

5

Spectral Interpretation of Decision Diagrams

5.1 Decision Diagrams

What are decision diagrams?

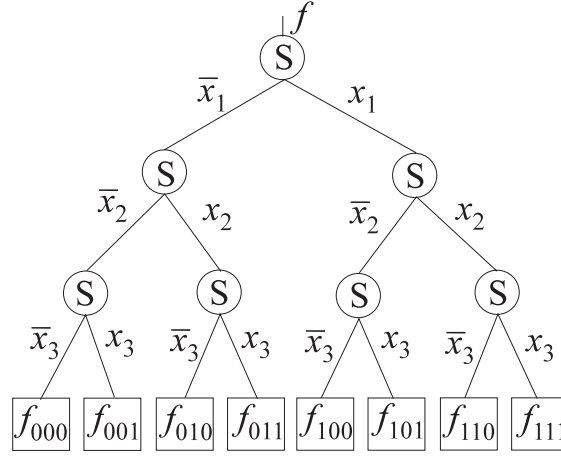
Definition 31 *Decision diagrams (DDs) are data structures for representing discrete functions.*

Switching functions and MV functions are particular examples of discrete functions on groups C_2^n and C_p^n , respectively.

DDs are derived by the reduction of decision trees (DTs)[195, 196]. The reduction of a DT into a DD for a given function f is possible due to some particular properties of f that assure some relationships between the function values, and is performed by exploiting these properties. Thus, DTs are the basic concept in DDs representation of discrete functions, since they do not depend on the properties of a particular function. The same class of DDs may differ considerably for different functions. Therefore, when discussing and comparing features of different classes of DDs, we should consider DTs from which DDs are derived to stay independent on the properties of particular functions.

What are DTs?

DTs were first used to represent discrete sets in 1935 in the Ph.D. Thesis by Prof. Dj. Kurepa [129]; hence they are called the Kurepa trees in mathematics. DTs were used to represent all switching functions of a given

Figure 5.1. BDT for $n = 3$.

number of variables [130, 204]. In the context of present applications, to represent a particular function, DTs were used to represent switching functions by Akers [7], and to represent MV functions by Thayse, Davio, and Deshamps [263], both in 1978.

The present interest in DDs is the result of a 1986 paper by Bryant [19], in which the binary DDs (BDDs) for switching functions were used. We will introduce and illustrate this concept using the example of three-variable switching functions.

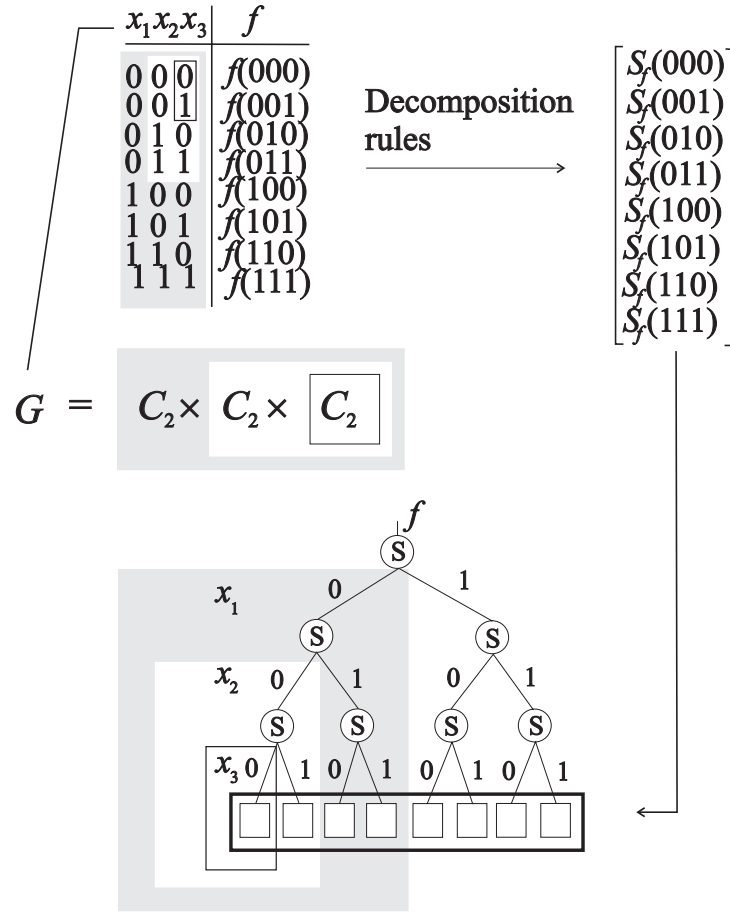
Example 45 Fig. 5.1 shows BDT for $n = 3$. This tree is also called the Shannon tree [187], by referring to the paper of Shannon [204], since it is the graphical representation of the decomposition of f performed by the recursive application of the Shannon decomposition rule (S-decomposition rule)

$$f = \bar{x}_i f_0 \oplus x_i f_1,$$

where $f_0 = f(x_i = 0)$, $f_1 = f(x_i = 1)$, for all the variables in f .

In a BDT, we consider non-terminal nodes and constant nodes represented by circles and squares, respectively. The non-terminal nodes to which the same variable is assigned form a level in the BDT. For each non-terminal node we consider the incoming and outgoing edges. The incoming edge at the root node is labeled by the represented function f . The outgoing edges are labeled by the values 0 and 1, or equivalently, the negative and positive literals \bar{x}_i and x_i . The constant nodes show the function values at the points (x_1, x_2, x_3) and are labeled by f_{ijk} , where $i, j, k \in \{0, 1\}$.

In a BDT, the values of constant nodes are the function values of f . Various DTs can be defined by using different decomposition rules to assign f to the DT. In these cases, the constant nodes represent the values S_f

Figure 5.2. Correspondence among G , f , and DT for f .

calculated from the values of f , which depend on the decomposition rules used assigned to non-terminal nodes in the DT.

Fig. 5.2 shows a correspondence among f , the truth-table for f , the group G where f is defined, and a decision tree for f . A DT expresses the symmetry on the vertical axes in the same way as the truth-table expresses the symmetry on the horizontal axes. The recursive structure of the truth-table, or equivalently, the group $G = C_2^n$, as shown in Fig. 5.2, maps into the recursive structure of the DT.

In a DT for a given f , if there are some constant or equal subvectors in the vector of the values of constant nodes, then the reduction can be performed. It is done by deleting and sharing isomorphic subtrees in the DT. In that way the DD for f is derived. In a DD, edges connecting nodes at non-successive levels may appear. The length of an edge connecting a

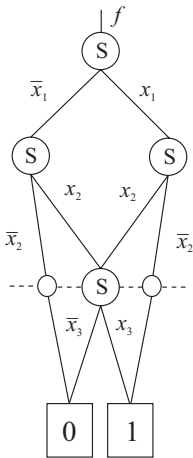


Figure 5.3. BDD for f in Example 46.

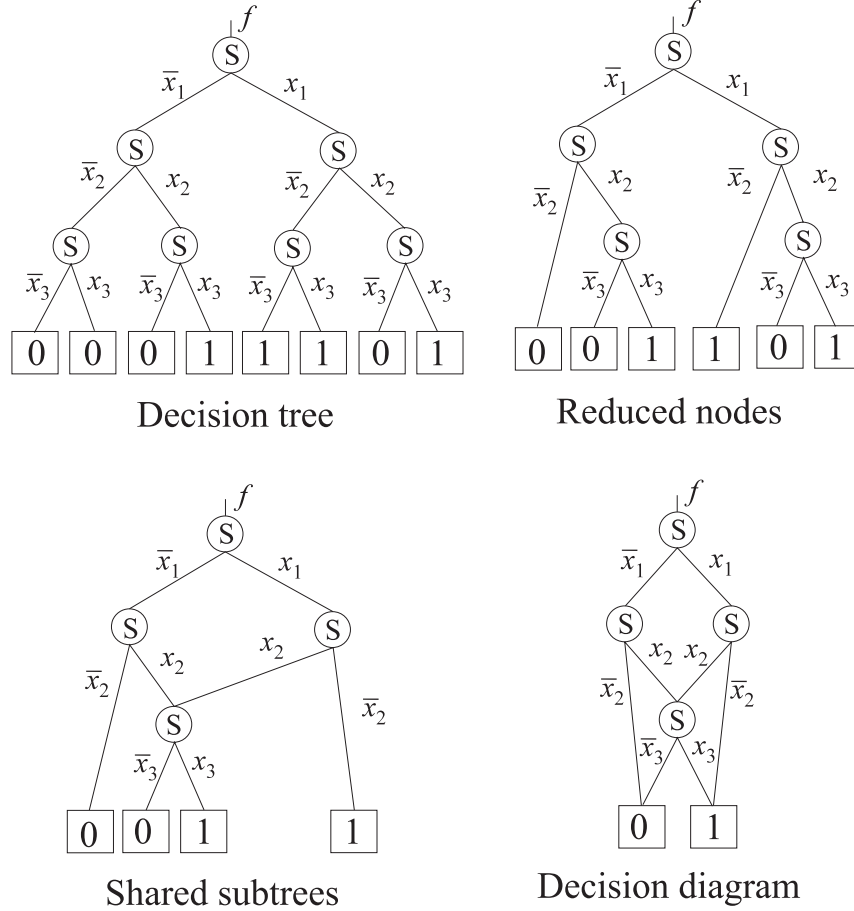
node at the i -th level with a node at the $(i+k)$ -th level is k . A point where such an edge crosses the imaginary line connecting nodes corresponding to the same variable is denoted as the cross point.

Definition 32 *In a reduced decision diagram, a cross point is a point where an edge longer than one crosses a level in the DD.*

Cross points permit us to take into account the impact of the deleted nodes. The following example illustrates the reduction procedure performed over a BDT to derive a BDD for f .

Example 46 In the BDT for $f(x_1, x_2, x_3) = x_1\bar{x}_2 \vee x_2x_3$ reduction by deleting nodes is possible since there are two constant subvectors representing $f_{000} = f_{001} = 0$ and $f_{100} = f_{011} = 1$. Therefore, the corresponding subtrees representing these subvectors reduce to constant nodes 0 and 1, respectively. In this BDT, there are two equal subvectors $[f_{010}, f_{011}]^T$ and $[f_{110}, f_{111}]^T$ equal $[0, 1]^T$. Therefore, the corresponding subtrees can be joined and the redundant subtree deleted. Fig. 5.3 shows a thus derived BDD for f . Fig. 5.4 explains the way in which the reduction is performed.

The reduction rules used in the reduction of the BDT in the above example are called the BDD reduction rules [196]. Depending on the decomposition rules at the nodes in the DT, different reduction rules are defined. For example, the positive Davio (pD) decomposition rule is defined as $f = 1 \cdot f_0 \oplus x_i(f_0 \oplus f_1)$. The negative Davio (nD) expansion rule is defined as $f = 1 \cdot f_1 \oplus \bar{x}_i(f_0 \oplus f_1)$. These rules and DTs derived by using them will be discussed further. We mention them here for the purpose of discussing different reduction rules used in DD representations of discrete functions.

Figure 5.4. Reduction of BDT into BDD for f in Example 46.

For the positive Davio (pD) decomposition and negative Davio (nD) decomposition rules zero-suppressed reduction rules are defined [151]. Fig. 5.5 shows the BDD and zero-suppressed BDD reduction rules. In this figure, part (a) and part (c) show reduction of nodes in Shannon (S), and positive Davio (pD) and negative Davio nodes (nD), respectively. Part (b) is equal in both BDD and zero-suppressed BDD reduction rules and represents sharing of isomorphic subtrees. In the reduction of Shannon nodes, the property $\bar{x}_i \oplus x_i = 1$ is exploited, and the incoming edge to the remaining node is labeled by 1 multiplied by the label of the incoming edge of the deleted node. In the reduction of Davio nodes, we use the property that multiplication of \bar{x}_i or x_i by 0 is zero, thus, it does not contribute to the value of f represented by the node. Fig. 5.6 shows the generalized BDD reduction rules that should be used in a general case of DDs where labels at the edges

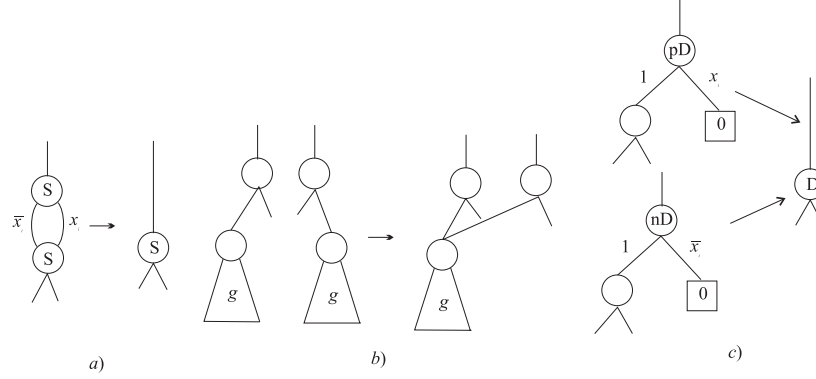


Figure 5.5. BDD reduction rules and zero-suppressed BDD reduction rules.

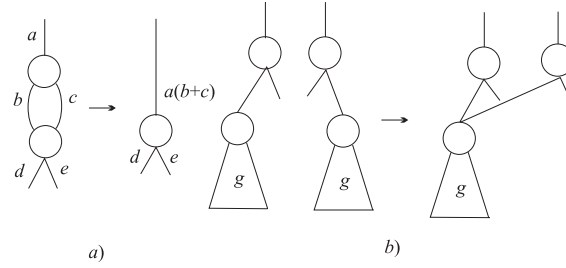
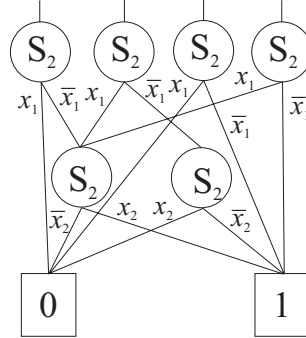


Figure 5.6. Generalized BDD reduction rules.

can take values different from logic 0 and 1 or integers 0 and 1 [243]. They involve the BDD reduction rules as a particular example.

From a given BDD, the represented function f is determined by starting from the constant nodes and performing the Shannon expression at the non-terminal nodes up to the root node. Equivalently, f is represented by the AND-EXOR expression with product terms determined as products of variables at the edges in the paths from the root node to the constant nodes showing the value 1.

Example 47 For the BDD in Fig. 5.3, the node at the level for x_3 shows the subfunction $0 \cdot \bar{x}_3 \oplus 1 \cdot x_3 = x_3$. The left node at the level for x_2 shows the subfunction $0 \cdot \bar{x}_2 \oplus x_2 x_3 = x_2 x_3$. Similarly, the subfunction represented by the subtree rooted at the right node for x_2 is $1 \cdot \bar{x}_2 \oplus x_2 x_3 = \bar{x}_2 \oplus x_2 x_3$. Finally, for the root node, $\bar{x}_1 \cdot x_2 x_3 \oplus x_1 \cdot (\bar{x}_2 \oplus x_2 x_3)$. Thus, this BDD represents the given function f in the form $f = \bar{x}_1 x_2 x_3 \oplus x_1 \bar{x}_2 \oplus x_1 x_2 x_3$. The product terms $\bar{x}_1 x_2 x_3$, $x_1 \bar{x}_2$, and $x_1 x_2 x_3$ correspond to the product of labels at the edges in the paths from the root node to the constant node 1.

Figure 5.7. Shared BDD for f in Example 48.

How do we represent multi-output functions?

Multi-output functions can be represented by shared BDDs (SBDDs) [151, 152, 153], with a separate root node for each output.

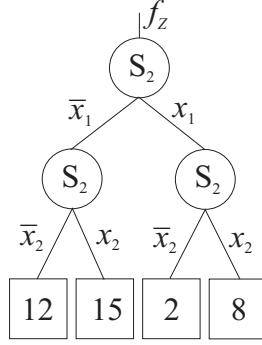
Example 48 Fig. 5.7 shows a shared BDD for the multi-output function $f = (f_4, f_3, f_2, f_1)$, where $f_1 = \bar{x}_1 x_2$, $f_2 = x_1 \oplus x_2$, $f_3 \bar{x}_1$, and $f_4 = \bar{x}_1 \vee x_2$.

Alternatively, a multi-output function can be represented by an equivalent integer-valued function f_z , as is explained in Examples 4 and 13 and represented by a single root-node word-level DD discussed in what follows. We will introduce this way of representing multi-output functions with the following example.

Example 49 The multi-output function in Example 48 can be represented by the integer-valued function $f_z(x) = 2^3 f_4 + 2^2 f_3 + 2 f_1 + f_0$. Functions f_4 , f_3 , f_2 , and f_1 are represented by the vectors $\mathbf{F}_4 = [1, 1, 0, 1]^T$, $\mathbf{F}_3 = [1, 1, 0, 0]^T$, $\mathbf{F}_2 = [0, 1, 1, 0]^T$, and $\mathbf{F}_1 = [0, 1, 0, 0]^T$. Therefore, $f_z(x)$ is represented by the vector

$$\mathbf{F}_Z = 8 \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \end{bmatrix} + 4 \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} + 2 \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 12 \\ 15 \\ 2 \\ 8 \end{bmatrix}.$$

This function can be represented by a decision tree that will have integers as constant nodes. The i -th bit in the binary representation for the values of constant nodes shows the value of the i -th output f_i . These DTs are denoted as Multi-terminal binary DDs (MTBDDs) [32] and will be discussed later. MTBDDs use the property that the Shannon decomposition can be formally extended to integers as $f = \bar{x}_i f_0 + x_i f_1$, where the \bar{x}_i is interpreted as $(1 - x_i)$. Fig. 5.8 shows MTBDD for \mathbf{F}_Z . The reduction into a DD is impossible, since there are no equal values in \mathbf{F}_Z .

Figure 5.8. MTBDD for f in Example 48.

We will mainly discuss single root-node DDs, assuming that the presentation extends in a straightforward way to shared DDs.

5.2 Spectral Interpretation of BDTs

In matrix notation, the Shannon decomposition rule $f = \bar{x}_i f_0 \oplus x_i f_1$ is written as

$$f = \begin{bmatrix} \bar{x}_i & x_i \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \end{bmatrix}.$$

First, we apply it to x_1 , which means to the root node in the DT. Thus, we write

$$f = \begin{bmatrix} \bar{x}_1 & x_1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \end{bmatrix}.$$

Then, we use the same rule for x_2 and

$$f = (\begin{bmatrix} \bar{x}_1 & x_1 \end{bmatrix} \otimes \begin{bmatrix} \bar{x}_2 & x_2 \end{bmatrix}) \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right) \begin{bmatrix} f_{00} \\ f_{01} \\ f_{10} \\ f_{11} \end{bmatrix}.$$

When we apply it for x_3 , then

$$f = (\begin{bmatrix} \bar{x}_1 & x_1 \end{bmatrix} \otimes \begin{bmatrix} \bar{x}_2 & x_2 \end{bmatrix} \otimes \begin{bmatrix} \bar{x}_3 & x_3 \end{bmatrix})$$

$$\begin{aligned}
& \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right) \begin{bmatrix} f_{000} \\ f_{001} \\ f_{010} \\ f_{011} \\ f_{100} \\ f_{101} \\ f_{110} \\ f_{111} \end{bmatrix} \\
&= \begin{matrix} & \bar{x}_1\bar{x}_2\bar{x}_3 & \bar{x}_1\bar{x}_2x_3 & \bar{x}_1x_2\bar{x}_3 & \bar{x}_1x_2x_3 & & & \\ & & & x_1\bar{x}_2\bar{x}_3 & x_1\bar{x}_2x_3 & x_1x_2\bar{x}_3 & x_1x_2x_3 & \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} & \begin{bmatrix} f_{000} \\ f_{001} \\ f_{010} \\ f_{011} \\ f_{100} \\ f_{101} \\ f_{110} \\ f_{111} \end{bmatrix} & \cdot \end{matrix}
\end{aligned}$$

Remark 6 A BDT represents f in the complete disjunctive form for f . In spectral interpretation, f is assigned to a BDT, by decomposition of f with respect to the trivial mapping described by the identity matrix, whose columns can be represented by minterms. Therefore, the constant nodes represent the values of f . Each path from the root node to the constant nodes corresponds to a basis function in the identity mapping. It is described by a minterm that is generated by the multiplication of labels at the edges.

5.3 Spectral Interpretation of FDTs

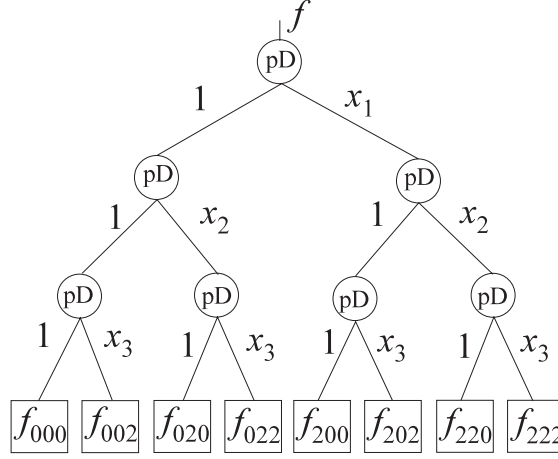
Functional DDs (FDDs) [113], also called positive polarity Reed-Muller DDs (PPRMDDs) [187], are derived by using the positive Davio decomposition

$$f = 1 \cdot f_0 \oplus x_i(f_0 \oplus f_1).$$

Fig. 5.9 shows FDT for $n = 3$. This DT is also denoted as the positive Davio tree, or positive polarity Reed-Muller DT [187]. In this figure, the index 2 denotes EXOR of cofactors indexed by 0 and 1, i.e., $f_2 = f_0 \oplus f_1$. Similarly, for constant nodes, f_{002} represents $f_{000} \oplus f_{001}$, and $f_{020} = f_{000} \oplus f_{010}$, etc.

In matrix notation,

$$f = \begin{bmatrix} 1 & x_i \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \end{bmatrix}.$$

Figure 5.9. FDT for $n = 3$.

After a recursive application of this decomposition rule to the nodes in FDT, we get

$$\begin{aligned}
 f &= \left(\begin{bmatrix} 1 & x_1 \end{bmatrix} \otimes \begin{bmatrix} 1 & x_2 \end{bmatrix} \otimes \begin{bmatrix} 1 & x_3 \end{bmatrix} \right) \\
 &\quad \left(\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \right) \mathbf{F} \\
 &= \mathbf{X}_r(3) \mathbf{R}(3) \mathbf{F}.
 \end{aligned}$$

The way that a FDT represents a given function f through the Reed-Muller spectrum for f , is explained by the following example.

Example 50 Fig. 5.10 shows the BDT for a function f of $n = 2$ variables. The constant nodes are elements of the truth-vector for $f(x_1, x_2)$; thus, $\mathbf{F} = [f_{00}, f_{01}, f_{10}, f_{11}]^T$. If at each node we perform calculations determined by the basic Reed-Muller matrix $\mathbf{R}(1)$, at the root node we get the Reed-Muller spectrum. The calculations at each node are performed over subfunctions represented by subtrees rooted at the processed node. Referring to the FFT-like algorithm for the Reed-Muller transform, at each node we perform the basic butterfly operation of the algorithm. Since at upper levels in the BDT, the calculations are performed over subfunctions, corresponding to componentwise calculations over subvectors in \mathbf{F} , instead of over function values, it follows that we perform an FFT-like algorithm over BDT. For this reason we get the Reed-Muller spectrum for f at the root node.

However, if we consider an FDT having the Reed-Muller coefficients f_{00} , $f_{02} = f_{00} \oplus f_{01}$, $f_{20} = f_{00} \oplus f_{10}$, and $f_{22} = f_{00} \oplus f_{01} \oplus f_{11}$, and perform the same calculations, we get the function values for f at the root node, since the Reed-Muller matrix is a self-inverse matrix. In FDTs, the labels at the

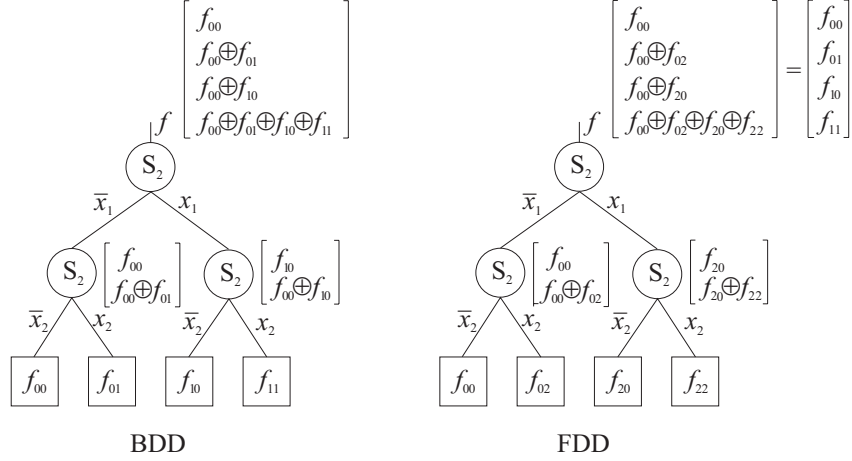


Figure 5.10. BDT and FDT.

edges are determined as symbolic notation for columns of $\mathbf{R}(1)$. Because of this, in determining f from the FDT by following labels at the edges, we actually perform the inverse Reed-Muller transform over the values of constant nodes of FDT, which are, therefore, selected as the Reed-Muller coefficients.

The following example explains the applications of FDDs.

Example 51 Consider the function $f(x_1, x_2) = \bar{x}_1\bar{x}_2 \oplus x_1x_2$, whose truth-vector is $\mathbf{F} = [1, 0, 0, 1]^T$. Fig. 5.11 shows the BDT for f and related BDD. We cannot reduce any non-terminal node, since there are no equal sub-vectors in \mathbf{F} . However, the Reed-Muller spectrum, calculated as $\mathbf{S}_{rf} = (\mathbf{R}(1) \otimes \mathbf{R}(1))\mathbf{F} = [1, 1, 1, 0]^T$, determines the PPRM for this function $f = 1 \oplus x_1 \oplus x_2$. Fig. 5.11 shows the FDT for f determined using the Reed-Muller spectrum. From zero-suppressed BDD reduction rules, we can share constant nodes for the left node at the level for x_2 , since both outgoing edges, labeled by 1 and x_2 , point to the same value 1. These rules allow us to delete the edge labeled by x_2 that points to the constant node 0, since $x_2 \cdot 0 = 0$ does not contribute to the Reed-Muller spectrum. In this way we determine the FDD representing f in the form of PPRM. We can use the generalized BDD rules, which allow us to delete the left node at the level for x_2 , since both outgoing edges point to the same value; however, we should redetermine the label at the incoming edge for the remaining node as is specified by these reduction rules. Transferring from the original domain into the Reed-Muller spectrum provides for the constant subvector $[1, 1]$ in \mathbf{S}_{rf} , which permits reduction of a non-terminal node.

Remark 7 In an FDT, each path from the root node to the constant nodes represents a Reed-Muller function, i.e., a column in the Reed-Muller ma-

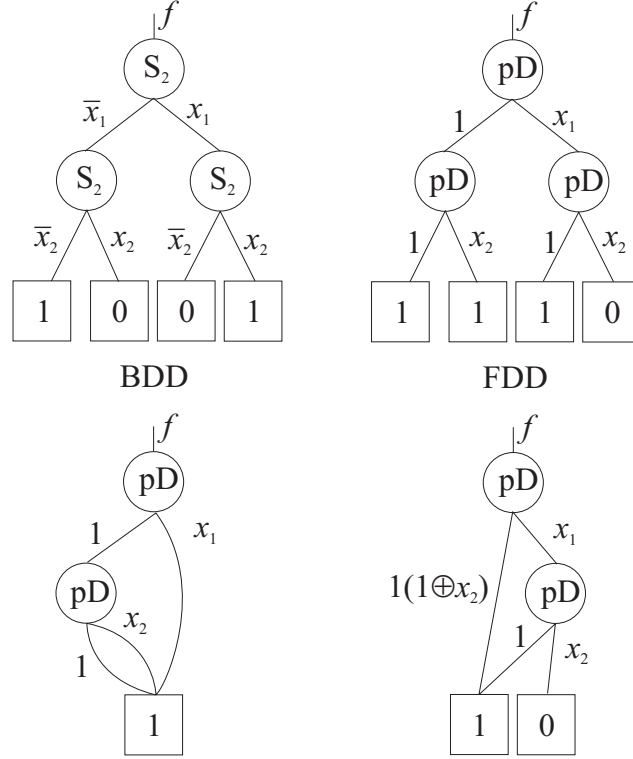


Figure 5.11. BDD, FDD, FDT, and FDD with zero-suppressed reduction rules, and FDD with generalized BDD rules.

trix. The constant nodes represent the Reed-Muller coefficients. Therefore, FDTs represent f in the form of the positive polarity Reed-Muller expression.

Statement 4 (Spectral interpretation of positive Davio trees)

1. Given a function f , to determine a positive Davio tree representing f , the Reed-Muller transform is performed.
2. Given a positive Davio tree, to determine an f which it represents, the (inverse) Reed-Muller transform is performed.

This inverse transform determines labels at the edges of a pD node, 1 and x_i . They are symbolic notation for the columns of $\mathbf{R}(1)$.

The use of the inverse Reed-Muller transform to determine f from the positive Davio tree is shadowed by the self-inverseness of the Reed-Muller

transform. However, a proper interpretation of the BDTs and FDTs permits the following statement.

Statement 5 *Given a switching function f , FDD for f is BDD for its Reed-Muller spectrum S_f and, vice versa, BDT for S_f is FDT for f .*

The same consideration trivially applies to BDDs. Compared to the positive Davio tree, the use of a direct transform to determine the Shannon tree of f and the inverse transform to read f from it, is double shadowed, since we are working with the identity transform that is also self-inverse.

This consideration about spectral interpretation of BDDs and FDDs permits us to state the two following theorems, basic for the spectral interpretation of DDs for discrete functions.

Theorem 3 *DTs are graphical representations of spectral transform expansions of $f \in P(G)$ with respect to a basis Q in $P(G)$. In a DT, each path from the root node to the constant nodes corresponds to a basic function in Q . The constant nodes represent the Q -spectral coefficients for f .*

Theorem 4 *A DT defined with respect to a basis Q represents at the same time f and the Q -spectrum of f .*

From this theorem, the statement below follows.

Statement 6 *Each decision tree representing a function f can be considered as the Shannon tree representing the Q -spectrum of f .*

This spectral interpretation of BDDs and FDDs implies definition of spectral transform decision trees (STDTs) [243] as a concept involving different DDs for particular specification of the basis Q .

Let $\mathbf{Q}(n)$ be a $(2^n \times 2^n)$ non-singular matrix with elements in P . Thus, the columns of \mathbf{Q} form a set of linearly independent functions. Since there are 2^n such functions, $\mathbf{Q}(n)$ determines a basis in $P(C_2^n)$.

Suppose that $\mathbf{Q}(n)$ is represented as the Kronecker product of n factors $\mathbf{Q}(1)$, i.e.,

$$\mathbf{Q}(n) = \bigotimes_{i=1}^n \mathbf{Q}(1).$$

For such a basis Q , the i -th basic matrix $\mathbf{Q}(1)$ defines an expansion of f with respect to the i -th variable

$$f = \mathbf{Q}^{-1}(1)\mathbf{Q}(1) \begin{bmatrix} f_0 \\ f_1 \end{bmatrix},$$

where \mathbf{Q}^{-1} is the matrix inverse of $\mathbf{Q}(1)$.

Definition 33 [243] *A spectral transform decision tree (STDT) is a decision tree assigned to f by the decomposition of f with respect to the basis Q .*

Definition 34 [243] *Spectral transform decision diagrams (STDDs) are decision diagrams derived by the reduction of the corresponding STDTs.*

Reduction of STDTs is performed by deleting and sharing isomorphic subtrees [196].

It is obvious that the definition of STDTs satisfies requirements in Theorem 3 and that different DDs are involved in STDTs for different specifications of the matrices $\mathbf{Q}(1)$.

After this discovery of the relationship between FDTs for f and BDTs for the Reed-Muller spectrum S_f as well as the definition of STDTs, the following spectral interpretation of various DDs was a simple piece of research work. The previous consideration extends directly to other bit-level DDs. They are also denoted as AND-EXOR related DDs [71, 72, 196].

What are bit-level DDs?

5.4 Bit-Level DDs

Definition 35 *Bit-level DDs are DDs in which the constant nodes are logic values 0 and 1.*

BDDs and FDDs are particular examples of bit-level DDs.

Different bit-level DDs for switching functions are defined by using different decomposition rules. In these DDs, besides the Shannon and positive Davio decomposition, the negative Davio decomposition defined by $f = 1 \cdot f_1 \oplus \bar{x}_i(f_0 \oplus f_1)$ is used.

Definition 36 *Consider the set K of matrices $\mathbf{I}(1)$, $\mathbf{R}(1)$, and $\bar{\mathbf{R}}(1)$, corresponding to the Shannon, positive Davio, and negative Davio decompositions, respectively:*

$$K = \left\{ \mathbf{I}(1) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \mathbf{R}(1) = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \bar{\mathbf{R}}(1) = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \right\}.$$

1. *Kronecker DTs (KDTs) [43] are defined with respect to the Kronecker transforms over $GF(2)$. In these transforms and in KDDs, the basic transform matrices $\mathbf{K}_i(1)$, and corresponding decomposition rules, are freely chosen in the set K for each variable.*
2. *Pseudo-Kronecker DDs (pseudo-KDDs) [187] are defined by freely choosing in the set K decomposition rules for each node irrespective of other nodes at the same level in the DT.*
3. *Fixed polarity Reed-Muller DTs (FPRMDTs) and pseudo-Reed-Muller DTs (PRMDTs) [187] are subsets of KDTs and PKDTs, respectively, allowing $\mathbf{R}(1)$ and $\bar{\mathbf{R}}(1)$, but not $\mathbf{I}(1)$, as decomposition rules.*

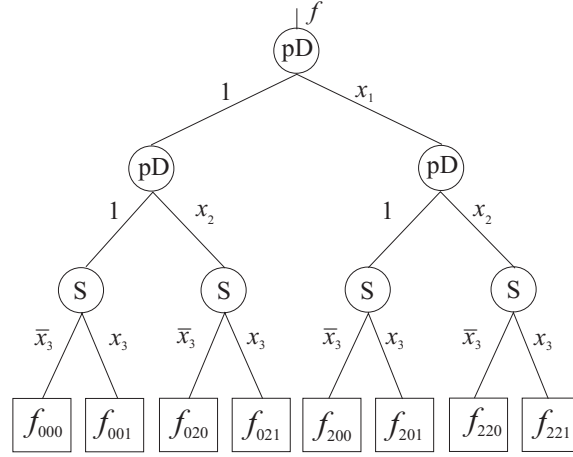
Figure 5.12. KDT for $n = 3$ in Example 52.**Example 52 (KDD)**

Fig. 5.12 shows KDT for $n = 3$ with positive Davio (pD) nodes for x_1 and x_2 and Shannon nodes (S)-nodes for x_3 . This KDT is the graphical representation of the decomposition of f with respect to the basis

$$\begin{aligned}\varphi_0 &= \bar{x}_3, \varphi_1 = x_3, \varphi_2 = x_2\bar{x}_3, \varphi_3 = x_2x_3, \\ \varphi_4 &= x_1\bar{x}_3, \varphi_5 = x_1x_3, \varphi_6 = x_1x_2\bar{x}_3, \varphi_7 = x_1x_2x_3,\end{aligned}$$

where the variables x_i take logic values 0 and 1.

Example 53 (PKDD)

Fig. 5.13 shows an example of a pseudo-Kronecker decision tree (PKDT) for $n = 3$ [187]. This PKDT is defined with respect to the basis

$$\begin{aligned}\varphi_0 &= \bar{x}_1\bar{x}_3, \varphi_1 = \bar{x}_1x_3, \varphi_2 = \bar{x}_1x_2, \varphi_3 = \bar{x}_1x_2x_3, \\ \varphi_4 &= x_1, \varphi_5 = x_1\bar{x}_3, \varphi_6 = x_1\bar{x}_2\bar{x}_3, \varphi_7 = x_1\bar{x}_2x_3.\end{aligned}$$

Assume that columns of the Reed-Muller matrix $\mathbf{R}(3)$ are denoted by $\mathbf{r}_i(x)$, $i = 0, \dots, 7$, respectively. The basic functions for PKDT in Fig. 5.13 are determined by the cyclic shift of $\mathbf{r}_5, \mathbf{r}_6, \mathbf{r}_7$ as follows:

$$\begin{aligned}\phi_0(x) &= \mathbf{r}_5(x+5), \phi_1(x) = \mathbf{r}_5(x+4), \phi_2(x) = \mathbf{r}_6(x+4), \phi_3(x) = \mathbf{r}_7(x+4), \\ \phi_4(x) &= \mathbf{r}_4(x), \phi_5(x) = \mathbf{r}_5(x+1), \phi_6(x) = \mathbf{r}_7(x+3), \phi_7(x) = \mathbf{r}_5(x),\end{aligned}$$

where $x+k$ denotes the shift for k places with respect to the addition of integers, and x is a number whose binary representation is (x_1, x_2, x_3) , thus, $x = (x_1, x_2, x_3)$.

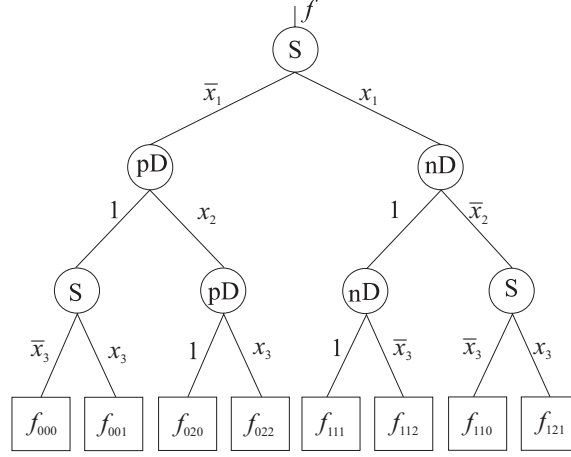
Figure 5.13. PKDT for $n = 3$ in Example 53.

Table 5.1 summarizes allowed bases and compares permitted changes in basic functions in terms of which various AND-EXOR expressions and corresponding DTs for functions in $GF(C_2^n)$ are defined.

5.5 Summary

1. Decision diagrams are graphic representations of Fourier series-like expressions for switching functions.
2. BDDs and FDDs are particular examples derived as graphic representations of disjunctive normal-form and positive polarity Reed-Muller expressions. In these DDs, basic functions are described by minterms and distinct products of variables. The products are determined over the power set of the set of variables. Thus, in FDDs, basis functions are described by the product of variables in the subsets of the power set.
3. BDDs and FDDs have non-terminal nodes defined in terms of Shannon and positive Davio nodes. Different Kronecker DDs are defined by freely choosing among the Shannon decomposition rule, positive Davio decomposition rule, and negative Davio decomposition rule for nodes at each level in the decision tree. In pseudo-Kronecker DDs, decomposition rules are freely chosen for each node irrespective of other nodes at the same level in the decision tree.
4. In DDs, constant nodes show values of spectral coefficients for the represented function in terms of the spectral transform used in definition of the related DTs. Basic functions are defined by the product

DD	Basis	Form of functions	Order of functions	Property
BDD	Trivial	Fixed	Fixed	Kronecker product representable
PPRM	Reed-Muller	Fixed	Fixed	Kronecker product representable
FPRM	Reed-Muller	Fixed	Changeable depending on levels	Permuted Reed-Muller
PRM	Pseudo-Reed-Muller	Fixed	Changeable depending on nodes	Permuted and cyclic shifted Reed-Muller
GRM	Combined Reed-Muller	Fixed	Changeable depending on nodes	Combination of the permuted Reed-Muller for different polarities
KDD	Combined Reed-Muller and trivial depending on levels	Changeable depending on levels	Changeable depending on levels	Kronecker product representable
PKDD	Pseudo-KRO	Changeable depending on nodes	Changeable depending on nodes	Permuted and cyclic shifted KRO

Table 5.1. Bases used in the AND-EXOR expressions and related DDs.

of the labels at the edges. Therefore, to assign a given function f to a DT, we perform the direct transform. In reading f from the DT, we perform the inverse transform. The same applies to DDs, since the reduction rules do not destroy the information content of DTs. It follows that a DD represents at the same time f and the spectrum for f with respect to the spectral transform used in definition of the related DT.

Spectral Interpretation of Decision Diagrams

Stankovic, R.; Astola, J.T.

2003, XX, 286 p., Hardcover

ISBN: 978-0-387-95545-2