$$

Notice that

$$\begin{aligned} id|_{S_1} &= (b_1 \circ \pi_1^{(1)} \times \cdots \times b_1 \circ \pi_k^{(1)}) \circ D_\sigma^k \\ &= (\mu_0^{-1} \circ b_2 \circ \mu^{-1} \circ \pi_1^{(1)} \times \cdots \times \mu_0^{-1} \circ b_2 \circ \mu^{-1} \circ \pi_k^{(1)}) \circ D_\sigma^k \\ &= (\underbrace{\mu_0^{-1} \times \cdots \times \mu_0^{-1}}_{k}) \circ (b_2 \circ \mu^{-1} \circ \pi_1^{(1)} \times \cdots \times b_2 \circ \mu^{-1} \circ \pi_k^{(1)}) \circ D_\sigma^k. \end{aligned}$$

Thus

$$\begin{aligned} & (b_2 \circ \mu^{-1} \circ \pi_1^{(1)} \times \cdots \times b_2 \circ \mu^{-1} \circ \pi_k^{(1)}) \circ D_\sigma^k \\ = \; & (\underbrace{\mu_0 \times \cdots \times \mu_0}_{k}) \circ id|_{S_1} \end{aligned}$$

and

$$\begin{aligned} & (b_2 \circ \pi_1^{(2)} \times \cdots \times b_2 \circ \pi_k^{(2)}) \circ D_\sigma^k(S_2) \\ = \; & (\underbrace{\mu_0 \times \cdots \times \mu_0}_{k}) \circ id|_{S_1}(S_1) \\ = \; & (\underbrace{\mu_0 \times \cdots \times \mu_0}_{k})(S_1) \\ = \; & S_2. \end{aligned}$$

It indicates that S_2 is b_2-stable, and $\#S_1 = \#S_2$.

Q.E.D.

Definition 1.10 (equivalent functions) *Functions $b_1, b_2 : T^k \to T$ are considered to be equivalent to each other (denoted by $b_1 \equiv b_2$) if there is a bijective mapping $\mu : T \to T$ such that $b_2 = \mu^{-1} \circ b_1 \circ (\underbrace{\mu \times \cdots \times \mu}_{k})$.*

Theorem 1.3 *Suppose $b_1 \equiv b_2$, $b_2 = \mu^{-1} \circ b_1 \circ (\underbrace{\mu \times \cdots \times \mu}_{k})$ for a bijective mapping $\mu : T \to T$. Assume F_h and $F_h^{'}$ to be uniform trees based on B_1 and B_2, respectively. Then*

- *both trees F_h and $F_h^{'}$ have the same assignment complexity;*
- *assume that $D_{b_1} = \{r/s \mid r \neq s \wedge r, s \in T\}$ and $D_{b_2} = \{\mu(r)/\mu(s) \mid r/s \in D_{b_1}\}$ are the diagnosis signal sets of B_1 and B_2, then both trees F_h and $F_h^{'}$ have the same test complexity.*

Proof: This theorem includes two propositions. Coming up next is the proof of the first proposition.

Suppose there is a bijective mapping $\mu : T \to T$ such that $b_2 = \mu^{-1} \circ b_1 \circ (\underbrace{\mu \times \cdots \times \mu}_{k})$.

Let f_h and f_h' denote $\phi_2(F_h)$ and $\phi_2(F_h')$, respectively. Then

$$\begin{aligned}
\phi_2(F_1) &= b_1 \\
\phi_2(F_{h+1}) &= f_{h+1} \\
&= b_1 \circ (f_h \times \cdots \times f_h) \\
\phi_2(F_1') &= b_2 \\
\phi_2(F_{h+1}') &= f_{h+1}' \\
&= b_2 \circ (f_h' \times \cdots \times f_h').
\end{aligned}$$

Define

$$\nu^{(1)} = \left(\underbrace{\mu^{-1} \times \cdots \times \mu^{-1}}_{k}\right) \tag{1.13}$$

$$\nu^{(h)} = \left(\underbrace{\nu^{(h-1)} \times \cdots \times \nu^{(h-1)}}_{k}\right), \qquad h = 2, 3, \ldots \tag{1.14}$$

Let

$$A_2^{(h)} = \nu^{(h)}\left(A_1^{(h)}\right) \tag{1.15}$$

Now we show that

$$\forall h \in \mathbf{N}\left\{f_h'\left(A_2^{(h)}\right) = \mu^{-1} \circ f_h\left(A_1^{(h)}\right)\right\}.$$

For $h = 1$,

$$\begin{aligned}
f_h'\left(A_2^{(h)}\right) &= f_1'\left(A_2^{(1)}\right) \\
&= b_2\left(\nu^{(1)}\left(A_1^{(1)}\right)\right) \\
&= \mu^{-1} \circ b_1 \circ (\underbrace{\mu \times \cdots \times \mu}_{k})\left((\underbrace{\mu^{-1} \times \cdots \times \mu^{-1}}_{k})\left(A_1^{(1)}\right)\right) \\
&= \mu^{-1} \circ b_1\left(A_1^{(1)}\right) \\
&= \mu^{-1} \circ f_h\left(A_1^{(h)}\right).
\end{aligned}$$

Suppose

$$\forall h \leq l\left\{f_h'\left(A_2^{(h)}\right) = \mu^{-1} \circ f_h\left(A_1^{(h)}\right)\right\}$$

holds. For $h = l + 1$, $A_1^{(l+1)}$ is a set of k^{l+1}-component vectors. Assume $A_i^{(l+1)} = A_{i,1}^{(l)} A_{i,2}^{(l)} \cdots A_{i,k}^{(l)}$, where $A_{i,j}^{(l)}$ ($i \in [1,2], j \in [1,k]$) is a set of k^l-component vectors.

$$f_h'\left(A_2^{(h)}\right) = f_{l+1}'\left(A_2^{(l+1)}\right)$$

$$
\begin{aligned}
&= b_2 \circ (\underbrace{f_l' \times \cdots \times f_l'}_{k}) \left(A_2^{(l+1)}\right) \\
&= b_2 \circ (\underbrace{f_l' \times \cdots \times f_l'}_{k}) \left(A_{2,1}^{(l)} \cdots A_{2,k}^{(l)}\right) \\
&= b_2 \left(f_l' \left(A_{2,1}^{(l)}\right) \cdots f_l' \left(A_{2,k}^{(l)}\right)\right) \\
&= \mu^{-1} \circ b_1 \circ (\underbrace{\mu \times \cdots \times \mu}_{k}) \left(\mu^{-1} \circ f_l \left(A_{1,1}^{(l)}\right) \cdots \mu^{-1} \circ f_l \left(A_{1,k}^{(l)}\right)\right) \\
&= \mu^{-1} \circ b_1 \circ \left(f_l \left(A_{1,1}^{(l)}\right) \cdots f_l \left(A_{1,k}^{l}\right)\right) \\
&= \mu^{-1} \circ b_1 \circ (\underbrace{f_l \times \cdots \times f_l}_{k}) \left(A_1^{(l+1)}\right) \\
&= \mu^{-1} \circ f_{l+1} \left(A_1^{(l+1)}\right) \\
&= \mu^{-1} \circ f_h \left(A_1^{(h)}\right).
\end{aligned}
$$

Now we show that whenever $A_1^{(h)}$ is a complete assignment set for F_h, $A_2^{(h)}$ defined by (1.15) is a complete assignment for F_h'.

For $h = 1$, it is trivial that $A_1^{(1)} \supset T^k$ implies that $\nu^{(1)}(A_1) \supset T^k$. Thus if $A_1^{(1)}$ is a complete assignment set for F_1, then $A_2^{(1)}$ is a complete assignment set for F_1'.

Assume that for $h \leq l$, if $A_1^{(h)}$ is a complete assignment set for F_h, then $A_2^{(h)}$ is a complete assignment set for F_h'.

Suppose $A_1^{(l+1)} = A_{1,1}^{(l)} A_{1,2}^{(l)} \cdots A_{1,k}^{(l)}$ is a complete assignment set for F_{l+1}. Since

$$F_{l+1} = B_1 \circ (\underbrace{F_l \times \cdots \times F_l}_{k})$$

and

$$f_{l+1} \left(A_1^{(l+1)}\right) = b_1 \left(f_l \left(A_{1,1}^{(l)}\right) \cdots f_l \left(A_{1,k}^{(l)}\right)\right) \tag{1.16}$$

then $A_{1,j}^{(l)}$ is a complete assignment of F_l for every $j \in [1, k]$ and

$$f_l \left(A_{1,1}^{(l)}\right) \cdots f_l \left(A_{1,k}^{(l)}\right)$$

is a complete assignment set for the cell B_1. This implies that $\nu^{(l)} \left(A_{1,j}^{(l)}\right)$ is a complete assignment set of F_l' for every $j \in [1, k]$ and

$$(\underbrace{\mu^{-1} \times \cdots \times \mu^{-1}}_{k}) \left(f_l \left(A_{1,1}^{(l)}\right) \cdots f_l \left(A_{1,k}^{(l)}\right)\right)$$

is a complete assignment set for the cell B_2.

Notice that

$$\begin{aligned}
f'_{l+1}\left(A_2^{(l+1)}\right) &= f'_{l+1}\left(\nu^{(l+1)} A_1^{(l+1)}\right) \\
&= f'_{l+1}\left(\underbrace{(\nu^{(l)} \times \cdots \times \nu^{(l)})}_{k}\left(A_{1,1}^{(l)} \cdots A_{1,k}^{(l)}\right)\right) \\
&= b_2\left(f'_l\left(\nu^{(l)}\left(A_{1,1}^{(l)}\right)\right) \cdots f'_l\left(\nu^{(l)}\left(A_{1,k}^{(l)}\right)\right)\right) \\
&= b_2\left(\mu^{-1} \circ f_l\left(A_{1,1}^{(l)}\right) \cdots \mu^{-1} \circ f_l\left(A_{1,k}^{(l)}\right)\right) \\
&= b_2\left(\underbrace{(\mu^{-1} \times \cdots \times \mu^{-1})}_{k}\left(f_l\left(A_{1,1}^{(l)}\right) \cdots f_l\left(A_{1,k}^{(l)}\right)\right)\right).
\end{aligned}$$

This indicates that $A_2^{(l+1)} = \nu^{(l+1)}\left(A_1^{(l+1)}\right)$ is a complete assignment set to F'_{l+1}. Thus we can state that if $A_1^{(h)}$ is a complete assignment set to F_h, then $A_2^{(h)}$ defined by (1.15) is certainly a complete assignment set to F'_h.

To prove the second proposition, we are required only to show that whenever a pattern $x \in T^k$ can propagate a diagnosis signal $r/s \in D_{b_1}$ through a cell B_1, the corresponding pattern $(\mu \times \cdots \times \mu)(x)$ can propagate the associated diagnosis signal $\mu(r)/\mu(s) \in D_{b_2}$ through a cell B_2. This is immediate since

$$\forall x, y \in T^k \left\{b_1(x) \neq b_1(y) \Longrightarrow b_2 \circ (\mu^{-1} \times \cdots \times \mu^{-1})(x) \neq b_2 \circ (\mu^{-1} \times \cdots \times \mu^{-1})(y)\right\}.$$

Q.E.D.

Example 1.2: The following tabular illustrates the definitions of binary functions $\wedge$ and $\vee$.

x	y	$\wedge$	$\vee$
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Let $T = \{0, 1\}$ and

$$\begin{aligned}
\mu_0 : &= \{0 \to 1, 1 \to 0\}, \\
\mu : &= \{(0,0) \to (1,1), (0,1) \to (1,0), (1,0) \to (0,1), (1,1) \to (0,0)\}.
\end{aligned}$$

It is easy to check that μ_0 is a bijective mapping from T to T, and μ from T^2 to T^2. Furthermore, $\mu = (\mu_0^{-1} \times \mu_0^{-1})$ and $\vee = \mu_0 \circ \wedge \circ \mu$.

Assume F to be a uniform tree based on $\wedge$, and F' is induced by replacing every $\wedge$ cell in F with a $\vee$ cell. Then F and F' have the same assignment and test complexity.

The following corollary is immediate from Theorem 1.3.

Corollary 1.2 *Assume $\mathcal{G} : (M, f) \longrightarrow (\mathcal{M}, g)$ to be a morphism. F_h and F'_h based on f and g, respectively, have the same test complexity if $\mathcal{G}$ is an isomorphism.*

Chapter 2

Assignment Complexity of Uniform Trees

This Chapter discusses the assignment complexity of the uniform tree, which is made up of identical cells realizing a function f. A complete assignment set for a tree with n primary input lines consists of a number of n-component patterns. When a complete assignment set is applied to the primary input lines of the tree, every internal f cell in the tree can be excited by all possible input combinations. The assignment complexity of a tree is defined as the cardinal number of the minimum complete assignment set of the tree. The assignment problem is a basic problem in the design, test and optimization of VLSI systems. We analyze the relationship between the property of f and the assignment complexity of the uniform tree and show that the assignment complexity of a balanced uniform tree with n primary input lines is either $\Theta(1)$ or $\Omega((\lg n)^\alpha)$ ($\alpha \in (0, 1]$). In the first case, the cardinal number of the minimum complete assignment set for a tree is constant and independent of the size and structure of the tree. In the second case, the assignment complexity depends on the number of the primary input lines of the tree. If a balanced uniform tree is based on a commutative function, then it is either $\Theta(1)$ or $\Theta(\lg n)$ assignable.

This Chapter consists of six sections. In section 2.1 we give a formal definition of the assignment complexity of uniform trees and make some conventions. Section 2.2 is on the sufficient and necessary condition of $\Theta(1)$ assignable uniform trees. Section 2.3 explores the jump of the assignment complexity from $\Theta(1)$ to $\Omega((\lg n)^\alpha)$. In section 2.4 we convert the assignment problem into the algebraic problem. Section 2.5 shows that a balanced uniform tree based on a commutative function is either $\Theta(1)$ or $\Theta(\lg n)$ assignable, and gives also an upper bound of the cardinality of the minimum complete assignment set for the $\Theta(1)$ assignable balanced uniform trees. Section 2.6 is on the complexity of deciding the assignment complexity of balanced uniform trees.

2.1 Assignment Complexity of Uniform Trees

Let M be a set of m symbols, and $f : M^k \longrightarrow M$ a surjective function. Without loss of generality we assume $M = \{1, 2, ..., m\}$. We use the symbol f to represent a function as well as a cell implementing the function. A uniform f-tree is made up of identical cells implementing the function f. The set of all f-trees is denoted by T_f. $T_f^{(n)}$ is used to denote a balanced uniform f-tree with n primary input lines. Fig. 2.1 shows a balanced tree. If every cell $C_{i,j}$ realizes the same function $f : M^k \longrightarrow M$, then it is a uniform tree.

We assign every line and cell in $T_f^{(n)}$ a unique level. The levels are arranged in ascending order from the primary output line to the primary input lines of $T_f^{(n)}$. The primary output line is assigned level 0. An f cell and all its input lines are assigned level $k+1$, if its output line is in level k. A tree is said to be of k-level, if it has k levels.

For the sake of convenience, we make some conventions. Throughout this thesis, $\{a, a, a\}$ and $\{a, a\}$ are recognized as two different *multiple* sets. The cardinal number of the former is three, and that of the latter is two. A multiple set can be changed into a conventional set by using operator $\top$. For example, $\top\{a, a, a\} = \top\{a, a\} = \{a\}$ and $\top\{b, c, b\} = \{b, c\}$. For a multiple set A, $\#A$ represents the elements number of A. For example, $\#\{a, a, a\} = 3$.

Let

$$\vec{I_j} = (I_{1j}, I_{2j}, ..., I_{tj})^T, \qquad j \in [1, k],\ I_{ij} \in M$$

$$(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \;:=\; \begin{pmatrix} I_{11}, I_{12}, ..., I_{1k} \\ I_{21}, I_{22}, ..., I_{2k} \\ \vdots \\ I_{t1}, I_{t2}, ..., I_{tk} \end{pmatrix} \tag{2.1}$$

According to function f we define a vector function $\tilde{f}$ as follows:

$$\tilde{f}(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \;:=\; \begin{pmatrix} f(I_{11}, I_{12}, ..., I_{1k}) \\ f(I_{21}, I_{22}, ..., I_{2k}) \\ \vdots \\ f(I_{t1}, I_{t2}, ..., I_{tk}) \end{pmatrix} \tag{2.2}$$

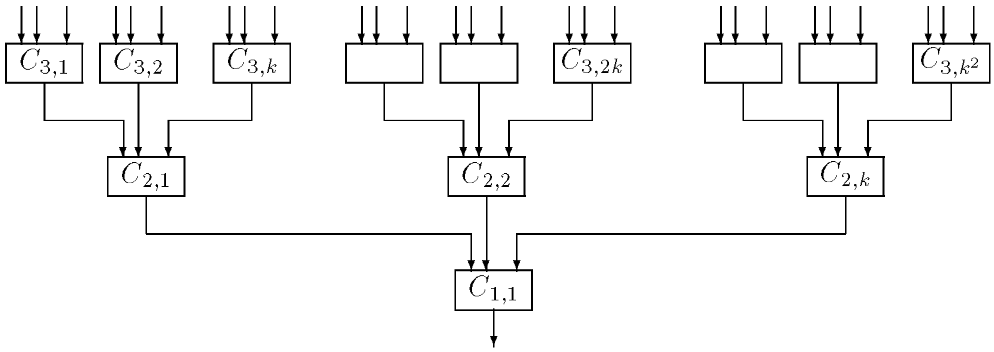


Fig. 2.1: A balanced tree

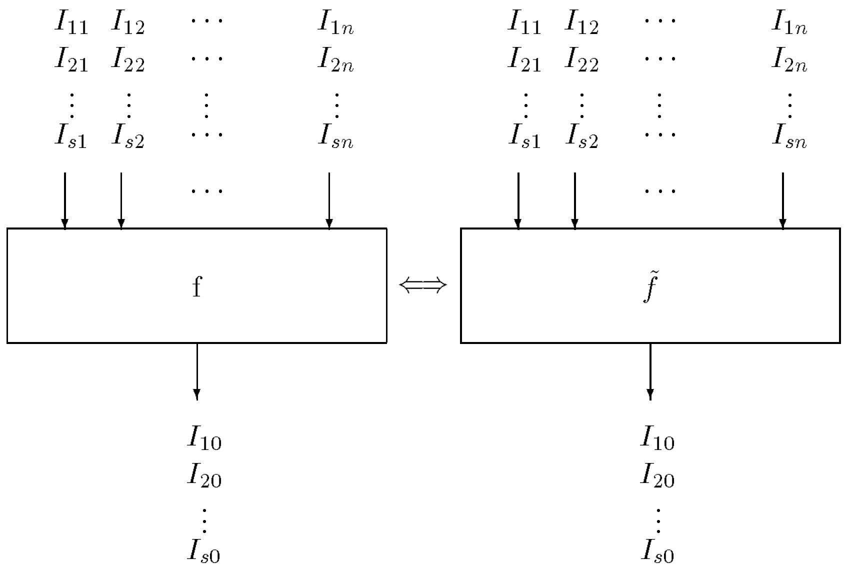


Fig. 2.2

It is easy to see that applying t k-component patterns to an f cell is equal to assigning k t-dimension vectors to the k input lines of the f cell. Fig 2.2 gives us a simple illustration.

Let $\mathcal{D}_t = \{(x_1, ..., x_t)^T \mid x_i \in M\}$, namely, the set of all t-dimension vectors ($t \in \mathbf{N}$). Given k vectors in $\mathcal{D}_t$, by using operator ∇ one can construct a set of t k-component patterns, and (2.3) is the formal definition of this operator.

$$\nabla(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \quad := \quad \{(I_{i1}, I_{i2}, ..., I_{ik}) \mid i \in [1, t]\} \tag{2.3}$$

Example 2.1: Function f_1 is defined as follows.

f_1	0	1
0	1	1
1	1	0

Let

$$\vec{I_0} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{pmatrix}, \quad \vec{I_1} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \quad \vec{I_2} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

Then

$$\tilde{f}_1\left(\vec{I}_1, \vec{I}_2\right) = \begin{pmatrix} f_1(1,1) \\ f_1(1,1) \\ f_1(1,0) \\ f_1(0,1) \\ f_1(0,0) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{pmatrix} = \vec{I}_0$$

and

$$\nabla\left(\vec{I}_1, \vec{I}_2\right) = \{(1,1),(1,1),(1,0),(0,1),(0,0)\}.$$

Assume that a $T_f^{(n)}$ consists of cells $C_{1,1}, C_{2,1}, C_{2,2}, ..., C_{k,l}, ...$, and cell $C_{i,j}$ is the jth cell in the ith level of $T_f^{(n)}$. Let A be a set of n-component patterns, and $\#A = t$. When all of the patterns in A are applied to the primary input lines of $T_f^{(n)}$, a t-component vector is delivered to every line in $T_f^{(n)}$. To apply all of the t patterns to the n primary input lines of $T_f^{(n)}$ is the same as to apply n t-component vectors to the n primary input lines, respectively. We use $\vec{I}_l(A,i,j)$ to denote the corresponding vector applied to the lth input line of an f cell $C_{i,j}$, and $\vec{I}_0(A,i,j)$ to denote the vector delivered to the output line of the cell. Then $\vec{I}_0(A,i,j) = \tilde{f}(\vec{I}_1(A,i,j), \vec{I}_2(A,i,j), \cdots, \vec{I}_k(A,i,j))$.

In order to classify the assignment complexity of uniform trees we give a formal definition of the complete assignment and the assignment complexity.

Definition 2.1 (assignment of uniform trees) $(\vec{I}_1(A,i,j), \vec{I}_2(A,i,j), \cdots, \vec{I}_k(A,i,j))$ *is called a complete assignment of cell* $C_{i,j}$ *if and only if*

$$M^k \subset \nabla(\vec{I}_1(A,i,j), \vec{I}_2(A,i,j), \cdots, \vec{I}_k(A,i,j)).$$

A is a complete assignment set of $T_f^{(n)}$ if and only if
$(\vec{I}_1(A,i,j), \vec{I}_2(A,i,j), \cdots, \vec{I}_k(A,i,j))$ is a complete assignment for every cell $C_{i,j}$ in $T_f^{(n)}$.

Fig. 2.3 shows a complete assignment set to $T_{f_1}^{(4)}$. By assigning five patterns $(1,1,1,1)$,$(0,1,1,1)$,$(1,1,0,1)$,$(1,0,1,0)$ and $(0,0,0,0)$ to the four primary input lines of $T_{f_1}^{(4)}$, one can guarantee that each of $(0,0)$, $(0,1)$, $(1,0)$ and $(1,1)$ can be applied to every cell in $T_{f_1}^{(4)}$. Thus we can state that the five patterns comprise a complete assignment set to $T_{f_1}^{(4)}$.

It is obvious that f has to be surjective to M, otherwise, one could not construct a complete assignment set for a tree system $T_f^{(n)}$ $(n > 2)$.

Definition 2.2 (assignment complexity of uniform trees) *The assignment complexity of balanced uniform tree $T_f^{(n)}$ is defined by the mapping*

$$\begin{aligned} AC_f &: T_f \longrightarrow \mathbf{N} \\ AC_f\left(T_f^{(n)}\right) &= \min\left\{\#A \;\middle|\; \begin{array}{c} A \text{ is a complete} \\ \text{assignment set for } T_f^{(n)} \end{array}\right\} \end{aligned} \quad (2.4)$$

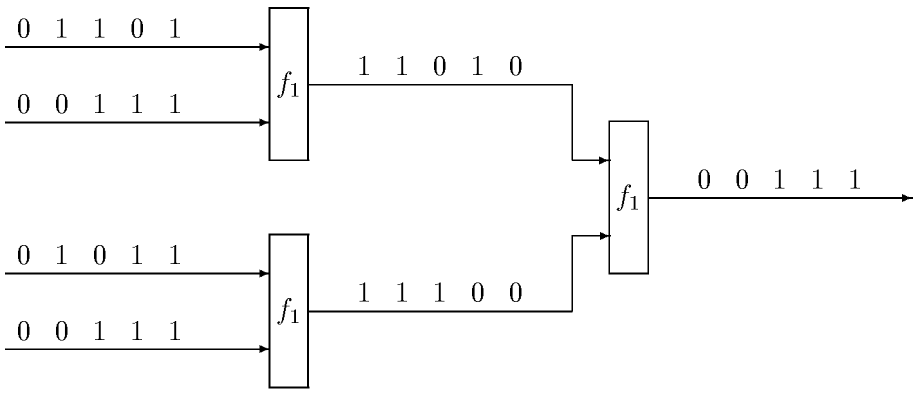


Fig. 2.3: Complete assignment for $T_{f_1}^{(4)}$

In a tree, all cells at the same level can be assigned simultaneously since their input lines are independent of each other. Furthermore, all cells at the same level can be excited completely by using m^k patterns. A straightforward conclusion is that all cells in $T_f^{(n)}$ can be excited completely by using $m^k\lceil\lg n\rceil$ patterns since $T_f^{(n)}$ has at the most $\lceil\lg n\rceil$ levels. Thus we have the following observation.

Observation 2.1 *For arbitrary surjective function $f : M^k \longrightarrow M$*

$$\begin{aligned} AC_f\left(T_f^{(n)}\right) &\leq m^k\lceil\lg n\rceil \\ &= O(\lg n) \end{aligned} \tag{2.5}$$

2.2 $\Theta(1)$ Assignable Trees

In this section we discuss the criteria of $\Theta(1)$ assignable uniform tree systems.

Lemma 2.1 *$T_f^{(n)}$ is $\Theta(1)$ assignable if there are a $t \in \mathbf{N}$ and a set $W \subset \mathcal{D}_t$ so that every $\vec{I_0} \in W$ can be generated by using k vectors $\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}$ which belong to W and comprise a complete assignment to an f cell. Put it formally,*

$$\exists t \in \mathbf{N} \exists W \subset \mathcal{D}_t \forall \vec{I_0} \in W \exists \vec{I_1}, \vec{I_2}, \cdots, \vec{I_k} \in W \left\{ \begin{array}{c} M^k \subset \nabla(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \\ \tilde{f}(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) = \vec{I_0} \end{array} \right\} \tag{2.6}$$

Proof: We prove that for every N-level $T_f^{(n)}$ we can construct a complete assignment set $A^{(N)}$ by assigning to every primary input line a vector in $W \in \mathcal{D}_t$. Then $\#A^{(N)}$ is equal to the constant t. This can be proven by using induction on the number of the level of the tree.

In case $N = 1$, the tree has only one cell. We choose an arbitrary $\vec{I_0} \in W$, then determine k vectors $\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k} \in W$ so that $M^k \subset \nabla(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k})$ and $\vec{f}(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) = \vec{I_0}$. It is clear that $\nabla(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k})$ is a complete assignment for the tree with only one cell.

Assume that for $N = i$ one can construct a complete assignment set $A^{(i)}$ for an i-level $T_f^{(n)}$ by assigning to every primary input line of $T_f^{(n)}$ a vector in W, and the vector assigned to the jth primary input line is $\vec{I}_{j,0} \in W$. Suppose $T_f^{(kn)}$ is of $i+1$ levels and is constructed by connecting every primary input line in $T_f^{(n)}$ to the output line of an f cell.

According to the assumption, there are $\vec{I}_{j,1}, \vec{I}_{j,2}, \cdots, \vec{I}_{j,k} \in W$ such that

$$M^k \subset \nabla(\vec{I}_{j,1}, \vec{I}_{j,2}, \cdots, \vec{I}_{j,k}) \ \wedge \ \vec{f}(\vec{I}_{j,1}, \vec{I}_{j,2}, \cdots, \vec{I}_{j,k}) = \vec{I}_{j,0}.$$

Hence $(\vec{I}_{j,1}, \vec{I}_{j,2}, \cdots, \vec{I}_{j,k})$ is a complete assignment for an f cell. When $\vec{I}_{j,1}, \vec{I}_{j,2}, \cdots, \vec{I}_{j,k}$ are applied to the k input lines of the cell linked directly to the jth input line in the level i, the vector offered to this input line is just $\vec{I}_{j,0}$. Thus we can construct a complete assignment to every cell in level $i+1$ by assigning to every primary input line in $T_f^{(kn)}$ a vector in W, and all of the vectors delivered to the lines in level i comprise $A^{(i)}$, which is a complete assignment set for $T_f^{(n)}$ as assumed. All of the vectors assigned to the lines in level $i+1$ comprise the $A^{(i+1)}$ which is a complete assignment set for $T_f^{(kn)}$. Thus we can conclude that $\#A^{(i)} = \#A^{(i+1)}$ and $T_f^{(n)}$ is $\Theta(1)$ assignable.

Q.E.D.

For $\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k} \in \mathcal{D}_t$, we regard $\left(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}\right)$ as a $t \times k$ matrix, and $\vec{I} \in \mathcal{D}_t$ as a 1 times k matrix.

Definition 2.3 (similar matrices) *Two matrices* $\left(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}\right)$ *and* $\left(\vec{I}_{i_1}, \vec{I}_{i_2}, \cdots, \vec{I}_{i_k}\right)$ *are said to be* similar *to each other, denoted by* $\left(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}\right) \sim \left(\vec{I}_{i_1}, \vec{I}_{i_2}, \cdots, \vec{I}_{i_k}\right)$, *if and only if the former can be changed to the latter by using row exchanges.*

For example,

$$\begin{pmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \\ 1 & 1 \end{pmatrix} \sim \begin{pmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 1 \\ 1 & 0 \\ 1 & 1 \end{pmatrix} \sim \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}$$

and

$$\begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{pmatrix} \sim \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \sim \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

It is easy to see that for three arbitrary matrices A_i, A_j, A_l, the following three statements hold.

1. $A_i \sim A_i$;
2. $A_i \sim A_j \Longrightarrow A_j \sim A_i$;
3. $A_i \sim A_j \wedge A_j \sim A_l \Longrightarrow A_i \sim A_l$.

Hence $\sim$ is an equivalence relation.

Corollary 2.1 *$T_f^{(n)}$ is $\Theta(1)$ assignable ($AC_f(T_f^{(n)}) = \Theta(1)$) if there are a $t \in \mathbf{N}$ and a set $W' \subset \mathcal{D}_t$ so that for every $\vec{I_0} \in W'$ there are $\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k} \in W'$, and they comprise a complete assignment and can be transferred into a vector* similar *to $\vec{I_0}$. Put it formally*

$$\exists t \in \mathbf{N} \exists W' \in \mathcal{D}_t \forall \vec{I_0} \in W' \exists \vec{I_1}, \vec{I_2}, \cdots, \vec{I_k} \in W' \left\{ \begin{array}{c} M^k \subset \nabla(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \\ \tilde{f}(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \sim \vec{I_0} \end{array} \right\} \qquad (2.7)$$

Proof: Given a set $W' \subset \mathcal{D}_t$ ($t \in \mathbf{N}$), we can always induce a set W so that

$$\forall \vec{I} \in \mathcal{D}_t \left\{ \exists \vec{I'} \in W' \left\{ \vec{I'} \sim \vec{I} \right\} \Longrightarrow \vec{I} \in W \right\} \ \wedge\ \forall \vec{I} \in W \exists \vec{I'} \in W' \left\{ \vec{I'} \sim \vec{I} \right\}.$$

The set W includes every vector which is *similar* to a vector in W'. It is obvious that W satisfies (2.6) if W' fulfills (2.7).

Q.E.D

As mentioned, applying t n-component patterns to the n primary input lines of $T_f^{(n)}$ is equal to applying n t-dimension vectors to the n primary input lines respectively.

Apply a complete assignment A to $T_f^{(n)}$. Let W be the set of the corresponding vectors applied to the primary input lines and the vectors delivered to other lines in every level of $T_f^{(n)}$. Set W can be partitioned into a number of equivalence classes according to the equivalence relation $\sim$. It is not hard to see that the larger the number of the equivalence classes in W, the greater the dimension t of the vectors in W. The dimension t is just the cardinality of A. We explore the relationship between the cardinality of A and the number of the equivalence classes in W.

Given a complete assignment A to an N-level $T_f^{(n)}$, we can construct N sets in the following way.

$$W_s(A) := \top \left\{ \vec{I_l}(A, i, j) \mid i \in [1, s], j \in [1, k^{i-1}], l \in [0, k] \right\}, \qquad s \in [1, N] \qquad (2.8)$$

$W_s(A)$ includes all vectors delivered to a line in level i ($i \in [1, s]$) and the vector delivered to the primary output line.

We partition $W_s(A)$ into equivalence classes according to the equivalence relation $\sim$, and use $\#W_s(A)/_\sim$ to denote the number of equivalence classes in $W_s(A)$. Observation 2.2 is obvious.

Observation 2.2 *Assume A to be a complete assignment set of an N-level $T_f^{(n)}$. Then*

1. $\forall s \in [2, N]\, \{W_{s-1}(A) \subseteq W_s(A) \subseteq \mathcal{D}_t\}$;

2. $\forall s \in [2, N]\, \{1 \leq \#W_{s-1}(A)/_\sim \leq \#W_s(A)/_\sim\}$.

Lemma 2.2 *Assume A to be a complete assignment set to an N-level $T_f^{(n)}$. Then $T_f^{(n)}$ is $\Theta(1)$ assignable if $\#W_1(A)/_\sim = 1$ or $\#W_s(A)/_\sim = \#W_{s-1}(A)/_\sim$ for an $s \in [2, N]$.*

Proof: Assume A to be a complete assignment set for an N-level $T_f^{(n)}$. In case $\#W_1(A)/_\sim = 1$, $W_1(A)$ includes only one equivalence class, and two arbitrary vectors in $W_1(A)$ are *similar* to each other. Suppose $W_1(A) = \{\vec{I_0}, \vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}\}$, and $\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}$ are the corresponding vectors applied to the k input lines of the f cell in the first level, and $\vec{I_0}$ is the vector delivered to the output line. $\#W_1(A)_\sim = 1$ means that

$$\forall l \in [1, k] \left\{\vec{I_l} \sim \vec{I_0}\right\} \wedge M^k \subset \nabla(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \wedge \tilde{f}(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) = \vec{I_0}.$$

This implies that

$$\forall \vec{I_0'} \in W_1(A) \exists \vec{I_1'}, \vec{I_2'}, \cdots, \vec{I_k'} \in W_1(A) \left\{ \begin{array}{c} M^k \subset \nabla(\vec{I_1'}, \vec{I_2'}, \cdots, \vec{I_k'}) \\ \tilde{f}(\vec{I_1'}, \vec{I_2'}, \cdots, \vec{I_k'}) \sim \vec{I_0'} \end{array} \right\}.$$

Thus $T_f^{(n)}$ is $\Theta(1)$ assignable according to Corollary 2.1.

Suppose $\#W_s(A)/_\sim = \#W_{s-1}(A)/_\sim$ for an $s \in [2, N]$. As mentioned,

$$\forall s \in [2, N]\, \{W_{s-1}(A) \subseteq W_s(A)\}.$$

This indicates that $W_s(A)$ and $W_{s-1}(A)$ have the same number of equivalence classes. It is not hard to see that

$$\forall \vec{I_0} \in W_s(A) \exists \vec{I_1}, \vec{I_2}, \cdots, \vec{I_k} \in W_s(A) \left\{ \begin{array}{c} M^k \subset \nabla(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \\ \tilde{f}(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \sim \vec{I_0} \end{array} \right\}.$$

Based on Corollary 2.1, $T_f^{(n)}$ is $\Theta(1)$ assignable.

Q.E.D.

Lemma 2.3 *Assume A to be a complete assignment set for an N-level $T_f^{(n)}$ $(N > 1)$, then*

$$\forall s \in [2, N]\, \{W_{s-1}(A) \subsetneq W_s(A)\} \tag{2.9}$$

holds if $T_f^{(n)}$ is not $\Theta(1)$ assignable.

Proof: Suppose $T_f^{(n)}$ is not $\Theta(1)$ assignable, and A is a complete assignment set for an N-level $T_f^{(n)}$. As mentioned, $W_s(A) \subset W_{s+1}(A)$ for all $s \in [2, N]$. If $W_s(A) = W_{s-1}(A)$ for an $s \in [2, N]$, then

$$\#W_s(A)/_\sim = \#W_{s-1}(A)/_\sim.$$

According to Lemma 2.2, $T_f^{(n)}$ is $\Theta(1)$ assignable. This contradicts the assumption directly.

Q.E.D.

Lemma 2.4 *For every complete assignment set A for an N-level $T_f^{(n)}$*

$$\forall s \in [2, N]\,\{\#W_s(A)/_\sim > s\} \tag{2.10}$$

holds if $T_f^{(n)}$ is not $\Theta(1)$ assignable.

Proof: Suppose $T_f^{(n)}$ is not $\Theta(1)$ assignable. According to Lemma 2.2,

$$\#W_1(A)/_\sim > 1 \wedge \forall s \in [2, N]\,\{\#W_s(A)/_\sim > \#W_{s-1}(A)/_\sim\}\,.$$

Therefore, $\#W_s(A)/_\sim > s$.

Q.E.D.

In order to prove Theorem 2.1 and 2.2, we define a set

$$P(M, t) := \left\{ (\underbrace{1, \cdots, 1}_{t_1}, \underbrace{2, \cdots, 2}_{t_2}, \cdots, \underbrace{m, \cdots, m}_{t_m})^T \;\middle|\; \sum_{1 \le i \le m} t_i = t \right\} \tag{2.11}$$

Every t-component vector in $\mathcal{D}_t$ is *similar* to a vector in $P(M, t)$.

Observation 2.3 *For every complete assignment set A ($\#A = t$) to an N-level $T_f^{(n)}$*

$$\forall \vec{I'} \in W_N(A) \exists \vec{I} \in P(M, t) \left\{ \vec{I'} \sim \vec{I} \right\} \;\wedge\; \#W_N(A)/_\sim \le \#P(M, t) \tag{2.12}$$

Theorem 2.1 *$T_f^{(n)}$ is $\Theta(1)$ assignable if and only if there are a $t \in \mathbf{N}$ and a set $W \subset \mathcal{D}_t$ so that*

$$\forall \vec{I_0} \in W \exists \vec{I_1}, \vec{I_2}, \cdots, \vec{I_k} \in W \left\{ M^k \subset \nabla(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \;\wedge\; \tilde{f}(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \sim \vec{I_0} \right\} \tag{2.13}$$

Proof: The *if* part follows Corollary 2.1 directly.

Assume $T_f^{(n)}$ to be $\Theta(1)$ assignable. Then there is a constant $t \in \mathbf{N}$, and one can construct a complete assignment set A of t patterns for an arbitrary $T_f^{(n)}$. Suppose $\lg n = N$ and $N > \#P(M, t)$. Since

$$\forall s \in [1, N]\,\{1 \le \#W_s(A)/_\sim \le \#P(M, t) < N\}\,,$$

there must be such an $s \in [2, N]$ that $\#W_s(A)/_\sim = \#W_{s-1}(A)/_\sim$. Thus we can state that there is an $s \in [1, N]$ so that

$$\forall \vec{I_0} \in W_s(A) \exists \vec{I_1}, \vec{I_2}, \cdots, \vec{I_k} \in W_s(A) \left\{ \begin{array}{c} M^k \subset \nabla(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \\ \tilde{f}(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \sim \vec{I_0} \end{array} \right\}.$$

Q.E.D

2.3 Jump from $\Theta(1)$ to $\Omega((\lg n)^{\frac{1}{m-1}})$

In this section we show that the assignment complexity of a $T_f^{(n)}$ is either $\Theta(1)$ or $\Omega((\lg n)^{\frac{1}{m-1}})$. In other words, there is a jump from $\Theta(1)$ to $\Omega((\lg n)^{\frac{1}{m-1}})$.

Coming up next we explore the upper boundary of $\#P(M,t)$.

Lemma 2.5 *For $\#M = m$,*

$$\#P(M,t) = \begin{pmatrix} t+m-1 \\ m-1 \end{pmatrix} \tag{2.14}$$

Proof: Let $p(\#M,t)$ denote $\#P(M,t)$. We prove this lemma by using induction on m, which is the cardinal number of M.

In case $m=1$, $p(1,t) = \begin{pmatrix} t \\ 0 \end{pmatrix}$ for all $t \in \mathbf{N}$. Suppose $p(m,t) = \begin{pmatrix} t+m-1 \\ m-1 \end{pmatrix}$ for $m \leq i$ and all $t \in \mathbf{N}$. For $m = i+1$, based on the inductive assumption in the last step.

$$\begin{aligned} p(m,t) &= p(i+1,t) \\ &= \sum_{0\leq j\leq t} p(i,t-j) \\ &= \sum_{0\leq j\leq t} \begin{pmatrix} t-j+i-1 \\ i-1 \end{pmatrix} \\ &= \sum_{i-1\leq j\leq t+i-1} \begin{pmatrix} j \\ i-1 \end{pmatrix} \\ &= \begin{pmatrix} t+i \\ i \end{pmatrix} \\ &= \begin{pmatrix} t+m-1 \\ m-1 \end{pmatrix}. \end{aligned}$$

Q.E.D.

Theorem 2.2 *$T_f^{(n)}$ is either $\Theta(1)$ or $\Omega((\lg n)^{\frac{1}{m-1}})$ assignable.*

Proof: Suppose $T_f^{(n)}$ is not $\Theta(1)$ assignable. It suffices to show that $\#A = \Omega((\lg n)^{\frac{1}{m-1}})$ for every complete assignment set A of $T_f^{(n)}$.

Assume $T_f^{(n)}$ to be of N levels. According to Lemma 2.4 and Observation 2.3,

$$N < \#W_N(A)/_\sim \leq \#P(M,t) \tag{2.15}$$

for every complete assignment set A. According to Lemma 2.5,

$$N < \begin{pmatrix} t+m-1 \\ m-1 \end{pmatrix}.$$

Then $t \geq N^{\frac{1}{m-1}} - m$ for $m > 1$. We know $N = \lceil \lg n \rceil$. Thus we can conclude that

$$\begin{aligned} t &= \Omega(N^{\frac{1}{m-1}}) \\ &= \Omega((\lg n)^{\frac{1}{m-1}}). \end{aligned}$$

Q.E.D.

The parameter m in Theorem 2.2 is the cardinality of M. For $M = \{0, 1\}$, the parameter m is 2. The following corollary is immediate from Observation 2.1 and Theorem 2.2.

Corollary 2.2 *Assume f to be a surjective function from $\{0,1\}^k$ to $\{0,1\}$. Then $T_f^{(n)}$ is either $\Theta(1)$ or $\Theta(\lg n)$ assignable.*

2.4 Problem Conversion

Theorem 2.1 gives a criterion of judging $\Theta(1)$ assignable uniform trees. And Theorem 2.2 explores the structure of assignment complexity. In this section, we give a new criterion for deciding the assignment complexity and convert the assignment problem of balanced uniform trees into the algebraic problem for exploring its aspects further.

We use $\vec{\mathbf{0}}$ to denote the all-zero vector and $\vec{\mathbf{1}}$ the all-one vector. Assume that L is a matrix, and $\vec{x}$, $\vec{b}$, $\vec{y}$, and $\vec{c}$ are vectors. When notations like

$$L\vec{x} = \vec{b}, \qquad \vec{y}L = \vec{c}$$

are used, we implicitly assume the compatibility of sizes and forms of L, $\vec{x}$, $\vec{y}$, and $\vec{c}$. If L is an $m \times n$-matrix, then $\vec{x}$ is a column vector with n components, $\vec{b}$ is a column vector with m components, $\vec{y}$ is a row vector of dimension m, and $\vec{c}$ is a row vector of dimension n. We define

$$\begin{aligned} (x_1, x_2, \cdots, x_n) \geq (y_1, y_2, \cdots, y_n) &\iff \forall i \in [1, n]\,\{x_i \geq y_i\} \\ (x_1, x_2, \cdots, x_n) > (y_1, y_2, \cdots, y_n) &\iff \begin{array}{c} (x_1, x_2, \cdots, x_n) \geq (y_1, y_2, \cdots, y_n) \\ \exists i \in [1, n]\,\{x_i > y_i\} \end{array} \\ |(x_1, x_2, \cdots, x_n)| &= \sum_{1 \leq i \leq n}^{n} |x_i| \end{aligned}$$

Assume that $s = \#M^k$ and $P_j = (p_j^{(1)}, p_j^{(2)}, \cdots, p_j^{(k)})$ denotes the jth element of M^k. Let Π_l ($l \in [1, k]$) be a projection of the lth component of P_j. For instance, $\Pi_l(P_j) = p_j^{(l)}$. Parameters $b_{ij}^{(l)}$ ($i \in [1, m], j \in [1, s], l \in [0, k]$) are defined as follows:

$$b_{ij}^{(0)} = \begin{cases} 1 & : \ f(P_j) = i \\ 0 & : \ \text{otherwise} \end{cases}, \qquad b_{ij}^{(l)} = \begin{cases} 1 & : \ \Pi_l(P_j) = i \\ 0 & : \ \text{otherwise} \end{cases}.$$

By using the above parameters we construct $k+1$ matrices $B^{(l)} = \left(b_{ij}^{(l)}\right)_{m \times s}$ ($l \in [0, k]$). It is obvious that every column of these matrices has only one nonzero element. Given an $i \in M$, M^k includes m^{k-1} elements P_j satisfying $\Pi_l(P_j) = i$. Hence, every row of $B^{(l)}$ ($l \in [1,k]$) has m^{k-1} nonzero components.

Before giving a new criterion for deciding the assignment complexity of uniform trees, we define the following terminology.

Two mappings

$$\begin{aligned} G &: \underbrace{\mathcal{D} \times \cdots \times \mathcal{D}}_{k} \longrightarrow \underbrace{\mathbf{N}_0 \times \cdots \times \mathbf{N}_0}_{s} \\ \mathcal{G} &: \underbrace{\mathbf{N}_0 \times \cdots \times \mathbf{N}_0}_{s} \longrightarrow \underbrace{\mathcal{D} \times \cdots \times \mathcal{D}}_{k} \end{aligned}$$

are defined as follows:

$$\begin{aligned} G(\vec{I}_1, \vec{I}_2, \cdots, \vec{I}_k) &= (x_1, x_2, \cdots, x_s)^T, \quad \text{if} \\ (\vec{I}_1, \vec{I}_2, \cdots, \vec{I}_k) &\sim (\underbrace{P_1, \cdots, P_1}_{x_1}, \underbrace{P_2, \cdots, P_2}_{x_2}, \cdots, \underbrace{P_s, \cdots, P_s}_{x_s})^T \,|_{\sum_{i=1}^{s} x_i = t} \end{aligned} \tag{2.16}$$

$$\mathcal{G}(x_1, x_2, \cdots, x_s) = (\underbrace{P_1, \cdots, P_1}_{x_1}, \underbrace{P_2, \cdots, P_2}_{x_2}, \cdots, \underbrace{P_s, \cdots, P_s}_{x_s})^T \tag{2.17}$$

Assume $\vec{I}_0 = \tilde{f}(\vec{I}_1, \vec{I}_2, \cdots, \vec{I}_k)$ for $(\vec{I}_1, \vec{I}_2, \cdots, \vec{I}_k) \in \mathcal{D}_t^k$. We call $B^{(l)}G(\vec{I}_1, \vec{I}_2, \cdots, \vec{I}_k)$ *characteristic vector* of $\vec{I}_l$ ($l \in [0,k]$), and use $Ch(\vec{I}_l)$ to denote it. Vector $\vec{I}_l$ belongs to $\mathcal{D}_t$, and its characteristic vector $Ch(\vec{I}_l)$ belongs to $\mathbf{N}_0^m$. We have the following observation.

Observation 2.4 *Given* $(\vec{I}_1, \vec{I}_2, \cdots, \vec{I}_k) \in \mathcal{D}_t^k$. *If*

$$\begin{aligned} Ch(\vec{I}_l) &= B^{(l)}G(\vec{I}_1, \vec{I}_2, \cdots, \vec{I}_k) \\ &= \left(c_1^{(l)}, c_2^{(l)}, \cdots, c_m^{(l)}\right)^T, \quad l \in [0,k], \end{aligned}$$

then

$$\forall l \in [0,k] \left\{ \vec{I}_l \sim (\underbrace{1, \cdots, 1}_{c_1^{(l)}}, \underbrace{2, \cdots, 2}_{c_2^{(l)}}, \cdots, \underbrace{m, \cdots, m}_{c_m^{(l)}})^{\mathrm{T}} \right\} \tag{2.18}$$

Theorem 2.3 $T_f^{(n)}$ *is* $\Theta(1)$ *assignable if and only if there is a finite set*

$$X = \{\vec{X}_i \mid \vec{X}_i \in \mathbf{N}^s\}$$

so that

$$\forall l \in [1,k] \{S_l(X) \subset S_0(X)\} \tag{2.19}$$

where

$$S_l(X) := \{B^{(l)}\vec{X}_i \mid \vec{X}_i \in X\}, \quad l \in [0,k] \tag{2.20}$$

Proof: We prove the *only if* part at first. Assume $T_f^{(n)}$ to be $\Theta(1)$ assignable. According to Theorem 2.1, there are a $t \in \mathbf{N}$ and a set $W \subset \mathcal{D}_t$ so that

$$\forall \vec{I_0} \in W \exists \vec{I_1}, \vec{I_2}, \cdots, \vec{I_k} \in W \left\{ M^k \subset \nabla(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \wedge \tilde{f}(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \sim \vec{I_0} \right\}.$$

We construct such a complete assignment set for $T_f^{(n)}$ that every $\vec{I_0} \in W$ is the output vector of a cell in $T_f^{(n)}$, and the input vector of another cell as well. Let X be the smallest set of vectors that includes every $G(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k})$ if $(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k})$ is the complete assignment for an f cell in $T_f^{(n)}$. Then $S_l(X) \subset S_0(X)$ for every $l \in [1, k]$.

Now we prove the *if* part. Suppose there is a finite set $X \subset \mathbf{N}^s$ and $S_l(X) \subset S_0(X)$ for every $l \in [1, k]$. Let $W' = \left\{ \tilde{f}(\mathcal{G}(\vec{X_i})) \mid \vec{X_i} \in X \right\}$. It is easy to show that

$$\forall \vec{I_0} \in W' \exists \vec{I_1}, \vec{I_2}, \cdots, \vec{I_k} \in W' \exists \vec{I'_1}, \vec{I'_2}, \cdots, \vec{I'_k} \in \mathcal{D}_t \left\{ \begin{array}{c} M^k \subset \nabla(\vec{I'_1}, \vec{I'_2}, \cdots, \vec{I'_k}) \\ \forall l \in [1, k]\{\vec{I_l} \sim \vec{I'_l}\} \\ \tilde{f}(\vec{I'_1}, \vec{I'_2}, \cdots, \vec{I'_k}) \sim \vec{I_0} \end{array} \right\}.$$

Let W be the smallest set such that

$$\forall \vec{I} \in \mathcal{D}_t \left\{ \exists \vec{I'} \in W' \left\{ \vec{I'} \sim \vec{I} \right\} \Longrightarrow \vec{I} \in W \right\}.$$

Then

$$\forall \vec{I_0} \in W \exists \vec{I_1}, \vec{I_2}, \cdots, \vec{I_k} \in W \left\{ M^k \subset \nabla(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \wedge \tilde{f}(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \sim \vec{I_0} \right\}.$$

According to Theorem 2.1, $T_f^{(n)}$ is $\Theta(1)$ testable.

Q.E.D.

Corollary 2.3 $T_f^{(n)}$ *is not* $\Theta(1)$ *assignable if*

$$B^{(0)}\vec{y} = B^{(l)}\vec{y}, \quad \vec{y} \geq \vec{1} \tag{2.21}$$

has no feasible solution for an $l \in [1, k]$.

Proof: Assume $T_f^{(n)}$ to be $\Theta(1)$ assignable. According to Theorem 2.3 there is such a set $X \subset \mathbf{N}^s$ that (2.19) and (2.22) hold. We show that if (2.22) holds for an $l \in [1, k]$, then (2.21) has a feasible solution for the given l.

$$S_l(X) \subset S_0(X) \tag{2.22}$$

In case $\#X = 1$ and $X = \{\vec{y}\}$, $\vec{y}$ is just a feasible solution of (2.21). Assume that in case $\#X = N$, (2.21) has a feasible solution if (2.22) holds for the given $l \in [1, k]$. For $\#X = N + 1$, there are three cases to be considered.

Case 1, $\#S_l(X) = \#S_0(X) = \#X$.

Case 2, $\#S_l(X) = \#S_0(X) < \#X$.

Case 3, $\#S_l(X) < \#S_0(X)$.

For the first case, $\#S_l(X) = \#S_0(X) = \#X$ and $S_l(X) = S_0(X)$, then $\vec{y} = \sum_{\vec{X}_i \in X} \vec{X}_i$ is a solution of (2.21).

For the second case, there must be $\vec{X}_i, \vec{X}_j \in X$ so that $\vec{X}_i \neq \vec{X}_j$ and $B^{(0)}\vec{X}_i = B^{(0)}\vec{X}_j$. Thus

$$S_l(X \setminus \vec{X}_i) \subset S_0(X \setminus \vec{X}_i) = S_0(X).$$

$X \setminus \vec{X}_i$ satisfies (2.22) also, and its cardinal number is N. This implies that (2.21) has a feasible solution.

For the third case, X must include such an $\vec{X}_i$ that $B^{(0)}\vec{X}_i \notin S_l(X)$. This indicates that

$$\begin{aligned} S_l(X) &\subset S_0(X \setminus X_0) \\ S_l(X \setminus \vec{X}_i) &\subset S_0(X \setminus \vec{X}_i) \end{aligned}$$

$X \setminus \vec{X}_i$ satisfies (2.22) for the given l, and its cardinal number is N.

Q.E.D

In the rest of this section, we present two basic theorems in linear programming. They will be used in the next section and Chapter 3.

Theorem 2.4 (Farkas' Lemma) *Assume A to be an $s \times t$ matrix.*

$$A\vec{x} = \vec{b}, \qquad \forall j \in [1,t]\{x_j \geq 0\} \tag{2.23}$$

has feasible solutions if and only if

$$\forall \vec{y} \in \mathbf{R}^s \left\{ \vec{y}A \geq \vec{\mathbf{0}} \Longrightarrow \vec{y}\vec{b} \geq 0 \right\} \tag{2.24}$$

The proof of **Farkas' Lemma** can be found almost in every linear programming book.

Theorem 2.5 *If*

$$A\vec{x} = \vec{\mathbf{0}}, \qquad \vec{x} \geq \vec{c} \tag{2.25}$$

has a feasible solution, then it has a feasible integer solution, provided that the terms of the constraint matrix A are all integers, and every component of $\vec{c}$ belongs to $\mathbf{N}_0$.

Proof: Assume that A is an $s \times t$ integer matrix and its rank is r. For $r \leq s$, we can determine an $r \times t$ matrix A' including r independent rows. Then (2.25) and

$$A'\vec{x} = \vec{\mathbf{0}}, \qquad \vec{x} \geq \vec{c} \tag{2.26}$$

have the same solution space.

It is obvious that (2.26) has a feasible solution if and only if

$$A'\vec{x} = -A'\vec{c}, \qquad \vec{x} \geq \vec{\mathbf{0}} \tag{2.27}$$

has a feasible solution.

Suppose that $B = (b_{ij})_{r\times r}$ is a nonsingular submatrix of A'. Without loss of generality, assume that B includes the first r columns of A'. Thus

$$x_i = \begin{cases} \text{the } i\text{th component of } -B^{-1}A'\vec{c} & : \quad i \in [1, r] \\ 0 & : \quad i \in [r+1, t] \end{cases} \tag{2.28}$$

define a basic solution of (2.27). It is clear that such a basic solution is a rational solution since the terms of the constraint matrix A and the constant vector $\vec{c}$ are all integers.

It has been proven that at least one of its basic solutions is feasible if (2.27) has a feasible solution [PaSt82]. It implies that (2.27) has a feasible rational solution if and only if it has a feasible solution. According to the relationship between (2.27) and (2.26), $\vec{x}$ is a feasible solution of (2.27) if and only if $\vec{x} + \vec{c}$ is a feasible solution of (2.26).

Given a feasible rational solution of (2.26), we can always construct a feasible integer solution since (2.26) is a homogeneous linear equation system and no component of $\vec{c}$ is negative.

Q.E.D

2.5 Commutative Trees

The $k+1$ matrices $B^{(l)}$ ($l \in [0, k]$) defined in section 2.4 are determined completely by the function definition of f. For commutative function f we have the following result.

Theorem 2.6 *Assume surjective function $f : M^k \longrightarrow M$ to be commutative. Then $T_f^{(n)}$ is $\Theta(1)$ assignable if and only if*

$$\begin{bmatrix} B^{(0)} - B^{(1)} \\ B^{(0)} - B^{(2)} \\ \vdots \\ B^{(0)} - B^{(k)} \end{bmatrix} \vec{y} = \begin{bmatrix} B^{(1)} - B^{(0)} \\ B^{(2)} - B^{(0)} \\ \vdots \\ B^{(k)} - B^{(0)} \end{bmatrix} \vec{\mathbf{1}}, \qquad \vec{y} \geq \vec{\mathbf{0}} \tag{2.29}$$

has a feasible solution.

Proof: We prove the *if* part at first. Suppose (2.29) has a feasible solution. This means that

$$\begin{bmatrix} B^{(0)} - B^{(1)} \\ B^{(0)} - B^{(2)} \\ \vdots \\ B^{(0)} - B^{(k)} \end{bmatrix} \vec{y} = \vec{\mathbf{0}}, \qquad \vec{y} \geq \vec{\mathbf{1}} \tag{2.30}$$

has a feasible solution. Furthermore, it has a feasible integer solution according to Theorem 2.5. Suppose $\vec{y} \in \mathbf{N}^s$ is a feasible integer solution of (2.30). Let $X = \{\vec{y}\}$. Then $S_l(X) \subset S_0(X)$ for all $l \in [1, k]$. According to Theorem 2.3, $T_f^{(n)}$ is $\Theta(1)$ assignable.

Now we turn to the proof of the *only if* part. Based on Farkas' Lemma, (2.29) has a feasible solution if and only if

$$\forall \vec{z} \in \mathbf{R}^{km} \left\{ \vec{z} \begin{bmatrix} B^{(0)} - B^{(1)} \\ B^{(0)} - B^{(2)} \\ \vdots \\ B^{(0)} - B^{(k)} \end{bmatrix} \geq \vec{\mathbf{0}} \Longrightarrow \vec{z} \begin{bmatrix} B^{(1)} - B^{(0)} \\ B^{(2)} - B^{(0)} \\ \vdots \\ B^{(k)} - B^{(0)} \end{bmatrix} \vec{\mathbf{1}} \geq 0 \right\} \tag{2.31}$$

Suppose (2.29) has no feasible solution. This means that

$$\exists \vec{z} \in \mathbf{R}^{km} \left\{ \vec{z} \begin{bmatrix} B^{(0)} - B^{(1)} \\ B^{(0)} - B^{(2)} \\ \vdots \\ B^{(0)} - B^{(k)} \end{bmatrix} > \vec{\mathbf{0}} \right\} \tag{2.32}$$

Thus we can choose a $\vec{z}$ so that for every $\vec{y} \geq \vec{\mathbf{1}}$

$$\vec{z} \begin{bmatrix} B^{(0)} - B^{(1)} \\ B^{(0)} - B^{(2)} \\ \vdots \\ B^{(0)} - B^{(k)} \end{bmatrix} \vec{y} > k \tag{2.33}$$

This implies that for an arbitrary complete assignment $(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k})$ to an f cell

$$\vec{z} \begin{bmatrix} B^{(0)} - B^{(1)} \\ B^{(0)} - B^{(2)} \\ \vdots \\ B^{(0)} - B^{(k)} \end{bmatrix} G(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) > k \tag{2.34}$$

since $G(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \geq \vec{\mathbf{1}}$ for the complete assignment $(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k})$.

According to Observation 2.4

$$\begin{aligned}
Ch(\vec{I_1}) &= B^{(1)}G(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) = B^{(2)}G(\vec{I_k}, \vec{I_1}, \cdots, \vec{I_{k-1}}) = \cdots = B^{(k)}G(\vec{I_2}, \vec{I_3}, \cdots, \vec{I_1}) \\
Ch(\vec{I_k}) &= B^{(1)}G(\vec{I_k}, \vec{I_1}, \cdots, \vec{I_{k-1}}) = B^{(2)}G(\vec{I_{k-1}}, \vec{I_k}, \cdots, \vec{I_{k-2}}) = \cdots = B^{(k)}G(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \\
&\vdots \\
Ch(\vec{I_2}) &= B^{(1)}G(\vec{I_2}, \vec{I_3}, \cdots, \vec{I_1}) = B^{(2)}G(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) = \cdots = B^{(k)}G(\vec{I_3}, \vec{I_4}, \cdots, \vec{I_2}).
\end{aligned}$$

This indicates that

$$\vec{z} \begin{bmatrix} B^{(1)} \\ B^{(2)} \\ \vdots \\ B^{(k)} \end{bmatrix} \left\{ G(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) + G(\vec{I_k}, \vec{I_1}, \cdots, \vec{I_{k-1}}) + \cdots + G(\vec{I_2}, \vec{I_3}, \cdots, \vec{I_1}) \right\}$$

$$= \vec{z} \begin{bmatrix} B^{(1)} + B^{(2)} + \cdots + B^{(k)} \\ B^{(1)} + B^{(2)} + \cdots + B^{(k)} \\ \vdots \\ B^{(1)} + B^{(2)} + \cdots + B^{(k)} \end{bmatrix} G(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}).$$

Assume $\vec{I_0} = \tilde{f}(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k})$. For commutative function f

$$\vec{I_0} = \tilde{f}(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) = \tilde{f}(\vec{I_k}, \vec{I_1}, \cdots, \vec{I}_{k-1}) = \cdots = \tilde{f}(\vec{I_2}, \vec{I_3}, \cdots, \vec{I_1}),$$

hence

$$Ch(\vec{I_0}) = B^{(0)}G(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) = B^{(0)}G(\vec{I_k}, \vec{I_1}, \cdots, \vec{I}_{k-1}) = \cdots = B^{(0)}G(\vec{I_2}, \vec{I_3}, \cdots, \vec{I_1}).$$

According to (2.34),

$$\vec{z} \begin{bmatrix} B^{(0)} \\ B^{(0)} \\ \vdots \\ B^{(0)} \end{bmatrix} kG(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \geq \vec{z} \begin{bmatrix} B^{(1)} + B^{(2)} + \cdots + B^{(k)} \\ B^{(1)} + B^{(2)} + \cdots + B^{(k)} \\ \vdots \\ B^{(1)} + B^{(2)} + \cdots + B^{(k)} \end{bmatrix} G(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) + k^2.$$

Thus we can state that if $T_f^{(n)}$ is not $\Theta(1)$ assignable, then

$$\exists \vec{z} \in \mathbf{R}^m \left\{ k\vec{z}B^{(0)}G(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \geq \vec{z}\left(B^{(1)} + B^{(2)} + \cdots + B^{(k)}\right) G(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) + k \right\}$$

holds for every complete assignment $(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k})$ to an f cell. In other words, if $T_f^{(n)}$ is not $\Theta(1)$ assignable then there is such a $\vec{z} \in \mathbf{R}^m$ that for every complete assignment $(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k})$ to an f cell

$$\exists i \in [1, k] \left\{ \vec{z}B^{(0)}G(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) \geq \vec{z}B^{(i)}G(\vec{I_1}, \vec{I_2}, \cdots, \vec{I_k}) + 1 \right\} \tag{2.35}$$

Suppose $T_f^{(n)}$ has k^N primary input lines. We determine a path, called *downhill path*, from the primary output line to a primary input line by using the following procedure.

1. Choose the cell with the primary output line as the first cell on the downhill path, and let $(\vec{I}_{1,1}, \vec{I}_{1,2}, \cdots, \vec{I}_{1,k})$ denote the complete assignment set to this cell. Then

$$\vec{z}B^{(0)}G(\vec{I}_{1,1}, \vec{I}_{1,2}, \cdots, \vec{I}_{1,k}) \geq \vec{z}B^{(0)}G(\vec{I}_{1,1}, \vec{I}_{1,2}, \cdots, \vec{I}_{1,k}) + 1 - 1.$$

2. Let $(\vec{I}_{l,1}, \vec{I}_{l,2}, \cdots, \vec{I}_{l,k})$ denote the complete assignment to the lth cell on the downhill path, and suppose

$$\vec{z}B^{(0)}G(\vec{I}_{1,1}, \vec{I}_{1,2}, \cdots, \vec{I}_{1,k}) \geq \vec{z}B^{(0)}G(\vec{I}_{l,1}, \vec{I}_{l,2}, \cdots, \vec{I}_{l,k}) + l - 1.$$

3. According to (2.35) we can always obtain such an i that

$$\vec{z}B^{(0)}G(\vec{I}_{l,1}, \vec{I}_{l,2}, \cdots, \vec{I}_{l,k}) \geq \vec{z}B^{(i)}G(\vec{I}_{l,1}, \vec{I}_{l,2}, \cdots, \vec{I}_{l,k}) + 1.$$

We choose the cell linked directly to the ith input line of the lth cell as the $(l+1)$th cell on the downhill path. $(\vec{I}_{l+1,1}, \vec{I}_{l+1,2}, \cdots, \vec{I}_{l+1,k})$ is the complete assignment to this cell. We can state that

$$\begin{aligned} \vec{z}B^{(0)}G(\vec{I}_{1,1}, \vec{I}_{1,2}, \cdots, \vec{I}_{1,k}) &\geq \vec{z}B^{(0)}G(\vec{I}_{l,1}, \vec{I}_{l,2}, \cdots, \vec{I}_{l,k}) + l - 1 \\ &\geq \vec{z}B^{(i)}G(\vec{I}_{l,1}, \vec{I}_{l,2}, \cdots, \vec{I}_{l,k}) + l \\ &= \vec{z}B^{(0)}G(\vec{I}_{l+1,1}, \vec{I}_{l+1,2}, \cdots, \vec{I}_{l+1,k}) + l. \end{aligned}$$

In this way, we can finally determine the Nth cell on the downhill path. Suppose $(\vec{I}_{N,1}, \vec{I}_{N,2}, \cdots, \vec{I}_{N,k})$ is the complete assignment set to this cell. According to the above calculation

$$\vec{z}B^{(0)}G(\vec{I}_{1,1}, \vec{I}_{1,2}, \cdots, \vec{I}_{1,k}) \geq \vec{z}B^{(0)}G(\vec{I}_{N,1}, \vec{I}_{N,2}, \cdots, \vec{I}_{N,k}) + N - 1.$$

Let $|\vec{y}|$ denote the sum of the absolute values of the components of $\vec{y}$. Then $|G(\vec{I}_{1,1}, \vec{I}_{1,2}, \cdots, \vec{I}_{1,k})|$ is the cardinal number of the complete assignment set to $T_f^{(N)}$.

$$\begin{aligned} \left|G(\vec{I}_{1,1}, \vec{I}_{1,2}, \cdots, \vec{I}_{1,k})\right| &\geq \frac{N}{|\vec{z}B^{(0)}|} \\ &= \Omega(N). \end{aligned}$$

We know that $N = \lceil \lg n \rceil$, and every $T_f^{(n)}$ is $O(\lg n)$ assigbable. Therefore, $T_f^{(n)}$ is $\Theta(\lg n)$ assignable.

Q.E.D.

The following Corollary is immediate from the above theorem.

Corollary 2.4 *Assume $f : M^k \longrightarrow M$ to be commutative, then $T_f^{(n)}$ is either $\Theta(1)$ or $\Theta(\lg n)$ assignable.*

Assume f to be commutative and $T_f^{(n)}$ $\Theta(1)$ assignable. The problem of searching for the minimum complete assignment set for $T_f^{(n)}$ is related to solving the following integer programming.

$$\min \sum_{1 \leq i \leq s} y_i$$

$$\begin{bmatrix} B^{(0)} - B^{(1)} \\ B^{(0)} - B^{(2)} \\ \vdots \\ B^{(0)} - B^{(k)} \end{bmatrix} \vec{y} = \begin{bmatrix} B^{(1)} - B^{(0)} \\ B^{(2)} - B^{(0)} \\ \vdots \\ B^{(k)} - B^{(0)} \end{bmatrix} \vec{\mathbf{1}}, \quad \vec{y} \geq \vec{\mathbf{0}} \tag{2.36}$$

The following theorem gives an upper bound of $\min \sum_{1 \leq i \leq s} y_i$.

Theorem 2.7 *The cardinality of the minimum complete assignment set for a $\Theta(1)$ assignable f tree system $T_f^{(n)}$ can be upper bounded by $k^2 m^{k-1}(2k)^{km-k-1}(m-1)^3 + m^k$, provided that f is a commutative function from M^k to M, and $\#M = m$.*

In order to prove this theorem, we prove the following lemma at first.

Lemma 2.6 *Assume integer matrix $B = (b_{ij})_{r \times r}$ to be of rank r, $\vec{c} = (c_1, ..., c_r)^T$.*

$$\begin{aligned} \max\{\sum_{1 \leq i \leq r} |b_{ij}| \; |j \in [1, r]\} &= \alpha \\ \max\{|c_i| \; |i \in [1, r]\} &= \beta \\ B\vec{y} &= \vec{c}. \end{aligned}$$

The absolute value of every component of $\vec{y}$ can be upper bounded by $r\alpha^{r-1}\beta$.

Proof of Lemma 2.6: Let B_{ij} denote the submatrix of B that is generated by omitting the ith row and the jth column of B. Then $det\ B_{ij}$ and $(-1)^{i+j}\ det\ B_{ij}$ are the minor and cofactor of the element b_{ij} in $det\ B$.

We show inductively that the absolute value of the determinant of a matrix $B = (b_{ij})_{l\times l}$, denoted by $det\ B$, is not greater than α^l.

For $l = 1$, $B = (b_{ij})_{1\times 1}$ and $|det\ B| \leq \alpha$.

Assume that for $l = N-1$, $|det\ B| \leq \alpha^{N-1}$.

For $l = N$, $B = (b_{ij})_{N\times N}$. By expanding the determinant of B along the first column of B, we have

$$\begin{aligned} |det\ B| &\leq \sum_{1\leq i\leq N} |b_{i1}B_{i1}| \\ &\leq \alpha^N. \end{aligned}$$

Let b'_{ij} stand for $\frac{(-1)^{i+j}\ det\ B_{ij}}{det\ B}$, then $B^{-1} = (b'_{ij})_{r\times r}$ and $|b'_{ij}| = \left|\frac{det\ B_{ij}}{det\ B}\right|$. Thus

$$\begin{aligned} |y_i| &= \left|\sum_{1\leq j\leq r} b'_{ij}c_j\right| \\ &\leq \sum_{1\leq j\leq r} \left|b'_{ij}c_j\right| \end{aligned}$$

For B_{ij} is an $(r-1)\times(r-1)$ matrix, $|det\ B_{ij}| \leq \alpha^{r-1}$. Furthermore, $|det\ B| \geq 1$, since b_{ij} $(i, j \in [1, r])$ are all integers. Hence $|b'_{ij}|$ is not greater than α^{r-1}. We have thus

$$|y_i| \leq r\alpha^{r-1}\beta.$$

□

Proof of Theorem 2.7: Let

$$\begin{aligned} B &= (b_{ij})_{km\times s} = \begin{bmatrix} B^{(0)} - B^{(1)} \\ B^{(0)} - B^{(2)} \\ \vdots \\ B^{(0)} - B^{(k)} \end{bmatrix} \\ \vec{c} &= (c_1, c_2, ..., c_{km})^T = -\begin{bmatrix} B^{(0)} - B^{(1)} \\ B^{(0)} - B^{(2)} \\ \vdots \\ B^{(0)} - B^{(k)} \end{bmatrix} \vec{1} \end{aligned}$$

It is easy to see that

$$\begin{aligned} b_{ij} = b^{(0)}_{lj} - b^{(t)}_{lj}, \quad i = (t-1)m + l, \quad l \in [1, m], \quad t \in [1, k] \\ c_i = -\sum_{j=1}^{s} b_{ij}, \quad i \in [1, km]. \end{aligned}$$

Then the linear programming (2.36) is the same as

$$\begin{aligned} \min \quad & \sum_{1 \leq \leq s} y_i \\ B\vec{y} \quad = \quad & \vec{c}, \quad \vec{y} \geq \vec{\mathbf{0}} \end{aligned} \tag{2.37}$$

Suppose the rank of the constraint matrix in (2.37) is r. According to the definitions of $B^{(0)}$ and $B^{(l)}$ ($l \in [1,k]$) given in section 2.4, $r \leq km - k$ since matrix $B^{(0)} - B^{(l)}$ can contain at the most $m-1$ independent row vectors for every $l \in [1,k]$. By exchanging the rows of B we can make the first r rows independent. Furthermore, assume, without loss of generality, the $r \times r$ submatrix, denoted by B_r, in the left top corner to be nonsingular. We solve

$$B_r\vec{y} = \vec{c}, \qquad \vec{c} = (c_1, c_2, ..., c_r)^T.$$

Vector $\vec{y}$ defined below is a basic solution of (2.37).

$$y_i = \begin{cases} \text{the } i\text{th component of} \quad B_r^{-1}\vec{c} & : \quad i \in [1,r] \\ 0 & : \quad i \in [r+1,t] \end{cases}$$

It is well known that at least one of the basic solutions of (2.37) is a feasible solution, provided that (2.37) has a feasible solution [PaSt82].

Given an $i = (t-1)m + l$, ($t \in [1,k], l \in [1,m]$), according to the definition of b_{ij}, $b_{ij} < 0$ only if $\Pi_t(P_j) = l \neq f(P_j)$. There are at most m^{k-1} indices $j \in [1,s]$ so that $b_{ij} < 0$. Inversely, $b_{ij} > 0$ only if $\Pi_t(P_j) \neq l = f(P_j)$. There are at most $m^k - m^{k-1}$ indices $j \in [1,s]$ so that $b_{ij} > 0$. Therefore, none of the absolute values of c_i ($i \in [1,km]$) is greater than $m^k - m^{k-1}$. That is, $\max\{c_i \mid i \in [1,km]\} \leq m^k - m^{k-1}$. The sum of the absolute values of all components in any column of B_r is not greater than $2k$. Based on the above lemma, $y_i \leq (km-k)(2k)^{km-k-1}(m^k - m^{k-1})$ since $r \leq km-k, \alpha \leq 2k$ and $\beta \leq m^k - m^{k-1}$. The sum of all y_i is not greater than $k^2m^{k-1}(2k)^{km-k-1}(m-1)^3$. Thus the cardinality of the minimum complete assignment set for $T_f^{(n)}$ can be upper bounded by the sum of $k^2m^{k-1}(2k)^{km-k-1}(m-1)^3$ and m^k.

Q.E.D.

2.6 Decidability

We have proven that a balanced uniform tree is either $O(1)$ or $\Omega((\lg n)^{\frac{1}{m-1}})$ assignable. In this section we discuss the complexity of deciding the assignment complexity, namely, the decidability. However, our discussion will be limited to commutative functions of two variables.

Assume f to be a commutative function from M^2 to M. Let $t = \#M^2$ and $P_j := (p_{j,l}, p_{j,r})$ be the jth element of M^2. We define

$$l_{ij} = \begin{cases} 1 & : \quad f(P_j) = i \neq p_{j,l} \\ -1 & : \quad f(P_j) \neq i = p_{j,l} \\ 0 & : \quad \text{otherwise} \end{cases} \qquad r_{ij} = \begin{cases} 1 & : \quad f(P_j) = i \neq p_{j,r} \\ -1 & : \quad f(P_j) \neq i = p_{j,r} \\ 0 & : \quad \text{otherwise} \end{cases}$$

By using the above parameters we construct two matrices $L_f = (l_{ij})_{m \times t}$ and $R_f = (r_{ij})_{m \times t}$. It is easy to see that $L_f = B^{(0)} - B^{(1)}$, and $R_f = B^{(0)} - B^{(2)}$. Based on Theorem 2.6 we can state that $T_f^{(n)}$ is $\Theta(1)$ assignable if and only if

$$\begin{bmatrix} L_f \\ R_f \end{bmatrix} \vec{y} = -\begin{bmatrix} L_f \\ R_f \end{bmatrix} \vec{\mathbf{1}}, \quad \vec{y} \geq \vec{\mathbf{0}} \tag{2.38}$$

has a feasible solution.

Let $\vec{l_j}$ and $\vec{r_j}$ denote the jth column of L_f and R_f, respectively.

Observation 2.5 *If f is a commutative function, then*

1. $\forall i \in [1,t] \exists j \in [1,t] \left\{ \vec{l_i} = \vec{r_j} \ \wedge \ \vec{r_i} = \vec{l_j} \right\}$.

2. *Every nonzero column of L_f and R_f has only two nonzero element. One of them is* 1, *another is* -1.

3. $\forall \vec{z} \in \mathbf{R}^m \forall j \in [1,t] \left\{ \vec{z}\vec{l_j} = (\vec{z} + \vec{\mathbf{1}})\vec{l_j} \ \wedge \ \vec{z}\vec{r_j} = (\vec{z} + \vec{\mathbf{1}})\vec{r_j} \right\}$.

In the following we will give a mothed of deciding whether(2.38) has a feasible solution.

Using the definition of f, we induce a digraph $G_f = (V, E)$ as follows:

$$V = \{i \mid i \in M\}, \qquad E = \{(i,j) \mid \exists k \in M \wedge f(i,k) = j\}$$

where (i,j) represents an arc from i to j.

It is not hard to see that the matrix L_f defined above is just the node-arc incident matrix of G_f. For commutative function f

$$\forall i,j \in M \ \exists k \in M \left\{ f(i,j) = f(j,i) = k \right\}.$$

This implies that

$$\forall i,j \in V \ \exists k \in V \left\{ (i,k),(j,k) \in E \right\}.$$

That is, two arbitrary vertices $i, j \in V$ are connected. G_f is a connected graph. We call G_f a strongly connected digraph, provided that for arbitrary $i_1, i_2 \in V$ there is at least a cyclical path leaving i_1 and entering i_2, then leaving i_2 and entering i_1. There exists a cyclical path traveling all vertexes in V if G_f is a strongly connected digraph.

Lemma 2.7 *Equation system* (2.38) *has a feasible solution if G_f is a strongly connected digraph.*

Proof: Suppose G_f is a strongly connected digraph. We know that

$$\exists \vec{z} \in \mathbf{R}^{2m} \left\{ \vec{z} \begin{bmatrix} L_f \\ R_f \end{bmatrix} \geq \vec{\mathbf{0}} \right\} \iff \exists \vec{z_1}, \vec{z_2} \in \mathbf{R}^m \left\{ \vec{z_1} L_f + \vec{z_2} R_f \geq \vec{\mathbf{0}} \right\}$$

Assume that

$$\exists \vec{z_1}, \vec{z_2} \in \mathbf{R}^m \left\{ \vec{z_1} L_f + \vec{z_2} R_f \geq \vec{\mathbf{0}} \right\}.$$

According to the first term of Observation 2.5

$$\forall \vec{z_1}, \vec{z_2} \in \mathbf{R}^m \forall i \in [1,t] \exists j \in [1,t] \left\{ \vec{z_1}\vec{l_i} + \vec{z_2}\vec{r_i} = \vec{z_2}\vec{l_j} + \vec{z_1}\vec{r_j} \right\}.$$

Hence

$$\forall \vec{z_1}, \vec{z_2} \in \mathbf{R}^m \left\{ \vec{z_1}L_f + \vec{z_2}R_f \geq \vec{\mathbf{0}} \Longleftrightarrow \vec{z_2}L_f + \vec{z_1}R_f \geq \vec{\mathbf{0}} \Longleftrightarrow (\vec{z_1}+\vec{z_2})[L_f + R_f] \geq \vec{\mathbf{0}} \right\}$$

$$\forall \vec{z_1}, \vec{z_2} \in \mathbf{R}^m \left\{ \vec{z_1}L_f + \vec{z_2}R_f \leq \vec{\mathbf{0}} \Longleftrightarrow \vec{z_2}L_f + \vec{z_1}R_f \leq \vec{\mathbf{0}} \Longleftrightarrow (\vec{z_1}+\vec{z_2})[L_f + R_f] \leq \vec{\mathbf{0}} \right\}$$

Let $(\vec{z_1}+\vec{z_2}) = (z_1, z_2, ..., z_m)$, and $c = \min\{z_i \mid i \in [1,m]\}$. Without loss of generality, suppose $z_1 = c$.

According to the third term of Observation 2.5

$$\begin{aligned} (z_1, z_2, ..., z_m)[L_f + R_f] \geq \vec{\mathbf{0}} \quad &\Longrightarrow \quad (0, z_2 - c, ..., z_m - c)[L_f + R_f] \geq \vec{\mathbf{0}} \\ &\Longrightarrow \quad \forall j \in [1,t] \left\{ (0, z_2 - c, ..., z_m - c)(\vec{l_j} + \vec{r_j}) \geq 0 \right\}. \end{aligned}$$

Since G_f is a strongly connected digraph, there is a cyclical path

$$1 \to i_1 \to i_2 \to \cdots \to i_k \to 1, \quad i_1, i_2, ..., i_k \in M$$

including all vertices of G_f. And $(i_k, 1)$ is an arc from vertex i_k to 1. This indicates that there is an $l \in M$ such that

$$f(i_k, l) = f(l, i_k) = 1 \neq i_k.$$

This implies that there is a $j \in [1,t]$ so that $P_j = (l, i_k)$ and

$$l_{1j} + r_{1j} \geq 1, \quad l_{i_k j} + r_{i_k j} \leq -1, \quad \forall i \in M \setminus \{1, i_k\} \{l_{ij} + r_{ij} \leq 0\}.$$

Then $z_{i_k} - c$ is equal to zero. Otherwise, $\vec{z}(\vec{l_j} + \vec{r_j}) < 0$, and $\vec{z}[L_f + R_f] \geq \vec{\mathbf{0}}$ could not hold. We can similarly show that $z_i - c = 0$ for all $i \in [1,m]$. In other words, $\vec{z} = (\underbrace{c, \cdots, c}_{m})$.

This indicates that

$$(\vec{z_1}+\vec{z_2})[L_f + R_f] = \vec{z_1}L_f + \vec{z_2}R_f = \vec{z_2}L_f + \vec{z_1}R_f = \vec{\mathbf{0}}.$$

and

$$\forall \vec{z} \in \mathbf{R}^m \left\{ \vec{z}[L_f + R_f] \geq \vec{\mathbf{0}} \Longrightarrow \vec{z} = \vec{\mathbf{0}} \right\}$$

Therefore, we can state that

$$\forall \vec{z} \in \mathbf{R}^{2m} \left\{ \vec{z} \begin{bmatrix} L_f \\ R_f \end{bmatrix} \geq \vec{\mathbf{0}} \Longrightarrow \vec{z} \begin{bmatrix} L_f \\ R_f \end{bmatrix} = \vec{\mathbf{0}} \right\}$$

and

$$\forall \vec{z} \in \mathbf{R}^{2m} \left\{ \vec{z} \begin{bmatrix} L_f \\ R_f \end{bmatrix} \geq \vec{\mathbf{0}} \Longrightarrow -\vec{z} \begin{bmatrix} L_f \\ R_f \end{bmatrix} \vec{\mathbf{1}} \geq 0 \right\} \tag{2.39}$$

hold if $G_f(V, E)$ is a strongly connected digraph. Following Farkas' Lemma, (2.38) has a feasible solution.

Q.E.D.

Lemma 2.8 *If G_f is not a strongly connected digraph, then there is a $\vec{z} \in \mathrm{N}_0^{\mathrm{m}}$ so that $\vec{z}L_f > \vec{\mathbf{0}}$, and* (2.38) *has no feasible solution*

Proof: Suppose G_f is not a strongly connected digraph. Then there are $i, j \in V$ so that i can reach j, but j can not reach i since G_f is a connected digraph. Let V_j be the largest set of vertices which j can reach, and $V_i = V \setminus V_j$. Thus there is no arc from V_j to V_i but at least one arc from V_i to V_j. Let L_f be the node-arc incident matrix of G_f. Every row of L_f is related to a vertex of V. We construct an m-component vector $\vec{z}$ and make each of its components correspond to a row of L_f. Then every component of $\vec{z}$ is related to a vertex in V. We set all the components related to vertices in V_j to be 1 and make others 0. For such a $\vec{z}$, $\vec{z}L_f > \vec{\mathbf{0}}$ holds. This implies that

$$\exists \vec{z} \in \mathbf{N}_0{}^m \left\{ \vec{z}L_f > \vec{\mathbf{0}} \right\}$$

and

$$\exists \vec{z} \in \mathbf{N}_0{}^{2m} \left\{ \vec{z} \begin{bmatrix} L_f \\ R_f \end{bmatrix} > \vec{\mathbf{0}} \ \wedge \ -\vec{z} \begin{bmatrix} L_f \\ R_f \end{bmatrix} \vec{\mathbf{1}} < 0 \right\}$$

This means that (2.38) has no feasible solution.

Q.E.D.

Corollary 2.5 *The integer linear programming* (2.38) *has feasible solutions if and only if*

$$\forall \vec{z} \geq \vec{\mathbf{0}} \left\{ \vec{z}L_f \geq \vec{\mathbf{0}} \Longrightarrow \vec{z}L_f = \vec{\mathbf{0}} \right\}.$$

Proof: Assume that $\vec{z} = (z_1, ..., z_m)$, and $c = \min\{z_i \mid i = 1, ..., m\}$. Let $\vec{z'} = (z_1 - c, ..., z_m - c)$. Because every column of L_f has the same number of 1 and -1, then $\vec{z}L_f = \vec{z'}L_f$. It implies that

$$\forall \vec{z} \in \mathbf{R}^m \exists \vec{z'} \geq \vec{\mathbf{0}} \left\{ \vec{z}L_f = \vec{z'}L_f \right\}.$$

Based on the above discussion, we can state that the integer linear programming (2.38) has feasible solutions if and only if $G_f(V, E)$ is a strongly connected digraph. $G_f(V, E)$ is a strongly connected digraph if and only if

$$\forall \vec{z} \geq \vec{\mathbf{0}} \left\{ \vec{z}L_f \geq \vec{\mathbf{0}} \Longrightarrow \vec{z}L_f = \vec{\mathbf{0}} \right\}.$$

Q.E.D.

In fact the two matrices L_f and R_f and digraph G_f are completely determined by the definition of function f. On the other hand, they characterize the property of function f. Therefore, we call them *characteristic matrices* and *characteristic digraph* of f.

The following theorem is immediate from Lemma 2.7 and 2.8. and Theorem 2.6.

Theorem 2.8 *Given a commutative function $f : \{1, 2, ..., m\}^2 \longrightarrow \{1, 2, ..., m\}$, the following three statements are equivalent*

1. G_f *is strongly connected.*

2. *The following integer programming has a feasible solution.*

$$\begin{bmatrix} L_f \\ R_f \end{bmatrix} \vec{y} = - \begin{bmatrix} L_f \\ R_f \end{bmatrix} \vec{\mathbf{1}}, \quad \vec{y} \geq \vec{\mathbf{0}}$$

3. $T_f^{(n)}$ *is* $O(1)$ *assignable.*

The computation complexity of deciding whether G_f is strongly connected is $O(m^2)$ [Meho84]. It is independent of the parameter n. Hence we have the following theorem.

Theorem 2.9 *The assignment complexity of a tree system* $T_f^{(n)}$ *based on a commutative function* f *from* M^2 *to* M $(m = \#M)$ *is* $O(m^2)$ *decidable. It is independent of the parameter* n*.*

When f is not commutative, the scene changes.

Example 2.2: Function h is defined as follows:

h	1	2	3
1	1	1	2
2	3	2	1
3	1	1	3

We can determine that

$$\begin{bmatrix} L_h \\ R_h \end{bmatrix} = \begin{bmatrix} 0 & 0 & -1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & -1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & -1 & -1 & 0 \\ 0 & 1 & 0 & -1 & 0 & 1 & 0 & 1 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & -1 & 1 & 0 & -1 & 0 & 0 & 0 \end{bmatrix}$$

It is easy to check that

$$(1,0,1,1,1,0) \begin{bmatrix} L_h \\ R_h \end{bmatrix} = (0,0,0,0,0,2,0,0,0).$$

This indicates that

$$\begin{bmatrix} L_h \\ R_h \end{bmatrix} \vec{y} = - \begin{bmatrix} L_h \\ R_h \end{bmatrix} \vec{\mathbf{1}}, \quad \vec{y} \geq \vec{\mathbf{0}}$$

has no feasible solution. However, both

$$L_h \vec{y} = -L_h \vec{\mathbf{1}}, \quad \vec{y} \geq \vec{\mathbf{0}}$$

and

$$R_h \vec{x} = -R_h \vec{\mathbf{1}}, \quad \vec{x} \geq \vec{\mathbf{0}}$$

have feasible solutions, respectively.

How to decide the assignment complexity of a balanced uniform tree based on a non-commutative function remains a problem.

Chapter 3

Test Complexity of Trees

The test complexity of tree circuits based on primitive gates of type AND, OR, NAND, NOR and NOT has been extensively studied [Haye71, Mark76]. Papers [AbGa81,BeHa90,BeSp91,BhHa86,SeKo77,Wu92a] discuss the test complexity problem of uniform trees and analogous circuits consisting of more complex identical nodes computing an associative or commutative function. In this Chapter we explore the test complexity structure of trees based on commutative functions. The test complexity of a tree is defined as the cardinality of the minimum complete test set of it and is measured as a function of the number of the primary input lines in the tree.

This Chapter shows that the test complexity of balanced trees based on commutative functions can be divided into $\Theta(1)$, $\Theta(\lg n)$, and $\Omega(n^{\alpha})$ ($\alpha \in (0,1]$) classes. This indicates that the test complexity of a tree can jump from one class to another, when its nodes are modified. It motivates us to analyze the test complexity of trees and study the possibility of modifying the trees and changing their test complexity from a high class to a low one.

In section 3.1, we give a formal definition of the fault and diagnosis signal, then define some notations. In section 3.2 we convert the test problem of the tree into the integer linear programming. In section 3.3, we discuss in detail the jump of the test complexity from $\Theta(1)$ to $\Omega(\lg n)$. The jump of the test complexity from $O(\lg n)$ to $\Omega(n^{\alpha})$ ($\alpha \in (0,1]$) will be studied in section 3.4. Section 3.5 deals with the arrangement problem which is a generalization of the assignment and test problems related to uniform trees.

3.1 Fault and Diagnosis Signal

J. P. Roth in [Roth66] introduces two symbols D and $\overline{D}$ to represent two fault diagnosis signals. The former has the value logic 1 in the normal circuit and logic 1 in the faulty circuit. Conversely, the latter has the logic 0 in the normal circuit and 1 in the faulty circuit. For stuck-at fault model there is no difficulty in using the two symbols to describe the fault sensitization, drive and propagation. In this and the next Chapters we discuss the test problem of VLSI systems performing symbolic computations, and adopt the cell definition fault model defined in section 1.2. Some symbols have to be defined to represent the corresponding faults and fault diagnosis signals for the cell definition fault model.

Let $M = \{0, 1, ..., m-1\}$ and f be a surjective and commutative function from M^2 to

M. Fig. 3.1 illustrates a $T_f^{(7)}$.

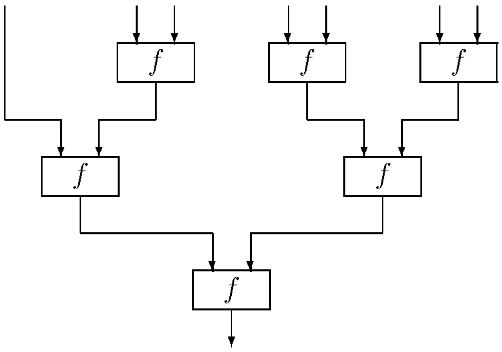


Fig. 3.1: A balanced tree $T_f^{(7)}$

Definition 3.1 (basic fault) *Assume $f(i,j) = k$ and $l \in M \setminus k$. The expression $(i,j) : k/l$ represents a* basic fault *in an f cell. Because of this fault, the cell outputs l for the input(assignment) (i,j) instead of the desired k.*

A tree is recognized to be defective if one of its cells is faulty. A cell is considered to be faulty if it has one or more basic faults. When (i,j) is assigned to the cell and if the practical output value is l rather than the desired k, we say the cell has certainly the basic fault $(i,j) : k/l$. The existence of such a fault can be judged by observing the response of the output of the cell to the assignment (i,j). In other words, (i,j) can sensitize the basic fault $(i,j) : k/l$ and deliver a diagnosis signal denoted by the expression k/l to the output line of the cell. The diagnosis signals total $m(m-1)$. The set of all basic diagnosis signals is

$$d_f := \{k/l \mid k \in M,\ l \in M \setminus k\} \tag{3.1}$$

In fact, one can exactly determine the function value $f(i,j)$ according to the definition of f, provided that both parameter i and j are known. Hence, we express the basic fault in form $(i,j)/l$. Given two parameters i and j, the correct function value $f(i,j)$ is uniquely defined. It is one of the m elements in M. Therefore, there are $m-1$ possible faults for a given pair $(i,j) \in M^2$. An assignment (i,j) applied to f can sensitize all the $m-1$ possible basic faults. Each of them corresponds to a basic fault. To test a cell completely, one has to apply all elements in M^2 to the cell. For m^2 distinct assignments, there are altogether $(m-1)m^2$ basic faults. The set of all basic faults is

$$F_f := \{(i,j)/l \mid (i,j) \in M^2,\ l \in M \setminus f(i,j)\} \tag{3.2}$$

Example 3.1: Function f_1 is defined as follows.

f_1	0	1
0	1	1
1	1	0

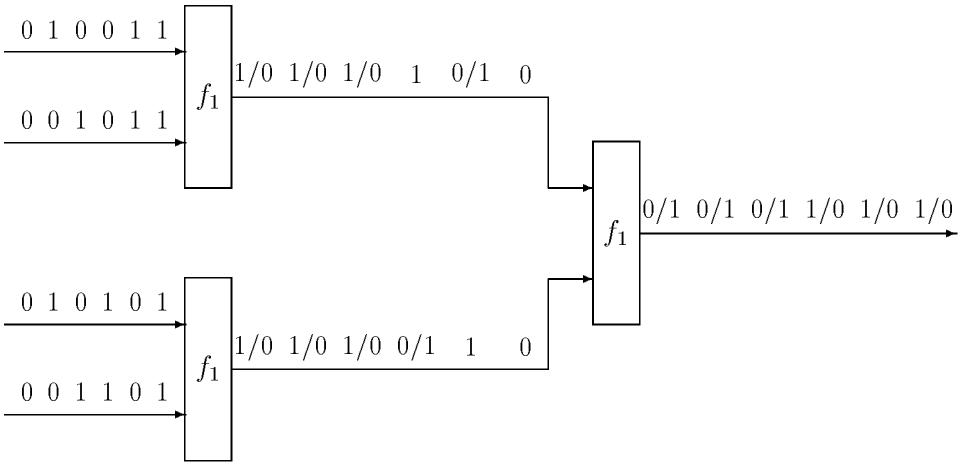


Fig. 3.2: Complete test for $T_{f_1}^{(4)}$

The set of all basic faults is

$$F_{f_1} = \{(0,0)/0,\ (0,1)/0,\ (1,0)/0,\ (1,1)/1\}.$$

A complete test set for a $T_f^{(n)}$ is first of all a complete assignment set. It has to not only assign all elements in M^2 to each of the cells in $T_f^{(n)}$ in order to sensitize every basic fault which possibly occurs in the cell, but also propagate the diagnosis signals to the primary output line for observing it. This is the difference between a complete assignment set and a complete test set.

Fig. 3.2 illustrates a complete test set for uniform tree $T_{f_1}^{(4)}$. This test set is made up of five patterns, and it can completely sensitize every basic fault and propagate the corresponding diagnosis signal to the primary output line. Its cardinality is larger than that of the minimal complete assignment set for the same tree $T_{f_1}^{(4)}$ illustrated by Fig. 2.2. However, it is one of the minimal complete test sets. In most cases, the cardinality of the minimum complete test set for a tree is larger than that of the minimum complete assignment set for the same tree.

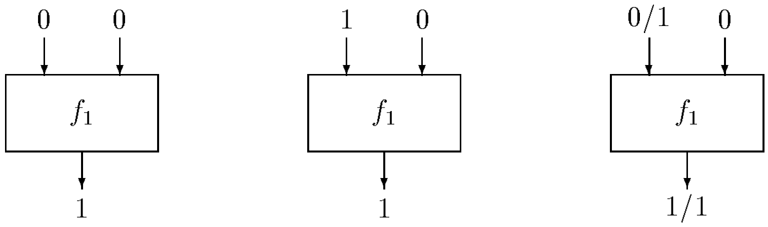
An assignment can be used to sensitize several basic faults and generate several diagnosis signals simultaneously. Consider S to be a subset of $M \setminus f(i,j)$, and we use the expression k/S to stand for a diagnosis signal pack which is made up of all diagnosis signals k/l $(l \in S)$. In case $S = \{l\}$, we substitute k/l for k/S. When $S = \emptyset$, we regard k/S as k, a fault-free signal. Similarly, we use the expression $(i,j)/S$ to represent a fault pack which is made up of all faults in form $(i,j)/l$ $(l \in S)$. For $f(i,j) = k$, we write k/S instead of $(i,j)/S$ when we are interested in the diagnosis signal pack instead of the fault pack, which can derive the diagnosis signal pack. We consider $(i,j)/l \in (i,j)/S$ if $l \in S$.

The terms *fault* and *diagnosis signal* will be used as substitutions for *fault pack* and *diagnosis signal pack*, respectively. The following is the diagnosis signal set.

$$D_f := \{k/S \mid k \in M,\ S \subset M \setminus k\} \tag{3.3}$$

The basic diagnosis signal set d_f is a subset of D_f. The diagnosis signal set D_{f_1} for f_1 defined in the Example 3.1 is $\{0/1,\ 1/0\}$.

Fig.3.3 (a) and (b) show that when $(0,0)$ and $(1,0)$ are applied to an f_1 cell, the corresponding outputs are all 1. This means that when $(0/1, 0)$ is applied to an f_1 cell the output is $1/1$ as shown in Fig. 3.3 (c). The diagnosis signal $0/1$ applied to the left input line of the f_1 cell disappears inside the cell. We say $(0/1, 0)$ is an incompatible assignment to f_1. An incompatible assignments should not be applied to a cell.



(a) (b) (c)

Fig. 3.3

In order to describe the compatibility of the diagnosis signals formally we define several functions.

$$\mathcal{E}_t(k/S) = k, \qquad \mathcal{E}_f(k/S) = S \quad \text{for all} \quad k/S \in D_f \tag{3.4}$$

By using them one can extract the correct value k and incorrect value pack S from a diagnosis signal k/S.

Let $P(M)$ stand for the power set of M, namely, the set of all subsets of M. Function $\mathcal{P}_f$ defined by (3.5)

$$\mathcal{P}_f(u,v) = \{f(\mathcal{E}_t(u), i), f(j, \mathcal{E}_t(v)) | i \in \mathcal{E}_f(v), j \in \mathcal{E}_f(u)\}, \quad (u,v) \in D_f^2 \tag{3.5}$$

is from D_f^2 to $P(M)$ and can be used to determine the set of faulty output values.

Definition 3.2 (D-condition) *We say that function $f : M^2 \longrightarrow M$ satisfies* D-condition *if*

$$\forall i,j \in M\, \{i \neq j \Longrightarrow \exists k,l \in M\, \{f(i,k) \neq f(j,k) \ \wedge\ f(l,i) \neq f(l,j)\}\}\,.$$

If function f does not satisfy D-condition, the f cell is redundant. Some faults in the f are untestable. The D-condition is a fundamental property of the testable functions. We will limit our discussion on the test problem to functions satisfying D-condition.

Definition 3.3 (compatible assignment) *$(u, v) \in D_f^2$ is said to be a compatible assignment to f if $\mathcal{P}_f(u, v)$ does not include $f(\mathcal{E}_t(u), \mathcal{E}_t(v))$.*

When f is not sensitive, there are certainly some incompatible assignments which can not be assigned to an f cell. For example, assume $i \neq j$ and $f(i, k) = f(j, k)$, then $f(i, k) \in \mathcal{P}_f(i/j, k)$, and $(i/j, k)$ is not a compatible assignment to f. When $(i/j,\ k)$ is applied to an f cell, the diagnosis signal i/j put to the left input line disappears in the cell, and one can not find its track at the output line at all. The signal k blockades the propagation of i/j through an f cell. Thus this assignment can not be assigned to an f cell. As shown in Fig. 3.3(c), $(0/1, 0)$ is not a compatible assignment to f_1.

According to the definition of f, one can determine the set of all compatible assignments, denoted by

$$V_f := \{(u,\ v) \mid (u, v) \in D_f^2 \textit{ is a compatible assignment to } f\} \tag{3.6}$$

The set of all compatible assignments for f_1 defined in the Example 3.1 is

$$V_{f_1} = \{(0/1, 1), (1, 0/1), (1/0, 1/0), (1, 1/0), (1/0, 1), (0, 0), (0, 1), (1, 0), (1, 1)\}$$

By assigning $(u, v) \in V_f$ to an f cell, both diagnosis signals u and v can be propagated through the cell, and the corresponding diagnosis signal received from the output line includes $f(\mathcal{E}_t(u), \mathcal{E}_t(v))/\mathcal{P}_f(u, v)$. Every fault $(\mathcal{E}_t(u), \mathcal{E}_t(v))/S$ $(\mathcal{P}_f(u, v) \subset S \subset M \setminus f(\mathcal{E}_t(u), \mathcal{E}_t(v)))$ in the cell can also be sensitized concurrently by this assignment. What S should be depends on the concrete arrangement of the fault sensitization and diagnosis signal propagation. In order to reflect this aspect, we use an ordered triple to represent a concrete assignment. The following set includes various assignments.

$$U_f = \{\ (u, v, S) \mid (u, v) \in V_f,\ \mathcal{P}_f(u, v) \subset S \subset M \setminus f(\mathcal{E}_t(u), \mathcal{E}_t(v))\ \} \tag{3.7}$$

When a concrete assignment (u, v, S) is applied to an f cell, the corresponding diagnosis signal delivered to the output line of the f cell can be determined by using function $\Phi_f : U_f \longrightarrow D_f$ defined by (3.8).

$$\Phi_f(u, v, S) = f(\mathcal{E}_t(u), \mathcal{E}_t(v))/S, \quad (u, v, S) \in U_f \tag{3.8}$$

Because f is commutative, $\Phi_f(u, v, S) = \Phi_f(v, u, S)$ for all $(u, v, S) \in U_f$.

Using U_f as the domain, we define two projections from U_f to D_f.

$$\Pi_l(u, v, S) = u,\ \ \Pi_r(u, v, S) = v, \quad \text{for all } (u, v, S) \in U_f \tag{3.9}$$

Let $\lambda = \#F_f$, $t = \#U_f$ and $s = \#D_f$. Order the elements of F_f, U_f and D_f, respectively. Let F_i denote the ith basic fault in F_f, A_j the jth element of U_f and u_k the kth element of D_f, respectively.

Given a basic fault $(i,j)/l$ and an assignment (u,v,S), we consider that (u,v,S) *test* $(i,j)/l$ if $i = \mathcal{E}_t(u)$, $j = \mathcal{E}_t(v)$ and $l \in S$. For $i \in [1,\lambda]$ and $j \in [1,t]$ we define

$$p_{ij} = \left\{ \begin{array}{lcl} 1 & : & A_j \quad \text{test} \quad F_i \\ 0 & : & \quad \text{otherwise} \end{array} \right. \tag{3.10}$$

If the assignment A_j can sensitize the fault F_i, then $p_{ij} = 1$. Otherwise $p_{ij} = 0$.

Definition 3.4 (complete cell test) $A = \{\underbrace{A_j, \cdots, A_j}_{x_j - time} \mid j \in [1,t]\}$ *is called a complete cell test, if*

$$\forall i \in [1,\lambda] \left\{ \sum_{1 \le j \le t} x_j p_{ij} \ge 1 \right\} \tag{3.11}$$

A complete cell test applied to an f cell can sensitize all basic fault in the cell.

When a complete test set is applied to a tree, all assignments to each of the cells in the tree comprise a complete cell test. If an input line is linked directly to the output line of another cell, the assignments to this input line must contain the total diagnosis signals received from that output line in order to propagate them to the primary output line for the observation.

Definition 3.5 (test complexity) *The test complexity of* $T_f^{(n)}$ *is defined by the mapping* $TC_f : T_f \longrightarrow \mathbf{N}$.

$$TC_f(T_f^{(n)}) = \min \left\{ \#A \mid A \ \text{ is a complete test set for } T_f^{(n)} \right\} \tag{3.12}$$

Assume that $A = \{\underbrace{A_j, \cdots, A_j}_{x_j - \text{time}} \mid A_j \in U_f\}$ is an assignment set applied to an f cell, $\#Q_l(A,u)$ and $\#Q_r(A,u)$ represent the number of u assigned to the left and right input lines of the cell, respectively. $\#Z(A,u)$ is used to denote the corresponding number of u obtained from the output line of f. For x_j multiple assignments of A_j applied to f, $\Phi_f(A_j)$ must appear at least x_j times on the output line of f. In order to describe the relations among $\#Q_l(A,u)$, $\#Q_r(A,u)$ and $Z(A,u)$ formally, we define the following match function

$$\Psi(u,v) = \left\{ \begin{array}{lcl} 1 & : & u = v \\ 0 & : & u \ne v \end{array} \right. \tag{3.13}$$

Suppose A contains x_j A_j for $j \in [1,t]$. In accordance with the above conventions,

$$\#Q_l(A,u) = \sum_{1 \le j \le t} x_j \Psi(\Pi_l A_j, u) \tag{3.14}$$

$$\#Q_r(A,u) = \sum_{1 \le j \le t} x_j \Psi(\Pi_r A_j, u) \tag{3.15}$$

$$\#Z(A,u) = \sum_{1 \le j \le t} x_j \Psi(\Phi_f(A_j), u) \tag{3.16}$$

hold for all $u \in D_f$.

As assumed in section 2.1, we allow a set to include the multiple elements, namely, the elements and their copies. For example, $\{a, a, b\}$ and $\{a, b\}$ are considered as two distinct sets. The cardinality for the former is three, while the cardinality for the latter is two.

We require two special operators $\uplus$ and $\top$ to treat our unconventional sets. The operator $\uplus$ is used to construct a new set by simply putting two sets together. For instance, $\{a, a, b\} \uplus \{a, b\} = \{a, a, a, b, b\}$. The operator $\top$ is used to form a conventional set by extracting all the distinct elements from a multiple set. For example, $\top\{a, a, b\} = \{a, b\}$. By using this operator one can classify a set and press an element and its copies to a single representative.

3.2 $\Theta(1)$ Testable and TLP$_f$

In this section we define an integer linear programming associated with the given function f and show that $T_f^{(n)}$ is $\Theta(1)$ testable if the integer linear programming associated with f has a feasible solution.

Assume that when $(u, v, S) \in U_f$ is applied to an f cell, the corresponding output of f is w. We can say that for this assignment we consume a u and a v on the left and right input lines of f, respectively, and we produce a w on the output line of f.

Consider concretely the consumption and production of the diagnosis signals u_i on the left input and output lines of an f cell for the assignment $A_j = (u, v, S)$. Assume $\Phi_f(A_j) = w$. There are total three cases.

Case 1) $u \neq u_i = w$: The production of u_i on the output line is surplus to the consumption of u_i on the left input line of the cell. For this assignment we win a u_i, and $\Psi(\Phi_f(A_j), u_i) - \Psi(\Pi_l A_j, u_i) = 1$.

Case 2) $u = u_i \neq w$: The production of u_i on the output line can not balance the consumption of u_i on the left input line of the cell. For this assignment we lose a u_i, and $\Psi(\Phi_f(A_j), u_i) - \Psi(\Pi_l A_j, u_i) = -1$.

Case 3) $u = u_i = w$ or $u \neq u_i \wedge w \neq u_i$: The production of u_i on the output line balances the consumption of u_i on the left input line of the cell for this assignment, and $\Psi(\Phi_f(A_j), u_i) - \Psi(\Pi_l A_j, u_i) = 0$.

Let $l_{ij} = \Psi(\Phi_f(A_j), u_i) - \Psi(\Pi_l A_j, u_i)$. The parameter $l_{ij} \in \{-1, 0, 1\}$ can reflect the forenamed three cases exactly.

The consumption of u_i on the right input line and the production of u_i on the output line of an f cell for the assignment $A_j = (u, v, S)$ can be divided into three cases similarly. Let $r_{ij} = \Psi(\Phi_f(A_j), u_i) - \Psi(\Pi_r A_j, u_i)$. The parameter $r_{ij} \in \{-1, 0, 1\}$ can reflect the corresponding three cases.

The following are the formal definitions of l_{ij} and r_{ij} for $i \in [1, s]$ and $j \in [1, t]$, where s and t are the cardinal numbers of the diagnosis signal set D_f and the assignment set U_f defined in section 3.1.

$$l_{ij} = \Psi(\Phi_f(A_j), u_i) - \Psi(\Pi_l A_j, u_i) = \begin{cases} 1 & : \quad \Pi_l A_j \neq u_i \wedge \Phi_f(A_j) = u_i \\ -1 & : \quad \Pi_l A_j = u_i \wedge \Phi_f(A_j) \neq u_i \\ 0 & : \quad \text{otherwise} \end{cases} \tag{3.17}$$

$$r_{ij} = \Psi(\Phi_f(A_j), u_i) - \Psi(\Pi_r A_j, u_i) = \begin{cases} 1 & : \quad \Pi_r A_j \neq u_i \wedge \Phi_f(A_j) = u_i \\ -1 & : \quad \Pi_r A_j = u_i \wedge \Phi_f(A_j) \neq u_i \\ 0 & : \quad \text{otherwise} \end{cases} \tag{3.18}$$

Observation 3.1 *For every assignment set* $A = \{\underbrace{A_j, \cdots, A_j}_{x_j - time} \mid j \in [1, t]\}$ *applied to an* f *cell*

$$\sum_{1 \leq j \leq t} x_j l_{ij} = \#Z(A, u_i) - \#Q_l(A, u_i)$$

and

$$\sum_{1 \leq j \leq t} x_j r_{ij} = \#Z(A, u_i) - \#Q_r(A, u_i)$$

for all $u_i \in D_f$.

Using these parameters we construct two $s \times t$ matrices $L = (l_{ij})_{s \times t}$ and $R = (r_{ij})_{s \times t}$. Let $\vec{l_j}$ and $\vec{r_j}$ denote the jth column vector of L and R, respectively. We call L and R *consumption-production* matrices related to the assignment set U_f.

If (3.19) has a feasible solution, then it has a rational solution since the terms of its constraint matrix and constant vector are all integers. Because (3.19) is a homogeneous linear equation system, one can construct a feasible integer solution for it with its feasible rational solutions.

$$\begin{bmatrix} \mathrm{L} \\ \mathrm{R} \end{bmatrix} \vec{x} = \vec{\mathbf{0}}, \quad \forall j \in [1, t]\{x_j \geq 0\} \tag{3.19}$$

Definition 3.6 (symmetrical circulation) *Multiple set*

$$A = \{\underbrace{A_j, \cdots, A_j}_{x_j - time} \mid j \in [1, t]\} \tag{3.20}$$

is said to be a symmetrical circulation *if the multiple of* A_j *is equal to the* j*th component* x_j *of* $\vec{x}$ *which is a feasible solution of* (3.19).

For an arbitrary symmetrical circulation A, $\#Q_l(A, u)$, $\#Q_r(A, u)$ and $\#Z(A, u)$ are equal to each other for all $u \in D_f$.

Lemma 3.1 *If* K *is a symmetrical circulation, then* $\overline{K} = \{(u, v, S) \mid (v, u, S) \in K\}$ *is a symmetrical circulation. If* K_1 *and* K_2 *are symmetrical circulations, then* $K_1 \uplus K_2$ *is also a symmetrical circulation.*

Proof : This lemma comprises two parts. At first we prove the first part. As mentioned in last section, $\Phi_f(u, v, S) = \Phi_f(v, u, S)$ for all $(u, v, S) \in U_f$. Then $\#Z(K, u) = \#Z(\overline{K}, u)$

for all $u \in D_f$. Notice that

$$\forall u \in D_f \left\{ \#Q_l(\overline{K}, u) = \#Q_r(K, u) = \#Q_l(K, u) = \#Q_r(\overline{K}, u) \right\}$$

Thus we can state that for every $u \in D_f$, $\#Q_l(\overline{K}, u)$, $\#Q_r(\overline{K}, u)$ and $\#Z(\overline{K}, u)$ are equal to each other.

Now we prove the second part. Let $K = K_1 \uplus K_2$. Then

$$\begin{aligned}
\#Q_l(K, u) &= \#Q_l(K_1, u) + \#Q_l(K_2, u) \\
&= \#Q_r(K_1, u) + \#Q_r(K_2, u) \\
&= \#Q_r(K, u) \\
&= \#Z(K_1, u) + \#Z(K_2, u) \\
&= \#Z(K, u)
\end{aligned}$$

for all $u \in D_f$. **Q.E.D.**

An immediate consequence of the above lemma is:

Corollary 3.1 *If K is a symmetrical circulation, then for a given constant k, $\overline{K} = \underbrace{K \uplus \cdots \uplus K}_{k}$ is also a symmetrical circulation.*

Using the parameters p_{ij} defined by(3.10) we construct a matrix $P = (p_{ij})_{\lambda \times t}$. When a complete test set are applied to a tree, the assignment set applied to each of the cells in the tree has to be a complete cell test. If there is a vector $\vec{x}$ satisfying both (3.19) and (3.11), namely,

$$\begin{bmatrix} L \\ R \end{bmatrix} \vec{x} = \vec{\mathbf{0}} \quad \text{and} \quad P\vec{x} \geq \vec{\mathbf{1}},$$

then A defined by (3.20) is a symmetrical circulation as well as a complete cell test for f. We call such a set A *circulative complete cell test*. To search for an optimal circulative complete cell test is equal to solve the following integer linear programming.

$$\begin{array}{rcl}
\min & \sum_{1 \leq j \leq t} & x_j \\
\begin{bmatrix} L \\ R \end{bmatrix} \vec{x} & = & \vec{\mathbf{0}} \\
P\vec{x} & \geq & \vec{\mathbf{1}}
\end{array} \tag{3.21}$$

In order to use the theory of the integer linear programming, we convert the above general form to the standard form by introducing a surplus variable x_{t+i} for each inequality

$$p_{i1}x_1 + p_{i2}x_2 + \cdots + p_{it}x_t \geq 1,$$

so that

$$p_{i1}x_1 + p_{i2}x_2 + \cdots + p_{it}x_t - x_{t+i} = 1$$

for $x_{t+i} \geq 0$. This yields the integer linear programming

$$\min \sum_{1 \leq j \leq t} x_j$$

$$\left[\begin{array}{c|c} L & \mathbf{0} \\ R & \mathbf{0} \\ \hline P & \text{-}I \end{array}\right] \vec{x} = \left[\begin{array}{c} \vec{\mathbf{0}} \\ \vec{\mathbf{0}} \\ \hline \vec{\mathbf{1}} \end{array}\right], \quad \forall j \in [1, t+\lambda]\{x_j \geq 0\} \tag{3.22}$$

where I is the identity matrix with r rows.

It is easy to see that (3.22) has a feasible solution if and only if (3.23) has a feasible solution, and (3.22) has a feasible integer solution if and only if (3.23) has a feasible integer solution.

$$\min \sum_{1 \leq j \leq t} x_j$$

$$\left[\begin{array}{c|c} L & \mathbf{0} \\ R & \mathbf{0} \\ \hline P & \text{-}I \end{array}\right] \vec{x} = \left[\begin{array}{c} \vec{\mathbf{0}} \\ \vec{\mathbf{0}} \\ \hline \vec{\mathbf{0}} \end{array}\right] \tag{3.23}$$

$$\forall j \in [1, t]\{x_j \geq 0\}, \qquad \forall j \in [t+1, t+\lambda]\{x_j \geq 1\}$$

Based on Theorem 2.5, (3.23) has a feasible integer solution if it has a feasible solution. This implies that (3.22) has a feasible integer solution if it has a feasible solution. Thereafter we consider only its feasible integer solution. We call (3.22) *test linear programming* of the f cell (abbreviated to TLP_f). The following observation is immediate from the above discussion.

Observation 3.2 TLP_f (3.22) *has a feasible solution if and only if the function f has a circulative complete cell test.*

Theorem 3.1 $T_f^{(n)}$ *is $\Theta(1)$ testable if its* TLP_f (3.22) *has a feasible solution.*

Proof : Suppose the TLP_f (3.22) has a feasible solution. We can find a circulative complete cell test

$$A = \{\underbrace{A_j, \cdots, A_j}_{x_j\text{-time}} \mid j \in [1, t]\}$$

for f, so that

$$\forall u \in D_f \{\#Q_l(A, u) = \#Q_r(A, u) = \#Z(A, u)\}.$$

When A is assigned to an f cell, a set $\{\Phi_f(A_j) \mid A_j \in A\}$ can be obtained from its output line. With two sets of this kind one can reconstruct a new complete cell test A for an f cell. Assume that the cardinality of A is κ. Given an f tree $T_f^{(n)}$, one can always construct κ n-component patterns which comprise a complete test set for the given $T_f^{(n)}$. The constant κ is independent of the parameter n, namely, the number of the primary input lines of $T_f^{(n)}$. $T_f^{(n)}$ is $\Theta(1)$ testable.

Q.E.D.

Corollary 3.2 *$T_f^{(n)}$ can be completely tested through $(\#M)^2$ n-component patterns if function $f:\ M^2 \longrightarrow M$ is sensitive.*

Proof : Assume $M = \{0, 1, ..., m-1\}$. Let M_i denote the subset $M \setminus i$ of M. Because f is sensitive, then $(i/M_i, j/M_j, M_{f(i,j)}) \in U_f$. Set

$$K = \{(i/M_i, j/M_j, M_{f(i,j)}) \mid (i,j) \in M^2\}$$

is a complete cell test. It is easy to see that

$$\forall i \in M\ \{\#Q_l(K, i/M_i) = \#Q_r(K, i/M_i) = \#Z(K, i/M_i) = m\}.$$

Therefore, K is a circulative complete cell test. We can further state that $T_f^{(n)}$ can be completely tested through $(\#M)^2$ n-component patterns.

Q.E.D.

Example 3.2: Function f_2 is defined below.

f_2	0	1
0	0	1
1	1	0

It is easy to see that f_2 is sensitive. Thus $T_{f_2}^{(n)}$ is $\Theta(1)$ testable according to the above corollary.

3.3 Jump from $\Theta(1)$ to $\Omega(\lg n)$

In this section we concentrate our attention on studying the jump of the test complexity from $\Theta(1)$ to $\Omega(\lg n)$.

Lemma 3.2 *If the* TLP$_f$ *(3.22) has no feasible solution, then there is a $\vec{y} \in \mathbf{R}^{2s}$ such that*

$$\vec{y}\begin{bmatrix} L \\ R \end{bmatrix} > \vec{\mathbf{0}} \ \wedge\ \exists i \in [1,\lambda] \forall j \in [1,t] \left\{ p_{ij} = 1 \Longrightarrow \vec{y}\begin{bmatrix} l_j \\ r_j \end{bmatrix} \geq 2 \right\} \tag{3.24}$$

Proof : Based on Farkas' Lemma (see Theorem 2.4), (3.22) has a feasible solution in $\mathbf{R}^{t+\lambda}$ if and only if for all $\vec{y} \in \mathbf{R}^{2s+\lambda}$

$$\vec{y}\left[\begin{array}{c|c} L & \mathbf{0} \\ R & \mathbf{0} \\ \hline P & -I \end{array}\right] \geq \vec{\mathbf{0}} \Longrightarrow \vec{y}\left[\begin{array}{c} \vec{\mathbf{0}} \\ \vec{\mathbf{0}} \\ \hline \vec{\mathbf{1}} \end{array}\right] \geq 0.$$

Notice that

$$\vec{y}\left[\begin{array}{c|c} L & \mathbf{0} \\ R & \mathbf{0} \\ \hline P & -I \end{array}\right] = \vec{\mathbf{0}} \quad \Longrightarrow \quad \vec{y}\left[\begin{array}{c} \vec{\mathbf{0}} \\ \vec{\mathbf{0}} \\ \hline \vec{\mathbf{1}} \end{array}\right] = 0$$

and

$$\vec{y}\left[\begin{array}{c|c} L & \mathbf{0} \\ R & \mathbf{0} \\ \hline P & \text{-}I \end{array}\right] \geq \vec{\mathbf{0}} \implies -(y_{2s+1}, ..., y_{2s+\lambda})\mathbf{I} \geq \vec{\mathbf{0}}$$

$$\implies (y_{2s+1}, ..., y_{2s+\lambda}) \leq \vec{\mathbf{0}}.$$

Assume that the TLP$_f$ (3.22) has no feasible solution. Then there is a $\vec{y} \in \mathbf{R}^{2s+\lambda}$ such that

$$\vec{y}\left[\begin{array}{c|c} L & \mathbf{0} \\ R & \mathbf{0} \\ \hline P & \text{-}I \end{array}\right] > \vec{\mathbf{0}} \wedge \vec{y}\left[\begin{array}{c} \vec{\mathbf{0}} \\ \vec{\mathbf{0}} \\ \hline \vec{\mathbf{1}} \end{array}\right] < 0.$$

L and R are two $s \times t$ matrices, and P is a $\lambda \times t$ matrix. The entries in P are all nonnegative. We can conclude that in this $\vec{y}$, $y_i \leq 0$ for all $i \in [2s+1, 2s+\lambda]$, and there is at least a $k \in [1, \lambda]$ such that $y_{2s+k} < 0$. Assume $k = 1$ and $\beta = y_{2s+1}$ without loss of generality. Thus there is a $\vec{y} = (y_1, ..., y_{2s}, \beta, y_{2s+2}, ..., y_{2s+\lambda})$ so that

$$\vec{y}\left[\begin{array}{c|c} L & \mathbf{0} \\ R & \mathbf{0} \\ \hline P & \text{-}I \end{array}\right] > \vec{\mathbf{0}} \wedge \vec{y}\left[\begin{array}{c} \vec{\mathbf{0}} \\ \vec{\mathbf{0}} \\ \hline \vec{\mathbf{1}} \end{array}\right] < 0.$$

For such a $\vec{y}$

$$\vec{y}\left[\begin{array}{c} L \\ R \end{array}\right] > \vec{\mathbf{0}} \ \wedge \ \forall j \in [1, t] \left\{ p_{1j} = 1 \Longrightarrow \vec{y}\left[\begin{array}{c} \vec{l_j} \\ \vec{r_j} \end{array}\right] > 0 \right\}.$$

This implies that given an arbitrary constant $\alpha \in \mathbf{R}^+$, there is a constant $c \in \mathbf{R}^+$ such that

$$c\vec{y}\left[\begin{array}{c} L \\ R \end{array}\right] > \vec{\mathbf{0}} \ \wedge \ \forall j \in [1, t] \left\{ p_{1j} = 1 \Longrightarrow c\vec{y}\left[\begin{array}{c} \vec{l_j} \\ \vec{r_j} \end{array}\right] \geq \alpha \right\}.$$

Taking 2 as α, we have this lemma. **Q.E.D.**

Lemma 3.3 *If the* TLP$_f$ (3.22) *has no feasible solution, then there is a vector* $(y_1, y_2, ..., y_s) \in \mathbf{R}^s$ *so that for every complete cell test A either*

$$\sum_{i=1}^{s} y_i \# Z(A, u_i) \geq \sum_{i=1}^{s} y_i \# Q_l(A, u_i) + 1$$

or

$$\sum_{i=1}^{s} y_i \# Z(A, u_i) \geq \sum_{i=1}^{s} y_i \# Q_r(A, u_i) + 1$$

holds.

Proof : Based on Lemma 3.2, if the TLP$_f$ (3.22) has no feasible solution, then there is a $\vec{z} \in \mathbf{R}^{2s}$ such that

$$\vec{z}\left[\begin{array}{c} L \\ R \end{array}\right] > \vec{\mathbf{0}} \;\wedge\; \exists i \in [1,\lambda]\; \forall j \in [1,t] \left\{ p_{ij} = 1 \Longrightarrow \vec{z}\left[\begin{array}{c} \vec{l_j} \\ \vec{r_j} \end{array}\right] \geq 2 \right\}.$$

Given a complete cell test $A = \{\underbrace{A_j, \cdots, A_j}_{x_j-\text{time}} \mid j \in [1,t]\}$, we construct a t-dimension vector $\vec{x}$ and take the multiple x_j of A_j in A as the jth component of $\vec{x}$. According to the definition of the complete cell test

$$\forall i \in [1,\lambda] \left\{ \sum_{1 \leq j \leq t} x_j p_{ij} \geq 1 \right\}.$$

There is certainly an $i \in [1,\lambda]$ and exists a $j \in [1,t]$ so that $p_{ij} = 1$, $x_j \geq 1$, and $\vec{z}\left[\begin{array}{c} \vec{l_j} \\ \vec{r_j} \end{array}\right] x_j \geq 2$. It is not hard to see that for such an $\vec{x}$

$$\exists i \in [1,\lambda] \forall j \in [1,t] \left\{ p_{ij} = 1 \Longrightarrow \vec{z}\left[\begin{array}{c} \vec{l_j} \\ \vec{r_j} \end{array}\right] \geq 2 \right\}$$

and

$$\vec{z}\left[\begin{array}{c} L \\ R \end{array}\right] \vec{x} \geq 2.$$

According to the definitions of L and R.

$$\begin{aligned} \vec{z}\left[\begin{array}{c} \mathrm{L} \\ \mathrm{R} \end{array}\right] \vec{x} &= \left(\vec{z}\left[\begin{array}{c} \vec{l_1} \\ \vec{r_1} \end{array}\right], \cdots, \vec{z}\left[\begin{array}{c} \vec{l_t} \\ \vec{r_t} \end{array}\right] \right) \vec{x} \\ &= \sum_{j=1}^{t} x_j \left(\sum_{i=1}^{s} z_i l_{ij} + \sum_{i=1}^{s} z_{s+i} r_{ij} \right) \\ &= \sum_{i=1}^{s} z_i \sum_{j=1}^{t} x_j l_{ij} + \sum_{i=1}^{s} z_{s+i} \sum_{j=1}^{t} x_j r_{ij}. \end{aligned}$$

According to Observation 3.1

$$\sum_{j=1}^{t} x_j l_{ij} = \#Z(A, u_i) - \#Q_l(A, u_i)$$

and

$$\sum_{j=1}^{t} x_j r_{ij} = \#Z(A, u_i) - \#Q_r(A, u_i),$$

we can state that for every complete cell test A of f

$$\begin{aligned}\sum_{1\le i\le s} z_i(\#Z(A,u_i)-\#Q_l(A,u_i))+\sum_{1\le i\le s} z_{s+i}(\#Z(A,u_i)-\#Q_r(A,u_i)) &= \vec{z}\begin{bmatrix}\mathrm{L}\\ \mathrm{R}\end{bmatrix}\vec{x}\\ &\ge 2.\end{aligned}$$

Using the given complete cell test A we construct a set $\overline{A}$ as follows:

$$\overline{A}=\{(u_i,u_k,S)\mid (u_k,u_i,S)\in A\}.$$

It is not hard to see that

$$\overline{A}=\left\{\underbrace{A_j,\cdots,A_j}_{x'_j-\mathrm{time}}\mid j\in[1,t]\right\}$$

and $(x'_1,...,x'_t)^T>\vec{\mathbf{0}}$. The multiples of $A_j\in\overline{A}(j\in[1,t])$ are all nonnegative. Thus

$$\begin{aligned}&\sum_{1\le i\le s} z_i(\#Z(\overline{A},u_i)-\#Q_l(\overline{A},u_i))+\sum_{1\le i\le s} z_{s+i}(\#Z(\overline{A},u_i)-\#Q_r(\overline{A},u_i))\\ &=\vec{z}\begin{bmatrix}\mathrm{L}\\ \mathrm{R}\end{bmatrix}(x'_1,...,x'_t)^T\\ &\ge 0.\end{aligned}$$

Because $\Phi_f(u,v,S)=\Phi_f(v,u,S)$ for all $(u,v,S)\in U_f$,

$$\forall u_i\in D_f\left\{\#Z(A,u_i)=\#Z(\overline{A},u_i)\right\}.$$

Furthermore, for all $u_i\in D_f$

$$\#Q_l(\overline{A},u_i)=\#Q_r(A,u_i)\ \wedge\ \#Q_r(\overline{A},u_i)=\#Q_l(A,u_i).$$

Hence

$$\begin{aligned}\sum_{1\le i\le s} z_i(\#Z(A,u_i)-\#Q_l(A,u_i))+\sum_{1\le i\le s} z_{s+i}(\#Z(A,u_i)-\#Q_r(A,u_i)) &\quad +\\ \sum_{1\le i\le s} z_i(\#Z(A,u_i)-\#Q_r(A,u_i))+\sum_{1\le i\le s} z_{s+i}(\#Z(A,u_i)-\#Q_l(A,u_i)) &\quad\ge\quad 2.\end{aligned}$$

In other words,

$$2\sum_{1\le i\le s}(z_i+z_{s+i})\#Z(A,u_i)\ge\sum_{1\le i\le s}(z_i+z_{s+i})(\#Q_l(A,u_i)+\#Q_r(A,u_i))+2.$$

Let $y_i=z_i+z_{s+i}$ for $i\in[1,s]$. It means that there is a vector $(y_1,y_2,...,y_s)\in\mathbf{R}^s$ and for every complete cell test A either

$$\sum_{1\le i\le s} y_i\#Z(A,u_i)\ge\sum_{1\le i\le s} y_i\#Q_l(A,u_i)+1$$

or

$$\sum_{1 \leq i \leq s} y_i \# Z(A, u_i) \geq \sum_{1 \leq i \leq s} y_i \# Q_r(A, u_i) + 1$$

holds, provided that the TLP_f (3.22) has no feasible solution.

Q.E.D.

Theorem 3.2 *$T_f^{(n)}$ is either $\Theta(1)$ or $\Omega(\lg n)$ testable.*

Proof : Suppose $T_f^{(n)}$ is not $\Theta(1)$ testable. Followed Theorem 3.1 the TLP_f (3.22) has no feasible solution. Based on Lemma 3.3 there is a vector $(y_1, y_2, ..., y_s) \in \mathbf{R}^s$ and for every complete cell test A, either

$$\sum_{i=1}^{s} y_i \# Z(A, u_i) \geq \sum_{i=1}^{s} y_i \# Q_l(A, u_i) + 1$$

or

$$\sum_{i=1}^{s} y_i \# Z(A, u_i) \geq \sum_{i=1}^{s} y_i \# Q_r(A, u_i) + 1$$

holds.

Suppose $T_f^{(n)}$ has 2^k primary input lines. We determine a path, called *downhill path*, from the primary output line to a primary input line by using the following procedure.

1. Choose the cell with the primary output line as the first cell on the downhill path, and let $A^{(1)}$ denote the assignment set to this cell. The following inequality holds.

$$\sum_{i=1}^{s} y_i \# Z(A^{(1)}, u_i) \geq \sum_{i=1}^{s} y_i \# Z(A^{(1)}, u_i) + 1 - 1.$$

2. Let $A^{(l)}$ denote the assignment to the lth cell on the downhill path, and assume that

$$\sum_{i=1}^{s} y_i \# Z(A^{(1)}, u_i) \geq \sum_{i=1}^{s} y_i \# Z(A^{(l)}, u_i) + l - 1$$

holds.

3. In case

$$\sum_{i=1}^{s} y_i \# Z(A^{(l)}, u_i) \geq \sum_{i=1}^{s} y_i \# Q_l(A^{(l)}, u_i) + 1,$$

we choose the cell linked directly to the left input line of the lth cell as the next cell on the downhill path. Otherwise

$$\sum_{i=1}^{s} y_i \# Z(A^{(l)}, u_i) \geq \sum_{i=1}^{s} \# Q_r(A^{(l)}, u_i) + 1$$

holds, and we choose the cell linked directly to the right input line of the lth cell as the next cell on this path. Let $A^{(l+1)}$ denote the assignment to this cell.

In the first case,

$$\begin{aligned}\sum_{1\le i\le s} y_i \#Z(A^{(l)},u_i) &\ge \sum_{1\le i\le s} y_i \#Q_l(A^{(l)},u_i)+1\\ &= \sum_{1\le i\le s} y_i \#Z(A^{(l+1)},u_i)+1.\end{aligned}$$

In the second case

$$\begin{aligned}\sum_{1\le i\le s} y_i \#Z(A^{(l)},u_i) &\ge \sum_{1\le i\le s} y_i \#Q_r(A^{(l)},u_i)+1\\ &= \sum_{1\le i\le s} y_i \#Z(A^{(l+1)},u_i)+1.\end{aligned}$$

We can state that for both cases

$$\begin{aligned}\sum_{1\le i\le s} y_i \#Z(A^{(1)},u_i) &\ge \sum_{1\le i\le s} y_i \#Z(A^{(l)},u_i)+l-1\\ &\ge \sum_{1\le i\le s} y_i \#Z(A^{(l+1)},u_i)+l.\end{aligned}$$

In this way, we can finally determine the kth cell on the downhill path. Let $A^{(k)}$ denote the assignment set to this cell. Followed the above calculation,

$$\sum_{1\le i\le s} y_i \#Z(A^{(1)},u_i) \ge \sum_{1\le i\le s} y_i \#Z(A^{(k)},u_i)+k-1$$

and

$$\begin{aligned}\#A^{(1)} &= \sum_{1\le i\le s} \#Z(A^{(1)},u_i)\\ &\ge \frac{k}{\max\{|y_i| \mid i=1,...,s\}}\\ &= \Omega(k)\end{aligned} \tag{3.25}$$

$T_f^{(n)}$ has at least a downhill path no shorter than $\lfloor \lg n \rfloor$. Hence $T_f^{(n)}$ is $\Omega(\lg n)$ testable. **Q.E.D.**

3.4 Jump from $O(\lg n)$ to $\Omega(n^\alpha)$

In this section we discuss the jump of the test complexity from $O(\lg n)$ to $\Omega(n^\alpha)$ ($\alpha \in (0,1]$).

To test an f cell completely, we have to deliver a set of diagnosis signals of type $u \in d_f$ to the output line of the cell. If there is a constant κ such that these diagnosis signals on all lines in the same level can be simultaneously propagated to the primary output line with κ patterns, then $T_f^{(n)}$ is $O(\lg n)$ testable.

Definition 3.7 (successor) *For $u \in D_f$, if there is a $(u, w, S) \in U_f$ such that $\Phi_f(u, w, S) = v$, we say that u leads to v directly through w and we use $u \xrightarrow{w} v$ to denote it.*

If $u_i \xrightarrow{w_i} u_{i+1}$ and $w_i \in W$ for all $i \in [1, l-1]$, then we say that u_1 leads to u_l in W and we use $u_1 \xrightarrow{W} u_l$ to denote it.

An element v is called a successor of u if $u \xrightarrow{D_f} v$.

For every $(u, v, S) \in U_f$, $u \xrightarrow{w} v$ implies that $w \xrightarrow{u} v$ since $\Phi_f(u, v, S) = \Phi_f(v, u, S)$.

Definition 3.8 (circle) *$W \subset D_f$ is called* circle *if*

$$\forall u, w \in W \left\{ u \xrightarrow{W} w \right\}.$$

Let $\Gamma(u) := \left\{ v \mid u \xrightarrow{D_f} v \right\}$ denote the set of successors of $u \in D_f$. If $\Gamma(u)$ contains a circle W, then we say that the element u can be driven into the circle W.

Definition 3.9 (embedded set) *We say that set $W \subset D_f$ can be embedded in $K \subset U_f$ if*

$$W \subset \{u \mid (u, v, S) \in K\} \cap \{v \mid (u, v, S) \in K\}.$$

Lemma 3.4 *For every circle W, there exists a symmetrical circulation K such that W can be embedded in K.*

Proof : At first we construct an assignment set A and a digraph G_f by using the following procedure.

1. Set $A := \emptyset$.

2. For two arbitrary elements $u, v \in W$, if there is a $(u, v, S) \in U_f$ such that $\Phi_f(u, v, S) = w$ and $w \in W$, then set $A := A \cup \{(u, v, S)\}$ and we add arcs (u, w) and (v, w) to the digraph G_f.

The digraph G_f is strongly connected since W is a circle.

Assume $\#A = t'$. According to A we construct a submatrix L' of L defined in section 3.2. Matrix L' contains the jth column of L if and only if A_j, the jth element of U_f, is included in A. Matrix L' contains ith row of L if and only if there is an assignment $(u, v, S) \in A$ so that $\Phi_f(u, v, S) = u_i$, and u_i is the ith element of D_f. Matrix R' is induced from R in a similar way. L' and R' are the consumption-production matrices (defined in section 3.2) related to A. The matrix L' is just the node-arc incidence matrix of the digraph G_f. The matrix R' can be obtained by exchanging the columns of L'. In other words,

$$\forall i \in [1, t'] \exists j \in [1, t'] \left\{ \vec{l'_i} = \vec{r'_j} \wedge \vec{r'_i} = \vec{l'_j} \right\}.$$

Based on Theorem 2.8,

$$\begin{bmatrix} L' \\ R' \end{bmatrix} \vec{x} = \vec{\mathbf{0}}, \quad \vec{x} \geq \vec{\mathbf{1}} \tag{3.26}$$

has a feasible integer solution if G_f is strongly connected.

According to the solution for (3.26) we can construct an assignment set

$$K = \{\underbrace{(u, v, S), \cdots, (u, v, S)}_{x_j\text{-time}} \mid j \in [1, t']\}$$

and make

$$\#Q_l(K, u) = \#Q_r(K, u) = \#Z(K, u)$$

true for all $u \in W$. Then K is a symmetrical circulation, and W can be embedded in K.

Q.E.D.

Theorem 3.3 *$T_f^{(n)}$ is $O(\lg n)$ testable, provided that $\Gamma(u_i)$ includes a circle for every $u_i \in d_f$.*

Proof : The input lines of all cells in the same level are independent of each other. The faults of all cells in the same level can be sensitized simultaneously. $T_f^{(n)}$ has at most $\lceil \lg n \rceil$ levels. Therefore, $T_f^{(n)}$ is $O(\lg n)$ testable, provided that there is a constant κ, and the whole diagnosis signals derived from the complete tests of all cells in the same level can be propagated to the primary output line with κ patterns.

According to the assumption, $\Gamma(u_i)$ contains a circle for every $u_i \in d_f$. According to Lemma 3.4, every circle can be embedded in a symmetrical circulation. Thus there is a constant κ_i, and all diagnosis signals of type u_i derived from the complete tests of all cells in the same level can be simultaneously propagated to the primary output line with κ_i patterns.

Let $\kappa = \sum_{1 \le i \le |d_f|} \kappa_i$. Then the whole diagnosis signals derived from the complete tests of all cells in the same level can be propagated to the primary output line with κ patterns.

Q.E.D.

Lemma 3.5 *Given a diagnosis signal set $W \subset D_f$, if*

$$\forall v \in W \exists u, w \in W \left\{ u \xrightarrow{w} v \right\} \tag{3.27}$$

then W contains a circle.

Proof : Assume that (3.27) holds. We prove this lemma by using induction on the cardinality of W.

For $\#W = 1$, $W = \{u\}$. Based on the assumption, $u \xrightarrow{u} u$. Then $W = \{u\}$ is a circle.

Suppose W contains a circle for $\#W < k$. For $W = \{u_1, u_2, ..., u_k\}$, there are two possible cases. W itself is either a circle or not.

At first we suppose W itself is not a circle. This implies that there are $u', v' \in W$, and u' can not lead to v' in W. Define a subset W' of W as follows:

$$W' = \left\{ w' \mid u' \xrightarrow{W} w' \right\}$$

W' consists of all elements which u' can lead to, and no element in W' can lead to an element in $W \setminus W'$. The cardinality of set $W \setminus W'$ is less than k, and

$$\forall v \in W \setminus W' \exists u, w \in W \setminus W' \left\{ u \xrightarrow{w} v \right\}.$$

According to the above assumption, $W \setminus W'$ contains a circle. Thus we can state that W does contain a circle. **Q.E.D.**

Assume that $T_f^{(n)}$ consists of cells $C_{1,1}$, $C_{2,1}$, $C_{2,2}$, ..., $C_{k,l}$, ..., and cell $C_{i,j}$ is the jth cell in the ith level of $T_f^{(n)}$. When an assignment p (an n-component pattern) is applied to $T_f^{(n)}$, a diagnosis signal is delivered to every line in $T_f^{(n)}$. We use $\mathcal{Q}(p, C_{i,j})$ and $\mathcal{Z}(p, C_{i,j})$ to denote the corresponding set of signals delivered to the input lines and the output line of cell $C_{i,j}$, respectively. Let

$$\mathcal{Q}_i(p) := \bigcup_{\substack{i \in [1,k] \\ j \in [1,2^{i-1}]}} \mathcal{Q}(p, C_{i,j}) \tag{3.28}$$

$$\mathcal{Z}_i(p) := \bigcup_{\substack{i \in [1,k] \\ j \in [1,2^{i-1}]}} \mathcal{Z}(p, C_{i,j}) \tag{3.29}$$

Here $\mathcal{Q}_i(p)$ and $\mathcal{Z}_i(p)$ are traditional sets containing no duplicated element. According to (3.28) and (3.29), $\mathcal{Z}_{i+1}(p) = \mathcal{Q}_i(p) \cup \mathcal{Z}_i(p)$ for all $i \in [1, k-1]$.

Lemma 3.6 *Assume that $\Gamma(u)$ contains no circle. When an assignment p is applied to a k-level $T_f^{(n)}$,*

$$\mathcal{Q}_k(p) \subset \Gamma(u) \Longrightarrow \mathcal{Q}_k(p) \not\subset \mathcal{Z}_k(p).$$

Proof: Suppose $\mathcal{Q}_k(p) \subset \Gamma(u)$ and $\mathcal{Q}_k(p) \subset \mathcal{Z}_k(p)$. The former implies that $\mathcal{Q}(p, C_{i,j}) \subset \Gamma(u)$, and the unique element of $\mathcal{Z}(p, C_{i,j})$ is a successor of u. Then $\mathcal{Z}_k(p) \subset \Gamma(u)$. The latter means that

$$\forall v \in \mathcal{Z}_k(p) \exists u, w \in \mathcal{Z}_k(p) \left\{ u \xrightarrow{w} v \right\},$$

and $\mathcal{Z}_k(p)$ contains a circle. This is a contradiction with the assumption that $\Gamma(u)$ contains no circle.

Q.E.D.

Lemma 3.7 *Assume that $\Gamma(u)$ contains no circle. When an arbitrary assignment p is applied to a k-level $T_f^{(n)}$,*

$$\mathcal{Q}_k(p) \subset \Gamma(u) \Longrightarrow |\mathcal{Q}_k(p) \cup \mathcal{Z}_k(p)| \geq k+1$$

holds.

Proof: We prove this lemma by using induction on the level k of $T_f^{(n)}$. Assume that $\Gamma(u)$ contains no circle and $\mathcal{Q}_k(p) \subset \Gamma(u)$.

For $k = 1$, the lemma is ture.

Assume that for $k < l$

$$\mathcal{Q}_k(p) \subset \Gamma(u) \Longrightarrow |\mathcal{Q}_k(p) \cup \mathcal{Z}_k(p)| \geq k + 1$$

holds. Then $|\mathcal{Q}_{l-1}(p) \cup \mathcal{Z}_{l-1}(p)| \geq l$.

For $k = l$, based on Lemma 3.6

$$\mathcal{Q}_l(p) \subset \Gamma(u) \Longrightarrow \mathcal{Q}_l(p) \not\subset \mathcal{Z}_l(p).$$

Because $\mathcal{Z}_l(p) = \mathcal{Q}_{l-1}(p) \cup \mathcal{Z}_{l-1}(p)$, we can state that

$$|\mathcal{Q}_l(p) \cup \mathcal{Z}_l(p)| \geq |\mathcal{Z}_l(p)| + 1 \geq l + 1.$$

Then we have the lemma. **Q.E.D.**

We know that $\#D_f = N$, and there are altogether N distinct diagnosis signals in D_f. A diagnosis signal $u \in d_f$ has at most N distinct successors. In other words, $\#\Gamma(u) \leq N$ for every $u \in d_f$. Assume $\#\Gamma(u) = k$. A pattern p applied to a k level balanced tree can simultaneously propagate diagnosis signals in $\Gamma(u)$ from at most $2^k - 1$ primary input lines to the primary output line if $\Gamma(u)$ contains no circle.

Theorem 3.4 *Balanced $T_f^{(n)}$ is either $O(\lg n)$ or $\Omega(n^\alpha)$ ($\alpha \in (0, 1]$) testable.*

Proof : Assume that $T_f^{(n)}$ is not $O(\lg n)$ testable. It implies that there is at least an element $u \in d_f$ such that $\Gamma(u)$ does not contain circles. Consider the propagation of a u diagnosis signal from each of the primary input lines to the primary output line. According to the above discussion, there is a constant $k \in \mathbf{N}$, and the total number of the diagnosis signals belonging to $\Gamma(u)$ will be multiplied at least by $\frac{2^k}{2^k-1}$ through every k levels. A balanced tree with n primary input lines has about $\lceil \lg n \rceil$ levels. Therefore, at least $\left(\frac{2^k}{2^k-1}\right)^{\frac{\lg n}{k}}$ patterns are necessary to propagate a u diagnosis signal from each of the primary input lines to the primary output line. Suppose A is a complete test set for $T_f^{(n)}$. The following formula holds.

$$\begin{aligned} \#A &\geq \left(\frac{2^k}{2^k - 1}\right)^{\frac{\lg n}{k}} \\ &= 2^{\frac{\lg n}{k} \lg \frac{2^k}{2^k-1}} \\ &= n^{1 - \frac{\lg 2^k - 1}{k}} \\ &= \Omega(n^\alpha), \end{aligned}$$

where $0 < \alpha = 1 - \frac{\lg 2^k - 1}{k} \leq 1$. Thus we can state that $T_f^{(n)}$ is $\Omega(n^\alpha)$ testable.

Q.E.D.

3.5 Arrangement Complexity

An arrangement set to a tree with n primary input lines consists of a number of n-component patterns, so that by applying them to the primary input lines of the tree, the input and output sets of every cell insider the tree has a *predefined property*. An arrangement problem is to apply an arrangement set to a given tree. This is a more general combinational problem expanded from the assignment and test problems related to tree VLSI systems.

For distinguishing from the Cartesian product M^k, we use $M^{(l)}$ to denote the set of all l-dimensional vectors. Given a function $b : M^k \longrightarrow M$, we define a vector function $b^{(l)}$ as the follows:

$$b^{(l)}(\vec{v}_1, \vec{v}_2, \cdots, \vec{v}_k) \;:=\; \begin{pmatrix} b(v_{11}, v_{12}, ..., v_{1k}) \\ b(v_{21}, v_{22}, ..., v_{2k}) \\ \vdots \\ b(v_{l1}, v_{l2}, ..., v_{lk}) \end{pmatrix}, \quad \vec{v}_i \in M^{(l)}$$

For the arrangement of a uniform tree $T_b^{(n)}$, we are concerned about the predefined property which is expected to be satisfied for every b-cell in the tree. We use *predicate* $\mathbf{P}_b$ to describe this property. The following is the formal definition of the arrangement problem. An example following the definition gives a further explanation on it.

Definition 3.10 (arrangement complexity)
Given a function $b : M^k \longrightarrow M$ and a predefined property P, we define a predicate $\mathbf{P}_b$ as follows

$$\mathbf{P}_b(\vec{v}_0, \vec{v}_1, \cdots, \vec{v}_k) = \begin{cases} \text{true}: & \begin{array}{c} \vec{v}_0 = b_b^{(l)}(\vec{v}_1, \cdots, \vec{v}_k) \;\; and \\ (\vec{v}_0, \vec{v}_1, \cdots, \vec{v}_k) \;\text{ has property } P \end{array} \\ \\ \text{false}: & \text{otherwise} \end{cases}$$

where $\vec{v}_i \in M^{(l)}$ and l is an arbitrary integer in $\mathbf{N}$. In addition, the predicate satisfies the following condition

$$(\vec{v}_0, \vec{v}_1, \cdots, \vec{v}_k) \sim (\vec{u}_0, \vec{u}_1, \cdots, \vec{u}_k) \Rightarrow \mathbf{P}_b(\vec{v}_0, \vec{v}_1, \cdots, \vec{v}_k) = \mathbf{P}_b(\vec{u}_0, \vec{u}_1, \cdots, \vec{u}_k), \quad \vec{v}_i \in M^{(l)}$$

We call n l-dimensional vectors in $M^{(l)}$ (l n-component patterns of M^n) a complete arrangement *set to $T_b^{(n)}$ if by applying them to the n primary input lines of $T_b^{(n)}$, $\mathbf{P}_b(\vec{v}_0, \vec{v}_1, \cdots, \vec{v}_k)$ is true for every b-cell in $T_b^{(n)}$. Here, $\vec{v}_0$ is the output vector and $\vec{v}_1, \cdots, \vec{v}_k$ are the input vectors of that b-cell.*

The arrangement complexity *l of $T_b^{(n)}$ is defined as the minimum of the dimensions of the complete arrangement sets to $T_b^{(n)}$.*

For example, if we define the predicate $\mathbf{P}_b$ as following

$$\mathbf{P}_b(\vec{v}_0, \vec{v}_1, \cdots, \vec{v}_k) = \begin{cases} \text{true} \;: & \begin{array}{c} \vec{v}_0 = b^{(l)}(\vec{v}_1, \cdots, \vec{v}_k) \\ M^k \subset \{(v_{i1}, \cdots, v_{ik}) \mid i = 1, 2, ..., l\} \end{array} \\ \\ \text{false} \;: & \text{otherwise} \end{cases}$$

where $\vec{v}_i \in M^{(l)}$, then the above arrangement problem becomes an assignment problem[Wu93a].

It is easy to show that the arrangement problem becomes a test problem when we give another interpretation to the function b and the predicate $\mathbf{P}_b$.

We can disscus the arrangement problem before discussing the assignment and test problems. However, people who are interested in the construction of the complete assignment and test sets for a given balanced uniform tree may appreciate the concrete analysis of the assignment and test problems.

In the following discussion, we reuse the three mapping families F_h, π^h and F^h defined in section 1.4. $\mathcal{A}$ denote the set of all uniform trees based on a cell B.

Theorem 3.5 *If $S \subset M^k$ is b-stable and a minimal arrangement set to a cell B, $\#S$ is an upper boundary of the minimum of the cardinalities of the complete arrangement sets to each of the trees in $\mathcal{A}$.*

Proof: Given a tree $\mathcal{T}' \in \mathcal{A}$, one can always find a balanced tree $\mathcal{T} \in \mathcal{A}$ so that $\mathcal{T}'$ can be embedded into $\mathcal{T}$. The arrangement complexity of $\mathcal{T}$ is an upper boundary of that of $\mathcal{T}'$. One can construct a complete arrangement set of a constant size for every tree in $\mathcal{A}$ if he can construct a complete arrangement set of a constant size for every balanced tree in $\mathcal{A}$. In other words, all trees in $\mathcal{A}$ are constant arrangeable if all balanced trees in $\mathcal{A}$ are constant arrangeable.

A balanced tree $F_{h+1} \in \mathcal{A}$ has the structure $B \circ (\underbrace{F_h \times \cdots \times F_h}_{k})$. Suppose $S \subset M^k$ is b-stable and an arrangement set to cell B. Then for every $h \in \mathbf{N}$

$$\begin{aligned} \phi_2(F^h) &= \phi_2((\underbrace{F_h \times \cdots \times F_h}_{k}) \circ \pi^h|_S) \\ &= id|_S \end{aligned}$$

and

$$(\underbrace{F_h \times \cdots \times F_h}_{k}) \circ \pi^h(S) = S.$$

This indicates that when $\pi^h(S)$ is applied to the primary input lines of F_{h+1}, the vectors applied to an arbitrary cell B inside F_{h+1} comprise an arrangement set. Thus $\pi^h(S)$ is just a complete arrangement set to F_{h+1}, and $\#S$ is an upper boundary on the minimum of the cardinalities of the complete arrangement sets to F_{h+1}.

Q.E.D.

The following corollary is obvious.

Corollary 3.3 *If $S \subset M^k$ is b-stable and an optimal arrangement set to cell B, $\pi^h(S)$ is an optimal complete arrangement set to F_{h+1}.*

Theorem 3.6 *The arrangement complexity of $T_b^{(n)}$ is $\Theta(1)$ if there are an $i \in \mathbf{N}$, a subset $S \subset M^{k^i}$ and k^i bijective mappings $\pi_1, ..., \pi_{k^i} : S \longrightarrow S$ so that S is a complete arrangement set to F_i and*

$$(\underbrace{F_i \times \cdots \times F_i}_{k^i}) \circ (\underbrace{\pi_1 \times \cdots \times \pi_{k^i}}_{k^i}) \circ D_\sigma^{k^i}(S) = S,$$

Proof: Suppose there are an $i \in \mathbf{N}$, a subset $S \subset M^{k^i}$ and k^i bijective mappings $\pi_1, ..., \pi_{k^i} : S \longrightarrow S$ so that S is a complete arrangement set to F_i and

$$(\underbrace{F_i \times \cdots \times F_i}_{k^i}) \circ (\underbrace{\pi_1 \times \cdots \times \pi_{k^i}}_{k^i}) \circ D_\sigma^{k^i}(S) = S. \tag{3.30}$$

Let $\kappa = k^i$ and $g = \phi_2(F_i)$. Based on (3.30), S is g-stable. Let $l = \#S$. We define

$$\mathbf{P}_g(\vec{v}_0, \vec{v}_1, \cdots, \vec{v}_\kappa) = \begin{cases} \text{true} : & \begin{array}{c} \vec{v}_0 = g^{(l)}(\vec{v}_1, \cdots, \vec{v}_\kappa) \\ (\vec{v}_1, \cdots, \vec{v}_\kappa) \text{ is a complete arrangement to } F_i \end{array} \\ & \\ \text{false} : & \text{otherwise} \end{cases}$$

where $\vec{v}_i \in M^{(l)}$.

According to the assumption, S is g-stable. Based on Theorem 3.5, we can state that $\#S$ is an upper boundary on the minimum of the cardinalities of the arrangement sets to $T_b^{(n)}$, and The arrangement complexity of $T_b^{(n)}$ is $\Theta(1)$.

Q.E.D.

The proofs of Theorem 3.5 and 3.6 are constructive. In fact, from them we can derive algorithms of constructing the minimal complete arrangement set for the balanced uniform tree based on function b.

In the rest of this section, we give another criteria of $\Theta(1)$ arrangeable uniform trees and show that the arrangement complexity of $T_b^{(n)}$ is either $\Theta(1)$ or $\Omega((\lg n)^\gamma)$ $(\gamma > 0)$.

According to the definition of the complete arrangement set, we can define a predicate $\mathbf{P}_b$ based on the given function $b : M^k \longrightarrow M$ and the predefined property P, so that n l-dimensional vectors of $M^{(l)}$ comprise a complete arrangement set to $T_b^{(n)}$ if and only if by applying them to the n primary input lines of $T_b^{(n)}$, $\mathbf{P}_b(\vec{v}_0, \vec{v}_1, \cdots, \vec{v}_k)$ is true for every b-cell inside $T_b^{(n)}$. Here, $\vec{v}_0$ is the output vector and $\vec{v}_1, \cdots, \vec{v}_k$ are the input vectors of that b-cell.

Lemma 3.8 *$T_b^{(n)}$ is $\Theta(1)$ arrangeable if there are an $l \in \mathbf{N}$ and a set $W \subset M^{(l)}$ such that for every $\vec{v}_0 \in W$ there are k vectors $\vec{v}_1, \vec{v}_2, \cdots, \vec{v}_k$ in W and $\mathbf{P}_b(\vec{v}_0, \vec{v}_1, \cdots, \vec{v}_k)$ is true. Put it formally,*

$$\exists l \in \mathbf{N} \exists W \subset M^{(l)} \forall \vec{v}_0 \in W \exists \vec{v}_1, \cdots, \vec{v}_k \in W \, \{\mathbf{P}_b(\vec{v}_0, \vec{v}_1, \cdots, \vec{v}_k)\} \tag{3.31}$$

Proof: Suppose

$$\exists l \in \mathbf{N} \exists W \subset M^{(l)} \forall \vec{v}_0 \in W \exists \vec{v}_1, \cdots, \vec{v}_k \in W \, \{\mathbf{P}_b(\vec{v}_0, \vec{v}_1, \cdots, \vec{v}_k)\} .$$

We prove that for every N-level $T_b^{(n)}$, there are n vectors $\vec{v}_1, \cdots, \vec{v}_n$ of W, and they comprise a complete arrangement set to $T_b^{(n)}$. This can be done through induction on the number of the level of $T_b^{(n)}$.

In the case $N = 1$, the tree has only one cell. We choose a vector $\vec{v}_0 \in W$ arbitrarily, then determine k vectors $\vec{v}_1, \vec{v}_2, \cdots, \vec{v}_k \in W$ so that $\mathbf{P}_b(\vec{v}_0, \vec{v}_1, \cdots, \vec{v}_k)$ is true. It is clear that $(\vec{v}_1, \cdots, \vec{v}_k)$ is an arrangement to cell B, a tree having only one cell.

Assume that for $N = i$, vectors $\vec{v}_{1,0}, \vec{v}_{2,0}, \cdots, \vec{v}_{k^i,0} \in W$ comprise a complete arrangement set to $T_b^{(k^i)}$. Suppose $T_b^{(k^{i+1})}$ is of $(i+1)$ levels and is constructed by connecting every primary input line in $T_b^{(k^i)}$ to the output line of a b-cell.

According to the assumption,

$$\forall j \in [1, k^i] \exists \vec{v}_{j,0}, \vec{v}_{j,1}, \cdots, \vec{v}_{j,k} \in W \left\{ \mathbf{P}_b(\vec{v}_{j,0}, \vec{v}_{j,1}, \cdots, \vec{v}_{j,k}) \right\}.$$

Hence, $(\vec{v}_{j,0}, \vec{v}_{j,1}, \cdots, \vec{v}_{j,k})$ is an arrangement to a b-cell. When $\vec{v}_{j,1}, \vec{v}_{j,2}, \cdots, \vec{v}_{j,k}$ are applied to the k input lines of the cell B directly linked to the jth input line in the level i, the vector offered to this input line is just $\vec{v}_{j,0}$. Thus we can state that the k^{i+1} l-dimensional vectors $\vec{v}_{1,1}, \vec{v}_{1,2}, \cdots, \vec{v}_{1,k}, \vec{v}_{2,1}, \cdots, \vec{v}_{k^i,k}$ comprise a complete arrangement set to $T_b^{(k^{i+1})}$.

Q.E.D.

Corollary 3.4 *$T_b^{(n)}$ is $\Theta(1)$ arrangeable if*

$$\exists l \in \mathbf{N} \exists W' \subset M^{(l)} \forall \vec{v'}_0 \in W' \exists \vec{v}_0, \vec{v}_1, \cdots, \vec{v}_k \in W' \left\{ \mathbf{P}_b(\vec{v}_0, \vec{v}_1, \cdots, \vec{v}_k) \wedge \vec{v'}_0 \sim \vec{v}_0 \right\} \tag{3.32}$$

Proof: Given a set $W' \subset M^{(l)}$ $(l \in \mathbf{N})$, we can always induce a set W so that

$$\forall \vec{v} \in M^{(l)} \left\{ \exists \vec{v'} \in W' \left\{ \vec{v'} \sim \vec{v} \right\} \Longrightarrow \vec{v} \in W \right\} \ \wedge \ \forall \vec{v} \in W \exists \vec{v'} \in W' \left\{ \vec{v'} \sim \vec{v} \right\}.$$

The set W includes every vector which is *similar* to a vector in W'. It is obvious that such a W satisfies (3.31) if W' fulfills (3.32).

Q.E.D

Assume that $T_b^{(n)}$ consists of b-cells $C_{1,1}, C_{2,1}, C_{2,2}, ..., C_{k,m}, ...$, and cell $C_{i,j}$ is the jth cell in the ith level of $T_b^{(n)}$. Let A denote n vectors in $M^{(l)}$, and apply A to this tree. Let W denote the corresponding set including A and all other vectors delivered to other lines in every level of $T_b^{(n)}$. We use (A, i, j, m) to represent the corresponding vector applied to the mth input line of $C_{i,j}$, and $(A, i, j, 0)$ to represent the vector delivered to the output line of $C_{i,j}$.

Given a complete arrangement set A to an N-level $T_b^{(n)}$, we determine N sets in the following way:

$$W_s(A) := \left\{ (A, i, j, m) \mid i \in [1, s], j \in [1, k^{i-1}], m \in [0, k] \right\}, \quad s \in [1, N] \tag{3.33}$$

$W_s(A)$ includes all vectors delivered to a line in level i $(i \in [1, s])$ and the vector delivered to the primary output line. We know that $\sim$ is an equivalence relation. $W_s(A)$ can be partitioned into equivalence classes according to the equivalence relation $\sim$, and we use $\#W_s(A)/_{\sim}$ to denote the number of equivalence classes in $W_s(A)$. The following observation is obvious.

Observation 3.3 *Assume A to be a complete arrangement set to an N-level $T_b^{(n)}$. Then*

1. $\forall s \in [2, N] \left\{ W_{s-1}(A) \subseteq W_s(A) \subseteq M^{(l)} \right\}$;

2. $\forall s \in [2, N] \left\{ 1 \leq \# W_{s-1}(A)/_\sim \leq \# W_s(A)/_\sim \right\}$.

Lemma 3.9 *Assume A to be a complete arrangement set to an N-level $T_b^{(n)}$. Then $T_b^{(n)}$ is $\Theta(1)$ arrangeable if $\# W_s(A)/_\sim = \# W_{s-1}(A)/_\sim$ for an $s \in [2, N]$.*

Proof: Assume A to be a complete arrangement set to an N-level $T_b^{(n)}$. Suppose $\# W_s(A)/_\sim = \# W_{s-1}(A)/_\sim$ for an $s \in [2, N]$. This indicates that $W_s(A)$ and $W_{s-1}(A)$ have the same number of equivalence classes, namely, $\# W_s(A)/_\sim = \# W_{s-1}(A)/_\sim$. As mentioned, $W_{s-1}(A) \subseteq W_s(A)$ for all $s \in [2, N]$. It is not hard to see that

$$\forall \vec{v'}_0 \in W_s(A) \exists \vec{v}_0, \vec{v}_1, \cdots, \vec{v}_k \in W_s(A) \left\{ \mathbf{P}_b(\vec{v}_0, \vec{v}_1, \cdots, \vec{v}_k) \ \wedge \ \vec{v'}_0 \sim \vec{v}_0 \right\}.$$

Based on Corollary 3.4, $T_b^{(n)}$ is $\Theta(1)$ arrangeable.

Q.E.D.

Lemma 3.10 *For every complete arrangement set A to an N-level $T_b^{(n)}$,*

$$\forall s \in [1, N] \left\{ \# W_s(A)/_\sim \geq s \right\} \tag{3.34}$$

if $T_b^{(n)}$ is not $\Theta(1)$ arrangeable.

Proof: Suppose $T_b^{(n)}$ is not $\Theta(1)$ arrangeable. According to Observation 3.3 and Lemma 3.9,

$$\# W_1(A)/_\sim \geq 1 \ \wedge \ \forall s \in [2, N] \left\{ \# W_s(A)/_\sim > \# W_{s-1}(A)/_\sim \right\}.$$

Therefore, $\# W_s(A)/_\sim \geq s$. **Q.E.D.**

Given an $l \in \mathbf{N}$, we can partition $M^{(l)}$ into a number of equivalence classes according to the equivalence relation $\sim$. Let $\# M^{(l)}/_\sim$ denote the number of equivalence classes of $M^{(l)}$.

Observation 3.4 *For every complete arrangement set A, which is made up of vectors in $M^{(l)}(l \in \mathbf{N})$, to an N-level $T_b^{(n)}$,*

$$1 \leq \# W_N(A)/_\sim \leq \# M^{(l)}/_\sim \tag{3.35}$$

Theorem 3.7 *$T_b^{(n)}$ is $\Theta(1)$ arrangeable if and only if there are an $l \in \mathbf{N}$ and a set $W \subset M^{(l)}$ such that*

$$\forall \vec{v'}_0 \in W \exists \vec{v}_0, \vec{v}_1, \cdots, \vec{v}_k \in W \left\{ \mathbf{P}_b(\vec{v}_0, \vec{v}_1, \cdots, \vec{v}_k) \ \wedge \ \vec{v'}_0 \sim \vec{v}_0 \right\} \tag{3.36}$$

Proof: The *if* part is immediate from Corollary 3.4. Assume $T_b^{(n)}$ to be $\Theta(1)$ arrangeable. There is a constant $l \in \mathbf{N}$, and one can determine a complete arrangement set A, which is made up of vectors in $M^{(l)}$, to an arbitrary $T_b^{(n)}$. Suppose $\lceil \lg n \rceil = N$ and $N > \#M^{(l)}/_\sim$. Since

$$\forall s \in [1, N] \left\{ \#W_s(A)/_\sim \leq \#M^{(l)}/_\sim < N \right\},$$

there must be such an $s \in [2, N]$ that $\#W_s(A)/_\sim = \#W_{s-1}(A)/_\sim$, and

$$\forall \vec{v'}_0 \in W_s(A) \exists \vec{v}_0, \vec{v}_1, \cdots, \vec{v}_k \in W_s(A) \left\{ \mathbf{P}_b(\vec{v}_0, \vec{v}_1, \cdots, \vec{v}_k) \wedge \vec{v'}_0 \sim \vec{v}_0 \right\}.$$

Q.E.D

Coming up next we show that the arrangement complexity of $T_b^{(n)}$ is either $\Theta(1)$ or $\Omega((\lg n)^\gamma)(\gamma > 0)$. In other words, there is a jump from $\Theta(1)$ to $\Omega((\lg n)^\gamma)$.

Theorem 3.8 *The arrangement complexity of $T_b^{(n)}$ is either $\Theta(1)$ or $\Omega((\lg n)^\gamma)$.*

Proof: Suppose the arrangement complexity of $T_b^{(n)}$ is not $\Theta(1)$, and A is a complete arrangement set to $T_b^{(n)}$. Let $s = \lceil \lg n \rceil$. Based on Lemma 3.10,

$$\#M^{(l)}/_\sim \geq \#W_s(A)/_\sim \geq s \geq \lg n.$$

It is not hard to see that $\#M^{(l)}/_\sim$ is equal to the number of ways of inserting $l - 1$ spaces into the sequence of $|M|$ 1's. According to Lemma 2.5,

$$\#M^{(l)}/_\sim = \left(\begin{array}{c} l + |M| - 1 \\ |M| - 1 \end{array} \right).$$

Notice that for $l > 1$,

$$l^{|M|} > \left(\begin{array}{c} l + |M| - 1 \\ |M| - 1 \end{array} \right).$$

This means that

$$l \geq s^{\frac{1}{|M|}} \geq (\lg n)^{\frac{1}{|M|}} \tag{3.37}$$

Hence, we have the theorem.

Q.E.D.

Function f is said to be commutative, if for every permutation $(q_1, q_2, ..., q_k)$ of $(1, 2, ..., k)$

$$\mathbf{P}_b(\vec{v_0}, \vec{v_1}, \cdots, \vec{v_k}) = \mathbf{P}_b(\vec{v}_0, \vec{v}_{q_1}, \vec{v}_{q_2}, \cdots, \vec{v}_{q_k}).$$

Theorem 3.9 *For commutative function b, the arrangement complexity of $T_b^{(n)}$ is $\Theta(1)$ if and only if there are an $i \in \mathbf{N}$, a subset $S \subset M^{k^i}$ and k^i bijective mappings $\pi_1, ..., \pi_{k^i} : S \longrightarrow S$ so that S is a complete arrangement set to F_i and*

$$\underbrace{(F_i \times \cdots \times F_i)}_{k^i} \circ \underbrace{(\pi_1 \times \cdots \times \pi_{k^i})}_{k^i} \circ D_\sigma^{k^i}(S) = S,$$

where σ denotes the type of S.

Proof: Suppose f is commutative. The *if* part is immediate from Theorem 3.6. We are required to treat only the *only if* part.

In the same way used to analyze the assignment complexity of balanced uniform trees based on commutative functions [Wu93a], we can show that the arrangement complexity of $T_b^{(n)}$ is $\Theta(1)$ only if

$$\exists l \in \mathbf{N} \exists \vec{v}_0, \vec{v}_1, \vec{v}_2, \cdots, \vec{v}_k \in M^{(l)} \left\{ \mathbf{P}_b(\vec{v}_0, \vec{v}_1, \vec{v}_2, \cdots, \vec{v}_k) \wedge \forall j \in [1, k] \left\{ \vec{v}_j \sim \vec{v}_0 \right\} \right\}. \quad (3.38)$$

Let l be the minimum number of all integers satisfying (3.38). Then there are an $i \in \mathbf{N}$ and k^i vectors $\vec{v}_1, \vec{v}_2, \cdots, \vec{v}_{k^i}$ in $M^{(l)}$ so that they are *similar* to each other and comprise a complete arrangement set to an i-level balanced uniform tree F_i. If we consider $(\vec{v}_1, \vec{v}_2, \cdots, \vec{v}_{k^i})$ as an $l \times k^i$ matrix, it contains no duplicate rows.

Let $\kappa = k^i$ and $v_{m,j}$ denote the mth component of $\vec{v_j}$. Define

$$S = \left\{ (v_{m,1}, v_{m,2}, \cdots, v_{m,\kappa}) \mid m \in [1, l] \right\}.$$

Suppose the output vector is $\vec{v}_0$ when S is applied to F_i. According to the assumption

$$\forall j \in [1, \kappa] \left\{ \vec{v}_j \sim \vec{v}_0 \right\}.$$

Put it differently, $\vec{v}_0$ can be transformed to every vector $\vec{v}_j$ ($j \in [1, \kappa]$) by exchanging its component positions. This means that there are κ bijective mappings

$$\pi_1, ..., \pi_\kappa : S \longrightarrow S$$

so that

$$\underbrace{(F_i \times \cdots \times F_i)}_{k^i} \circ \underbrace{(\pi_1 \times \cdots \times \pi_{k^i})}_{k^i} \circ D_\sigma^{\kappa}(S) = S,$$

where σ denotes the type of S. S is a complete arrangement set to F_i, and the arrangement complexity of F_h is $\Theta(1)$ ($h \in \mathbf{N}$).

For an arbitrarily given $T_b^{(n)}$, we can always find an F_h, so that F_h covers $T_b^{(n)}$. Thus the arrangement complexity of $T_b^{(n)}$ is $\Theta(1)$.

Q.E.D.

Chapter 4

Test Complexity of Uniform Tree Circuits

This Chapter deals with the test complexity of balanced uniform tree circuits. A uniform tree circuit is a special uniform tree based on a boolean function from $\{0,1\}^m$ to $\{0,1\}$. In section 4.1 we make some conventions and prove that a balanced uniform tree circuit is either $\Theta(1)$ or $\Omega(\lg n)$ testable. In section 4.2 we show that the test complexity of balanced uniform tree circuits based on commutative functions can be divided into $\Theta(1)$, $\Theta(\lg n)$ and $\Omega(n^r)$ $(r \in (0,1])$ testable classes. In section 4.3 we prove that uniform tree circuits based on unate functions are all $\Omega(n^r)$ $(r \in (0,1])$ testable, and a balanced uniform tree circuit based on a monotonic function is $\Theta(n^r)$ $(r \in (0,1])$ testable. The section 4.4 shows that the test complexity of uniform tree circuits based on general functions has more classes.

4.1 Uniform Tree Circuits

Let $B = \{0,1\}$ and f be a surjective function from B^m to B. We call a uniform tree based on f uniform tree circuit.

In this Chapter, the *cell definition fault* model is assumed. A fault in a cell changes the function assigned to the cell. Furthermore, we assume that there is only one faulty cell in the whole tree circuit that causes a discrepancy between the practical output and the expected one for some inputs.

Let $\overline{1} = 0$ and $\overline{0} = 1$. A $T_f^{(n)}$ is recognized to be defective if one of its cells is faulty. An f cell is considered to be faulty if for an element $X \in B^m$ applied to it, the corresponding output is $\overline{f(X)}$ instead of the desired $f(X)$. In order to detect this fault, one has to apply X to the faulty cell and sensitize that fault, then drive a diagnosis signal to the output line. When the output line of the faulty cell is not primary, one has to propagate the diagnosis signal to the primary output line for the observation.

For $X \in B^m$, we use $H_w(X)$ to denote its *Hamming weight* which is the number of 1 components in X. For $X, Y \in B^m$, we use $H_d(X,Y)$ to denote their *Hamming distance* which is the number of bits in which X and Y differ.

We partition B^m into $m+1$ classes in the following way.

$$B_i = \{X \mid X \in B^m, H_w(X) = i\} \quad \text{for } i \in [0, m].$$

B_i denotes the ith class including every X whose Hamming weight is i.

We use symbols D and $\overline{D}$ to denote two different *diagnosis signals.* The diagnosis signal D has the value 1 in the normal circuit and 0 in the faulty circuit. The other diagnosis signal $\overline{D}$ has the value 0 in the normal circuit and 1 in the faulty circuit [BrFr76].

Assume that $H_d(X,Y) = 1$, $f(Y) = \overline{f(X)}$, and X and Y differ only in their ith component x_i and y_i. We say that the assignment X can propagate a D signal on the ith input line of an f cell if $x_i = 1$, otherwise a $\overline{D}$ signal. The assignment X can deliver a D signal to the output line of the f cell if $f(X) = 1$, otherwise a $\overline{D}$ signal.

Definition 4.1 (precedent) *Assume that $u, v \in \{D, \overline{D}\}$ are two diagnosis signals. We say that u is a precedent of v if u can be transformed into v through an f cell.*

We use $u \leadsto v$ to denote that u is a precedent of v, and $u \not\leadsto v$ to denote that u is not a precedent of v.

Definition 4.2 (C-property) *$X \in B^m$ is logic 1 critical if*

$$\forall Y \in B^m \left\{ Y > X \wedge H_d(X,Y) = 1 \Longrightarrow f(Y) = \overline{f(X)} \right\}.$$

$X \in B^m$ is logic 0 critical if

$$\forall Y \in B^m \left\{ Y < X \wedge H_d(X,Y) = 1 \Longrightarrow f(Y) = \overline{f(X)} \right\}.$$

$X \in B^m$ is full critical if it is logic 1 as well as logic 0 critical.

We say a function f has C-property(cancellation) if every $X \in B^m$ is full critical.

The C-property is a special case of the sensitive property defined in section 1.4. A full critical assignment X can propagate a diagnosis signal on every input line of an f cell and can propagate diagnosis signals from m input lines.

Example 4.1: Function f is defined as follows:

f	0	1
0	0	1
1	1	0

As shown in Fig. 4.1, when $(0,1)$ is applied to an f cell the output is 1. By changing the input pair from $(0,1)$ to either $(1,1)$ or $(0,0)$, the output will changes from 1 to 0. This means that by applying the assignment $(0,1)$ to an f cell, one can propagate a $\overline{D}$ from the left input line and a D from the right input line to the output line of the f cell. The assignment $(0,1)$ is full critical. It can be shown easily that f defined in this example has C-property.

Definition 4.3 (diagnosis information) *A diagnosis signal $u \in \{D, \overline{D}\}$ on a line has one unit of diagnosis information for the line. Assign X to an f cell. We say that the corresponding diagnosis signal received from the output line of the f cell can contain $\frac{i}{m}$ units of diagnosis information for each of the m input lines, provided that with the assignment X one can propagate a diagnosis signal from at most i input lines.*

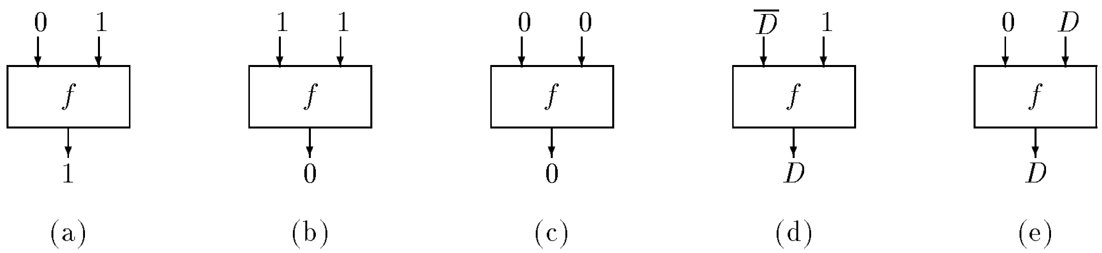


Fig. 4.1

By using the concept of the *diagnosis information* one can measure the propagatability of the diagnosis signals for an assignment X applied to an f cell.

Lemma 4.1 *$\forall X, Y \in B^m$ $\{H_w(X) = H_w(Y) \Longrightarrow f(X) = f(Y)\}$ if f has C-property.*

The proof for this lemma is trivial.

□

It is obvious that if f has C-property, then

$$\begin{aligned} \forall i \in [0, m-1] \forall X \in B_i \forall Y \in B_{i+1} \{f(X) &= \overline{f(Y)}\} \\ \forall i \in [0, m-2] \forall X \in B_i \forall Y \in B_{i+2} \{f(X) &= f(Y)\}. \end{aligned}$$

This states that every assignment $X \in B_i$ applied to an f cell can propagate i D as well as $(m-i)$ $\overline{D}$ signals on the input lines of the cell if f has C-property.

Theorem 4.1 *If f has C-property, then $T_f^{(n)}$ is $\Theta(1)$ testable, else $T_f^{(n)}$ is $\Omega(\lg n)$ testable.*

Proof : Suppose f has C-property. Then f is sensitive. Based on Corollary 1.1 $T_f^{(n)}$ is $\Theta(1)$ testable. In the following we dispose the *only if* part.

Assume that f has no C-property. Then there are $X \in B_i$ and $Y \in B_{i+1}$, so that $f(X) = f(Y)$. It means that either of X and Y applied to an f cell can propagate at most $m-1$ diagnosis signals. Let ν denote the minimum number of diagnosis signals received from the primary output line of an l-level uniform tree circuit. We can inductively prove that $\nu \geq l$.

For $l = 1$, $T_f^{(n)}$ is a single cell, and $\nu \geq 1$. Assume that $\nu \geq i$ holds for $l = i$. For $l = i+1$, each of the m input lines of the cell with the primary output line is linked directly to the output line of an i-level tree circuit, and at least i diagnosis signals on it has to be propagated. On the assumption that there are two distinct assignments $X, Y \in B^m$, so that $H_d(X, Y) = 1$ and $f(X) = f(Y)$, either of X and Y can propagate at most $m-1$ diagnosis signals. In order to test $T_f^{(n)}$ completely, every element of B^m has to be applied to each of the cells of $T_f^{(n)}$. Thus $\nu > \frac{m \times i}{m}$. Put it differently, $\nu \geq i+1$.

A balanced uniform tree circuit with n primary input lines has at least $\lceil \log_m n \rceil$ levels. It is an immediate conclusion that $T_f^{(n)}$ is $\Omega(\lg n)$ testable.

Q.E.D.

Assume that a complete test set of an f cell delivers ν_0 $\overline{D}$ signals and ν_1 D signals to the output line. If there is a constant t, so that with t patterns one can propagate ν_0 $\overline{D}$ signals and ν_1 D diagnosis signals assigned to every input line of cells at the same level of $T_f^{(n)}$ to the primary output line, then we say that every level of $T_f^{(n)}$ is constant testable. It is obvious that a balanced uniform tree circuit is $O(\lg n)$ testable if every level is constant testable.

Lemma 4.2 *$T_f^{(n)}$ is $\Omega(n^r)$ $(r \in (0,1])$ testable if there is no full critical element in B^m.*

Proof : Assume that there is no full critical element in B^m. Every assignment applied to an f cell can propagate at most $m-1$ diagnosis signals. Then through every level the number of diagnosis signals on lines will be multiplied by at least $\frac{m}{m-1}$. It implies that to propagate a diagnosis signal assigned to every primary input line we have to deliver at least $(\frac{m}{m-1})^{\log_m n}$ diagnosis signals to the primary output line of $T_f^{(n)}$. Clearly,

$$(\frac{m}{m-1})^{\log_m n} = n^{1-\log_m(m-1)}, \quad 0 < 1 - \log_m(m-1) \leq 1.$$

The lemma follows.

Q.E.D.

4.2 Commutative Tree Circuits

We have proved that a $T_f^{(n)}$ is either $\Theta(1)$ or $\Omega(\lg n)$ testable in the last section. In this section we show that a $T_f^{(n)}$ based on a commutative function is either $O(\lg n)$ or $\Omega(n^r)$ $(r \in (0,1])$ testable. In other words, the test complexity of balanced uniform tree circuits based on commutative functions can be divided into three classes, namely, $\Theta(1)$, $\Theta(\lg n)$ and $\Omega(n^r)$ $(r \in (0,1])$.

Assume C_f to be the set of all critical elements in B^m. We define a function $\mathcal{P}_f$ from C_f to the integer field.

$$\mathcal{P}_f(X) = H_w(X) - mf(X), \quad X \in C_f \tag{4.1}$$

According to the above definition we have Observation 4.1 and 4.2.

Observation 4.1 *For a given boolean function $f : B^m \longrightarrow B$, the following statements are always true.*

1. $\forall X \in C_f \{-m \leq \mathcal{P}_f(X) \leq m\}$;
2. $\forall X \in C_f \left\{\mathcal{P}_f(X) = -m \Longrightarrow X = \vec{\mathbf{0}} \wedge f(\vec{\mathbf{0}}) = 1\right\}$;
3. $\forall X \in C_f \left\{-m < \mathcal{P}_f(X) < 0 \Longrightarrow 0 < H_w(X) < m \wedge f(\vec{\mathbf{0}}) = 1\right\}$;

4. $\forall X \in C_f \left\{\mathcal{P}_f(X) = 0 \Longrightarrow X = \vec{\mathbf{0}} \wedge f(\vec{\mathbf{0}}) = 0 \vee X = \vec{\mathbf{1}} \wedge f(\vec{\mathbf{0}}) = 1\right\}$;

5. $\forall X \in C_f \left\{0 < \mathcal{P}_f(X) < m \Longrightarrow 0 < H_w(X) < m \wedge f(\vec{\mathbf{0}}) = 0\right\}$;

6. $\forall X \in C_f \left\{\mathcal{P}_f(X) = m \Longrightarrow X = \vec{\mathbf{1}} \wedge f(\vec{\mathbf{0}}) = 0\right\}$;

Observation 4.2 *Suppose boolean function g is defined as follows:*

$$\forall x_1, ..., x_m \in B \left\{g(x_1, ..., x_m) = \overline{f(\overline{x_1}, ..., \overline{x_m})}\right\}$$

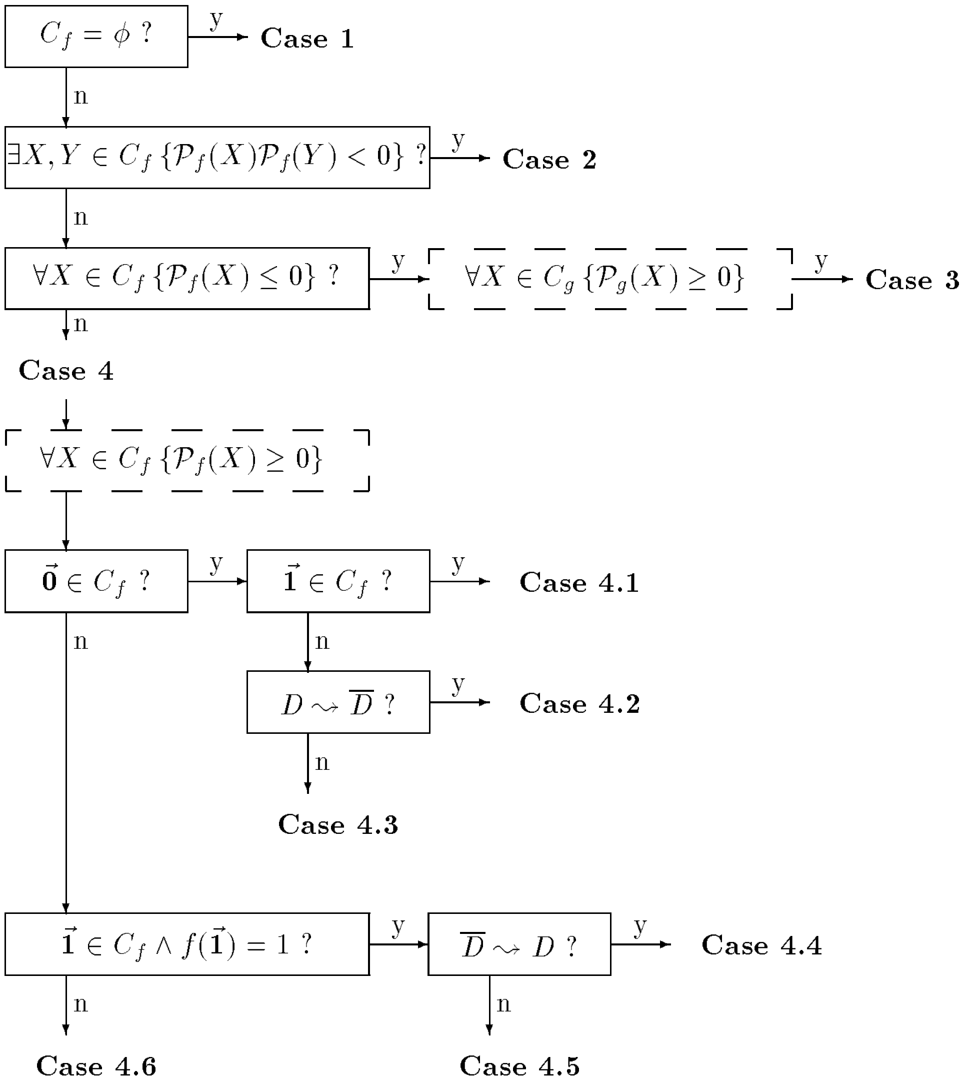
1. *f and g are equivalent to each other ($f \equiv g$), and $T_f^{(n)}$ and $T_g^{(n)}$ have the same test complexity;*
2. *$C_f \neq \phi \Longleftrightarrow C_g \neq \phi$;*
3. *$\forall X \in C_f(X) \{\mathcal{P}_f(X) \leq 0\} \Longleftrightarrow \forall X \in C_g(X) \{\mathcal{P}_g(X) \geq 0\}$.*

By using set C_f and function $\mathcal{P}_f$ we can determine that a given commutative boolean function is in one of the several cases as shown in Fig. 4.2.

In the following we explore the property of the given boolean function f in each of these cases.

Case 1)	$C_f = \phi$;
Case 2)	$\exists X, Y \in C_f \{\mathcal{P}_f(X)\mathcal{P}_f(Y) < 0\}$;
Case 3)	$\forall X \in C_g \{\mathcal{P}_g(X) \geq 0\}$;
Case 4)	$\forall X \in C_f \{\mathcal{P}_f(X) \geq 0\}$;
Case 4.1)	$\forall X \in C_f \{\mathcal{P}_f(X) \geq 0\} \wedge \vec{\mathbf{0}}, \vec{\mathbf{1}} \in C_f$
	$\Longrightarrow \vec{\mathbf{0}}, \vec{\mathbf{1}} \in C_f$;
Case 4.2)	$\forall X \in C_f \{\mathcal{P}_f(X) \geq 0\} \wedge \vec{\mathbf{0}} \in C_f \wedge \vec{\mathbf{1}} \notin C_f \wedge D \rightsquigarrow \overline{D}$
	$\Longrightarrow \vec{\mathbf{0}} \in C_f \wedge f(\vec{\mathbf{0}}) = 0 \wedge D \rightsquigarrow \overline{D}$;
Case 4.3)	$\forall X \in C_f \{\mathcal{P}_f(X) \geq 0\} \wedge \vec{\mathbf{0}} \in C_f \wedge \vec{\mathbf{1}} \notin C_f \wedge D \not\rightsquigarrow \overline{D}$
	$\Longrightarrow \vec{\mathbf{1}} \notin C_f \wedge D \not\rightsquigarrow \overline{D}$;
Case 4.4)	$\forall X \in C_f \{\mathcal{P}_f(X) \geq 0\} \wedge \vec{\mathbf{0}} \notin C_f \wedge \vec{\mathbf{1}} \in C_f \wedge f(\vec{\mathbf{1}}) = 1 \wedge \overline{D} \rightsquigarrow D$
	$\Longrightarrow \vec{\mathbf{1}} \in C_f \wedge f(\vec{\mathbf{1}}) = 1 \wedge \overline{D} \rightsquigarrow D$;
Case 4.5)	$\forall X \in C_f \{\mathcal{P}_f(X) \geq 0\} \wedge \vec{\mathbf{0}} \notin C_f \wedge \vec{\mathbf{1}} \in C_f \wedge f(\vec{\mathbf{1}}) = 1 \wedge \overline{D} \not\rightsquigarrow D$
	$\Longrightarrow \vec{\mathbf{0}} \notin C_f \wedge \overline{D} \not\rightsquigarrow D$;
Case 4.6)	$\forall X \in C_f \{\mathcal{P}_f(X) \geq 0\} \wedge \vec{\mathbf{0}} \notin C_f \wedge (\vec{\mathbf{1}} \notin C_f \vee f(\vec{\mathbf{1}}) = 0)$
	$\Longrightarrow \forall X \in C_f \{f(X) = 0\} \wedge \vec{\mathbf{0}} \notin C_f$
	$\Longrightarrow \forall X \in C_f \{\mathcal{P}_f(X) > 0\}$.

If function f belongs to case 1, then $T_f^{(n)}$ is $\Omega(n^r)$ ($r \in (1, 0]$) testable based on Lemma 4.2. In the following we investigate the test complexity for other cases.



$C_f = \phi$?
y
Case 1
n
$\exists X, Y \in C_f \{\mathcal{P}_f(X)\mathcal{P}_f(Y) < 0\}$?
y
Case 2
n
$\forall X \in C_f \{\mathcal{P}_f(X) \leq 0\}$?
y
$\forall X \in C_g \{\mathcal{P}_g(X) \geq 0\}$
y
Case 3
n
Case 4
$\forall X \in C_f \{\mathcal{P}_f(X) \geq 0\}$
$\vec{\mathbf{0}} \in C_f$?
y
$\vec{\mathbf{1}} \in C_f$?
y
Case 4.1
n
n
$D \rightsquigarrow \overline{D}$?
y
Case 4.2
n
Case 4.3
$\vec{\mathbf{1}} \in C_f \wedge f(\vec{\mathbf{1}}) = 1$?
y
$\overline{D} \rightsquigarrow D$?
y
Case 4.4
n
n
Case 4.6
Case 4.5

Fig. 4.2

Lemma 4.3 *$T_f^{(n)}$ is $O(\lg n)$ testable if there are two full critical elements $X, Y \in B^m$ so that $\mathcal{P}_f(X)\mathcal{P}_f(Y) < 0$.*

Proof : For a commutative function f, if $H_w(X) = H_w(Y)$, then $f(X) = f(Y)$, and X and Y have the same propagatability of the diagnosis signal.

Without loss of generality, we assume that $f(X) = 1$, $f(Y) = 0$, $H_w(X) = i$ and $H_w(Y) = j$. Then $\mathcal{P}_f(X) = i - m$, and $\mathcal{P}_f(Y) = j$. Both $m - i$ and j are greater than zero.

With j assignments in B_i, one can propagate $j \times i$ D signals and $j(m-i)$ $\overline{D}$ signals on input lines of an f cell and deliver j D signals to the output line.

With $m-i$ assignments in B_j, one can propagate $(m-i)j$ D signals and $(m-i)(m-j)$ $\overline{D}$ signals on input lines of an f cell and deliver $(m-i)$ $\overline{D}$ signals to the output line.

With j assignments in B_i and $m-i$ assignments in B_j, one can propagate $m \times j$ D signals and $m(m-i)$ $\overline{D}$ signals on m input lines of an f cell.

The conclusion is that by applying j assignments in B_i and $m-i$ assignments in B_j to an f cell, one can propagate j D signals and $(m-i)$ $\overline{D}$ signals from each of the m input lines and deliver j D signals and $(m-i)$ $\overline{D}$ signals to the output line. There is a constant t, and all diagnosis signals derived from testing all cells at the same level in $T_f^{(n)}$ can be propagated to the primary output line with t n-bit patterns. Thus, one can test every level of $T_f^{(n)}$ through t n-bit patterns. $T_f^{(n)}$ is $O(\lg n)$ testable.

Q.E.D.

Lemma 4.4 *$T_f^{(n)}$ is $O(\lg n)$ testable if $\vec{\mathbf{0}}, \vec{\mathbf{1}} \in C_f$.*

Proof: Suppose that both $\vec{\mathbf{0}}$ and $\vec{\mathbf{1}}$ belong to C_f. By using assignment $\vec{\mathbf{0}}$ one can propagate a $\overline{D}$ from each of the m input lines of an f cell, while by using $\vec{\mathbf{1}}$ one can propagate a D from each of the m input lines of an f cell. Every level of $T_f^{(n)}$ is constant testable, and $T_f^{(n)}$ is $O(\lg n)$ testable.

Q.E.D.

Lemma 4.5 *$T_f^{(n)}$ is $O(\lg n)$ testable if $\vec{\mathbf{0}} \in C_f \ \wedge \ f(\vec{\mathbf{0}}) = 0$, and D is a precedent of $\overline{D}$ signal.*

Proof : Suppose $\vec{\mathbf{0}} \in C_f \wedge f(\vec{\mathbf{0}}) = 0$, and D is a precedent of $\overline{D}$ signal. With $\vec{\mathbf{0}}$ one can propagate a $\overline{D}$ from each of the m input lines of an f cell, and deliver a $\overline{D}$ to the output line of the cell. With a proper assignment one can transform a D signal assigned to an input line of an f cell into a $\overline{D}$. This indicates that with m proper assignments one can separately transform m D signals assigned to m input lines of an f cell into m $\overline{D}$ signals. Hence $m+1$ assignments are enough to propagate a D signal as well as a $\overline{D}$ from every input line of all cells at the same level to the primary output line. Thus every level of $T_f^{(n)}$ is constant testable, and $T_f^{(n)}$ is $O(\lg n)$ testable.

Q.E.D.

Lemma 4.6 *$T_f^{(n)}$ is $\Omega(n^r)$ $(r \in (0,1])$ testable if $\vec{\mathbf{1}} \notin C_f$, and D signal is not a precedent of $\overline{D}$.*

Proof : Without loss of generality, assume $T_f^{(n)}$ to be an l-level tree circuit with m^l primary input lines. On the assumption that D is not a precedent of $\overline{D}$, and no D signal on the primary input lines can be transformed into $\overline{D}$ signal.

Consider only the propagation of a D signal from each of the primary input lines to the primary output line. To propagate a D signal from each of the m input lines of a cell to its output line, at least $\frac{m}{m-1}$ assignments are necessary and $\frac{m}{m-1}$ D signals will be delivered to the output line. Assume that $(\frac{m}{m-1})^i$ patterns are necessary to propagate a D signal from each of the primary input lines of an i-level balanced uniform tree circuit to its primary output line and deliver $(\frac{m}{m-1})^i$ D signals to the primary output line. Thus we can state that at least $(\frac{m}{m-1})^{i+1}$ patterns are necessary to propagate a D signal from each of the primary input lines of an $i+1$ levels balanced uniform tree circuit to its primary output line and deliver $(\frac{m}{m-1})^{i+1}$ D to the primary output line. Since

$$\left(\frac{m}{m-1}\right)^{\log_m n} = n^{1-\log_m(m-1)} \quad \text{and} \quad 0 < 1 - \log_m(m-1) \leq 1,$$

the lemma follows.

Q.E.D.

Lemma 4.7 *$T_f^{(n)}$ is $O(\lg n)$ testable if $\vec{\mathbf{1}} \in C_f \;\wedge\; f(\vec{\mathbf{1}}) = 1$, and $\overline{D}$ is a precedent of D signal.*

The proof for this lemma is similar to that for Lemma 4.5.

□

Lemma 4.8 *$T_f^{(n)}$ is $\Omega(n^r)$ $(r \in (0,1])$ testable if $\vec{\mathbf{0}} \notin C_f$, and $\overline{D}$ signal is not a precedent of D.*

The proof for this lemma is similar to that for Lemma 4.6.

□

Lemma 4.9 *$T_f^{(n)}$ is $\Omega(n^r)$ $(r \in (0,1])$ testable if $\mathcal{P}_f(X) > 0$ for every full critical element $X \in B^m$.*

Proof : Suppose $\mathcal{P}_f(X) > 0$ for every full critical element $X \in B^m$. This implies that $\vec{\mathbf{0}} \notin C_f$ and $f(X) = 0$ for every full critical element $X \in B^m$. An arbitrary full critical element X can propagate at most $m-1$ $\overline{D}$ signals and can only generate a $\overline{D}$.

A nonfull critical assignment can propagate at most $m-1$ diagnosis signals.

Without loss of generality, we assume that $T_f^{(n)}$ has m^l primary input lines and l is even. We prove bellow that a diagnosis signal on the primary output line can contain at most $\left(\frac{m^2-1}{m^2}\right)^{\frac{l}{2}}$ units of diagnosis information for every primary input line.

For $l = 0$, it is trivial. Assume that for $l = 2i$, a diagnosis signal on the primary output line of $T_f^{(n)}$ can contain at most $\left(\frac{m^2-1}{m^2}\right)^i$ units of diagnosis information for every primary input line.

Suppose $l = 2i+2$, and $T_f^{(n)}$ is an l-level uniform tree circuit. In $T_f^{(n)}$ each input line of a cell at the second level is linked directly to the output line of a $2i$-level balanced uniform tree circuit based on f.

On the assumption that $\mathcal{P}_f(X) > 0$ for every full critical assignment $X \in B^m$, then a D signal on the output line of an f cell can only be derived from a nonfull critical assignment, and contain at most $\frac{m-1}{m}$ units of diagnosis information for every input line of the cell. This indicates that a D signal on either the primary output line or lines in level 1 can contain no more than $\frac{m-1}{m}$ units of diagnosis information for every connected line in level 2.

A $\overline{D}$ signal on the primary output line can be derived from either a full critical assignment or nonfull critical assignment applied to the cell in level 1. For the latter case, such a $\overline{D}$ signal can contain no more than $\frac{m-1}{m}$ units of diagnosis information for every line in level 1. For the former case, the $\overline{D}$ signal is derived from a full critical assignment X which can propagate $H_w(X)$ D signals and $m - H_w(X)$ $\overline{D}$ signals. Then this $\overline{D}$ signal can contain no more than $\frac{\frac{m-1}{m}H_w(X)+m-H_w(X)}{m}$ units of diagnosis information for every connected line in level 2.

Clearly,

$$\begin{aligned}\frac{\frac{m-1}{m}H_w(X) + m - H_w(X)}{m} &= \frac{m^2 - H_w(X)}{m} \\ &\leq \frac{m^2-1}{m^2}.\end{aligned}$$

since $\mathcal{P}_f(X) > 0$ and $H_w(X) \geq 1$ for every $X \in C_f$ according to the assumption.

This implies that a diagnosis signal on the primary output line can contain at most $\frac{m^2-1}{m^2}$ units of diagnosis information for every connected input line of a cell at the second level, and $\left(\frac{m^2-1}{m^2}\right)^{i+1}$ units of diagnosis information for every primary input line at most. Then $\left(\frac{m^2}{m^2-1}\right)^{\frac{l}{2}}$ patterns are necessary to propagate one unit of diagnosis information for every primary input line to the primary output line. Notice that

$$\left(\frac{m^2}{m^2-1}\right)^{\frac{l}{2}} = n^{1-0.5\log_m^{(m^2-1)}}.$$

Then we have Lemma 4.9.

Q.E.D.

Theorem 4.2 *$T_f^{(n)}$ is either $O(\lg n)$ or $\Omega(n^r)$ $(r \in (0,1])$ testable.*

Proof : Given an f function from B^m to B, only the following four cases can happen.

1. $C_f = \phi$, and $T_f^{(n)}$ is $\Omega(n^r)$ testable according to Lemma 4.2;

2. $\exists X, Y \in C_f\, \{\mathcal{P}_f(X)\mathcal{P}_f(Y) < 0\}$, and $T_f^{(n)}$ is $O(\lg n)$ testable followed Lemma 4.3;

3. $\forall X \in C_f\, \{\mathcal{P}_f(X) \leq 0\}$, and $T_f^{(n)}$ has the same test complexity as $T_g^{(n)}$ based on Theorem 1.3. For function g, $\forall X \in C_g\, \{\mathcal{P}_g(X) \geq 0\}$.

4. $\forall X \in C_f \{\mathcal{P}_f(X) \geq 0\}$, and $T_f^{(n)}$ is either $O(\lg n)$ or $\Omega(n^r)$ testable followed Lemma 4.4 through 4.9.

Q.E.D.

The following corollary is an immediate consequence of Theorem 4.1 and 4.2.

Corollary 4.1 . *The test complexity of balanced uniform tree circuits based on commutative functions can be divided into three classes, namely,* $\Theta(1)$, $\Theta(\lg n)$ *and* $\Omega(n^r)$ $(r \in (0, 1])$.

4.3 Unate Tree Circuits

The test complexity of unate circuits based on gates of type AND and OR is discussed in detail in [Aker73,Redd73]. In this section we study the test complexity of uniform tree circuits based on unate functions.

Let $\oplus$ denote the boolean operator EXOR and define $X \oplus Y = (x_1 \oplus y_1, ..., x_m \oplus y_m)$ for $X, Y \in B^m$.

Definition 4.4 (unate function) *Function* $f(X)$ *is unate in* x_i *if there is a* $b_i \in B$ *so that*

$$\begin{aligned}\forall X \in B^m \{(x_1, ..., x_{i-1}, b_i \oplus x_i, x_{i+1}, ..., x_m) &\leq (x_1, ..., x_{i-1}, b_i \oplus y_i, x_{i+1}, ..., x_m) \\ \Longrightarrow f(x_1, ..., x_{i-1}, x_i, x_{i+1}, ..., x_m) &\leq f(x_1, ..., x_{i-1}, y_i, x_{i+1}, ..., x_m)\}.\end{aligned}$$

Function f *is considered to be positive unate in* x_i *if* $b_i = 0$, *and positive unate in* $\overline{x_i}$ *if* $b_i = 1$. *Function* f *is considered to be a unate function if it is unate in every* x_i $(i \in [1, m])$.

According to the above definition, if f is unate then there is a $b_f \in B^m$ such that

$$\forall X, Y \in B^m \{b_f \oplus X \leq b_f \oplus Y \Longrightarrow f(X) \leq f(Y)\} \tag{4.2}$$

We call such a b_f *characteristic vector* of the function f. The unate function f is considered to be monotonic if b_f is either $(1, ..., 1)$ or $(0, ..., 0)$.

Given a unate function f, we construct two subsets of B^m,

$$\mathcal{E} = \{X \mid f(X) = 1 \wedge \forall Y \in B^m \{b_f \oplus Y < b_f \oplus X \Longrightarrow f(Y) < f(X)\}\} \tag{4.3}$$

$$\mathcal{N} = \{X \mid f(X) = 0 \wedge \forall Y \in B^m \{b_f \oplus Y > b_f \oplus X \Longrightarrow f(Y) > f(X)\}\} \tag{4.4}$$

Set $\mathcal{E}$ is an *antichain* in which no two elements are comparable. Set $\mathcal{N}$ is also an antichain. We call them D and $\overline{D}$ *critical antichain* of unate function f, respectively. It is easy to see that a diagnosis signal can only be propagated by using an element in either $\mathcal{E}$ or $\mathcal{N}$.

Lemma 4.10 *Let* b_f *be the characteristic vector of unate function* f. *If the ith component* b_i *of* b_f *is* 0, *then a* D *signal on the ith input line of an* f *cell can only be propagated by using elements in* $\mathcal{E}$, *and a* $\overline{D}$ *signal on the ith input line of an* f *cell can only be propagated by using elements in* $\mathcal{N}$; *If the ith component* b_i *of* b_f *is* 1, *then a* D *signal on the ith input line of an* f *cell can only be propagated by using elements in* $\mathcal{N}$, *and a* $\overline{D}$ *signal on the ith input line of an* f *cell can only be propagated by using elements in* $\mathcal{E}$;

Proof: Suppose $b_f = (b_1, b_2, ..., b_m)$. Without loss of generality, we consider the first component b_1. Assume $b_1 = 0$. For arbitrary $x_2, ..., x_m \in B$,

$$(0, b_2, ..., b_m) \oplus (1, x_2, ..., x_m) > (0, b_2, ..., b_m) \oplus (0, x_2, ..., x_m).$$

According to the definition of unate function

$$f(0, x_2, ..., x_m) = 1 \Longrightarrow f(1, x_2, ..., x_m) = 1$$
$$f(1, x_2, ..., x_m) = 0 \Longrightarrow f(0, x_2, ..., x_m) = 0$$

This indicates that the $\overline{D}$ signal on the first input line of an f cell can not be propagated by using an assignment in $\mathcal{E}$, and the D signal on the first input line of an f cell can not be propagated by using an assignment in $\mathcal{N}$.

In a similar way we can show that if $b_1 = 1$, then the $\overline{D}$ signal on the first input line of an f cell can not be propagated by using an assignment in $\mathcal{N}$, and the D signal on the first input line of an f cell can not be propagated by using an assignment in $\mathcal{E}$.

Q.E.D.

Lemma 4.11 *One of $\mathcal{E}$ and $\mathcal{N}$ contains no full critical element.*

Proof: Assume that $X = (x_1, x_2, ..., x_m)$ is a full critical element in $\mathcal{E}$. Let b_f be the characteristic vector of function f, and $b_f \oplus X = (a_1, a_2, ..., a_m)$. We show that $a_i = 1$ for every $i \in [1, m]$.

Suppose $b_f \oplus X = (0, a_2, ..., a_m)$. Let $Y = (\overline{x_1}, x_2, ..., x_m)$. Then $b_f \oplus Y = (1, a_2, ..., a_m)$, and $f(Y) = 0$ since X is full critical. It means that

$$b_f \oplus Y > b_f \oplus X \;\wedge\; f(Y) < f(X)$$

This contradicts (4.2) directly.

Thus $b_f \oplus X = (1, ..., 1)$ if $X \in \mathcal{E}$ is a full critical element. In a similar way we can show that $b_f \oplus Y = (0, ..., 0)$ if $Y \in \mathcal{N}$ is a full critical element.

Assume that there is a full critical element X in $\mathcal{E}$. Then

$$b_f \oplus X = (1, ..., 1) \;\wedge\; \forall Y \in B^m \left\{ b_f \oplus Y \in B_{m-1} \Longrightarrow f(Y) = 0 \right\}.$$

This indicates that

$$\forall Y \in B^m \left\{ b_f \oplus Y \in B^m \setminus B_m \Longrightarrow f(Y) = 0 \right\},$$

particularly

$$\forall Y, Z \in B^m \left\{ b_f \oplus Y \in B_0 \;\wedge\; b_f \oplus Z \in B_1 \Longrightarrow f(Y) = f(Z) = 0 \right\}.$$

This implies that there is no full critical element in $\mathcal{N}$.

Similarly, we can show that there is no full critical element in $\mathcal{E}$ if there is a full critical element in $\mathcal{N}$.

Q.E.D.

Theorem 4.3 *$T_f^{(n)}$ based on a unate function is $\Omega(n^r)$ $(r \in (0,1])$ testable.*

Proof: Assume that f is a unate function, then it has a characteristic vector $b_f \in B^m$ such that

$$\forall X, Y \in B^m \{b_f \oplus X < b_f \oplus Y \Longrightarrow f(X) \leq f(Y)\}.$$

Given a $b_f \in B^m$, only the following three cases can happen.
Case 1) $H_w(b_f) = 0$;
Case 2) $1 \leq H_w(b_f) \leq m-1$;
Case 3) $H_w(b_f) = m$.

For the first case, the corresponding D and $\overline{D}$ critical antichains are the following.

$$\mathcal{E} = \{X \mid f(X) = 1 \wedge \forall Y \in B^m \{Y < X \Longrightarrow f(Y) < f(X)\}\} \tag{4.5}$$
$$\mathcal{N} = \{X \mid f(X) = 0 \wedge \forall Y \in B^m \{Y > X \Longrightarrow f(Y) > f(X)\}\} \tag{4.6}$$

The $D(\overline{D})$ signals on the input lines of an f cell can be propagated only with an element in $\mathcal{E}(\mathcal{N})$ and can only be transformed into $D(\overline{D})$ signals. Based on the above lemma, one of $\mathcal{E}$ and $\mathcal{N}$ contains no full critical element.

Without loss of generality, assume that $\mathcal{E}$ contains no full critical element. By using an element one can propagate a D from at most $m-1$ input lines of an f cell. Assume that the minimum number of D signals on every input line of a cell in level i is $\nu_1^{(i)}$, then the minimum number of $\overline{D}$ signals delivered to an arbitrary line in level $i+1$ is not smaller than $\frac{m}{m-1}\nu_1^{(i)}$, and the minimum number of D signals delivered to an arbitrary line in level $i+2$ is not smaller than $\frac{m}{m-1}\nu_1^{(i)}$.

Assume that $T_f^{(n)}$ is an l-level uniform tree circuit, and we propagate a D signal from each of the n primary input lines to the primary output line. It is easy to see that the number of D signals delivered to the primary output line is not smaller than $(\frac{m}{m-1})^l$, and the $T_f^{(n)}$ is $\Omega(n^r)$ $(r \in (0,1])$ testable.

For the third case, the corresponding D and $\overline{D}$ critical antichains are the following.

$$\mathcal{E} = \{X \mid f(X) = 1 \wedge \forall Y \in B^m \{Y > X \Longrightarrow f(Y) < f(X)\}\} \tag{4.7}$$
$$\mathcal{N} = \{X \mid f(X) = 0 \wedge \forall Y \in B^m \{Y < X \Longrightarrow f(Y) > f(X)\}\} \tag{4.8}$$

The $D(\overline{D})$ signals on the input lines of an f cell can be propagated only with an element in $\mathcal{N}(\mathcal{E})$ and can only be transformed into $\overline{D}(D)$ signals. Based on Lemma 4.11, one of $\mathcal{E}$ and $\mathcal{N}$ contains no full critical element.

Without loss of generality, assume that $\mathcal{N}$ contains no full critical element. Suppose the minimum numbers of D and $\overline{D}$ signals on every input line of a cell in level i are $\nu_1^{(i)}$ and $\nu_0^{(i)}$, respectively. In order to propagate them to the primary output line, the minimum number of $\overline{D}$ signals delivered to every output line of a cell in level i can not be smaller than $\frac{m}{m-1}\nu_1^{(i)}$. The minimum number of $\overline{D}$ signals delivered to every input line of a cell in level $i+2$ is not smaller than $\frac{m}{m-1}\nu_1^{(i)}$.

Assume that $T_f^{(n)}$ is a $2l$-level uniform tree circuit, and we propagate a D signal from each of the n primary input lines to the primary output line. It is easy to see that the number of D signals delivered to the primary output line is not smaller than $(\frac{m}{m-1})^l$, and the $T_f^{(n)}$ is $\Omega(n^r)$ $(r \in (0,1])$ testable.

Now we are only required to consider the second case, and the corresponding D and $\overline{D}$ critical antichains are presented in (4.3) and (4.4).

Suppose $1 \leq H_w(b_f) \leq m-1$. According to Lemma 4.10, if b_i, the ith component of b_f, is 0, then every $D(\overline{D})$ signal on the ith input line of an f cell can be propagated only with an element in $\mathcal{E}(\mathcal{N})$, else every $D(\overline{D})$ signal on the ith input line of an f cell can be propagated only with an element in $\mathcal{N}(\mathcal{E})$.

If neither $\mathcal{E}$ nor $\mathcal{N}$ has a full critical element, then every fault signal received from the output line of an f cell can contain at most $\frac{m-1}{m}$ units of diagnosis information for every input line of the cell. We can easily show that $T_f^{(n)}$ is $\Omega(n^r)$ $(r \in (0,1])$ testable.

Suppose there is a full critical element in $\mathcal{E}$. As assumed, there are at least a 0 and a 1 component in b_f. Without loss of generality, we suppose $b_1 = 0$ and $b_2 = 1$. Then D signals on the first input line and $\overline{D}$ signals on the second input line of an f cell can be propagated only with an element in $\mathcal{E}$, and $\overline{D}$ signals on the first input line and D signal on the second input line of an f cell can be propagated only with an element in $\mathcal{N}$.

Let $\nu_1^{(i)}$ and $\nu_0^{(i)}$ denote the number of D and $\overline{D}$ signals on the ouput of an f cell in ith level. The following formulas hold

$$\begin{aligned} \nu_1^{(i+1)} &\geq \max\{\nu_1^{(i)}, \nu_0^{(i)}\} \\ \nu_0^{(i+1)} &\geq \frac{m}{m-1}\min\{\nu_1^{(i)}, \nu_0^{(i)}\}. \end{aligned}$$

Assume $\nu_0^{(1)} = \nu_1^{(1)} = 1$. Then we can prove that

$$\begin{aligned} \nu_1^{(2i)} &\geq \left(\frac{m}{m-1}\right)^i \\ \nu_0^{(2i)} &\geq \left(\frac{m}{m-1}\right)^i \\ \nu_1^{(2i+1)} &\geq \left(\frac{m}{m-1}\right)^i \\ \nu_0^{(2i+1)} &\geq \left(\frac{m}{m-1}\right)^{i+1} \end{aligned}$$

hold for $i \in \left[0, \frac{\log_m n}{2}\right]$.

Let $\nu = \nu_1^{(l)} + \nu_0^{(l)}$. Then $\nu = \Omega((\frac{m}{m-1})^{\frac{l}{2}})$. Set $r = \frac{1-\log_m(m-1)}{2}$. Thus $\nu = \Omega(n^r)$ $(r \in (0,1])$.

Q.E.D.

Theorem 4.3 gives a low boundary of the test complexity of balanced uniform tree circuits based on unate functions. Monotonic functions are special unate functions. We call a tree circuit based on monotonic function monotonic circuit. In the following, we study in more detail the test complexity of the monotonic tree circuit family.

Assume that $T_f^{(n)}$ is an l-level uniform tree circuit. Let $\nu_1^{(i)}$ and $\nu_0^{(i)}$ denote the minimum numbers of D and $\overline{D}$ signals delivered to every input line of an f cell in level $l-i$, respectively. We will show that a rough relationship among these parameters can be described through the inequality (4.9).

$$\begin{aligned} \begin{bmatrix} \nu_1^{(i)} \\ \nu_0^{(i)} \end{bmatrix} &\geq \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} \begin{bmatrix} \nu_1^{(i-1)} \\ \nu_0^{(i-1)} \end{bmatrix} + \begin{bmatrix} c_1 \\ c_0 \end{bmatrix} \\ \begin{bmatrix} \nu_1^{(i)} \\ \nu_0^{(i)} \end{bmatrix} &\geq \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}^i \begin{bmatrix} \nu_1^{(0)} \\ \nu_0^{(0)} \end{bmatrix} + \sum_{j=0}^{i-1} \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}^j \begin{bmatrix} c_1 \\ c_0 \end{bmatrix} \end{aligned} \tag{4.9}$$

The parameters $a_{00}, a_{01}, a_{10}, a_{11}, c_0$ and c_1 are all determined by the definition of the function f.

When the equality in (4.9) is satisfiable, then (4.9) represents that the number of diagnosis signals on the output line of a cell is a linear function of the number of diagnosis signals on input lines of the cell. We call (4.9) *recurrence formula of the test complexity* of f. The matrix and constant vector in (4.9) are called *rotation matrix* and *translation vector* of f, respectively.

A monotonic function is monotonic increase, if its characteristic vector is $(0, ..., 0)$, else is monotonic decrease. At first we study the test complexity of uniform tree circuits based on monotonic increase functions.

Theorem 4.4 *$T_f^{(n)}$ based on a monotonic increase function is $\Theta(n^r)$ $(r \in (0,1])$ testable.*

Proof: For monotonic increase function f, the characteristic vector b_f is $(0, ..., 0)$ and

$$\forall X, Y \in B^m \, \{Y < X \Longrightarrow f(Y) \leq f(X)\}.$$

The corresponding D and $\overline{D}$ critical antichains are presented in (4.5) and (4.6), respectively.

It is not difficult to see that every element in $\mathcal{E}$ is 0 critical, while every element in $\mathcal{N}$ is 1 critical. A D signal can only be propagated through assignments in $\mathcal{E}$, while a $\overline{D}$ signal can only be propagated through assignments in $\mathcal{N}$. $\mathcal{E}$ and $\mathcal{N}$ have no common element.

For $Y_j = (y_1, \cdots, y_i, \cdots, y_m) \in B^m$ we define $\Pi_i Y_j = y_i$, and $\overline{Y_j} = (\overline{y_1}, \cdots, \overline{y_i}, \cdots, \overline{y_m})$.

Let $s = \#\mathcal{E}$ and $t = \#\mathcal{N}$. Define

$$\alpha^{-1} = \max \left\{ \min \left\{ \sum_{X_j \in \mathcal{E}} z_j \Pi_i X_j \left(\sum_{1 \leq j \leq s} z_j \right)^{-1} \middle| \; i \in [1, m] \right\} \middle| \; z_j \in \mathbf{N}, \; j \in [1, s] \right\}$$

$$\beta^{-1} = \max \left\{ \min \left\{ \sum_{Y_j \in \mathcal{N}} z_j \Pi_i \overline{Y_j} \left(\sum_{1 \leq j \leq t} z_j \right)^{-1} \middle| \; i \in [1, m] \right\} \middle| \; z_j \in \mathbf{N}, \; j \in [1, t] \right\}$$

By repeatedly applying an assignment $X_j \in \mathcal{E}$ z_j times to an f cell one can propagate $z_j \Pi_i X_j$ D signals on the ith input line of the cell. By applying $\sum_{1 \leq j \leq s} z_j$ assignments in $\mathcal{E}$

one can propagate $\sum_{X_j \in \mathcal{E}} z_j \Pi_i X_j$ D signals on the ith input line of the cell. The formula

$$\min \left\{ \sum_{X_j \in \mathcal{E}} z_j \Pi_i X_j \left(\sum_{1 \le j \le s} z_j \right)^{-1} \middle| \; z_j \in \mathbf{N}, i \in [1, m] \right\}$$

defines the minimum rate between the minimum number of D signals propagated from an input line and the number of assignments used in $\mathcal{E}$. The parameter α^{-1} represents the maximum efficiency of propagating a D signal on an input line of an f cell by using an assignment in $\mathcal{E}$. Similarly, β^{-1} represents the maximum efficiency of propagating a $\overline{D}$ signal on an input line of an f cell by using an assignment in $\mathcal{N}$.

To propagate a D signal on every input line of an f cell one has to use $\lceil \alpha \rceil$ assignments in $\mathcal{E}$ and deliver $\lceil \alpha \rceil$ D diagnosis signals to the output line. Similarly, to propagate a $\overline{D}$ signal on every input line of an f cell one has to use $\lceil \beta \rceil$ assignments in $\mathcal{N}$ and deliver $\lceil \beta \rceil$ $\overline{D}$ signals to the output line.

It is easy to see that $\alpha, \beta \ge 1$ and $1 < \alpha\beta \le m^2$.

No assignment $X \in B^m \setminus (\mathcal{E} \cup \mathcal{N})$ can be used to propagate diagnosis signals. However, every assignment $X \in B^m$ has to be applied to an f cell in order to test the cell completely.

Let $c_1 = \sum_{X \in B^m} f(X) - s$ and $c_0 = \sum_{X \in B^m} \overline{f(X)} - t$. Then $c_1 + c_0$ equals the number of elements which are included in neither $\mathcal{E}$ nor $\mathcal{N}$.

In order to propagate $\nu_1^{(i-1)}$ D signals from each of the m input lines of an f cell and to test the cell itself completely one has to drive $\nu_1^{(i-1)}\alpha + c_1$ D signals to the output line. In order to propagate $\nu_0^{(i-1)}$ $\overline{D}$ signals from each of the m input lines of an f cell and to test the cell itself completely one has to deliver $\nu_0^{(i-1)}\beta + c_0$ $\overline{D}$ signals to the output line.

We can determine that the recurrence formula of the test complexity of f is

$$\begin{bmatrix} \nu_1^{(i)} \\ \nu_0^{(i)} \end{bmatrix} = \begin{bmatrix} \alpha & 0 \\ 0 & \beta \end{bmatrix} \begin{bmatrix} \nu_1^{(i-1)} \\ \nu_0^{(i-1)} \end{bmatrix} + \begin{bmatrix} c_1 \\ c_0 \end{bmatrix}.$$

Assume $T_f^{(n)}$ to be an l-level tree circuit, and $\nu_0^{(0)} = \nu_1^{(1)} = 1$. Put it differently, only a D signal and a $\overline{D}$ signal are assigned to every primary input lines. Thus we have

$$\begin{aligned}
\begin{bmatrix} \nu_1^{(i)} \\ \nu_0^{(i)} \end{bmatrix} &= \begin{bmatrix} \alpha & 0 \\ 0 & \beta \end{bmatrix}^i \begin{bmatrix} \nu_1^{(0)} \\ \nu_0^{(0)} \end{bmatrix} + \sum_{j=0}^{i-1} \begin{bmatrix} \alpha & 0 \\ 0 & \beta \end{bmatrix}^j \begin{bmatrix} c_1 \\ c_0 \end{bmatrix} \\
&= \begin{bmatrix} \alpha^i \\ \beta^i \end{bmatrix} + \sum_{j=0}^{i-1} \begin{bmatrix} c_1\alpha^j \\ c_0\beta^j \end{bmatrix} \qquad \text{for } i \in [1, l] \\
\nu_1^{(l)} &= \begin{cases} \Theta(\alpha^l) & : \alpha > 1 \\ \Theta(l) & : \alpha = 1 \wedge c_1 > 0 \\ \Theta(1) & : \alpha = 1 \wedge c_1 = 0 \end{cases} \\
\nu_0^{(l)} &= \begin{cases} \Theta(\beta^l) & : \beta > 1 \\ \Theta(l) & : \beta = 1 \wedge c_0 > 0 \\ \Theta(1) & : \beta = 1 \wedge c_0 = 0 \end{cases}
\end{aligned}$$

We know that $l \geq \log_m n$. Let $\lambda = \log_m \alpha$ and $\mu = \log_m \beta$. We have

$$\nu_1^{(l)} = \begin{cases} \Theta(n^\lambda) & : \alpha \geq 1 \\ \Theta(\log_m n) & : \alpha = 1 \wedge c_1 > 0 \\ \Theta(1) & : \alpha = 1 \wedge c_1 = 0 \end{cases}$$

$$\nu_0^{(l)} = \begin{cases} \Theta(n^\mu) & : \beta \geq 1 \\ \Theta(\log_m n) & : \beta = 1 \wedge c_0 > 0 \\ \Theta(1) & : \beta = 1 \wedge c_0 = 0 \end{cases}$$

As mentioned, either α or β is greater than 1 for a monotonic function f. Let $\nu = \nu_1^{(l)} + \nu_0^{(l)}$ and $r = \max\{\lambda, \mu\}$. Then $\nu = \Theta(n^r)$ ($r \in (0, 1]$).

Q.E.D.

In case f is commutative and monotonic increase, then there is a $k \in [0, m]$ so that

$$\forall j \in [0, m] \, \{\{j < k \Longrightarrow \forall X \in B_j \, \{f(X) = 0\}\} \;\wedge\; \{j \geq k \Longrightarrow \forall Y \in B_j \, \{f(Y) = 1\}\}\}.$$

Such a function is called "threshold k" function from B^m to B (see page 110 in [Hotz74]). The following is its formal definition.

$$f(x_1, \ldots, x_m) = \begin{cases} 1 & : \sum_{1 \leq i \leq m} x_i \geq k \\ 0 & : \text{otherwise} \end{cases}, \qquad (x_1, ..., x_m) \in B^m$$

The D and $\overline{D}$ critical antichains are

$$\mathcal{E} = \left\{(x_1 \cdots x_m) \;\middle|\; \sum x_i = k\right\} \quad \text{and} \quad \mathcal{N} = \left\{(x_1 \cdots x_m) \;\middle|\; \sum x_i = k - 1\right\}.$$

Every assignment in $\mathcal{E}$ can propagate *a* D signal from k input lines of an f cell, while every assignment in $\mathcal{N}$ can propagate a $\overline{D}$ from $m - k + 1$ input lines of an f cell. The terms of the rotation matrix and the translation vector can be determined as follows:

$$\alpha = \frac{m}{k}, \qquad \beta = \frac{m}{m - k + 1}$$

$$c_1 = \sum_{k+1 \leq i \leq m} \binom{m}{i}, \qquad c_0 = \sum_{0 \leq i \leq k-2} \binom{m}{i}.$$

Assume $T_f^{(n)}$ to be an l-level uniform tree circuit based on the "threshold k" function f. Function f is monotonic increase. In the same way used to prove Theorem 4.4 we can determine the parameters $\nu_1^{(l)}$ and $\nu_0^{(l)}$ and show that

$$\begin{aligned} \nu &= \nu_1^{(l)} + \nu_0^{(l)} \\ &= \Theta(\alpha^{\log_m n}) + \Theta(\beta^{\log_m n}) \\ &= \Theta(n^{1 - \log_m k}) + \Theta(n^{1 - \log_m (m-k+1)}). \end{aligned}$$

Take $r = \max\{1 - \log_m k, 1 - \log_m(m - k + 1)\}$. We have $\nu = \Theta(n^r)$.

Example 4.1: Estimate the test complexity of $T_f^{(n)}$ based on the boolean function $f(x, y) = x \wedge y$.

We can determine that the D and $\overline{D}$ critical antichains are

$$\mathcal{E} = \{(1, 1)\} \quad \text{and} \quad \mathcal{N} = \{(0, 1), (1, 0)\}.$$

The recurrence formula of the test complexity of f is

$$\begin{bmatrix} \nu_1^{(i)} \\ \nu_0^{(i)} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} \nu_1^{(i-1)} \\ \nu_0^{(i-1)} \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

Thus

$$\begin{aligned} \nu_0^{(l)} &= 2^l + c_0 \sum_{0 \le j \le l-1} 2^j \\ &= 2^{l+1} - 1 \\ \nu_1^{(l)} &= 1, \\ \nu &= \nu_1^{(l)} + \nu_0^{(l)} \\ &= 2n. \end{aligned}$$

It has been shown that $T_f^{(n)}$ based on $f(x, y) = x \wedge y$ can be tested completely with $n+1$ patterns [Haye71]. This conclusion is slightly different from our result since the fault models used here and adopted in [Haye71] are not the same. In [Haye71] only stuck-at-1 and stuck-at-0 faults are considered, i.e., the assignment $(0, 0)$ does not need to be assigned to an f cell. Here we assume the cell definition fault model, so that every assignments in B^2 needs to be assigned to an f cell. For instance, we hold that there can exist a fault in an f cell, and only the assignment $(0, 0)$ can sensitize it. This implies that $c_0 = 1$. The value of $\nu_0^{(l)}$ is equal to the sum of two terms 2^l and $c_0 \sum_{0 \le j \le l-1} 2^j$. If we set $c_0 = 0$ and omit the second term, then $\nu_0^{(l)} = 2^l$, and $\nu = n + 1$. This new result is completely consistent with that discovered in [Haye71].

In the same way we can determine that $\nu = 2n$ for the case $f(x, y) = x \vee y$.

Based on Theorem 4.4, $\nu = \nu_1^{(l)} + \nu_0^{(l)}$ can be exactly evaluated, provided that f is a monotonic increase function. In case f is not a monotonic increase function, it may be possible to find a monotonic increase function g and to embed f into g. The test complexity of $T_g^{(n)}$ can be determined. Thus the test complexity of $T_f^{(n)}$ can be estimated indirectly.

Example 4.2: Estimate the test complexity of $T_h^{(n)}$ based on function $h(x, y) = \overline{x \wedge y}$.

It is clear that h is not a monotonic increase function. We define a function g as follows.

$$\begin{aligned} g(x, y, u, v) &= h(h(x, y), h(u, v)) \\ &= x \wedge y \vee u \wedge v. \end{aligned}$$

Obviously, $g(x, y, u, v)$ is a monotonic increase function, and h can be embedded into g.

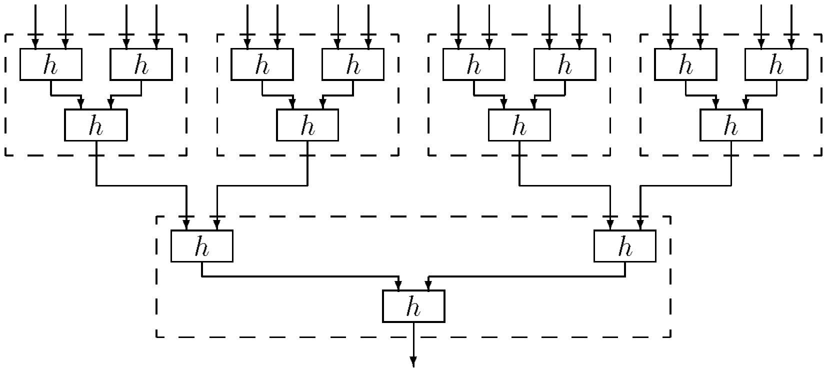


Fig. 4.2: A uniform $T^{(16)}$

Fig. 4.2 illustrates a $T^{(16)}$. Every dash box represents a g cell. If we regard a dash box as a basic cell, then $T^{(16)}$ is a two-level uniform tree circuit based on the monotonic function g. Otherwise, $T^{(16)}$ can be recognized as a four-level uniform tree circuit based on the function h.

We can determine that the D and $\overline{D}$ critical antichains and the recurrence formula of the test complexity of g are the following.

$$\begin{aligned}
\mathcal{E} &= \{(1,1,0,0),(0,0,1,1)\}, \quad \mathcal{N} = \{(1,0,1,0),(0,1,0,1)\} \\
\begin{bmatrix} \nu_1^{(i)} \\ \nu_0^{(i)} \end{bmatrix} &= \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} \nu_1^{(i-1)} \\ \nu_0^{(i-1)} \end{bmatrix} + \begin{bmatrix} 5 \\ 7 \end{bmatrix}.
\end{aligned}$$

Assume $\log_4 n = l$, then

$$\begin{aligned}
\nu_1^{(l)} &= 2^l + 5\sum_{i=0}^{l-1} 2^i \\
&= 3 \cdot 2^{l+1} - 5. \\
\nu_0^{(l)} &= 2^l + 7\sum_{i=0}^{l-1} 2^i \\
&= 2^{l+3} - 7. \\
\nu &= \nu_0^{(l)} + \nu_1^{(l)} \\
&= \Theta(n^{\frac{1}{2}}).
\end{aligned}$$

$T^{(n)}$ based on h is $\Theta(n^{\frac{1}{2}})$ testable since $T^{(n)}$ based on g is $\Theta(n^{\frac{1}{2}})$ testable.

Coming up next we study the case in which f is a monotonic decrease function.

Theorem 4.5 *$T_f^{(n)}$ based on a monotonic decrease function f is $\Theta(n^r)$ ($r \in (0,1]$) testable.*

Proof : The proof for this theorem is similar to that for Theorem 4.3 in the most ways. For monotonic decrease function f, the characteristic vector b_f is $(1, ..., 1)$ and

$$\forall X, Y \in B^m \left\{X > Y \Longrightarrow f(X) \leq f(Y)\right\}.$$

The corresponding D and $\overline{D}$ critical antichains are presented in (4.7) and (4.8), respectively.

It is obvious that all D signals can only be propagated with assignments in $\mathcal{N}$, while all $\overline{D}$ signals can only be propagated with assignments in $\mathcal{E}$.

In the same way used to prove Theorem 4.4 we define the parameters α, β, c_0 and c_1 and the recurrence formula of the test complexity of f. We have

$$\begin{aligned}
\begin{bmatrix} \nu_1^{(i)} \\ \nu_0^{(i)} \end{bmatrix} &= \begin{bmatrix} 0 & \alpha \\ \beta & 0 \end{bmatrix} \begin{bmatrix} \nu_1^{(i-1)} \\ \nu_0^{(i-1)} \end{bmatrix} + \begin{bmatrix} c_1 \\ c_0 \end{bmatrix} \\
\nu_0^{(i)} &= \nu_1^{(i-1)}\beta + c_0 \\
\nu_1^{(i)} &= \nu_0^{(i-1)}\alpha + c_1
\end{aligned}$$

for $i \in [1, l]$.

Assume $\nu_0^{(0)} = \nu_1^{(0)} = 1$. We can inductively prove that

$$\begin{aligned}
\nu_0^{(2i)} &= \alpha^i\beta^i + c_0\sum_{j=0}^{i-1}(\alpha\beta)^j + c_1\beta\sum_{j=0}^{i-1}(\alpha\beta)^j \\
\nu_1^{(2i)} &= \alpha^i\beta^i + c_1\sum_{j=0}^{i-1}(\alpha\beta)^j + c_0\alpha\sum_{j=0}^{i-1}(\alpha\beta)^j \\
\nu_0^{(2i+1)} &= \alpha^i\beta^{i+1} + c_0\sum_{j=0}^{i}(\alpha\beta)^j + c_1\beta\sum_{j=0}^{i-1}(\alpha\beta)^j \\
\nu_1^{(2i+1)} &= \alpha^{i+1}\beta^i + c_1\sum_{j=0}^{i}(\alpha\beta)^j + c_0\alpha\sum_{j=0}^{i-1}(\alpha\beta)^j
\end{aligned}$$

hold for $2i \in [0, l-1]$.

Assume that $l = 2k$. Then

$$\begin{aligned}
\nu &= \nu_0^{(2k)} + \nu_1^{(2k)} \\
&= \Theta((\alpha\beta)^{\frac{l}{2}}).
\end{aligned}$$

We know that $l \geq \log_m n$. Thus

$$\begin{aligned}
\nu &= \Theta\left((\alpha\beta)^{\frac{\log_m^n}{2}}\right) \\
&= \Theta\left(n^{\frac{\log_m^{\alpha\beta}}{2}}\right).
\end{aligned}$$

Let $r = \frac{\log_m^{\alpha\beta}}{2}$, then $r \in (0, 1]$, and $\nu = \Theta(n^r)$.

Q.E.D.

The function h given in Example 4.2 is a monotonic decrease function. We can determine that the D and $\overline{D}$ critical antichains and the recurrence formula of the test complexity of h are following.

$$\mathcal{E} = \{(1,0),(0,1)\}, \quad \mathcal{N} = \{(1,1)\}$$

$$\begin{bmatrix} \nu_1^{(i)} \\ \nu_0^{(i)} \end{bmatrix} = \begin{bmatrix} 0 & 2 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \nu_1^{(i-1)} \\ \nu_0^{(i-1)} \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

Here $m = 2$, and $\alpha\beta = 2$. According to Theorem 4.4,

$$\begin{aligned} \nu &= \Theta(n^{\frac{\log_2^{\alpha\beta}}{2}}) \\ &= \Theta(n^{\frac{1}{2}}). \end{aligned}$$

$T_h^{(n)}$ is $\Theta(n^{\frac{1}{2}})$ testable.

4.4 Summary

In section 4.2 we have proven that the test complexity of balanced uniform tree circuits based on commutative functions can be divided into $\Theta(1)$, $\Theta(\lg n)$ and $\Omega(n^r)$ $(r \in (0,1])$ testable classes. In section 4.3 we have shown that tree circuits based on unate functions are all $\Omega(n^r)$ $(r \in (0,1])$ testable. The test complexity of uniform tree circuits based on general functions has more classes.

Example 4.3: Table 4.1 defines a function f. We show below that $T_f^{(n)}$ is $\Theta((\lg n)^2)$ testable.

Table 4.1

$f(x_1,x_2,x_3)$	x_1	x_2	x_3
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
1	1	0	0
0	1	0	1
0	1	1	0
1	1	1	1

Table 4.2

$f(x_1,x_2,x_3)$	x_1	x_2	x_3
$\overline{D}$	$\overline{D}$	0	0
0	0	0	1
0	0	1	0
$\overline{D}$	$\overline{D}$	1	1
D	1	$\overline{D}$	$\overline{D}$
$\overline{D}$	1	$\overline{D}$	1
$\overline{D}$	1	1	$\overline{D}$
D	D	D	D

From Table 4.2 we can see that the assignments $(0,0,1)$ and $(0,1,0)$ can not be used to propagate diagnosis signals. In order to propagate a diagnosis signal one has to choose one of the six assignments $(0,0,0)$, $(0,1,1)$, $(1,0,0)$, $(1,0,1)$, $(1,1,0)$ and $(1,1,1)$, as shown by Table 4.2.

With the assignment $(1,1,1)$, one can propagate a D signal on each of the three input lines of an f cell. With the assignment $(1,0,0)$ one can transform $\overline{D}$ signals on the second and third input lines of an f cell into D signal. The $\overline{D}$ signals on the first input line of an f cell can only be propagated with the assignment $(0,1,1)$ or $(0,0,0)$. The $\overline{D}$ signals on the second and third input line can be propagated with the assignment $(1,0,0)$ as well as with the assignment $(1,0,1)$ and $(1,1,0)$, respectively. We had better use the assignment $(1,0,0)$ to transform them into D signals since D signals on each of the three input lines of an f cell can be propagated simultaneously. But the $\overline{D}$ signal on the first input line of an f cell can not be transformed into D, no matter how to choose the assignment.

To test a cell completely, all elements in B^3 have to be applied to the cell. It is wise to use each of the eight assignment at least once to propagate some D and $\overline{D}$ diagnosis signals, then use the assignment $(1,0,0)$ to transform the rest of $\overline{D}$ on the second and third input lines into D diagnosis signals. The rest of $\overline{D}$ diagnosis signals on the first input line of the cell can be propagated by repeatedly using either the assignment $(0,0,0)$ or $(0,1,1)$.

Let $\nu_1^{(i)}$ and $\nu_0^{(i)}$ denote the minimum numbers of the D and the $\overline{D}$ signals on the output line of an f cell in level i. This indicates that

$$\begin{bmatrix} \nu_1^{(i)} \\ \nu_0^{(i)} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \nu_1^{(i-1)} \\ \nu_0^{(i-1)} \end{bmatrix} + \begin{bmatrix} 0 \\ 5 \end{bmatrix} \quad \text{for } i \in [1,l].$$

Let $\nu_1^{(0)} = \nu_0^{(0)} = 1$. We can inductively prove that

$$\begin{aligned} \nu_1^{(i)} &= \frac{5i^2 - 3i + 2}{2} \\ \nu_0^{(i)} &= 5i + 1 \quad \text{for } i \in [1,l]. \end{aligned}$$

In order to test an l-level $T_f^{(n)}$ completely, we have to deliver

$$\begin{aligned} \nu &= \nu_1^{(l)} + \nu_0^{(l)} \\ &= \frac{5l^2 + 7l + 4}{2} \end{aligned}$$

diagnosis signals to the primary output line.

As we know, $l = \Theta(\lg n)$, thus we can state that $T_f^{(n)}$ is $\Theta((\lg n)^2)$ testable.

Chapter 5

Synthesis of $O(\lg n)$ Testable Trees

We have shown that the test complexity of the balanced uniform trees can be divided into $\Theta(1)$, $\Theta((\lg n)^\alpha)$ and $\Omega(n^\alpha)$ ($\alpha \in (0,1]$) classes. Two approaches for minimizing the test complexity of circuits are proposed in [Haye74] and [SaRe74]. By using them one can always synthesize a $\Theta(1)$ testable circuit for every function. Their common idea is to use extra test points to make every internal gates in the circuit under test directly controllable and observable. Such modifications change the logic structure of the circuit and necessitate many extra input and output pins. In practice, the number of access terminals is strongly limited. At the present time, the test complexity in order of $O(\lg n)$ is acceptable. Maybe it is a realistic attitude to synthesize an $O(\lg n)$ testable tree within the restriction on the quantity of extra gates and the number of additional access terminals without changing the tree-like logical structure of the system.

This Chapter explores that a balanced uniform $T^{(n)}$ based on a so called *kernel sensitive function*(kernel sensitive function will be formally defined in section 5.1) is always $O(\lg n)$ testable.

Let M be a set of m symbols. We assume $M = \{0, 1, ..., m-1\}$, without loss of generality. This Chapter presents a systematic method of synthesizing kernel sensitive functions from non-kernel sensitive functions from M^2 to M, and shows that every balanced $T^{(n)}$ based on surjective functions from M^2 to M can be embedded in an $O(\lg n)$ testable tree $\mathcal{T}^{(n)}$. This indicates that one can trade the hardware overhead for the low test complexity without changing the tree-like structure of the circuit system. This strategy is meaningful since the cost of the hardware has been decreasing while the cost of the test has been increasing. In comparison with other methods of reducing the test complexity, this method requires more extra gates and less extra input and output pins. With the development of VLSI technology the gate density of VLSI is increasing much more rapidly than the number of access terminals. Thus this method is also promising.

This Chapter is structured in the following way. In section 5.1 we make some conventions and define the *kernel sensitive function*. Section 5.2 presents a systematic method of synthesizing kernel sensitive functions from non-kernel sensitive functions. In section 5.3 we will show how to synthesize an $O(\lg n)$ testable tree and embed a given balanced tree in it.

5.1 Kernel Sensitive Function

In this Chapter we use the terminologies defined in Chapter 3. In the following we present again some conventions.

The set of all basic diagnosis signals is

$$d_f := \{k/l \mid k \in M,\ l \in M \setminus k\} \tag{5.1}$$

The diagnosis signal set is

$$D_f := \{k/S \mid k \in M,\ S \subset M \setminus k\} \tag{5.2}$$

One of its subsets is

$$\Delta_f := \{k/S \mid k \in M,\ S = M \setminus k\} \tag{5.3}$$

Using D_f^2 as the domain, we define a function $\mathcal{P}_f$ as follows:

$$\mathcal{P}_f(i/S_i, j/S_j) = \{f(i,k), f(l,j) \mid k \in S_j, l \in S_i\}, \quad (i/S_i, j/S_j) \in D_f^2 \tag{5.4}$$

$\mathcal{P}_f(i/S_i, j/S_j)$ is a subset of M. When the diagnosis signal pair $(i/S_i, j/S_j)$ is applied to an f cell, the set of error outputs must include $\mathcal{P}_f(i/S_i, j/S_j)$.

Definition 5.1 *Function* $f:\ M^2 \longrightarrow M$ *is* sensitive *if*

$$\forall i,j,k \in M\ \{f(i,j) = f(i,k) \Longleftrightarrow j = k\ \wedge\ f(j,i) = f(k,i) \Longleftrightarrow j = k\} \tag{5.5}$$

If a cell implements a sensitive function, then any change of one of its input signals will cause a change of its output signal.

Definition 5.2 (compatible pair) *A pair* $(i/S_i, j/S_j) \in D_f^2$ *is said to be compatible, if* $\mathcal{P}_f(i/S_i, j/S_j)$ *does not include* $f(i,j)$.

When f is not sensitive, there are certainly some diagnosis signal pairs which are not compatible and can not be assigned to an f cell. For example, suppose $i \neq j$ and $f(i,k) = f(j,k)$. Then $\mathcal{P}_f(i/j,\ k) = \{f(j,k)\}$, and $f(i,k) \in \mathcal{P}_f(i/j,\ k)$. This indicates that the pair $(i/j, k)$ is not compatible. When $(i/j,\ k)$ is applied to an f cell, the diagnosis signal i/j disappears in the cell, and one can not find its track at the output line at all. The signal k blockades the propagation of the diagnosis signal i/j through an f cell. Thus this pair can not be assigned to an f cell.

According to the definition of f, one can determine the set of all compatible pairs.

$$V_f = \{(i/S_i, j/S_j) \mid (i/S_i, j/S_j) \in D_f^2,\ (i/S_i, j/S_j) \text{ is compatible}\} \tag{5.6}$$

When $(i/S_i, j/S_j) \in V_f$ is applied to a cell, both diagnosis signals i/S_i and j/S_j can be propagated through the cell. The corresponding diagnosis signal received from the output must contain $f(i,j)/\mathcal{P}_f(i/S_i, j/S_j)$.

Definition 5.3 (stable kernel) *A set* W *is called* stable kernel *if every pair* $(i/S_i, j/S_j)$ *of* W^2 *is compatible, and the diagnosis signal* $f(i,j)/\mathcal{P}_f(i/S_i, j/S_j)$ *belongs to* W.

In case W is a stable kernel, one can assign every pair $(i/S_i, j/S_j) \in W^2$ to an f cell, and the corresponding diagnosis signal delivered to the output line of the f cell belongs to W.

Lemma 5.1 $\Delta_h := \{i/H_i \mid i \in H, H_i = H \setminus i\}$ *is a stable kernel if function*

$$f|_{H^2} : H^2 \longrightarrow H$$

is sensitive.

Proof: Assume $f|_{H^2} : H^2 \longrightarrow H$ to be sensitive. Then

$$\forall (i,j) \in H^2 \left\{ f(i,j) \notin \mathcal{P}_f(i/H_i, j/H_j) \wedge \mathcal{P}_f(i/H_i, j/H_j) = H_{f(i,j)} \right\}.$$

Especially for every pair $(i/H_i, j/H_j) \in \Delta_h^2$, $(i/H_i, j/H_j)$ is compatible, and the diagnosis signal $f(i,j)/\mathcal{P}_f(i/H_i, j/H_j)$ belongs to Δ_h. Thus Δ_h is a stable kernel.

Q.E.D.

Observation 5.1 *Assume W to be a stable kernel. By using a pattern one can propagate a diagnosis signal of W from each of the lines in the same level to the primary output line of $T^{(n)}$ simultaneously.*

Definition 5.4 (signal drive) *We say that:*

- *One can transform i/S_i directly into k/S_k from the left side by using j/S_j at the right side if $(i/S_i, j/S_j) \in V_f$, $f(i,j) = k$ and $\mathcal{P}_f(i/S_i, j/S_j) \subset S_k$.*
- *One can transform i/S_i directly into k/S_k from the right side by using j/S_j at the left side if $(j/S_j, i/S_i) \in V_f$, $f(j,i) = k$ and $\mathcal{P}_f(j/S_j, i/S_i) \subset S_k$.*
- *$u \in \mathcal{D}_f$ can be driven into a set $W \subset \mathcal{D}_f$ if there are $w_l, w_r \in W$, and u can be directly transformed into w_l and w_r from the left and right side, or there are $w_l, w_r \in D_f$ such that u can be directly transformed into w_l and w_r from the left and right side, and both w_l and w_r can be driven into W.*

In case one can transform u directly into w from both the left and right sides by using v at the another side, we say that u can be directly transformed into w by using v, denoted by $u \xrightarrow{v} w$.

From the above definition we can see that one can transform i/S_i directly into k/S_k from the left(right) side by using $j \in \mathcal{M}$ at the right(left) side if one can transform i/S_i directly into k/S_k from the left(right) side by using $j/S_j \in \mathcal{D}_f$ at the right(left) side.

Definition 5.5 (kernel sensitive) *Function $f : \mathcal{M}^2 \longrightarrow \mathcal{M}$ is* kernel sensitive *if there is a set $H \subset \mathcal{M}$ such that function $f|_{H^2} : H^2 \longrightarrow H$ is sensitive and every basic diagnosis signal in d_f can be driven into*

$$\Delta_h := \{i/H_i \mid i \in H \wedge H_i = H \setminus i\} \tag{5.7}$$

Example 5.1: Functions $f_1 : \{0,1\}^2 \longrightarrow \{0,1\}$, $g_1 : \{0,1,2\}^2 \longrightarrow \{0,1,2\}$, and $h_1 : \{1,2\}^2 \longrightarrow \{1,2\}$ are defined as follows:

f_1	0	1
0	0	1
1	1	1

g_1	0	1	2
0	0	1	1
1	1	1	2
2	1	2	1

h_1	1	2
1	1	2
2	2	1

The function f_1 is not kernel sensitive, and $d_{f_1} = \{0/1,\ 1/0\}$. The basic diagnosis signal set of g_1 is

$$d_{g_1} = \{0/1,\ 0/2,\ 1/0,\ 1/2,\ 2/0,\ 2/1,\}$$

Based on g_1 we can induce a sensitive function $h_1 = g_1|_{\{1,2\}^2}$. And $\Delta_{h_1} = \{1/2, 2/1\}$. It is not hard to see that every element in d_{g_1} can be driven into Δ_{h_1}. In more detail,

$$0/1 \xrightarrow{2} 1/2,\quad 1/0 \xrightarrow{2} 2/1,\quad 0/2 \xrightarrow{1} 1/2,\quad 2/0 \xrightarrow{1} 2/1$$

This indicates that g_1 is kernel sensitive.

Theorem 5.1 *Balanced uniform $T^{(n)}$ based on a function $g : \mathcal{M}^2 \longrightarrow \mathcal{M}$ is $O(\lg n)$ testable if g is kernel sensitive.*

Proof: Suppose g is kernel sensitive. Given a constant k, $T^{(2^k)}$ is $O(1)$ testable. It is sufficient to prove that all cells in the level l $(l \geq k)$ of $T^{(n)}$ are $O(1)$ testable. All input lines linked to cells in the same level of a tree system are independent of each other. Thus all faults in all cells in the same level of $T^{(n)}$ can be sensitized simultaneously. If there is a constant κ so that one can propagate every basic diagnosis signal from each of the lines in the level $l(l \geq k)$ to the primary output line of $T^{(n)}$ by using κ patterns, then all cells in the same level are $O(1)$ testable and the balanced uniform $T^{(n)}$ is $O(\lg n)$ testable.

For a kernel stable function $g : \mathcal{M}^2 \longrightarrow \mathcal{M}$, there is a set $H \subset \mathcal{M}$ so that function $h = g|_{H^2}$ is sensitive, and $\Delta_h := \{i/H_i \mid i \in H \wedge H_i = H \setminus i\}$ is a stable kernel. Furthermore, there is a constant k and every diagnosis signal in d_g can be driven into Δ_h through k transformations since D_g contains a finite number of elements.

Consider driving a given diagnosis signal $u_1 \in d_g$ from the jth primary input line to the primary output line of a balanced uniform tree $T^{(2^k)}$. Without loss of generality, we assume $j = 1$, and suppose further that by using $v_i \in \mathcal{M}$ at the right side one can transform u_i directly into u_{i+1} for $i \in [1, k]$, and $u_{k+1} \in \Delta_h$.

We enumerate the primary input lines of $T^{(2^k)}$ from the left to the right, and assign u_1 and v_1 to the first and second primary lines. Furthermore, we assign $w_i \in \mathcal{M}$ $(i \in [3, 2^k])$ to the ith primary input line of $T^{(2^k)}$ so that

$$\begin{aligned} &g(w_3, w_4) = v_2 \\ &g\underbrace{(\cdots g(}_{i-1} w_{2^{i-1}+1}, w_{2^{i-1}+2}), \cdots, g(w_{2^i-1}, w_{2^i}) \cdots) = v_i, \qquad i \in [3, k] \end{aligned}$$

These can always be done since function g is surjective.

It is easy to see that by assigning $u_1, v_1, w_3, ..., w_{2^k}$ to the 2^k primary input lines of $T^{(2^k)}$ one can propagate the diagnosis signal u_1 from the first primary input line to the primary output line of $T^{(2^k)}$, and the corresponding diagnosis signal u_{k+1} delivered to the primary output line of $T^{(2^k)}$ belongs to Δ_h.

This indicates that one can drive every basic diagnosis signal from each of the lines in the level l of $T^{(n)}$ $(l \geq k, n \geq 2^k)$ into Δ_h with 2^k patterns. As mentioned in Observation 5.1, one can propagate a diagnosis signal in a stable kernel from each of the lines in the same level to the primary output line simultaneously.

Let $\kappa = 2^k|d_f|$. All cells in the level l $(l \geq k)$ of $T^{(n)}$ can be tested by using κ n-component patterns, and $T^{(2^k)}$ is $O(1)$ testable for the given constant k. Thus $T^{(n)}$ is $O(\lg n)$ testable.

Q.E.D.

5.2 Synthesis of Functions

In this section we show that every non-kernel sensitive function can be embedded in a kernel sensitive function.

Definition 5.6 *Assume that $f : M^2 \longrightarrow M$ and $g : \mathcal{M}^2 \longrightarrow \mathcal{M}$ are two surjective functions. If there is a monomorphism $\mathcal{F} : (M, f) \longrightarrow (\mathcal{M}, g)$, then we say that f can be embedded in g, or g can cover f through $\mathcal{F}$.*

Suppose there are two monomorphisms

$$\mathcal{F}_1 : (M_1, f_1) \longrightarrow (M_2, f_2) \quad \text{and} \quad \mathcal{F}_2 : (M_2, f_2) \longrightarrow (M_3, f_3).$$

Then $\mathcal{F}_2 \circ \mathcal{F}_1$ is a monomorphism from (M_1, f_1) to (M_3, f_3). This implies that f can be embedded in h if f can be embedded in g, and g can be embedded in h.

Assume that f can be embedded in g through a monomorphism $\mathcal{F} : (M, f) \longrightarrow (\mathcal{M}, g)$, then we can define an epimorphism $\mathcal{G} : (\mathcal{M}, g) \longrightarrow (M, f)$ so that

$$\forall a \in M \, \{\mathcal{G}(\mathcal{F}(a)) = a\}$$

and

$$\forall a, b \in M \, \{f(a, b) = \mathcal{G}(g(\mathcal{F}(a), \mathcal{F}(b)))\}.$$

Theorem 5.2 *If f can be embedded in g through a monomorphism $\mathcal{F}$, then there is an epimorphism $\mathcal{G}$ such that*

$$\forall a, b, c, d \in M \, \{f(f(a, b), f(c, d)) = \mathcal{G}(g(g(\mathcal{F}(a), \mathcal{F}(b)), g(\mathcal{F}(c), \mathcal{F}(d))))\} \tag{5.8}$$

Proof : On the assumption that f can be embedded in g through an monomorphism $\mathcal{F} : (M, f) \longrightarrow (\mathcal{M}, g)$, then there is an epimorphism $\mathcal{G} : (\mathcal{M}, g) \longrightarrow (M, f)$, and

$$\forall a \in M \, \{\mathcal{G}(\mathcal{F}(a)) = a\} \quad \text{and} \quad \forall a, b \in M \, \{\mathcal{F}(f(a, b)) = g(\mathcal{F}(a), \mathcal{F}(b))\}$$

hold always. Thus

$$\begin{aligned}
f(a,b) &= \mathcal{G}(g(\mathcal{F}(a),\mathcal{F}(b))) \\
\mathcal{F}(f(a,b)) &= \mathcal{F}(\mathcal{G}(g(\mathcal{F}(a),\mathcal{F}(b)))) \\
&= g(\mathcal{F}(a),\mathcal{F}(b)) \\
f(f(a,b),f(c,d)) &= \mathcal{G}(g(\mathcal{F}(f(a,b)),\mathcal{F}(f(c,d)))) \\
&= \mathcal{G}(g(g(\mathcal{F}(a),\mathcal{F}(b)),g(\mathcal{F}(c),\mathcal{F}(d))))
\end{aligned}$$

can hold, and we can conclude that

$$\forall a,b,c,d \in M \, \{f(f(a,b),f(c,d)) = \mathcal{G}(g(g(\mathcal{F}(a),\mathcal{F}(b)),g(\mathcal{F}(c),\mathcal{F}(d))))\}.$$

Q.E.D.

This theorem states that a tree based on g can be used as a substitution for a tree based on f, if f can be embedded in g.

Example 5.2: Consider functions f_1 and g_1 defined in Example 5.1 and embed the former in the latter.

Let $M_1 = \{0,1\}$ and $\mathcal{M}_1 = \{0,1,2\}$. A monomorphism $\mathcal{F}$ can be defined as follows:

$$\mathcal{F} : (M_1, f_1) \longrightarrow (\mathcal{M}_1, g_1), \quad \mathcal{F}(0) = 0, \ \mathcal{F}(1) = 1.$$

Based on $\mathcal{F}$ we can define an epimorphism

$$\mathcal{G} : (\mathcal{M}_1, g_1) \longrightarrow (M_1, f_1), \quad \mathcal{G}(0) = 0, \ \mathcal{G}(1) = \mathcal{G}(2) = 1.$$

This indicates that f_1 can be embedded in g_1, and $T_{g_1}^{(n)}$ can simulate the function of the uniform tree $T_{f_1}^{(n)}$.

Since g_1 is kernel sensitive, $T_{g_1}^{(n)}$ is $O(\lg n)$ testable followed Theorem 5.1. However, $T_{f_1}^{(n)}$ is $\Theta(n)$ testable [Haye71].

In the rest of this section we show that every function f can be embedded in a kernel sensitive function.

Lemma 5.2 *Every function f from $\{0,1\}^2$ to $\{0,1\}$ can be embedded in a kernel sensitive function.*

Proof: There are altogether 16 distinct functions from $\{0,1\}^2$ to $\{0,1\}$. Functions $f_1, ..., f_8$ are defined as follows:

f_1	0	1
0	0	1
1	1	1

f_2	0	1
0	1	1
1	1	0

f_3	0	1
0	0	1
1	1	0

f_4	0	1
0	0	1
1	0	0

f_5	0	1
0	0	0
1	1	0

f_6	0	1
0	1	0
1	1	0

f_7	0	1
0	1	1
1	0	0

f_8	0	1
0	0	0
1	0	0

We define $\overline{0} = 1$, $\overline{1} = 0$, and function $f_i(x,y) = \overline{f_{i-8}(x,y)}$ for $i \in [9,16]$.

Based on these functions we induce two functions g and g' as follows:

g	0	1	2	3
0	$f(0,0)$	$f(0,1)$	3	2
1	$f(1,0)$	$f(1,1)$	2	3
2	3	2	2	3
3	2	3	3	2

g'	0	1	2	3
0	$f'(0,0)$	$f'(0,1)$	2	3
1	$f'(1,0)$	$f'(1,1)$	3	2
2	2	3	3	2
3	3	2	2	3

It is easy to check that both g and g' are kernel sensitive, provided that

$$f \in \{f_i \mid i \in [1,8]\}, \quad f' \in \{f_i \mid i \in [9,16]\}.$$

This indicates that every function $f_i : \{0,1\}^2 \to \{0,1\}$ can be embedded in a kernel sensitive function.

Q.E.D.

Theorem 5.3 *Every function can be embedded in a kernel sensitive function.*

Proof: Due to Lemma 5.2, it is sufficient to consider only functions from M^2 to M for $\#M > 2$. Let $H = \{m, m+1, ..., 2m-1\}$ and $\mathcal{M} = M \cup H$. Given a function $f : M^2 \longrightarrow M$, we induce a function $g : \mathcal{M}^2 \longrightarrow \mathcal{M}$ as follows:

$$g(i,j) = \begin{cases} f(i,j) & : \quad (i,j) \in M^2 \\ (j-i)\%m + m & : \quad (i,j) \in M \times H \\ (i-j)\%m + m & : \quad (i,j) \in H \times M \\ (-j-i-1)\%m + m & : \quad (i,j) \in H^2 \end{cases} \tag{5.9}$$

Let $h = g|_{H^2}$. The set of basic diagnosis signals of h is $d_h := \{i/j \mid i,j \in H \wedge i \neq j\}$. It is not hard to see that h is sensitive, and $\Delta_h := \{i/H_i \mid i \in [1,m] \wedge H_i = H \setminus i\}$ is a stable kernel. We can check that

$$\forall i,j \in \mathcal{M}\, \{i \neq j \Longrightarrow \exists k \in H\, \{g(i,k) \neq g(j,k) \wedge g(i,k)/g(j,k) \in d_h\}\} \tag{5.10}$$

$$\forall i,j \in \mathcal{M}\, \{i \neq j \Longrightarrow \exists k \in H\, \{g(k,i) \neq g(k,j) \wedge g(k,i)/g(k,j) \in d_h\}\} \tag{5.11}$$

This implies that every basic diagnosis signal $i/j \in d_g$ can be transformed from the left side as well as from the right side into an element in d_h. We know that every element in d_h can be driven into Δ_h. Thus every element in d_g can be driven into the stable kernel Δ_h, and g is kernel sensitive.

Q.E.D.

Example 5.3: Assume f to be a function from $\{0,1,2\}^2$ to $\{0,1,2\}$. A function g is defined as follows:

g	0	1	2	3	4	5
0	$f(0,0)$	$f(0,1)$	$f(0,2)$	3	4	5
1	$f(1,0)$	$f(1,1)$	$f(1,2)$	5	3	4
2	$f(2,0)$	$f(2,1)$	$f(2,2)$	4	5	3
3	3	5	4	5	4	3
4	4	3	5	4	3	5
5	5	4	3	3	5	4

Using $H = \{3,4,5\}^2$ as the domain we can induce a function $h = g|_{H^2}$. The function h is sensitive and $\Delta_h = \{3/\{4,5\}, 4/\{5,3\}, 5/\{3,4\}\}$. It is easy to see that every diagnosis signal in d_g can be driven into Δ_h. This indicates that the function g is kernel sensitive.

It is obvious that if an f cell has $2k$ binary input pins and k binary output pins, then a so induced g cell has $2k+2$ binary input pins and $k+1$ binary output pins.

5.3 Synthesis of Trees

In section 5.2 we have shown that every function can be embedded in a kernel sensitive function and have introduced a method of synthesizing kernel sensitive functions from non-kernel sensitive functions. In this section we will show that this method can be used to synthesize an $O(\lg n)$ testable tree for every tree comprising a number of different cells.

It is well known that any fanout free tree circuit can be realized with NAND and NOR gates. Therefore, the synthesis of kernel sensitive functions for NAND and NOR functions is essential. At first we consider the synthesis of a kernel sensitive function for NAND function.

Assume $M_2 = \{0,1\}$ and $\mathcal{M}_2 = \{0,1,2,3\}$. Let $f_2 : M_2^2 \longrightarrow M_2$ denote NAND function. It can be formally defined as follows:

f_2	0	1
0	1	1
1	1	0

The set of basic diagnosis signals of f_2 is $d_{f_2} = \{0/1,\ 1/0\}$. John P. Hayes had shown that $T_{f_2}^{(n)}$ is $\Omega(n^{\frac{1}{2}})$ testable [Haye71].

Based on f_2 we induce a function $g_2 : \mathcal{M}_2^2 \longrightarrow \mathcal{M}_2$ as follows:

g_2	0	1	2	3
0	1	1	2	3
1	1	0	3	2
2	2	3	3	2
3	3	2	2	3

h_2	2	3
2	3	2
3	2	3

The set of basic diagnosis signals of g_2 is

$$d_{g_2} = \{0/1,\ 0/2,\ 0/3, 1/0,\ 1/2,\ 1/3, 2/0,\ 2/1,\ 2/3,\ 3/0,\ 3/1,\ 3/2\}$$

Based on g_2 we induce a function $h_2 = g_2|_{\{2,3\}^2}$. Function h_2 is sensitive and

$\Delta_{h_2} = \{2/3,\ 3/2\}$. Every element in d_{g_2} can be driven into Δ_{h_2}. In more detail,

$$0/1 \xrightarrow{2} 2/3,\ \ 1/0 \xrightarrow{2} 3/2,\ \ 0/2 \xrightarrow{2} 2/3,\ \ 2/0 \xrightarrow{2} 3/2,\ \ 1/3 \xrightarrow{2} 3/2$$
$$3/1 \xrightarrow{2} 2/3,\ \ 0/3 \xrightarrow{0} 1/3 \xrightarrow{2} 3/2,\ \ 3/0 \xrightarrow{0} 3/1 \xrightarrow{2} 2/3$$
$$1/2 \xrightarrow{1} 0/3 \xrightarrow{0} 1/3 \xrightarrow{2} 3/2,\ \ 2/1 \xrightarrow{1} 3/0 \xrightarrow{0} 3/1 \xrightarrow{2} 2/3$$

Thus g_2 is kernel sensitive.

Let

$$\mathcal{F} : (M_2, f_2) \longrightarrow (\mathcal{M}_2, g_2),\quad \mathcal{F}(0) = 0,\ \mathcal{F}(1) = 1$$

and

$$\mathcal{G} : (\mathcal{M}_2, g_2) \longrightarrow (M_2, f_2),\quad \mathcal{G}(0) = 0,\ \mathcal{G}(1) = \mathcal{G}(2) = \mathcal{G}(3) = 1.$$

We can check that $\mathcal{F}$ and $\mathcal{G}$ are monomorphism and epimorphism, respectively. This indicates that f_2 can be embedded in g_2.

In a similar way we can show that the NOR function f_9 can be embedded in function g_9 defined below.

f_9	0	1
0	1	0
1	0	0

g_9	0	1	2	3
0	1	0	2	3
1	0	0	3	2
2	2	3	3	2
3	3	2	2	3

According to g_9 we can induce a function $g_9|_{\{2,3\}^2}$ equivalent to h_2. Every basic diagnosis signal of g_9 can be transformed into Δ_{h_2}. Then g_9 is also kernel sensitive.

Theorem 5.4 *Every fanout free circuit $T^{(n)}$ can be embedded in an $O(\lg n)$ testable tree $\mathcal{T}^{(n)}$.*

Proof : Assume that $T^{(n)}$ is a balanced tree circuit made up of gates of type NAND and NOR. We can replace every NAND and NOR gate by g_2 and g_9 cell, respectively. In this way we can obtain a balanced tree $\mathcal{T}^{(n)}$ made up of cells of type g_2 and g_9.

Cell g_2 has 4^2 distinct input pairs. For each of the 4^2 input pairs there are three possible error outputs, namely, three possible faults. There are altogether 3×4^2 possible faults. The three faults associated to an input pair can be sensitized by the same pattern. However, the corresponding three diagnosis signals perhaps can not be propagated to the primary output line by using the same pattern, and we may have to use three patterns to test the three faults associated to the input pair separately. One pattern can test at least a fault. This means that cell g_2 can be completely tested by using 3×4^2 assignments, and each assignment delivers a basic diagnosis signal which can be transformed into an element in Δ_{h_2} through three transformations. Function g_9 has the same property.

Since h_2 is a sensitive function, every pair of diagnosis signals in $\Delta_{h_2}^2$ can be propagated simultaneously through both g_2 and g_9 cells. This indicates that all cells in the same level of $\mathcal{T}^{(n)}$ can be completely tested by using $2^3 \times 3 \times 4^2$ patterns. A balanced tree system $\mathcal{T}^{(n)}$ has $\Theta(\lg n)$ levels, then it is $O(\lg n)$ testable.

Q.E.D.

Theorem 5.5 *Every tree based on surjective functions from M^2 to M can be embedded in an $O(\lg n)$ testable tree $\mathcal{T}^{(n)}$ based on surjective functions from $\mathcal{M}^2$ to $\mathcal{M}$.*

Proof: Due to Theorem 5.4, it is sufficient to consider only the cases $\#M > 2$. According to Theorem 5.3, one can induce a kernel sensitive function $g_i : \mathcal{M}^2 \longrightarrow \mathcal{M}$ for every function $f_i : M^2 \longrightarrow M$. From every kernel sensitive function g_i, one can induce a sensitive function $h = g_i|_{H^2}$.

Given a $T^{(n)}$ based on f_i, we construct a $\mathcal{T}^{(n)}$ by replacing every f_i cell by a g_i cell. Since $\#\mathcal{M} = 2m$, for a given input pair to g_i there are $2m-1$ possible faults. All diagnosis signals in d_{g_i} can be driven into the stable kernel $\Delta_h = \{i/H_i \mid i \in H \wedge H_i = H \setminus i\}$ through one transformation. Every pair of diagnosis signals in Δ_h^2 can be simultaneously propagated through every g_i cell. This indicates that all cells in the same level of $\mathcal{T}^{(n)}$ can be completely tested with $2^1 \times (2m-1)(2m)^2$ patterns. Thus all cells in the same level are $\Theta(1)$ testable, and $\mathcal{T}^{(n)}$ is $O(\lg n)$ testable.

Q.E.D.

Chapter 6

An Approach to Pseudoexhaustive Testing

6.1 Introduction

Assume that a multiple-primary-output circuit C has n primary input lines, and each of its primary output lines depends on at most k primary input lines. The generation of a pseudoexhaustive test set for the circuit C is equal to the construction of an n-column $0,1$ matrix such that every k-column submatrix contains 2^k distinct row vectors, put it differently, every k-column projection is surjective on $\{0,1\}^k$. In order to simplify the description, we use $L(n,k)$ to denote this problem as well as the row number of the desired $0,1$ matrix for two given integers n and k. The solution of $L(n,k)$ has also applications to the design of fault tolerant computing systems [Frie84, LSGH87, Wu90, Wu91].

Extensive research has been done on this subject. Tang and Woo[TaWo83] have found a method with $O(n^{\frac{k}{2}})$ upper bound for $L(n,k)$. Although their method could be used to form an acceptable solution for small n and k, it is unsuitable for large n and k. A constructive and almost optimal solution for $k = 2$ has been introduced in [CKMZ83]. In that paper a constructive solution with $O((\log n)^{k-1})$ upper bound for $L(n,k)$ is also described for general cases. Friedman, J. considers a related problem in [Frie84]. By using his result, one can construct a solution for $L(n,k)$, and the scale of $L(n,k)$ can be upper bounded by $\frac{2^{2k\log k+3k}k^4}{\log k}\log n$ [BeSi88]. It is clear that this approach will be of advantage when n is very large. This subject has also been discussed in detail in [BeSi88]. Several strategies for $L(n,k)$ have also been introduced and analyzed in [LSGH87].

In this Chapter we present a new approach based on the partition theory. By using this approach one can derive an acceptable solution for small k and practical n, and the magnitude of $L(n,k)$ can be upper bounded by $O((\log n)^{2\log k-1})$. Its highlight is that by using this approach one can reduce an $L(n,k)$ problem to a set of identical $L(N,k)$ problems ($N < (\frac{k^2}{4}+1)^2$). A nearly optimal solution for $L(n,k)$ can be constructed by combining optimal solutions for $L(N,k)$. The computational complexity of constructing the solution for $L(n,k)$ is $O(n(\log n)^{2\log k})$. In section 6.2, we introduce the basic idea of our approach. Section 6.3 presents the Partition Algorithm for constructing a special partition set with which one can reduce an $L(n,k)$ to $r(r = \lfloor 0.25k^2\rfloor + 1)$ identical $L(q,k)$, whereby q is approximately equal to $\sqrt{n}$. By using this approach recursively, an $L(n,k)$ can be reduced to a set of identical $L(N,k)$ problems. In section 6.4, $L(N,k)$ problem with small parameter N will be discussed. The application of our approach to pseudoexhaustive test generation for VLSI circuits is presented in section 6.5. In that section we analyze also the computational complexity of our approach.

6.2 Divide and Conquer

Definition 6.1 (**$P(n,k)$ Property**) *Given a set $S = \{1, 2, ..., n\}$, a partition set of r partitions*

$$P_i = \{p_{i,1}, p_{i,2}, ..., p_{i,c_i}\}, \qquad i \in [1, r]$$

of S has the $P(n,k)$ property if and only if for two arbitrary disjoint subsets $U, V \subset S$ with $|U \cup V| \leq k$, there is at least one partition P_i ($i \in [1,r]$), such that for every cell $p_{i,j}(j \in [1, c_i])$ of P_i either $p_{i,j} \cap U$ or $p_{i,j} \cap V$ is empty, put it formally,

$$\forall U, V \subseteq S \left\{ \begin{array}{c} U \cap V = \phi \wedge \ \mid U \cup V \mid \leq k \\ \Longrightarrow \exists i \in [1, r] \forall p_{i,j} \in P_i \, \{p_{i,j} \cap U = \phi \vee p_{i,j} \cap V = \phi\} \end{array} \right\} \tag{6.1}$$

The relationship between the $P(n,k)$ property and $L(n,k)$ has been discussed in [LSGH87]. We reformulate it as follows.

Lemma 6.1 *If a partition set of r partitions $P_i = \{p_{i,1}, p_{i,2}, ..., p_{i,c_i}\}$ ($i \in [1,r]$) has the $P(n,k)$ property, then*

$$L(n,k) \leq \sum_{i=1}^{r} L(\mid P_i \mid, k) - r + 1 \tag{6.2}$$

Proof: Suppose a partition set of r partitions $P_i = \{p_{i,1}, p_{i,2}, ..., p_{i,c_i}\}$ has the $P(n,k)$ property, and the matrix $\left(b_{lj}^{(i)}\right)_{l_i \times c_i}$ is a solution for $L(c_i, k)(i \in [1, r])$. Then each $l_i \times k$ submatrix of $\left(b_{lj}^{(i)}\right)_{l_i \times c_i}$ contains all 2^k distinct vectors of $\{0,1\}^k$. With these matrices we can construct a matrix $(a_{mz})_{L \times n}$ by using the Synthesis Algorithm presented in Fig. 6-1.

In the Synthesis Algorithm there are three nested loops. The outermost loop corresponds to the r partitions. For every partition P_i, an $l_i \times n$ matrix is constructed. The middle loop controls the rows of the matrix $\left(b_{lj}^{(i)}\right)_{l_i \times c_i}$. For every row $\left(b_{l1}^{(i)}, b_{l2}^{(i)}, ..., b_{lc_i}^{(i)}\right)$ the innermost loop generates an n-component row vector $(a_{m1}, a_{m2}, ..., a_{mn})$. The value of $b_{lj}^{(i)}$ is assigned to a_{mz}, when $z \in [1, n]$ is in the cell $p_{i,j}$ of the partition $P_i = \{p_{i,1}, ..., p_{i,j}, ..., p_{i,c_i}\}$. The whole algorithm constructs a $\sum_{1 \leq i \leq r} l_i$ rows and n columns matrix A, of which every $L \times k$ submatrix contains all 2^k distinct row vectors in $\{0,1\}^k$. Furthermore, we can make every matrix $\left(b_{lj}^{(i)}\right)_{l_i \times c_i}$ contain a zero row vector $(0, 0, ..., 0)$. Thus A contains altogether

```
/* Given r partitions {p_{i,1}, ..., p_{i,c_i}} of n-element set S
  and r matrices (b_{lj}^{(i)})_{l_i×c_i}, i = 1, ..., r,
  construct a matrix (a_{ij})_{L×n} by using (b_{lj}^{(i)})_{l_i×c_i}.   */
  m := 0;
  for i = 1, 2, ..., r
     for l = 1, 2, ..., l_i
     {
        m := m + 1;
        for j = 1, 2, ..., c_i
           ∀z ∈ p_{i,j}   a_{mz} := b_{lj}^{(i)};
     };
  remove the reduplicate row vectors from (a_{ij})_{L×n};
```

Fig. 6-1. Synthesis Algorithm

r zero row vectors. We keep one of them and eliminate the others. There may be also other kind of duplicate row vectors, which can be eliminated. Finally, we can obtain a matrix $(a_{mz})_{L\times n}$, with

$$L(n,k) \leq \sum_{i=1}^{r} L(\mid P_i \mid, k) - r + 1.$$

Q.E.D.

It is clear that the value on the right side of (6.2) depends on the parameter r and $L(\mid P_i \mid, k)$. We would like to generate a partition set, which makes the value on the right side of (6.2) relatively small. There is a tradeoff between r and $\mid P_i \mid$. There might exist many partitions of S. These partitions could be combined to form many partition sets having the $P(n,k)$ property. But it is very difficult to choose an optimal one from them.

Now we define a special partition set.

Definition 6.2 (simplex partition set) *A set of r partitions*

$$P_i = \{p_{i,1}, p_{i,2}, ..., p_{i,c_i}\}, \qquad i = 1, 2, ..., r$$

of the set S is called a simplex partition set *of S, if two arbitrary elements of S can share a cell in at most one of the r partitions, put it formally,*

$$\forall i, l \in [1, r] \forall p_{i,j} \in P_i \forall p_{l,m} \in P_l \{ i \neq l \Longrightarrow \mid p_{i,j} \cap p_{l,m} \mid \leq 1 \} \tag{6.3}$$

Lemma 6.2 *Partition set $\{P_1, P_2, ..., P_r\}$ has the $P(n,k)$ property, if it is a simplex partition set of S and*

$$r > \lfloor 0.25k^2 \rfloor \tag{6.4}$$

Proof: Assume U and V to be two subsets of S, and let

$$| U |= k_1, \ \ | V |= k_2, \ \ k_1 + k_2 \leq k, \ \ | U \cap V |= 0.$$

Suppose $\{P_1, P_2, ..., P_r\}$ is a simplex partition set of S. Two arbitrary elements of S can share a cell in at most one of the r partitions. Each of the elements in V can share a cell with some elements of U in at most k_1 partitions. Thus the elements of V can share some cells with the elements of U in at most $k_1 \times k_2$ partitions. Therefore, when $r > \max\{k_1 \times k_2 | k_1 + k_2 \leq k\}$, the above partition set can certainly have the $P(n, k)$ property. We know that in the integer domain

$$\max\{k_1 \times k_2 \mid k_1 + k_2 \leq k\} = \lfloor 0.25k^2 \rfloor.$$

From the assumption (6.4) follows the lemma.

Q.E.D.

$\lfloor 0.25k^2 \rfloor + 1$ is an important parameter for our approach, and we use r to denote it in the rest of this paper.

6.3 Simplex Partition Algorithm

There are many approaches to construct a simplex partition set of S. In this section we propose an algorithm to generate a special one. Suppose c is an integer not smaller than $\sqrt{n}$. We construct $c+1$ partitions of S as follows:

$$\begin{aligned}
P_0 &= \{p_{0,1}, p_{0,2}, ..., p_{0,m}\}, \quad m = \left\lceil \frac{n}{c} \right\rceil, \\
p_{0,j} &= \left\{ z \left| \left\lceil \frac{z}{c} \right\rceil = j, z \in S \right. \right\}, \quad j = 1, 2, ..., m && (6.5) \\
P_i &= \{p_{i,1}, p_{i,2}, ..., p_{i,c}\}, \quad i = 1, 2, ..., c, \\
p_{i,j} &= \left\{ z \left| z - i \times \left\lfloor \frac{z-1}{c} \right\rfloor \equiv j \bmod c, z \in S \right. \right\}, \quad j = 1, 2, ..., c && (6.6)
\end{aligned}$$

P_0 splits $[1:n]$ into $\lceil \frac{n}{c} \rceil$ integer intervals of sizes c, respectively, i.e., $P_0 = \{[1:c], [c+1, 2c], ...\}$. The above partitions can be generated by the Simplex Partition Algorithm presented in Fig. 6-2.

Example 3.1: Construct a simplex partition set for $S = \{1, 2, ..., 9\}$ and $c = 3$.

According to (6.5) the first partition is the following:

$$P_0 = \{\{1, 2, 3\}, \{4, 5, 6\}, \{7, 8, 9\}\}.$$

Based on (6.6) we can construct P_1, P_2 and P_3 as follows:

$$\begin{aligned}
P_1 &= \{\{1, 5, 9\}, \{2, 6, 7\}, \{3, 4, 8\}\}, \\
P_2 &= \{\{1, 6, 8\}, \{2, 4, 9\}, \{3, 5, 7\}\}, \\
P_3 &= \{\{1, 4, 7\}, \{2, 5, 8\}, \{3, 6, 9\}\}.
\end{aligned}$$

/* Given an n-element set S and an integer $c \geq \sqrt{n}$,
construct $c+1$ partitions of S. */
CONSTRUCT P_0:
for all $z \in S$
  if $\lceil \frac{z}{c} \rceil = j$
    put i **into cell** $p_{0,j}$;
CONSTRUCT P_i:
for $i = 1, 2, ..., c$
  for all $z \in S$
    if $z - i \times \lfloor \frac{z-1}{c} \rfloor \equiv j \bmod c$
      put z **into cell** $p_{i,j} \in P_i$;

Fig. 6-2. Simplex Partition Algorithm

$\{P_0, P_1, P_2, P_3\}$ is a simplex partition set of S.

Lemma 6.3 *If integer $c \geq \sqrt{n}$ is not a factor of $a \times b$ for arbitrary integers $a \in [1, c-1]$ and $b \in \left[1, \lfloor \frac{n-1}{c} \rfloor\right]$, then the partition set generated by the Simplex Partition Algorithm is a simplex partition set of S.*

Proof: Assume P_{i_1} and P_{i_2} to be two distinct partitions of the above $c+1$ partitions, $p_{i_1,j_1} \in P_{i_1}$ and $p_{i_1,j_2} \in P_{i_2}$. It is obvious that if i_1 or i_2 is 0, p_{i_1,j_1} and p_{i_2,j_2} have at most one common element. In other cases, an element z in p_{i_1,j_1} has to satisfy

$$z - i_1 \times \left\lfloor \frac{z-1}{c} \right\rfloor \equiv j_1 \bmod c$$

and an element z in p_{i_2,j_2} has to satisfy

$$z - i_2 \times \left\lfloor \frac{z-1}{c} \right\rfloor \equiv j_2 \bmod c$$

In other words, p_{i_1,j_1} and p_{i_2,j_2} have a common element if and only if there is an integer $z \in S$ that satisfies equation

$$\begin{cases} z - i_1 \times \left\lfloor \frac{z-1}{c} \right\rfloor \equiv j_1 \bmod c \\ \\ z - i_2 \times \left\lfloor \frac{z-1}{c} \right\rfloor \equiv j_2 \bmod c \end{cases} \qquad z \in [1, n] \tag{6.7}$$

Without loss of generality, assume $i_1 > i_2$. Then $i_1 - i_2 \in [1, c-1]$. Substitute z' for $\left\lfloor \frac{z-1}{c} \right\rfloor$. The solution z for (6.7) is unique if the solution z' for (6.8) is unique since the values of z leading to the same solution z' form an integer interval of size c. Given $i_1, j_1, i_2, j_2 \in [1, c]$, each such interval contains at most one z satisfying (6.7).

$$(i_1 - i_2) \times z' \equiv (j_2 - j_1) \bmod c \qquad z' \in \left[0, \left\lfloor \frac{n-1}{c} \right\rfloor\right] \tag{6.8}$$

If there are different solutions $z_1' > z_2'$ for (6.8), then

$$(i_1 - i_2) \times (z_1' - z_2') \equiv 0 \bmod c \qquad z_1' - z_2' \in \left[1, \left\lfloor \frac{n-1}{c} \right\rfloor\right] \tag{6.9}$$

That is to say, $(i_1 - i_2) \times (z_1' - z_2')$ could be divided by c. This is in contradiction with our assumption. For the case $i_1 < i_2$, a contradiction can also be derived. Hence there is at most one solution for equation (6.8), and the partition set generated by the Simplex Partition Algorithm is a simplex partition set of S.

Q.E.D.

The following corollary is obvious.

Corollary 6.1 *Assume $c \geq r - 1$ and let $\{P_0, P_1, ..., P_c\}$ be a simplex partition set. Then every set of r partitions out of this simplex partition set has the $P(n, k)$ property.*

□

It is easy to see that (6.8) has at most one solution $z' \in \left[0, \left\lfloor \frac{n-1}{c} \right\rfloor\right]$ if $(i_1 - i_2)$ and c have no common factor.

Corollary 6.2 *For every integer $c \geq \sqrt{n}$, P_0, P_{i_1} and P_{i_2} generated by the Simplex Partition Algorithm comprise a simplex partition set, provided that $|i_1 - i_2|$ and c have no common factor.*

□

Lemma 6.4 *If there is a prime $c \geq \max\{r - 1, \sqrt{n}\}$, we can construct a simplex partition set having the $P(n, k)$ property, and the following inequality holds.*

$$L(n, k) \leq r \times L(c, k) - r + 1.$$

This is a direct conclusion of Lemma 6.3, Corollary 6.1 and Lemma 6.1.

□

We can use the above method recursively to simplify $L(n, k)$. If we can find primes $n_1, n_2, ..., n_l$, and these primes satisfy the conditions:

$$\begin{aligned} n_1 &\geq \max\{r - 1, \sqrt{n}\}, \\ n_{i+1} &\geq \max\{r - 1, \sqrt{n_i}\}, \qquad i \in [0, l-1] \end{aligned}$$

then we can conclude that

$$\begin{aligned} L(n, k) &\leq r \times L(n_1, k) \leq r^2 \times L(n_2, k) \\ &\leq \cdots \leq r^l \times L(n_l, k) \end{aligned} \tag{6.10}$$

Theorem 6.1 *Given two integers n and $k (n \geq k)$,*

$$\begin{aligned} L(n,k) &\leq r^i \times L\left(4 \times \left\lceil n^{2^{-i}} \right\rceil, k\right) \\ &\leq 2 \binom{k^2+4}{\lfloor 0.5k \rfloor} \left(\frac{2\log n}{\log(0.25k^2+1)}\right)^{\lceil \log(0.25k^2+1) \rceil} \end{aligned} \tag{6.11}$$

holds for every integer $i \in [0, \lfloor \log\log n - \log\log r \rfloor]$.

Proof: Let $i \in [0, \lfloor \log\log n - \log\log r \rfloor]$. Choose i primes $n_1, n_2, ..., n_i$ as follows:

$$\begin{aligned} n_1 &= \min\{q \mid q \geq \max\{r-1, \sqrt{n}\} \ \ \& \ \ q \ \text{ is a prime}\}, \\ n_{j+1} &= \min\{q \mid q \geq \max\{r-1, \sqrt{n_j}\} \ \ \& \ \ q \ \text{ is a prime}\}, \quad j \in [1, i-1]. \end{aligned}$$

There is at least one prime in $\left[\left\lceil n^{2^{-1}} \right\rceil, 2\left\lceil n^{2^{-1}} \right\rceil\right]$ since there exists at least one prime $q \in [n, 2n]$ for arbitrary integer n [Huxl72].

Notice that

$$\left\lceil n^{2^{-i}} \right\rceil = \underbrace{\left\lceil \left\lceil \cdots \left\lceil \left\lceil n^{2^{-1}} \right\rceil^{2^{-1}} \right\rceil \cdots \right\rceil^{2^{-1}} \right\rceil}_{i-\text{time}} \tag{6.12}$$

and $\lceil a \rceil \times \lceil b \rceil \geq \lceil a \times b \rceil$, if $a, b \geq 1$. Therefore

$$\begin{aligned} n_1 &\leq 4 \times \left\lceil n^{2^{-1}} \right\rceil, \\ n_j &\leq 4 \times \underbrace{\left\lceil \cdots \left\lceil \left\lceil n^{2^{-1}} \right\rceil^{2^{-1}} \right\rceil \cdots \right\rceil}_{j-\text{time}} \\ &= 4 \times \left\lceil n^{2^{-j}} \right\rceil, \qquad\qquad \text{for} \quad j \in [1, i] \end{aligned}$$

and

$$L(n,k) \leq r^i \times L\left(4 \times \left\lceil n^{2^{-i}} \right\rceil, k\right).$$

We know

$$\begin{aligned} r &= 2^{\log r}, \\ r^{\lfloor \log\log n - \log\log r \rfloor + 1} &\leq \left\lfloor \left(\frac{2\log n}{\log r}\right)^{\log r} \right\rfloor \\ \log\log\left(n^{2^{-\lfloor \log\log n - \log\log r \rfloor}}\right) &\leq 1 + \log\log r \\ n^{2^{-\lfloor \log\log n - \log\log r \rfloor}} &\leq r^2. \end{aligned}$$

Using the method proposed in [TaWo83], we can estimate that $L(n,k) \leq 2\binom{n}{\lfloor 0.5k \rfloor}$.

Let $i = \lfloor \log\log n - \log\log r \rfloor$. Then

$$\begin{aligned} L(n,k) &\leq r^i \times L\left(4 \times \left\lceil n^{2^{-i}} \right\rceil, k\right) \\ &\leq r^i \times L(4r^2, k). \end{aligned}$$

As assumed above, there is at least one prime $q \in [2r, 4r]$. This implies that $L(4r^2, k) \leq r \times L(4r, k)$, and

$$
\begin{aligned}
L(n,k) &\leq r^{i+1} \times L(4r,k) \\
&\leq 2\binom{4r}{\lfloor 0.5k \rfloor} r^{\lfloor \log\log n - \log\log r \rfloor + 1} \\
&\leq 2\binom{k^2+4}{\lfloor 0.5k \rfloor} \left(\frac{2\log n}{\log(0.25k^2+1)}\right)^{\lceil \log(0.25k^2+1) \rceil}.
\end{aligned}
$$

Q.E.D.

We have tested that there is at least one prime in $[n^2, (n+1)^2]$ for $n \in [1, 10^5]$, Considering practical applications, inequality (6.13) is also meaningful.

$$
L(n,k) \leq r^i \times L\left(\left(\left\lceil n^{2^{-i-1}} \right\rceil + 2\right)^2, k\right) \tag{6.13}
$$

for $i \in [0, \log\log n - \log\log r]$, $n, k \in [1, 10^{10}]$.

Theorem 6.2 *$L(n,2) \leq 8\log n$ and $L(n,3) \leq 8(\log n)^{\log 3}$ for $n \geq 3$.*

Proof: For arbitrary integer $c \geq \max\{2, \sqrt{n}\}$, partitions P_0, P_1, P_2 generated by the Simplex Partition Algorithm comprise a simplex partition set of an n-element set according to Corollary 6.2. Since $r := \lfloor 0.25k^2 \rfloor + 1$ is not greater than 3 for $k \leq 3$, then P_0, P_1, P_2 has the $P(n,3)$ property based on Lemma 6.2. Following Lemma 6.1,

$$
\begin{aligned}
L(n,k) &\leq r^1 \times L\left(\left\lceil n^{2^{-1}} \right\rceil, k\right) \\
&\leq r^i \times L\left(\left\lceil n^{2^{-i}} \right\rceil, k\right)
\end{aligned}
$$

for $k \leq 3$ and every $i \in [0, \log\log n - \log\log r]$. Notice that

$$
n = 2^{2^{\log\log n}} \leq 2^{2^{\lceil \log\log n \rceil}} = 4^{2^{\lceil \log\log n \rceil - 1}}.
$$

Thus

$$
\begin{aligned}
L(n,2) &\leq L(2^{2^{\lceil \log\log n \rceil}}, 2) \\
&\leq 2^{\lceil \log\log n \rceil} L(2^{2^{\lceil \log\log n \rceil - \lceil \log\log n \rceil}}, 2) \\
&\leq 2 \cdot \log n \cdot L(2,2). \\
L(n,3) &\leq L(4^{2^{\lceil \log\log n \rceil - 1}}, 3) \\
&\leq 3^{\lceil \log\log n \rceil - 1} L(4^{2^{\lceil \log\log n \rceil - 1 - \lceil \log\log n \rceil + 1}}, 3) \\
&\leq 3^{\lfloor \log\log n \rfloor} L(4,3). \\
&\leq 3^{\log\log n} L(4,3).
\end{aligned}
$$

It is easy to see $L(2,2) = 4$ and $L(4,3) = 8$. Then we have the theorem.

Q.E.D.

6.4 Basic Problems

Using the Simplex Partition Algorithm presented in Fig. 6-2, we can reduce an $L(n,k)$ problem to a set of $L(N,k)$ problems, where N is smaller than r^2. We call these small problems *basic problems*. Special strategies should be adopted to search for solutions for them. For tow given integers N and k, we can construct many partition sets having the $P(N,k)$ property. Through different partition sets we can derive different solutions for $L(N,k)$. We give an example to demonstrate the importance of choosing the partition set appropriately.

Example 4.1: Let $S = \{1, 2, ..., 27\}$ and $k = 6$. Solve $L(27,6)$.

The parameter $r := \lfloor 0.25 \times k^2 \rfloor + 1$ is equal to 10, and we give two solutions for $L(27,6)$.

1) Solve $L(27,6)$ by using the partition set

$$\begin{aligned} P_i &= \{p_{i,1}, p_{i,2}, ..., p_{i,14}\}, \quad i \in [1,10], \\ p_{i,j} &= \{a_j, b_j | a_j + b_j \equiv i \bmod 27, a_j, b_j \in [1,27]\}, \quad j \in [1,14]. \end{aligned}$$

With this partition set, we can reduce a single $L(27,6)$ problem to ten $L(14,6)$ problems. Using the method of Tang and Woo [TaWo83], we can evaluate that

$$\begin{aligned} L(14,6) &\leq 455 \\ L(27,6) &\leq 10 \times L(14,6) \\ &\leq 10 \times 455 \\ &= 4550. \end{aligned}$$

2) Solve $L(27,6)$ by using the partition set

$$\begin{aligned} P_0 &= \{\{1,2,...,9\}, \{10,11,...,18\}, \{19,20,...,27\}\}; \\ P_i &= \{p_{i,1}, p_{i,2}, ..., p_{i,9}\}, \quad i \in [1,9], \\ p_{i,j} &= \left\{ z \,\middle|\, z - i \times \left\lfloor \frac{z-1}{9} \right\rfloor \equiv j \bmod 9, z \in [1,27] \right\}, \quad j \in [1,9]. \end{aligned}$$

This is a simplex partition set. By using this partition set, we can reduce the same $L(27,6)$ to an $L(3,6)$ and nine $L(9,6)$ problems. $L(9,6)$ is not greater than 120 according to the method in [TaWo83]. $L(3,6)$ is equal to 8. The upper boundary for the solution of $L(27,6)$ can be established as follows:

$$\begin{aligned} L(27,6) &\leq L(3,6) + 9 \times L(9,6) \\ &\leq 8 + 9 \times 120 \\ &= 1088. \end{aligned}$$

The above example demonstrates clearly that different partition sets could derive very different consequences. The approach of Tang and Woo is very suitable to $L(N,k)$ problem, when N is not larger than $2k$. We can adopt their method to construct a solution for $L(N,k)$, in case $N \leq 2k$. The solution scale generated by using their method can be estimated easily, then the solution scale of $L(N,k)$ can be evaluated. With these additional heuristic information we can use our method to reduce $L(n,k)$ to some special basic problems, and use the method of Tang and Woo to generate solutions for them, finally form a good solution for $L(n,k)$. In some cases, we would like to reduce $L(n,k)$ to a set of identical basic problems for simplifying the set partition and the solution synthesis. This strategy may slightly increase the solution scale for $L(n,k)$.

6.5 Application and Computation Complexity

The solution of $L(n,k)$ can be applied to many fields. In this section, an example is given to show its application to the pseudoexhaustive test generation for VLSI circuits. Finally we analyze the computational complexity of this method.

Example 5.1: Consider a multiple-primary-output circuit C with 9 primary input lines, of which every primary output line depends on at most three primary input lines. It is required to generate a pseudoexhaustive test set for this circuit. This is an $L(9,3)$ problem. We solve it in three steps:

Step 1: *Reduce* $L(9,3)$ *to three identical instances of* $L(3,3)$*;*

Step 2: *Solve* $L(3,3)$*;*

Step 3: *Construct a solution for* $L(9,3)$ *using the solution for* $L(3,3)$.

Let $S = \{1, 2, ..., 9\}$. The following three partitions constitute a simplex partition set of S.

$$\begin{aligned} P_0 &= \{\{1,2,3\},\{4,5,6\},\{7,8,9\}\}, \\ P_1 &= \{\{1,5,9\},\{2,6,7\},\{3,4,8\}\}, \\ P_2 &= \{\{1,6,8\},\{2,4,9\},\{3,5,7\}\}. \end{aligned}$$

For this example, the parameter $r := \lfloor 0.25 \times 3^2 \rfloor + 1$ is equal to 3. According to Lemma 6.2, the above simplex partition set has the $P(9,3)$ property. An $L(9,3)$ can be reduced to three identical $L(3,3)$ by using this partition set.

The solution for $L(3,3)$ is just the following 8×3 matrix $(b_{ij})_{8\times 3}$.

$$(b_{ij})_{8\times 3} = \begin{bmatrix} 0&0&0\\0&0&1\\0&1&0\\0&1&1\\1&0&0\\1&0&1\\1&1&0\\1&1&1 \end{bmatrix}, \quad (a_{mz})_{8\times 9} = \begin{bmatrix} 0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&1&1&1\\0&0&0&1&1&1&0&0&0\\0&0&0&1&1&1&1&1&1\\1&1&1&0&0&0&0&0&0\\1&1&1&0&0&0&1&1&1\\1&1&1&1&1&1&0&0&0\\1&1&1&1&1&1&1&1&1 \end{bmatrix}$$

The remaining task is to construct a 9-column 0, 1-matrix based on the above partition set and the solution for $L(3,3)$. This can be done by using the Synthesis Algorithm

$$(a_{mz})_{20\times 9} = \begin{bmatrix} 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&1&1 \\ 0&0&0&1&1&1&0&0&0 \\ 0&0&0&1&1&1&1&1&1 \\ 1&1&1&0&0&0&0&0&0 \\ 1&1&1&0&0&0&1&1&1 \\ 1&1&1&1&1&1&0&0&0 \\ 1&1&1&1&1&1&1&1&1 \\ & & & & & & & & \\ 0&0&1&1&0&0&0&1&0 \\ 0&1&0&0&0&1&1&0&0 \\ 0&1&1&1&0&1&1&1&0 \\ 1&0&0&0&1&0&0&0&1 \\ 1&0&1&1&1&0&0&1&1 \\ 1&1&0&0&1&1&1&0&1 \\ & & & & & & & & \\ 0&0&1&0&1&0&1&0&0 \\ 0&1&0&1&0&0&0&0&1 \\ 0&1&1&1&1&0&1&0&1 \\ 1&0&0&0&0&1&0&1&0 \\ 1&0&1&0&1&1&1&1&0 \\ 1&1&0&1&0&1&0&1&1 \end{bmatrix}$$

presented in Fig. 6-1.

Based on the partition P_0, we can generate an 8×9 matrix $(a_{mz})_{8\times 9}$. In this matrix the first, second and third columns are equal, since the three elements $1, 2, 3$ of S belong to one cell in P_0. For the same reason, the fourth, fifth and sixth columns are identical and the seventh, eighth and ninth columns are identical, too.

In the same way we can construct two 8×9 matrices for both partitions P_1 and P_2. Putting the three 8×9 matrices together and omitting the duplicate rows, finally we get a 20×9 matrix $(a_{mz})_{20\times 9}$.

In matrix $(a_{mz})_{20\times 9}$, the first 8 rows have been constructed according to P_0, the following 6 rows based on P_1 and the last 6 rows corresponding to P_2. In $(a_{mz})_{20\times 9}$, every 3-column projection contains 2^3 distinct vectors in $\{0,1\}^3$, therefore, the row vectors of $(a_{mz})_{20\times 9}$ constitute a pseudoexhaustive test set for the circuit C.

Example 5.2: Determine an upper bound for $L(1024, 9)$.

For $k = 9$, $r = \lfloor 0.25 \times k^2 \rfloor + 1 = 21$. 37 is a prime greater than $\sqrt{1024}$. According to Lemma 6.4

$$L(1024, 9) \leq 21 \times L(37, 9) - 21 + 1.$$

Using the approach of Tang and Woo[TaWo83], we can determine that

$$\begin{aligned} L(37,9) &\le \binom{37}{\lfloor \frac{9}{2} \rfloor} + \binom{37}{9 - \lfloor \frac{9}{2} \rfloor - 1} \\ &= 2\binom{37}{4} \\ &= 132090. \\ L(1024,9) &\le 21 \times 132090 - 21 + 1 \\ &= 2773870. \end{aligned}$$

The upper bound of $L(1024, 9)$ strongly depends upon that of $L(37, 9)$. Following [CKMZ83], $L(n,k) \le \lceil k2^k \ln n \rceil$ for $n \ge 2$. Thus $L(37,9) \le \lceil 9 \cdot 2^9 \ln 37 \rceil$, and $2\binom{37}{4}$ is much larger than the smallest upper boundary for $L(37, 9)$. In case one can generate an optimal solution for $L(37, 9)$, he can construct a nearly optimal solution for $L(1024, 9)$.

For k to be 3, 5, 7 and 9, the parameter r equals 3, 7, 13 and 21, respectively. The upper bounds of $L(n, k)$ corresponding to different parameters n and k are listed in Table 6-1. It shows that the method presented in this paper has considerable advantage for small k and practical size of n.

In the rest of this section we discuss the computational complexity of our approach. Let $\mathcal{C}(n,k)$ denote the quantum of the computation for generating a solution of $L(n, k)$. Assume that $L(n, k)$ can be reduced to $L(q, k)$ directly. Then $\mathcal{C}(n, k)$ is the sum of the computation quantities of $\mathcal{C}(q, k)$ and those for constructing r partitions of an n-element set to reduce $L(n, k)$ to $L(q, k)$ and for synthesizing a solution of $L(n, k)$ with that of $L(q, k)$. It is not hard to see that the complexity of the Simplex Partition Algorithm is $O(rn)$. The Synthesis Algorithm has three loops. The outermost loop is corresponding to r partitions. For every P_i, it cycles one time. The middle loop is limited by the row number of the 0, 1-matrix $\left(b_{l_j}^{(i)}\right)_{l_i \times c_i}$ for $L(q, k)$. We know $l_i \le L(q, k)$. The innermost loop checks each of the n elements which cell it belongs to in the partition P_i. Then the complexity of the Synthesis Algorithm is $O(r \times L(q,k) \times n)$.

We have thus

Theorem 6.3

$$\mathcal{C}(n,k) = O\left(n \binom{k^2+4}{\lfloor 0.5k \rfloor} \left(\frac{2\log n}{\log(0.25k^2+1)}\right)^{\lceil \log(0.25k^2+1) \rceil}\right) \tag{6.14}$$

Proof: Based on Theorem 6.1, an $L(n, k)$ can be reduced to basic problems $L(n_i, k)$ ($n_i \le 4r$) through a number of steps.

$$L(n,k) \longrightarrow L(n_1,k) \longrightarrow L(n_2,k) \longrightarrow \cdots \longrightarrow L(n_i,k), \qquad n_j \;\le\; 4\left\lceil n^{2^{-j}} \right\rceil.$$

Thus

$$\mathcal{C}(n,k) \;=\; O(rn) + \mathcal{C}(n_1,k) + O(rn \times L(n_1,k))$$

$$L(16,3) \le 3 \cdot L(4,3) - 2 \le 3 \cdot 2 \cdot \binom{4}{1} - 2 = 22$$

$$L(64,3) \le 3 \cdot L(8,3) - 2 \le 3 \cdot 2 \cdot \binom{8}{1} - 2 = 46$$

$$L(256,3) \le 3 \cdot L(16,3) - 2 \le 64$$

$$L(1024,3) \le 3 \cdot L(32,3) - 2 \le 3(3 \cdot L(6,3) - 2) - 2 \le 9 \cdot 2 \cdot \binom{6}{1} - 8 = 100$$

$$L(16,5) \le 2 \cdot \binom{16}{2} = 240$$

$$L(64,5) \le 7 \cdot L(11,5) - 6 \le 7 \cdot 2 \cdot \binom{11}{2} - 6 = 764$$

$$L(256,5) \le 7 \cdot L(17,5) - 6 \le 7 \cdot 2 \cdot \binom{17}{2} - 6 = 1898$$

$$\begin{aligned} L(1024,5) &\le 7 \cdot L(37,5) - 6 \le 7(7 \cdot L(7,5) - 6) - 6 \\ &\le 49 \cdot 2 \cdot \binom{7}{2} - 48 = 2010 \end{aligned}$$

$$L(16,7) \le 2 \cdot \binom{16}{3} = 1120$$

$$L(64,7) \le 13 \cdot L(13,7) - 12 \le 13 \cdot 2 \cdot \binom{13}{3} - 12 = 7424$$

$$L(256,7) \le 13 \cdot L(17,7) - 12 \le 13 \cdot 2 \cdot \binom{17}{3} - 12 = 17668$$

$$\begin{aligned} L(1024,7) &\le 13 \cdot L(37,7) - 12 \le 13(13 \cdot L(13,7) - 12) - 12 \\ &\le 169 \cdot 2 \cdot \binom{13}{3} - 168 = 96500 \end{aligned}$$

$$L(16,9) \le 2 \cdot \binom{16}{4} = 3640$$

$$L(64,9) \le 21 \cdot L(23,9) - 21 \le 21 \cdot 2 \cdot \binom{23}{4} - 21 = 371890$$

$$L(256,9) \le 21 \cdot L(23,9) - 21 \le 21 \cdot 2 \cdot \binom{23}{4} - 21 = 371890$$

$$L(1024,9) \le 21 \cdot L(37,9) - 21 \le 21 \cdot 2 \cdot \binom{37}{4} - 21 = 2773870$$

Table 6-1. Upper Bounds of $L(n,k)$ for Different n and k

$$
\begin{aligned}
&= O(rn \times L(n_1,k)) + O(rn_1) + \mathcal{C}(n_2,k) + O(rn_1 \times L(n_2,k)) \\
&= \mathcal{C}(4r,k) + O(rn \times L(n_1,k)) + O\left(\sum_{1 \le j \le i-1} rn_j \times L(n_{j+1},k) \right) \\
&= \mathcal{C}(4r,k) + O(rn \times L(n_1,k)) + O\left(\sum_{1 \le j \le i-1} r \left\lceil n^{2^{-j}} \right\rceil \times L(n_{j+1},k) \right).
\end{aligned}
$$

Using the method proposed in [TaWo83], we can estimate that

$$
\begin{aligned}
\mathcal{C}(4r,k) &\le 4r \times 2 \binom{4r}{\lfloor 0.5k \rfloor} \\
&= (2k^2+8) \binom{k^2+4}{\lfloor 0.5k \rfloor} \\
&= O\left(k^2 \binom{k^2+4}{\lfloor 0.5k \rfloor} \right).
\end{aligned}
$$

It is not hard to see that for $n > 2$ and $i \le \lfloor \log n \rfloor + 1$

$$
\sum_{1 \le j \le i-1} \left\lceil n^{2^{-j}} \right\rceil = O(n).
$$

Furthermore, $n_1 \le 4n^{2^{-1}}$. Based on Theorem 6.1,

$$
\begin{aligned}
L(n_1,k) &\le r^{\lfloor \log\log n - \log\log r \rfloor} L(4 \left\lceil n^{2^{-\lfloor \log\log n - \log\log r \rfloor - 1}} \right\rceil, k) \\
&\le r^{\lfloor \log\log n - \log\log r \rfloor} L(4r,k).
\end{aligned}
$$

This means that

$$
\begin{aligned}
\mathcal{C}(n,k) &= O(rn \times L(n_1,k)) \\
&= O\left(nr^{\lfloor \log\log n - \log\log r \rfloor + 1} \binom{4r}{\lfloor 0.5k \rfloor} \right) \\
&= O\left(n \binom{k^2+4}{\lfloor 0.5k \rfloor} \left(\frac{2\log n}{\log(0.25k^2+1)} \right)^{\lceil \log(0.25k^2+1) \rceil} \right).
\end{aligned}
$$

Q.E.D

Table 6-2 shows the upper bounds of $L(n,k)$ and $\mathcal{C}(n,k)$ for different approaches. The bounds for $L(n,k)$ is listed in the second column, and those for $\mathcal{C}(n,k)$ in the third column.

	$L(n,k)$	$\mathcal{C}(n,k)$
TaWo83	$2\binom{n}{\lfloor 0.5k \rfloor}$	$O\left(n\binom{n}{\lfloor 0.5k \rfloor}\right)$
CKMZ83	$2^k(\log n)^{k-1}$	$O(2^k n(\log n)^k)$
Frie84	$2^{2k\log k+3k}k^4 \cdot \frac{\log n}{\log k}$	$O\left(n^{k(k-1)}\right)$
H. Wu	$2\binom{4r}{\lfloor 0.5k \rfloor}\left(\frac{2\log n}{\log r}\right)^{\lceil \log r \rceil}$	$O\left(n\binom{4r}{\lfloor 0.5k \rfloor}\left(\frac{2\log n}{\log r}\right)^{\lceil \log r \rceil}\right)$

Table 6-2. Upper Bounds of $L(n,k)$ and $\mathcal{C}(n,k)$ where $r = 0.25k^2 + 1$

Chapter 7

Monomial Oriented Pseudorandom Test

7.1 Introduction

Pseudorandom pattern generation techniques have two important applications to VLSI test. One is to compute a short random test preceding the long and laborious deterministic test to catch easily detected faults [Breu71], the other is to design built-in self test circuitries [AbCe83, AgCe81, BuSi82, KMZ79]. In the first case we pursue its low test generation cost and the good "practical fault coverage", namely, the fault coverage of the first hundreds or thousands patterns. In the second case we seek its potential test generation capability and the cheap overhead of the hardware implementations. If the circuit structure is known, one can perhaps construct an appropriate biased random test generator with the available information to achieve a great improvement. The input signal biased random test and pattern biased random test [Hart91] are typical examples [SLC71, Wund87].

Assume a pseudorandom pattern generator $G(n)$ to be of n bits and $n \geq 30$. The so called three basic properties of $G(n)$ are the maximum-length property, the window property and the run property [BMS87]. For the application to the test generation of VLSI, these properties are not essential indeed, because only the first N ($N << 2^n$) patterns generated by $G(n)$ can be used in practice. It would be nice, when the first N patterns could contain the most important patterns which can either cover the most part of the concerned faults or detect some faults having big error latencies [ShMc75]. The weighted random pattern generations are just based on this thought. In order to generate properly weighted test patterns, the circuit analysis have to be done, and the corresponding generator requires more hardware overhead. In some cases, the circuit analysis result may show that both logic 1 and logic 0 input probabilities are quit balanced for every primary input lines. Conventionally, one adopts the uniform pseudorandom technique for such kind of circuits. The results with this technique sometimes are disappointing. The practical fault coverages are often low, when the circuits under test have a large number of primary input lines.

It occurs often that to test a random test resistant fault one has to apply a certain combination to some primary input lines, while other primary input lines are free. We can say that this fault can only be tested by using a certain monomial. There exists certainly a constant $k \leq n$ such that every fault can be tested by a k-monomial. According to these hints, we propose some monomial oriented pseudorandom techniques. These techniques can be used either to generate random test patterns to precede the deterministic test or to design built-in self test circuits. For the latter application, the hardware overhead is acceptable and the practical fault coverage is also good. It is particularly suitable to pseudorandom test generations for circuits with a large number of primary input and output lines. The experiments of these techniques on various benchmark circuits have given a considerable good results in terms of both the fault coverage and hardware overhead.

7.2 k-Monomial and Its Probability

Let $B = \{0,1\}, \mathcal{N} = \{0,1,...,n-1\}$, and $S = \left\{(i_1, i_2, ..., i_k) \mid (i_1, ..., i_k) \subset \mathcal{N}^k, i_1 < \cdots i_k\right\}$.

Definition 7.1 *A k -monomial over B^n is a expression of $X_{i_1}^{a_1}...X_{i_k}^{a_k}$, where $(i_1, ..., i_k) \in S$ are pairwise different indices and $(a_1, ..., a_k) \in B^k$.*

A representative of the k-monomial $X_{i_1}^{a_1}...X_{i_k}^{a_k}$ is a vector in B^n. Its i_1th, ..., i_kth components are $a_1, ..., a_k$, respectively, and the other components are free.

It is easy to see that $|S| = \begin{pmatrix} n \\ k \end{pmatrix}$. Let $L = |S|$. We order the elements of S from 1 to L. Given an element $(i_1, ..., i_k)$ of S we can construct 2^k distinct k -monomials $X_{i_1}^{a_1} \cdots X_{i_k}^{a_k}$. Let $M = \{X_{i_1}^{a_1} \cdots X_{i_k}^{a_k} \mid (i_1, ..., i_k) \in S, a_i \in B\}$. As mentioned, there exists a constant $k \leq n$ and for every irredundant fault there is at least a k -monomial $X_{i_1}^{a_1} \cdots X_{i_k}^{a_k}$ such that each of its representative can detect this fault. Then we focus our attention on the generation of the representative of the k-monomials.

Assume $(i_1, i_2, ..., i_k)$ to be the ith element of S and $a_1...a_k$ is the binary code of j. We denote the monomial $X_{i_1}^{a_1} \cdots X_{i_k}^{a_k}$ by $M_{i,j}$. There are total $L \times 2^k$ distinct k-monomials. Assume $p(M_{i,j}, l)$ to be the probability that one of the representative of $M_{i,j}$ is included in the first l patterns generated by $G(n)$. Then

$$p(M_{i,j}, l) = 1 - \left(1 - \frac{1}{2^k}\right)^l \tag{7.1}$$

Let $N = L \times 2^k$. We can determine the following measurement

$$F(l) = \frac{1}{N} \sum_{i=1}^{L} \sum_{j=1}^{2^k} p(M_{i,j}, l) \tag{7.2}$$

and call it *imaginary fault coverage*. $F(l)$ reflects the practical fault coverage.

Formula (7.1) is right if the random sampling is taken with replacement. For the pseudorandom pattern generation, a pattern which has already been generated will not appear until next period. Then the imaginary fault coverage evaluation based on formula (7.1) is not precise. The method proposed in [ChMc87] can be used to estimate $p(M_{i,j}, l)$ more accurately. The corresponding result is displayed by the following formula:

$$p(M_{i,j}, l) = 1 - \prod_{s=0}^{l-1} \frac{2^n - 2^{n-k} - s}{2^n - s} \tag{7.3}$$

In case $p(M_{i,j}, l)$ is independent of the indices i and j,

$$F(l) = p(M_{i,j}, l) \tag{7.4}$$

Theorem 7.1 *In case $p(M_{i,j}, l)$ is independent of the indices i and j, $F(l)$ is a monotonic decrease function of n.*

Proof: It is obvious that $\frac{2^n - 2^{n-k} - s}{2^n - s}$ is a monotonic increase function for given integers s and $k \leq n$. Thus $F(l)$ is a monotonic decrease function of n.

Q.E.D.

7.3 Expected Test Length

In this section we discuss the expected test length l for the monomial oriented pseudorandom test generation. We consider the following random game.

Assume that there are u balls in a black **Box** . Among them are v black balls. We sample balls from the black **Box** without replacement. Take j as a random variable. Suppose $P(j, x, u, v)$ represents the probability that we have just sampled x black balls after jth sampling, and the last sampled ball is black.

Example 7.1: Assume u, v and x to be 5, 1, and 1, respectively. We can determine that

$$\begin{aligned}
P(1,1,5,1) &= \frac{1}{5}, \\
P(2,1,5,1) &= \frac{4}{5} \cdot \frac{1}{4} = \frac{1}{5}, \\
P(3,1,5,1) &= \frac{4}{5} \cdot \frac{3}{4} \cdot \frac{1}{3} = \frac{1}{5}, \\
P(4,1,5,1) &= P(5,1,5,1) = \frac{1}{5}.
\end{aligned}$$

Example 7.2: Assume u, v and x to be 10, 2, and 1, respectively. We have

$$\begin{aligned}
P(1,1,10,2) &= \frac{2}{10} = \frac{9}{45}, \\
P(2,1,10,2) &= \frac{8}{10} \cdot \frac{2}{9} = \frac{8}{45}, \\
P(3,1,10,2) &= \frac{8}{10} \cdot \frac{7}{9} \cdot \frac{2}{8} = \frac{7}{45}, \\
P(i,1,10,2) &= \frac{10-i}{45}, \qquad i \in [1,9].
\end{aligned}$$

By generalizing the above examples, we obtain:

$$P(j,x,u,v) = 0 \quad \text{for} \quad j < x \quad \text{or} \quad j > u - v + x \tag{7.5}$$

$$P(x,x,u,v) = \prod_{i=0}^{x-1} \frac{v-1}{u-1} \tag{7.6}$$

and

$$P(j,x,u,v) = \left(\begin{array}{c} j-1 \\ j-x \end{array} \right) \prod_{i=0}^{x-1} \frac{v-i}{u-i} \prod_{t=0}^{j-x-1} \frac{u-v-t}{u-x-t} \quad \text{for } x < j \leq u-v+x \tag{7.7}$$

Furthermore

$$\begin{aligned} \sum_{j=1}^{u} P(j,x,u,v) &= \sum_{j=x}^{u-v+x} P(j,x,u,v) \\ &= \prod_{i=0}^{x-1} \frac{v-i}{u-i} + \sum_{j=x+1}^{u-v+x} \left(\begin{array}{c} j-1 \\ j-x \end{array} \right) \prod_{i=0}^{x-1} \frac{v-i}{u-i} \prod_{t=0}^{j-x-1} \frac{u-v-t}{u-x-t} \\ &= 1 \end{aligned} \tag{7.8}$$

and

$$\sum_{j=x+1}^{u-v+x} \left(\begin{array}{c} j-1 \\ j-x \end{array} \right) \prod_{i=0}^{x-1} \frac{v-i}{u-i} \prod_{t=0}^{j-x-1} \frac{u-v-t}{u-x-t} = 1 - \prod_{i=0}^{x-1} \frac{v-i}{u-i} \tag{7.9}$$

Let $E(j,x,u,v)$ and $V(j,x,u,v)$ denote the mean and variance of random variable j. From the above formulas we can estimate them as follows:

$$\begin{aligned} E(j,x,u,v) &= \sum_{j=1}^{\infty} jP(j,x,u,v) \\ &= \sum_{j=x}^{u-v+x} jP(j,x,u,v) \\ &= x \prod_{i=0}^{x-1} \frac{v-i}{u-i} + \sum_{j=x+1}^{u-v+x} j \left(\begin{array}{c} j-1 \\ j-x \end{array} \right) \prod_{i=0}^{x-1} \frac{v-i}{u-i} \prod_{t=0}^{j-x-1} \frac{u-v-t}{u-x-t} \\ &= x \prod_{i=0}^{x-1} \frac{v-i}{u-i} + \sum_{j=x+1}^{u-v+x} x \left(\begin{array}{c} j-1 \\ j-x \end{array} \right) \prod_{i=0}^{x-1} \frac{v-i}{u-i} \prod_{t=0}^{j-x-1} \frac{u-v-t}{u-x-t} \\ &+ \sum_{j=x+1}^{u-v+x} (j-x) \left(\begin{array}{c} j-1 \\ j-x \end{array} \right) \prod_{i=0}^{x-1} \frac{v-i}{u-i} \prod_{t=0}^{j-1} \frac{u-v-t}{u-x-t} \\ &= x + \sum_{j=x+1}^{u-v+x} (j-x) \left(\begin{array}{c} j-1 \\ j-x \end{array} \right) \prod_{i=0}^{x-1} \frac{v-i}{u-i} \prod_{t=0}^{j-x-1} \frac{u-v-t}{u-x-t}. \end{aligned}$$

Notice that

$$
\begin{aligned}
& \sum_{j=x+1}^{u-v+x}(j-x)\left(\begin{array}{c} j-1 \\ j-x \end{array}\right)\prod_{i=0}^{x-1}\frac{v-i}{u-i}\prod_{t=0}^{j-x-1}\frac{u-v-t}{u-x-t} \\
= & \sum_{j=x+1}^{u-v+x} x\left(\begin{array}{c} j-1 \\ j-(x+1) \end{array}\right)\prod_{i=0}^{x-1}\frac{v-i}{u-i}\prod_{t=-1}^{j-(x+1)-1}\frac{u-(v+1)-t}{u-(x+1)-t} \\
= & x\sum_{j=x+1}^{u-(v+1)+x+1}\left(\begin{array}{c} j-1 \\ j-(x+1) \end{array}\right)\prod_{i=0}^{x}\frac{v+1-i}{u-i}\cdot\frac{u-x}{v+1}\prod_{t=0}^{j-(x+1)-1}\frac{u-(v+1)-t}{u-(x+1)-t}\cdot\frac{u-v}{u-x} \\
= & \frac{u-v}{v+1}x\sum_{j=x+1}^{u-(v+1)+x+1}\left(\begin{array}{c} j-1 \\ j-(x+1) \end{array}\right)\prod_{i=0}^{x}\frac{v+1-i}{u-i}\prod_{t=0}^{j-(x+1)-1}\frac{u-(v+1)-t}{u-(x+1)-t} \\
= & \frac{x(u-v)}{v+1}.
\end{aligned}
$$

Then

$$E(j,x,u,v)=\frac{x(u+1)}{v+1} \tag{7.10}$$

The following Corollary is obvious.

Corollary 7.1 *In respect of the calculation without regard to the domain of x, we can state that:*

$$E(j,x,w,v)=E(j,E(j,x,u,v),w,u) \tag{7.11}$$

for $v \leq u \leq w$.

In the same way, we can calculate the second moment $E(j^2,x,u,v)$, of random variable j.

$$
\begin{aligned}
E(j^2,x,u,v) &= \sum_{j=1}^{u} j^2 P(j,x,u,v) \\
&= x^2\prod_{i=0}^{x-1}\frac{v-i}{u-i}+\sum_{j=x+1}^{u-v+x} j^2\left(\begin{array}{c} j-1 \\ j-x \end{array}\right)\prod_{i=0}^{x-1}\frac{v-i}{u-i}\prod_{t=0}^{j-x-1}\frac{u-v-t}{u-x-t} \\
&= x^2\prod_{i=0}^{x-1}\frac{v-i}{u-i}+\sum_{j=x+1}^{u-v+x} xj\left(\begin{array}{c} j \\ j-x \end{array}\right)\prod_{i=0}^{x-1}\frac{v-i}{u-i}\prod_{t=0}^{j-x-1}\frac{u-v-t}{u-x-t} \\
&= \frac{x^2(u+1)}{v+1}\prod_{i=0}^{x}\frac{v+1-i}{u+1-i} \\
&+ \sum_{j=x+2}^{u-v+x+1} x(j-1)\left(\begin{array}{c} j-1 \\ j-(x+1) \end{array}\right)\prod_{i=0}^{x}\frac{v+1-i}{u+1-i}\cdot\frac{u+1}{v+1}\prod_{t=0}^{j-x-2}\frac{u-v-t}{u-x-t} \\
&= \frac{x(u+1)}{v+1}\sum_{j=x+1}^{u+1-(v+1)+x+1}(j-1)P(j,x+1,u+1,v+1)
\end{aligned}
$$

$$
\begin{aligned}
&= \frac{x(u+1)}{v+1}\cdot\frac{(x+1)(u+2)-v-2}{v+2} \\
&= \frac{x(u+1)(x(u+2)+u-v)}{(v+1)(v+2)} \qquad (7.12)
\end{aligned}
$$

Clearly

$$
\begin{aligned}
Var(j,x,u,v) &= E(j^2,x,u,v) - E^2(j,x,u,v) \\
&= \frac{x(u+1)(x(u+2)+u-v)}{(v+1)(v+2)} - \frac{x^2(u+1)^2}{(v+1)^2} \\
&= \frac{x(u+1)(u-v)}{(v+1)(v+2)}(1-\frac{x}{v+1}) \qquad (7.13)
\end{aligned}
$$

Now we return to the original topic. Let $u = 2^n$ and $v = 2^{n-k}$. Then u and v denote the cardinalities of B^n and $X_{i_1}^{a_1}\cdots X_{i_k}^{a_k}$, respectively. We have

$$
E(j,x,2^n,2^{n-k}) = \frac{x(2^k+2^{k-n})}{1+2^{k-n}} \qquad (7.14)
$$

and

$$
\begin{aligned}
Var(j,x,2^n,2^{n-k}) &= \frac{x(2^n+1)(2^n-2^{n-k})}{(2^{n-k}+1)(2^{n-k}+2)}(1-\frac{x}{2^{n-k}+1}) \\
&= \frac{x(2^k+2^{k-n})(2^k-1)}{(1+2^{k-n})(1+2^{k-n+1})}(1-\frac{x}{2^{n-k}+1}) \qquad (7.15)
\end{aligned}
$$

$E(j,1,2^n,2^{n-k})$ is just the expected test length. That is to say, $E(j,1,2^n,2^{n-k})$ patterns generated continuously by $G(n)$ can be expected to include a representative for every k-monomial. The following theorem is obvious.

Theorem 7.2 *$E(j,1,2^n,2^{n-k})$ and $Var(j,1,2^n,2^{n-k})$ are monotonic increase functions of n for the given integer $k < n$.*

□

According to Theorem 7.1 and 7.2, we can conclude that the long period of the pseudorandom pattern generator $G(n)$ is negative for the practical fault coverage and the expected test length.

7.4 Experiments

We have already had an impression that the long period of the pseudorandom pattern generator is not of benefit to the practical fault coverage, while the short period can not guarantee the high potential fault test capability. It is really a contradiction. LFSR is one of the most popular pseudorandom pattern generation techniques. Some tactics can be used to comprise the forenamed contradiction in LFSR technique to certain extent to improve the practical fault coverage.

7.4.1 Multiple LFSRs

Assume that

$$\begin{aligned} y_{n_i}^{(t+1)} &= c_{n_i} y_{n_{i+1}-1}^{(t)}, \\ y_j^{(t+1)} &= y_{j-1}^{(t)} + c_j y_{n_{i+1}-1}^{(t)}, \qquad j \in [n_i+1, n_{i+1}-1] \end{aligned}$$

for $i \in [1, m]$ define m LFSRs. Their periods are $p_1, p_2, ..., p_m$, respectively. If mLFSR is an n-bit pattern generator constructed by concatenating the m LFSRs together, then the period of mLFSR is the smallest common multiple of $p_1, p_2, ..., p_m$.

It is not hard to see that if all the m LFSRs are based on primitive polynomials, and their periods are prime to each other, then the period of mLFSR is not far smaller than 2^n. Therefore, mLFSR can have a very high potential test generation capability. Table 7.1 shows the practical fault coverages of mLFSRs.

In Table 7.1, we list the experimental results on benchmark circuits. For each circuit, we adopt three or more generators to generate random test set for single stuck-at faults. One of them is the traditional LFSR based on a single primitive polynomial. Its degree n is equal to the number of the primary input lines of the circuit under test. Such an LFSR has a period of $2^n - 1$. Others are mLFSRs constructed by concatenating m LFSRs together. The rows marked sLFSR and mLFSR($m \geq 2$) represent the percents of the fault coverages for the test sets produced by using a single LFSR and mLFSRs, respectively. The number of the primary input lines of the circuit under test is listed in the column labeled INP. The numbers listed in the brackets following mLFSR denote the degrees of m primitive polynomials used to construct an mLFSR. The seeds for all of the generators are 1010... This table shows that the practical fault coverage of mLFSR is quit different from that of sLFSR with such kind of seeds. Such differences are rather distinct with seed 1000... So far as regards the practical fault coverage, the multiple LFSR generators are obviously better than the single LFSR generators. The hardware overhead of mLFSR is not much more expensive than that of sLFSR. Hence it is also suitable for built-in self test circuits.

7.4.2 Multiple Seeds

If we can fit an mLFSR with several seeds, the practical fault coverage can be improved further. Table 7.2 demonstrates the experimental results on some benchmark circuits. The columns labeled with #Seeds, #Redundant, and #Untested represent the numbers of used seeds, redundant faults and untested faults, respectively. The simulations were performed by a cell oriented fault simulator [HSS92]. From this table we can see that except C7552 and C2670, all combinational benchmark circuits can be completely tested with at most three seeds. In order to embed the test set in a BIST environment, one can adopt the technique proposed in [AgCe81] to construct a test generator consisting of a ROM and mLFSR. Fig. 7.1 demonstrates the main logic structure of such a test generator called *store and generate BIT*.

When many seeds are required to reach 100% fault coverage, the method proposed in [AkJa89] can be used to embed those seeds in a circuit comprising a counter and an

XOR array. Fig. 7.2 illustrates one of the possible embedding formats. The k-bit LFSR and XOR array can generate total $2^k - 1$ seeds. For each of these seeds, the l-bit LFSR controls the mLFSR to produce $2^l - 1$ patterns. Table 7.3 shows the experimental results on benchmark circuits with this technique. The fault coverage reaches to 100% for circuit C2670 after 10 test patterns have been embedded in the counter-XOR-array.

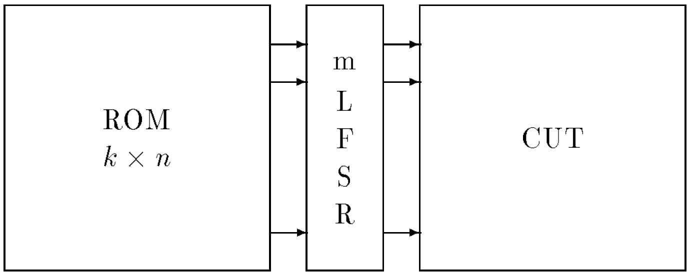


Fig. 7.1 Store and Generate BIT

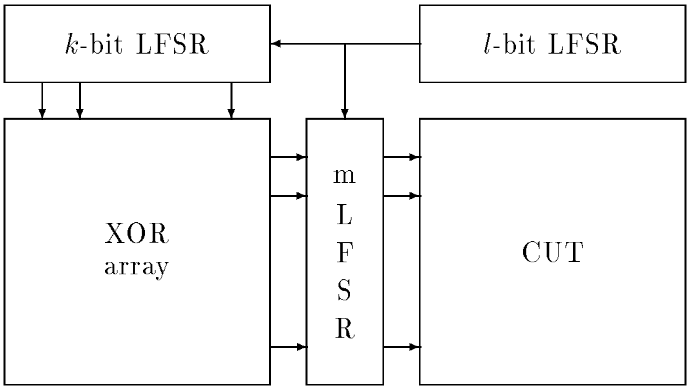


Fig. 7.2 Embedding Format

Circuits	INP	RTPGs	Fault Coverages (%)				
			32	64	128	256	1024
C6288	32	sLFSR	98.0	99.2	99.3	99.3	99.3
		3LFSR(9,11,12)	98.4	99.2	99.3	99.3	99.3
		4LFSR(6,7,9,10)	98.5	99.3	99.3	99.3	99.3
C432	36	sLFSR	52.4	74.0	80.8	87.2	91.6
		3LFSR(11-13)	64.1	75.1	83.3	89.7	91.6
		4LFSR(7,8,10,11)	64.5	74.7	85.3	88.8	91.2
C1908	33	sLFSR	67.5	73.1	76.5	83.7	93.3
		3LFSR(10,12)	72.1	76.6	80.0	86.0	93.1
		4LFSR(6,8-10)	60.1	64.9	73.1	81.7	94.2
C499	41	sLFSR	61.2	75.1	84.4	90.7	97.0
		3LFSR(12,14,15)	72.9	81.0	87.8	90.9	96.9
		4LFSR(8,10-12)	63.2	82.6	87.2	91.5	96.5
C3540	50	sLFSR	48.1	57.3	71.8	81.3	88.7
		2LFSR(24-26)	44.9	56.8	75.1	82.7	88.9
		3LFSR(15,17,18)	61.5	71.6	81.7	85.8	89.1
		4LFSR(11-14)	67.7	75.1	82.2	86.0	89.3
		5LFSR(8-12)	66.6	75.9	82.6	85.9	89.1
C880	60	sLFSR	53.6	62.8	71.4	76.8	83.5
		2LFSR(29,31)	59.0	67.5	73.1	86.2	96.9
		3LFSR(19-21)	75.2	80.2	85.5	92.1	97.9
		4LFSR(13,14,16,17)	82.0	86.1	88.6	93.7	97.9
		5LFSR(10-14)	77.9	86.8	91.1	95.0	98.1
C2670	157	sLFSR	38.1	42.2	52.5	60.7	69.3
		5LFSR(29-31,33,34)	54.1	64.7	71.0	76.2	81.9
		7LFSR(19-22,24-26)	58.8	68.5	72.8	77.6	82.0
		9LFSR(13-17,19-22)	66.5	71.4	75.6	79.1	82.0
		11LFSR(9-16,18-20)	65.5	72.0	77.3	79.9	81.9
C5315	178	sLFSR	33.4	43.4	50.5	60.1	80.4
		6LFSR(27-31,33)	68.2	79.7	88.1	96.1	98.4
		8LFSR(18,19,21-26)	68.6	84.5	92.6	96.0	98.5
		10LFSR(13-19,21-23)	67.7	84.2	89.1	94.4	98.4
		12LFSR(9-16,18-21)	72.7	84.0	91.5	96.5	98.6

Table 7.1: Faults Coverage Comparison Between sLFSRs and mLFSRs

Circuits	INP	#Seeds	Periods	#Redundant	#Untested	Fault Coverages(%)
C432	36	1	2^{10}	40	0	100
C499	41	2	2^{10}	8	0	100
C880	60	3	2^{12}	0	0	100
C1355	41	2	2^{10}	8	0	100
C1908	33	3	2^{12}	9	1	100
C2670	233	10	2^{12}	117	0	93.01
C3540	50	3	2^{12}	137	0	100
C5315	178	3	2^{12}	59	0	100
C6288	32	1	2^{10}	34	0	100

Table 7.2: Faults Coverages for Multiple-Seed mLFSRs

Circuits	INP	k	l	#Redundant	Fault Coverages(%)
C432	36	1	10	40	100
C499	41	2	10	8	100
C880	60	2	12	0	100
C1355	41	2	10	8	100
C1908	33	2	12	9	100
C2670	233	10	12	117	100
C3540	50	2	12	137	100
C5315	178	2	12	59	100
C6288	32	1	10	34	100

Table 7.3: Faults Coverages for Multiple-Seed Embedding mLFSRs

Concluding Remarks

The object set $\mathcal{A}$ of X-category $(\mathcal{T}, \mathcal{A}, Q, Z, \circ, \times)$ includes all combinational circuits. Our discussion on the test complexity concerns only on the irredundant circuits in $\mathcal{A}$ and assumes that we can always find a test pattern for every single fault in the concerned circuit.

If we want to limit $\mathcal{A}$ to the set of all testable circuits, then we have to redefine the functions Q and Z so that for two elements $B_i, B_j \in \mathcal{A}$, $B_j \circ B_i$ has meaning if and only if $Z(B_i)$ contains a complete test set for B_j, and at least one of the complete test sets of B_i can be propagated through B_j.

It is NP-complete to decide whether a given single stuck-at fault in a circuit is detectable[IbSa75, FuTo82, Fuji85]. Thus it seems to be very unlikely that there is an efficient algorithm for deciding whether $B_j \circ B_i$ is testable.

Given a testable circuit $C \in \mathcal{A}$, how to generate a complete test set for it depends on its structure, size and the available information about C. In case C is a uniform tree based on a function f, the generation of the optimum complete test set is related to the analysis of its test complexity. We can state that one can always construct an optimum complete test set for C whenever he can determine its test complexity exactly.

The construction of a complete assignment set to a uniform tree is the first step towards the generation of a complete test set to the tree. Table 8.1 illustrates the classification of the assignment complexity of balanced uniform trees.

General	Commutative
$\Theta(1)$	$\Theta(1)$
$\Theta((\lg n)^{\alpha})$ $\alpha \in \left[\frac{1}{m-1}, 1\right]$	$\Theta(\lg n)$

Table 8.1 Classification of the Assignment Complexity of Balanced Uniform Trees

Tables 8.2 and 8.3 show the classification of the test complexity of balanced uniform trees. A uniform tree is either $\Theta(1)$ or $\Omega((\lg n)^{\beta})$ $(\beta > 0)$ testable, and the test complexity of uniform trees based on commutative functions can be divided into $\Theta(1)$, $\Theta(\lg n)$ and $\Omega(n^{\alpha})$. To decide whether a uniform tree C is $\Theta(1)$ testable is equal to judge if there is a finite set $X \subset \mathbf{N}^t$ such that

$$\forall i \in [1, k] \, \{\{A_i\vec{x} \mid \vec{x} \in X\} \subset \{A_0\vec{x} \mid \vec{x} \in X\}\},$$

where A_i $(i \in [0, k]$ are 0, 1 matrices associated with the definition of the function f.

For a balanced uniform tree $T_f^{(n)}$ based on a commutative function f, the above problem is equal to that of deciding whether a linear equation system associated with function f has a feasible solution. In case $T_f^{(n)}$ is $\Theta(1)$ testable, to generate the optimum test set for $T_f^{(n)}$ one has to solve an integer programming associated with the function.

General	Commutative
$\Theta(1)$	$\Theta(1)$
$\Theta((\lg n)^{\beta})$ $\beta > 0$	$\Theta(\lg n)$
$\Theta(n^{\alpha})$ $\alpha \in (0, 1]$	$\Theta(n^{\alpha})$

Table 8.2 Classification of the Test Complexity of Balanced Uniform Trees Based on Function $f : M^2 \longrightarrow M$

<table>
<tr><th>General</th><th>Commutative</th><th>Unate</th></tr>
<tr><td>$\Theta(1)$</td><td>$\Theta(1)$</td><td rowspan="3">$\Theta(n^{\alpha})$
$\alpha \in (0, 1]$</td></tr>
<tr><td>$\Theta((\lg n)^{\gamma})$
$\gamma \geq 1$</td><td>$\Theta(\lg n)$</td></tr>
<tr><td>$\Theta(n^{\alpha})$
$\alpha \in (0, 1]$</td><td>$\Theta(n^{\alpha})$
$\alpha \in (0, 1]$</td></tr>
</table>

Table 8.3 Classification of the Test Complexity of Balanced Uniform Trees Based on Function $f : \{0, 1\}^k \longrightarrow \{0, 1\}$

In Chapter 3, we expand the assignment and test problems into a more general combinational problem, namely, the so called arrangement problem. The assignment and test problems can be considered as two instances of the arrangement problem.

Let F_i denote the i-level balanced uniform tree based on f. We have shown that a sufficient condition for the arrangement complexity of $T_f^{(n)}$ to be $\Theta(1)$ is the following:

There are an $i \in \mathbf{N}$, a subset $S \subset M^{k^i}$ and k^i bijective mappings $\pi_1, ..., \pi_{k^i} : S \longrightarrow S$ so that S is a complete arrangement of F_i and

$$\underbrace{(F_i \times \cdots \times F_i)}_{k^i} \circ \underbrace{(\pi_1 \times \cdots \times \pi_{k^i})} \circ D_\sigma^K(S) = S,$$

where σ represents the type of S.

For symmetrical function f, the above condition is even necessary. However, whether it is the necessary condition for general functions is still a open problem.

Chapter 5 shows that an arbitrarily given tree can be embedded in an $O(\lg n)$ testable tree. It is also an interesing subject to synthesize a hardware optimal $O(\lg n)$ testable tree for a given $\Omega(n^r)(r > 0)$ testable tree.

If C has a number of primary output lines and n primary input lines, and every primary output line depends on at most k primary input lines ($k < n$), the pseudoexhaustive test may be a suitable approach for C. We use $L(n, k)$ to denote the pseudoexhaustive test problem for such a circuit. The algorithm presented in Chapter 6 reduces a big problem $L(n, k)$ to a small $L(N, k)$ ($N << n$). If one can generate an optimal solution for $L(N, k)$, then he can construct an considerable good solution for $L(n, k)$. The remains problem is how to generate an optimal solution for $L(N, k)$.

In case pseudorandom test is required, it is very worth choosing a suitable random pattern generator. We have shown that monomial oriented pseudorandom pattern generators are better than the traditional single LFSR, and may be a right alternative. An interesting theoretical and practical problem is how to construct the real monomial oriented pseudorandom pattern generators.

Bibliography

[AbCe83] M. E. Aboulhamid, and E. Cerny. A Class of Test Generators for Built -In Testing. *IEEE Trans. on Computers,* C-32(10), pp.957-959, 1983.

[AbGa81] J. A. Abraham, and D. D. Gajski. Design of Testable Structures Defined by Simple Loops. *IEEE Trans. on Computers*, C-30, pp.875-884, 1981.

[AgCe81] V. K. Agarwal, E. Cerny. Store and Generate Built-in-Testing Approach. *Proc. 11th FTCS* , June 1981, pp.35-40.

[Aker73] S. B. Akers. Universal Test Sets for Logic Networks. *IEEE Trans. on Computers*, Vol. C-22 No.9 pp.835-839, 1973.

[AkJa89] S. B. Akers, and W. Jansz. Test Set Embedding in a Built-In Self-Test Environment. *Proc. 1989 ITC* , pp.257-263.

[Beck87] B. Becker. An Easily Testable Optimal Time VLSI-Multiplier. *ACTA Informatica*, Vol. 24, pp.363-380, 1987.

[Beck88] B. Becker. Efficient Testing of Optimal-Time Adders. *IEEE Trans. on Computers*, C-37, pp.1113-1121, 1988.

[BeHa90] B. Becker, and J. Hartmann. Optimal-Time Multipliers and C-Testability. *Proceedings of the 2nd Annual Symposium on Parallel Algorithms and Architectures*, pp.146-154, 1990.

[Benn84] R. G. Bennetts. Design of Testable Logic Circuits. Addison Wesley, 1984.

[BeSi88] B. Becker, and Hans-Ulrich Simon. How Robust Is the n-Cube? *Information and Computation* Vol. 77 No. 2, pp.162-178, 1988.

[BeSp91] B. Becker, and U. Sparmann. Computations over Finite Monoids and their Test Complexity. *Theoretical Computer Science,* pp.225-250, 1991.

[BhHa86] D. Bhattacharya, and J. P. Hayes. Fast and Easily Testable Implementation of Arithmetic Functions. *Proceedings of the 16th International Symposium on Fault Tolerant Computing Systems*, pp.324-329, July 1986.

[BMS87] P. H. Bardell, W. H. McAnney and J. Savir. **Built-In Test for VLSI: Pseudorandom Techniques**, John Wiley & Sons, New York 1987.

[BrFr76] M. A. Breuer, A.D. Friedman. **Diagnosis & Reliable Design of Digital Systems**. Computer Science Press, INC. 1976.

[Breu71] M. A. Breuer. A Random and an Algorithmic Technique for Fault Detection Test Generation for Sequential Circuits. *IEEE Trans. on Computers*, Vol. C-20(11), pp.1364-1370, 1971.

[BSMS81] Z. Barzilai, J. Savir, G. Markowsky, and M. G. Smith. The Weighted Syndrome Sums Approach to VLSI Testing. *IEEE Trans. on Computers*, Vol. C-30(12), pp. 996-1000, 1981.

[BuSi82] M. G. Buehler, and M. W. Sievers. Off-Line Built-In Test Techniques for VLSI Circuits. *Computer*, 15(6), pp.69-82, 1982.

[ChMc87] Cary K. Chin, and Edward J. McCluskey. Test Length for Pseudorandom Testing. *IEEE Trans. on Computers*, Vol. C-36, No. 2, pp.252-256, 1987.

[CKMZ83] A. K. Chandra, Lawrence T. Kou, George Markowsky, and Shmuel Zaks. On Sets of Boolean n-Vectors with all k-Projections Surjective. *Acta Informatic* Vol. 20, pp.103 - 111, 1983.

[Eldr59] R. D. Eldred. Test Routines Based on Symbolic Logical Statements. *J. Assoc. Comput. Mach.* **6**(1), pp.33-36, 1959.

[Frie84] J. Friedman. Constructing $O(n \log n)$ Size Monotone Formulae for the k-th Elementary Symmetric Polynomial of n Boolean Variables. *Proceedings 25th IEEE Found. of Comput. Sci.*, pp.506-515, 1984.

[Fuji85] H. Fujiwara. **Logic Testing and Design for Testability.** MIT Press Series in Computer Systems, 1985.

[FuTo82] H. Fujiwara, and S. Toida. The Complexity of Fault Detection: An Approach to Design for Testability. *Proc. 12th Int. Symp. FTC*, pp.101-108, 1982.

[Hart91] J. Hartmann. The Random Testability of the n-Inputs AND Gate. *Proceeding of STACS 91, 8th Annual Symposium on Theoretical Aspects of Computer Science*, pp.488-498, 1991.

[Hart93] J. Hartmann. On Numerical Weight Optimization for Random Testing. *Proceeding of the joint EDAC and EUROASIC Conf.* 1993.

[Haye71] J. P. Hayes. On Realizations of Boolean Functions Requiring a Minimal or Near-Minimal Number of Tests. *IEEE Trans. on Computers* Vol. C-20, No. 12, pp.1506-1513, 1971.

[Haye74] J. P. Hayes. On Modifying Logic Networks to Improve their Diagnosability. *IEEE Trans.on Computers*, C-23(1): pp.56-62, 1974.

[Hotz65] G. Hotz. Eine Algebraisierung des Syntheseproblems von Schaltkreisen. *EIK* **1**, 185-205, 209-231, 1965.

[Hotz74] G. Hotz. **Schaltkreistheorie**. Walter de Gruyter · Berlin · New York 1974.

[Hotz90] G. Hotz. **Einführung in die Informatik**. B. G. Teubner Stuttgart 1990.

[HSFH87] L. K. Horning, J. M. Soden, R. R. Fritzemeier, and C. F. Hawkins. Measurements of Quiescent Power Supply Current for CMOS ICs in Production Testing. *Proceeding of ITC 1987,* pp.300-309.

[HSS92] J. Hartmann, B. Schieffer, and U. Sparmann. Cell Oriented Fault Simulation. *Proceedings of the European Simulation Multiconference 92*, pp. 424-429, 1992.

[Huel72] M. N. Huxley. **The Distribution of Prime Numbers.** (pp.119)OXFORD, CLARENDON PRESS: 1972.

[IbSa75] O. H. Ibarra, S. K. Sahni. Polynomially Complete Fault Detection Problems. *IEEE Trans. on Computers*, C-24, pp.242-249, 1975.

[KMZ79] B. Könemann, J. Mucha, and G. Zwiehoff. Built-in Logic Block Observation Techniques. *Proceeding of ITC*, pp.37-41, 1979.

[LaFi80] R. E. Ladner, and M. J. Fisher. Parallel Prefix Computation. *Journal of the ACM*, 27(4), pp.831-838, 1980.

[Lala85] P. K. Lala. **Fault Tolerant and Fault Testable Hardware Design**. Prentice Hall, 1985.

[LSGH87] M. Livingston, Q. Stout, N. Graham and F. Harary. Subcube Fault-Tolerance in Hypercubes. *Technical Report*, Computing Research Laboratory, University of Michigan, Sept. 1987

[Mark76] G. Markowsky. A Straightforward Technique for Producing Minimal Multiple Fault Test Sets for Fanout-Free Combinational Circuits. *Technical Report RC 6222, IBM T. J. Watson Research Center*, Yorktown Heights, 1976.

[MaSu82] Y. K. Malaiya, and S.Y.H. Su. A New Fault Model and Testing Technique for CMOS Devices. *Proceeding of 1982 ITC*, pp.25-34

[Mehl84] K. Mehlhorn, **Data Structure and Algorithms 2: Graph Algorithms and NP-Completeness**, Springer-Verlag, 1984.

[PaSt82] Christos H. Papadimitriou, and K, Steiglitz, **Combinatorial Optimization**– Algorithms and Complexity, Prentice-Hall, Inc., 1982.

[Redd73] S. Reddy. Complete Test Sets for Logic Functions. *IEEE Trans. on Computers* Vol. C-22, No. 11 pp.1016-1020, 1973.

[Roth66] J. P. Roth. Diagnosis of Automata Failures: A Calculus and a Method. IBM J. Res. Dev. 10: pp278-281, 1966.

[SaRe74] K. K. Saluja, and S. M. Reddy. On Minimally Testable Logic Networks. *IEEE Trans. on Computers*, C-23(1): pp.552-554, 1974.

[SeKo77] S. C. Seth, and K.L. Kodandapani. Diagnosis of Faults in Linear Tree Networks. *IEEE Trans. on Computers*, C-26(1), pp.29-33, 1977.

[ShMc75] J. J. Shedletsky, and E. J. McCluskey. The Error Latency of a Fault in a Combinational Digital Circuit. *5th Annual FTCS*, pp.210-214, 1975.

[SLC71] H. D. Schnurmann, E. Lindbloom, and R. G. Carpenter. The Weighted Random Test-Pattern Generator. *IEEE Trans. on Computers*, Vol. C-24(7), pp.695-700, 1971.

[TaWo83] D. T. Tang, and L. S. Woo. Exhaustive Test Pattern Generation with Constant Weight Vectors. *IEEE Trans. on Computers*, Vol. C-32, No. 12. pp.1145-1150, 1983.

[WCMc87] K. D. Wagner, C. K. Chin, and Edward J. McCluskey. Pseudorandom Testing. *IEEE Trans. on Computers*, Vol. C-36, No. 3, pp.332-343, 1987.

[Wu90] H. Wu, "Pseudoexhaustive Test Generators", Technical Report, SFB 124, TP, B1 19/1990, Fachbereich 14, Informatik, Universität des Saarland, Germany.

[Wu91] H. Wu. On $L \times n$ Boolean Matrices with All $L \times k$ Submatrices Having 2^k Distinct Row Vectors. *Proceedings, EDAC 91*, pp.528-532, 1991.

[Wu92a] H. Wu. On Tests of Uniform Tree Circuits. *Proceeding of CONPAR VAPP V*, pp.527-538, 1992.

[Wu92b] H. Wu, "On the Test Complexity of Uniform Tree Circuits", Technical Report, SFB 124-B1, 06/1992, FB-14, Informatik, Universität des Saarlandes, Germany.

[Wu92c] H. Wu, "On the Test Complexity of Tree VLSI Systems", Technical Report, SFB 124-B1, 08/1992, FB-14, Informatik, Universität des Saarlandes, Germany.

[WuSp92] H. Wu, U. Sparmann. On the Assignments Complexity of Tree VLSI Systems. *Proceeding of 7th International Symposium on Computer and Information Sciences*, pp.97-103, 1992.

[Wu93a] H. Wu. On the Assignments Complexity of Uniform Trees. *Proceeding of International Symposium on Symbolic and Algebraic Computation, ISSAC' 93*, pp.95-104.

[Wu93b] H. Wu. Synthesis of $O(\lg n)$ Testable Trees. *Proceeding of the ninth International Conference on Fundamental of Computation Theory, FCT 93*, pp.452-461.

[Wu93c] H. Wu. On n-Column $0,1$-Matrices with All k-Projections Surjective, *Acta Informatic* Vol. 31, pp.285 - 299, 1994.

[Wund87] H. J. Wunderlich. On Computing Optimized Probabilities for Random Tests. *Proceeding 24th DAC*, pp.392-398, 1987.