ELSEVIER

Theoretical Computer Science 262 (2001) 633–647

Theoretical Computer Science

www.elsevier.com/locate/tcs

Tree-shellability of Boolean functions

Yasuhiko Takenaga[a,*], Kouji Nakajima[b], Shuzo Yajima[c]

[a] *Department of Computer Science, University of Electro-Communications, Chofu, Tokyo 182-8585, Japan*

[b] *Department of Information Science, Graduate School of Engineering, Kyoto University, Japan*

[c] *Faculty of Informatics, Kansai University, Japan*

Received 9 April 1999; revised 13 July 2000; accepted 7 August 2000

Communicated by S. Miyano

Abstract

In this paper, we define tree-shellable and ordered tree-shellable Boolean functions. A tree-shellable function is a positive Boolean function such that the number of prime implicants equals the number of paths from the root node to a 1-node in its binary decision tree representation. A tree-shellable function is easy to dualize and good for a kind of reliability computation. We show their basic properties and clarify the relations between several shellable functions, i.e. shellable, tree-shellable, ordered tree-shellable, aligned and lexico-exchange functions. We also discuss on tree-shellable quadratic functions. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Shellability; Dualization; Binary decision tree; Prime implicant; Boolean function

1. Introduction

It is important to clarify the properties of Boolean functions in various fields of computer science. Prime implicant is a very important concept on the theory of Boolean functions. Each prime implicant of a positive Boolean function is essential. Thus, every positive Boolean function is uniquely represented by an irredundant DNF and each term of the irredundant DNF corresponds to a minimum true point.

A shellable Boolean function is a positive Boolean function whose irredundant DNF representation satisfies that, for any k, first k product terms become orthogonal without changing the function by adding negative literals to each term. Shellable Boolean functions play an important role in many fields. The notion of shellability was originally used in the theory of simplicial complexes and polytopes (for example in [8, 10]). More recently, it is studied for its importance on reliability theory (for example in [1, 2, 16]).

* Corresponding author.

E-mail address: takenaga@cs.uec.ac.jp (Y. Takenaga).

PII: S0304-3975(00)00369-8

In this paper, we define a tree-shellable function and an ordered tree-shellable function as restricted shellable Boolean functions. The notion of tree-shellability makes it possible to define several subclasses of shellable Boolean functions in terms of tree-shellability. A tree-shellable function is a positive Boolean function defined by the relation between its prime implicants and binary decision tree (BDT) representation: there exists a BDT representation such that the number of prime implicants equals the number of paths from the root to a leaf labeled 1 in the BDT. An ordered tree-shellable function has the similar relation with an ordered BDT (OBDT), which is a BDT such that, on all the paths, variables appear according to a total order of variables.

A shellable function has the following good properties when it is given with the order of terms to make it shellable. First, if a Boolean function f is shellable, one can easily solve the following problem.

[Union of Product Problem] [2]

Input: $Pr[x_i = 1](1 \leqslant i \leqslant n), f(x_1, \ldots, x_n)$

Output: $Pr[f(x_1, \ldots, x_n) = 1]$,

where $Pr[A]$ represents the probability of event A. This is the problem of computing the reliability of some kind of systems. Each variable represents the state of a subsystem. A subsystem is operative if the variable has value 1. If a Boolean function f is shellable, one can easily compute the exact value of $Pr[f = 1]$ using the orthogonal DNF representation of f.

Second, if a Boolean function f is shellable, it is easy to compute the dual of f. The dual of a Boolean function $f(x_1, \ldots, x_n)$ is defined by $f^{\mathrm{d}} = \overline{f(\overline{x_1}, \ldots, \overline{x_n})}$. It is not known if the DNF representation of the dual f^{d} can be computed from the DNF representation of f in time polynomial to the input and output size. So the problem is still interested in by many researches (for example in [4, 11, 13]). The classes of Boolean functions which can be dualized in polynomial time include 2-monotonic (or regular) functions (which include threshold functions) [7, 9, 15], aligned functions [5], positive k-DNFs [12], matroid functions [14], etc.

If f is tree-shellable and the BDT representation of a Boolean function f is given, it is possible to compute the BDT representation of f^{d} only by exchanging a 1-edge and a 0-edge for every variable node and exchanging labels 1 and 0 for every leaf node. For ordered tree-shellable functions, it is possible to obtain an OBDT representation of an ordered tree-shellable function f in polynomial time when the shelling variable order is given, and it is easy to compute a positive DNF representation of f^{d} from the OBDT representation of f.

In this paper, we first define tree-shellable and ordered tree-shellable functions and show some basic properties of them. Next, we clarify relations among various shellable functions. Various subclasses of shellable Boolean functions have been proposed, e.g. lexico-exchange function [2], aligned function [5]. An aligned function clearly has both of the above good properties and, in addition, it is possible to check whether a Boolean function is aligned in polynomial time. However, the class of aligned functions is smaller than the ones defined in this paper. We show that the implications between shellable and tree-shellable functions, tree-shellable and ordered tree-shellable functions,

ordered tree-shellable and aligned functions are proper. We also show that ordered tree-shellability is equivalent to the lexico-exchange property. At last, we discuss on the tree-shellability of quadratic functions and the shelling variable order of ordered tree-shellable quadratic functions.

2. Preliminaries

2.1. Basic notations

Let $B=\{0,1\}$, n be a natural number, and $[n]=\{1,2,\ldots,n\}$. Let $[0]=\emptyset$. Let π be a permutation on $[n]$. π represents a total order of integers in $[n]$. Let $\pi(i)$ be the ith element of π. If s appears before t with respect to π, we denote $s \prec_\pi t$. For $S \subseteq [n]$, $\min_\pi(S)$ and $\max_\pi(S)$ for order π is defined as follows:

$$\min_\pi(S) = h \quad \text{if } h \in S \text{ and } h \prec_\pi i \quad \text{for all } i \in S\backslash\{h\},$$

$$\max_\pi(S) = h \quad \text{if } h \in S \text{ and } i \prec_\pi h \quad \text{for all } i \in S\backslash\{h\}.$$

If π is clear from the context, we can simply write $s \prec t$, $\min(S)$ or $\max(S)$.

Let I_s, I_t be distinct subsets of $[n]$ and π be a permutation on $[n]$. If $I_s \cap \{\pi(1),\ldots,\pi(i-1)\} = I_t \cap \{\pi(1),\ldots,\pi(i-1)\}$, and either $I_s \cap \{\pi(i),\ldots,\pi(n)\} = \emptyset$ or $\pi(i) \in I_s, \pi(i) \notin I_t$ hold for some $i \in [n]$, we denote $I_s \prec_L I_t$. The order $\prec_L$ is called *lexicographical order*.

2.2. Disjunctive normal form Boolean formula

Let $f(x_1,\ldots,x_n)$ and $g(x_1,\ldots,x_n)$ be Boolean functions. We denote $f \geqslant g$ if $f(x)=1$ for any assignment $x \in \{0,1\}^n$ which makes $g(x)=1$. An *implicant* of f is a product term $\bigwedge_{i\in I} x_i \bigwedge_{j\in J} \overline{x_j}$ which satisfies $\bigwedge_{i\in I} x_i \bigwedge_{j\in J} \overline{x_j} \leqslant f$, where $I, J \subseteq [n]$. An implicant which satisfies $\bigwedge_{i\in I-\{s\}} x_i \bigwedge_{j\in J} \overline{x_j} \not\leqslant f$ for any $s \in I$ and $\bigwedge_{i\in I} x_i \bigwedge_{j\in J-\{t\}} \overline{x_j} \not\leqslant f$ for any $t \in J$ is called a *prime implicant* of f. A product term which contains each of the n variables in either positive or negative literal is called a *minterm*.

An expression of the form $f = \bigvee_{k=1}^{m} (\bigwedge_{i\in I_k} x_i \bigwedge_{j\in J_k} \overline{x_j})$ is called a *disjunctive normal form Boolean formula* (DNF), where $I_k, J_k \subseteq [n]$ and $I_k \cap J_k = \emptyset$ for $k=1,\ldots,m$. $T_k = \bigwedge_{i\in I_k} x_i \bigwedge_{j\in J_k} \overline{x_j}$ is called a term of f.

A DNF is called an *orthogonal* DNF (ODNF), if $(I_k \cap J_l) \cup (I_l \cap J_k) \neq \emptyset$ for every pair of terms T_k, T_l $(k \neq l)$. If f is represented as an ODNF, at most one term of f has value 1 for any assignment.

A *positive* DNF (PDNF) is a DNF such that $J_k = \emptyset$ for all k. If f can be represented as a PDNF, it is called a positive Boolean function. For simplicity, we call that I_k is a term of a positive function. A PDNF is called *irredundant* if $I_k \subseteq I_l$ is not satisfied for any k, l $(1 \leqslant k, l \leqslant m, k \notin l)$. For an irredundant PDNF, let $PI(f)$ be the set of all I_k. $PI(f)$ represents the prime implicants of f. For $I_k \in PI(f)$, a minterm $\bigwedge_{i\in I_k} x_i \bigwedge_{j\notin I_k} \overline{x_j}$ is called a *minimum true point* of f. In the following of this paper, we consider only

positive functions and we assume that a function is given as an irredundant PDNF $f = \bigvee_{k=1}^{m} \bigwedge_{i \in I_k} x_i$.

2.3. Binary decision tree

A *binary decision tree* (BDT) is a labeled tree that represents a Boolean function. The leaf nodes of a BDT are labeled by 0 or 1 and any other node is labeled by a variable. Each node except leaf nodes has two outgoing edges, which are called a *0-edge* and a *1-edge*. The value of the function is given by traversing from the root node to a leaf node. At a node, one of the outgoing edges is selected according to the assignment for the variable. The value of the function is 0 if the label of the leaf is 0, and 1 if the label is 1.

A path from the root node to a leaf node labeled 1 is called a 1-path. On every 1-path, each variable appears at most once. A path P of a BDT is represented by a sequence of literals. If the kth edge on a 1-path P is the 1-edge (0-edge, resp.) from the node labeled by x_i, positive literal x_i (negative literal $\overline{x_i}$, resp.) is the kth element of P. For simplicity, we denote $\tilde{x}_i \in P$ when $\tilde{x}_i$ is included in the sequence representing P, where $\tilde{x}_i$ is either x_i or $\overline{x_i}$. Let P^k, $P^{(k)}$ denote the kth element of P and the prefix of P with length k, respectively.

When the 0-edge and the 1-edge of node v point to the nodes representing the same function, v is called to be a *redundant node*. A BDT which has no redundant node is called a *reduced BDT*. In the following of this paper, a BDT means a reduced BDT. If there is a total order of variables π which is consistent with the order of variables in any path of a BDT, it is called an *ordered* BDT (OBDT). The total order of variables for an OBDT is called the *variable order*. Let $S(P)$ be the set of variables that appear in path P in either a positive literal or a negative literal. An OBDT which satisfies $S(P) = \{\pi(1), \ldots, \pi(|S(P)|)\}$ for every path P is called a *leveled* BDT.

3. Shellable Boolean functions

3.1. Shellable function

Definition. Let f be a positive Boolean function represented by a PDNF $f = \bigvee_{k=1}^{m} \bigwedge_{i \in I_k} x_i$ and π_T be an order of terms of f. f is *shellable* with respect to π_T if there exist $J_1, \ldots, J_m (\subseteq [n])$ which satisfy the following conditions:
(1) For any l $(1 \leqslant l \leqslant m)$, $\bigvee_{k=1}^{l} \bigwedge_{i \in I_{\pi_T(k)}} x_i = \bigvee_{k=1}^{l} (\bigwedge_{i \in I_{\pi_T(k)}} x_i \bigwedge_{j \in J_{\pi_T(k)}} \overline{x_j})$.
(2) For any s, t such that $1 \leqslant s < t \leqslant m$, $(I_s \cap J_t) \cup (I_t \cap J_s) \neq \emptyset$.
f is shellable if there exists π_T such that f is shellable with respect to π_T. π_T is called the *term order* of f.

3.2. Lexico-exchange function

Definition (Ball and Provan [2])**.** Let f be a positive Boolean function represented by a PDNF and π be an order of variables of f. f is *lexico-exchange* with respect to

π, if, for every pair I_i, I_j such that $I_i \prec_L I_j$, there exists I_l which satisfies $I_l \prec_L I_j$ and $I_l \backslash I_j = \{h\}$, where $h = \min(I_i \backslash I_j)$. f is lexico-exchange if there exists π such that f is lexico-exchange with respect to π.

It is proved in [6] that it is NP-hard to check if a PDNF is lexico-exchange.

4. Tree-shellable Boolean functions

4.1. Tree-shellable function

Definition. A positive Boolean function f is *tree-shellable* when it can be represented by a BDT with exactly $|PI(f)|$ 1-paths.

Proposition 1. *If $f = \bigvee_{k=1}^{m} \bigwedge_{i \in I_k} x_i$ is tree-shellable, there exists a BDT T representing f which satisfies the following conditions*:
- *T has m 1-paths $P_1, \ldots, P_m$.*
- *Each P_k corresponds to a term I_k by the rule that $i \in I_k$ iff $x_i \in P_k$.*

Proof. It is clear from the definition that there exists a BDT which satisfies the first condition. Thus, we have only to prove that the second condition holds on the BDT. Let $P_1, \ldots, P_m$ be the paths of a BDT representing f which has m 1-paths. Let $pos(P_k)$($neg(P_k)$, resp.) be the set of indices of variables whose positive (negative, resp.) literals are in P_k.

First, assume that, for some k, there exists a 1-path P_l which satisfies $pos(P_l) \subsetneq I_k$. Then, minterm $\bigwedge_{i \in pos(P_l)} x_i \bigwedge_{j \notin pos(P_l)} \overline{x_j}$ must be 0 because no prime implicant make it to be 1. However, P_l makes it to be 1 in T. Thus, there exists no P_l which satisfies $pos(P_l) \subsetneq I_k$.

Next, assume that, for some k, there exists no 1-path P_l which satisfies $pos(P_l) \subseteq I_k$. Then, no 1-path makes minterm $\bigwedge_{i \in I_k} x_i \bigwedge_{j \notin I_k} \overline{x_j}$ to be 1 because, for any 1-path P_l, there exists a variable in $pos(P_l)$ which appears in the minterm as a negative literal. However, the minterm must be 1 because it is a minimum true point of f. Thus, for any k, there exists a 1-path P_l which satisfies $pos(P_l) \subseteq I_k$.

To satisfy both of them, there must be a 1-path P_l satisfying $pos(P_l) = I_k$. As T has exactly m 1-paths, each 1-path corresponds to a term by the rule that $pos(P_k) = I_k$. □

When a BDT T represents f, we call a 1-path P which satisfies $pos(P) = I_i$ the *main path* of I_i.

Definition. For a BDT T, let π_T be an order of 1-paths such that $i \prec_{\pi_T} j$ iff $P_i^{(k-1)} = P_j^{(k-1)}, P_i^k$ is a positive literal and P_j^k is a negative literal. Let $f = \bigvee_{k=1}^{m} \bigwedge_{i \in I_k} x_i$ be tree-shellable and a BDT T representing f has m paths. Then we call that f is *tree-shellable with respect to π_T* and π_T is the *shelling term order* of f.

The next proposition shows that tree-shellability is a kind of shellability as its name shows.

Proposition 2. *Let a positive Boolean function* $f = \bigvee_{k=1}^{m} \bigwedge_{i\in I_k} x_i$ *be tree-shellable with respect to* π_T, *and* P_k *be the* 1*-path of the BDT representing* f *that corresponds to* I_k. *Then*

$$\bigvee_{k=1}^{l} \bigwedge_{i\in I_{\pi_T(k)}} x_i = \bigvee_{k=1}^{l} \left(\bigwedge_{i\in pos(P_{\pi_T(k)})} x_i \bigwedge_{j\in neg(P_{\pi_T(k)})} \overline{x_j} \right)$$

for any l $(1 \leqslant l \leqslant m)$.

This proposition is obvious from the property of a BDT. If a 1-edge is always the right edge, the shelling term order is obtained by ordering the 1-paths in a BDT from the right one to the left one.

If a BDT T represents $f = \bigvee_{k=1}^{m} \bigwedge_{i\in I_k} x_i$ and has exactly m paths, T witnesses that f is tree-shellable. The next two corollaries deal with such f and T.

Corollary 3. *If a BDT* T *witnesses that* f *is tree-shellable, for any* 1*-path* P_k *of* T *and any* $\overline{x_s} \in P_k$, *there exists* l *which satisfies* $P_k^{(t-1)} = P_l^{(t-1)}$, $P_k^t = \overline{x_s}$, $P_l^t = x_s$ *and* $I_l \subsetneq I_k \cup \{s\}$ *for some* t.

Proof. Assume w.l.o.g. that $\pi_T(k) = k$ for any k and that P_k is the main path of $I_k (1 \leqslant k \leqslant m)$. Then

$$f = \bigvee_{k=1}^{m} \bigwedge_{i\in I_k} x_i = \bigvee_{k=1}^{m} \left(\bigwedge_{i\in pos(P_k)} x_i \bigwedge_{j\in neg(P_k)} \overline{x_j} \right)$$

holds because T witnesses that f is tree-shellable.

The idea of this proof is as follows. As P_k $(k \neq 1)$ includes negative literals, P_k cannot make all the minterms in I_k to be 1. Thus, the remaining minterms should be made 1 by the other 1-paths. We prove that the corollary should hold in order to make the remaining minterms to be 1.

The following discussion holds for any 1-path P_k and $\overline{x_s} \in P_k$. Minterm $\bigwedge_{i\in I_k\cup\{s\}} x_i \bigwedge_{j\notin I_k\cup\{s\}} \overline{x_j}$ $(= z)$ is 1 because $\bigwedge_{i\in I_k} x_i \geqslant z$. However, because $s \in neg(P_k)$, $\bigwedge_{i\in pos(P_k)} x_i \bigwedge_{j\in neg(P_k)} \overline{x_j} \not\geqslant z$. Thus, there must exist a path $P_l (l \neq k)$ which makes z to be 1. To make z to be 1 by P_l, $pos(P_l) \subseteq I_k \cup \{s\}$ and $neg(P_l) \subseteq [n] - (I_k \cup \{s\})$ should hold. Therefore, there is only one variable x_s that may appear in P_k and P_l as different literals. Hence, P_l is on P_k until the node labeled by x_s. Therefore, $P_k^{(t-1)} = P_l^{(t-1)}, P_k^t = \overline{x_s}$ and $P_l^t = x_s$ hold for some t.

As shown above, $pos(P_l) \subseteq I_k \cup \{s\}$ holds. However, $I_l \neq I_k \cup \{s\}$ because f is given as an irredundant DNF. Thus, $I_l \subsetneq I_k \cup \{s\}$. □

Corollary 4. *Let* T *be a BDT such that* T *has* $m = |PI(f)|$ 1*-paths* $P_1, P_2, \ldots, P_m$ *and* $pos(P_k) = I_k$ *for any* k $(1 \leqslant k \leqslant m)$. *Then* T *witnesses that* f *is tree-shellable if for any* 1*-path* P_k *of* T *and any* $\overline{x_s} \in P_k$, *there exists* l *which satisfy* $P_k^{(t-1)} = P_l^{(t-1)}$, $P_k^t = \overline{x_s}$, $P_l^t = x_s$ *and* $I_l \subsetneq I_k \cup \{s\}$ *for some* t.

Proof. As T has $|PI(f)|$ 1-paths, we have only to prove that T represents f. Assume w.l.o.g. that π_T satisfies $\pi_T(k)=k$ for any k. We prove by induction on k that

$$\bigvee_{i=1}^{k}\bigwedge_{q\in I_i} x_q = \bigvee_{i=1}^{k}\left(\bigwedge_{q\in pos(P_i)} x_q \bigwedge_{r\in neg(P_i)} \overline{x_r}\right)$$

holds for any k $(1\leqslant k\leqslant m)$. When $k=m$, the above equation means that T represents f.

When $k=1$, P_1 makes $\bigwedge_{q\in I_1} x_q$ to be 1 because the rightmost 1-path P_1 includes no negative literal.

Assume that the above equation holds for $k=j-1$. We prove that it also holds for $k=j$. P_j represents a product term $\bigwedge_{q\in pos(P_j)} x_q \bigwedge_{r\in neg(P_j)} \overline{x_r}$. As

$$\bigwedge_{q\in I_j} x_q = \left(\bigwedge_{q\in pos(P_j)} x_q \bigwedge_{r\in neg(P_j)} \overline{x_r}\right) \bigvee \left(\bigvee_{r\in neg(P_j)} \left(\left(\bigwedge_{q\in I_j} x_q\right) \bigwedge x_r\right)\right)$$

holds, $\bigvee_{r\in neg(P_j)}((\bigwedge_{q\in I_j} x_q)\bigwedge x_r)$ must be made to be 1 by $P_1,\ldots,P_{j-1}$. Assume that, for any $r\in neg(P_j)$, there exists l which satisfies $P_j^{(t-1)}=P_l^{(t-1)}$, $P_j^t=\overline{x_r}$, $P_l^t=x_r$ and $I_l\subsetneq I_j\cup\{r\}$ for some t. Then $I_l\subsetneq I_j\cup\{r\}$ means that $\bigwedge_{q\in I_l} x_q \geqslant (\bigwedge_{q\in I_j} x_q)\bigwedge x_r$ holds. Moreover, $P_j^{(t-1)}=P_l^{(t-1)}$, $P_j^t=\overline{x_r}$ and $P_l^t=x_r$ means that $1\leqslant l\leqslant j-1$. Thus,

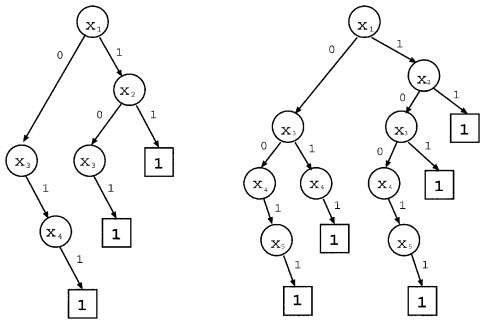
$$\begin{aligned}\bigvee_{i=1}^{j}\bigwedge_{q\in I_i} x_q &= \left(\bigvee_{i=1}^{j-1}\bigwedge_{q\in I_i} x_q\right) \bigvee \left(\bigwedge_{q\in pos(P_j)} x_q \bigwedge_{r\in neg(P_j)} \overline{x_r}\right)\\ &\quad \bigvee \left(\bigvee_{r\in neg(P_j)} \left(\left(\bigwedge_{q\in I_j} x_q\right) \bigwedge x_r\right)\right)\\ &= \left(\bigvee_{i=1}^{j-1}\bigwedge_{q\in I_i} x_q\right) \bigvee \left(\bigwedge_{q\in pos(P_j)} x_q \bigwedge_{r\in neg(P_j)} \overline{x_r}\right).\end{aligned}$$

Using the induction hypothesis,

$$\begin{aligned}&\left(\bigvee_{i=1}^{j-1}\bigwedge_{q\in I_i} x_q\right) \bigvee \left(\bigwedge_{q\in pos(P_j)} x_q \bigwedge_{r\in neg(P_j)} \overline{x_r}\right)\\ &= \left(\bigvee_{i=1}^{j-1}\left(\bigwedge_{q\in pos(P_i)} x_q \bigwedge_{r\in neg(P_i)} \overline{x_r}\right)\right) \bigvee \left(\bigwedge_{q\in pos(P_j)} x_q \bigwedge_{r\in neg(P_j)} \overline{x_r}\right)\\ &= \bigvee_{i=1}^{j}\left(\bigwedge_{q\in pos(P_i)} x_q \bigwedge_{r\in neg(P_i)} \overline{x_r}\right).\end{aligned}$$

Therefore the above equation holds for $k=j$. □

We can check the tree-shellability of a given PDNF using Corollary 4. We first give the label x_s of the root node and check if there exists I_l which satisfies $I_l\subsetneq I_k\cup\{s\}$



(a) $f=x_1x_2+x_1x_3+x_3x_4$ (b) $g=x_1x_2+x_1x_3+x_3x_4+x_4x_5$

Fig. 1. An example of an ordered tree-shellable function.

for any I_k which satisfies $x_s \notin I_k$. If the condition is not satisfied for some I_k, there exists no BDT whose root is labeled by x_s that witnesses that f is tree-shellable. If the condition is satisfied, examine recursively if two subfunctions $f|_{x_i=0}$ and $f|_{x_i=1}$ are both tree-shellable. If both of them are tree-shellable, f is tree-shellable and there exists a BDT with the root node labeled by x_i which witnesses that a given function f is tree-shellable. Otherwise, we choose the other variables one by one as the label of the root node. As this algorithm checks all the possible BDTs, this algorithm requires exponential time of n.

4.2. Ordered tree-shellable function

Definition. A positive Boolean function f is *ordered tree-shellable with respect to* π if it can be represented by an OBDT with variable order π which has exactly $|PI(f)|$ 1-paths. f is *ordered tree-shellable* if there exists π such that f is ordered tree-shellable with respect to π. We call π to be the *shelling variable order* of f.

Fig. 1(a) is an example of an ordered tree-shellable function. Note that the leaves with label 0 and the edges to them are omitted in this figure. f is ordered tree-shellable with respect to variable order $x_1x_2x_3x_4$. Fig. 1(b) is an example of a function which is not ordered tree-shellable. With variable order $x_1x_2x_3x_4x_5$, the BDT representing g has five 1-paths, which does not equal the number of prime implicants. It is not difficult to check that g is not ordered tree-shellable with respect to any other variable order.

As an ordered tree-shellable function is tree-shellable, Propositions 1 and 2 also hold for ordered tree-shellable functions. The shelling term order of an ordered tree-shellable function is equivalent to the lexicographical order based on the shelling variable order.

Theorem 5. *A positive Boolean function* $f = \bigvee_{k=1}^{m} \bigwedge_{i\in I_k} x_i$ *is ordered tree-shellable with respect to* π *iff the following condition holds.*

For any I_k *and any* $s \notin I_k$, $s \prec_\pi \max(I_k)$, *either*

(i) *there does not exist* I_l *such that* $(I_k \cup \{s\}) \cap \{\pi(1), \pi(2), \ldots, s\} = I_l \cap \{\pi(1), \pi(2), \ldots, s\}$, *or*

(ii) *if such prime implicants exist, at least one of them satisfies* $I_l \subsetneq I_k \cup \{s\}$.

Proof. (If) We assume w.l.o.g. that the variable order π satisfies $\pi(k) = k$ for any k, and the lexicographical order of terms π_T satisfies $\pi_T(k) = k$ for any k.

First, we show how to construct an OBDT from f. To construct an OBDT, we add terms one by one in lexicographical order. The algorithm to determine the path P_k that corresponds to I_k is as follows. Note that '$\cdot$' means concatenation of two sequences.

Construct OBDT

1. $P_k = \varepsilon$ (empty sequence)
2. For $i = 1$ to $\max(I_k)$ repeat 3 and 4.
3. If $i \in I_k$, then $P_k = P_k \cdot x_i$.
4. If $i \notin I_k$ and there already exists a path which starts with $P_k x_i$, then $P_k = P_k \cdot \overline{x_i}$.

Next, we have to show that the OBDT constructed by this algorithm witnesses that f is ordered tree-shellable if f satisfies the condition of this theorem. Its proof is similar to that of Corollary 4 by the following reason. Condition (i) of this theorem means that $\overline{x_s} \notin P_k$. Otherwise, $\overline{x_s} \notin P_k$ and I_l of this theorem satisfies $P_k^{(t-1)} = P_l^{(t-1)}$, $P_k^t = \overline{x_s}$ and $P_l^t = x_s$ for some t. Thus, it is equivalent to Corollary 4.

(Only if) It is proved similarly to Corollary 3. □

Using Theorem 5, it is possible to check the ordered tree-shellability of a Boolean function by an algorithm similar to that for tree-shellability. The only difference is that all the subtrees must have the same variable order in the case of ordered tree-shellable functions. This algorithm also requires exponential time because it checks all the variable orders.

4.3. Aligned function

Definition (Boros [5])**.** Let f be a positive Boolean function represented by a PDNF $f = \bigvee_{k=1}^{m} \bigwedge_{i \in I_k} x_i$. f is *aligned with respect to* π if, for every I_k and for every i such that $i \notin I_k$ and $i \prec_\pi \max(I_k)$, there exists I_l $(k \neq l)$ such that $I_l \subseteq \{i\} \cup (I_k \setminus \{\max(I_k)\})$. f is *aligned* if there exists π such that f is aligned with respect to π.

The result in [5] can be written as follows if we use our terms. □

Theorem 6. *A positive Boolean function* f *is aligned with respect to* π *iff there exists a leveled OBDT with variable order* π *which represents* f *and has exactly* $|PI(f)|$ 1*-paths.*

This theorem means that the class of aligned functions is a subclass of ordered tree-shellable functions. It can be seen as another definition of an aligned function. It is

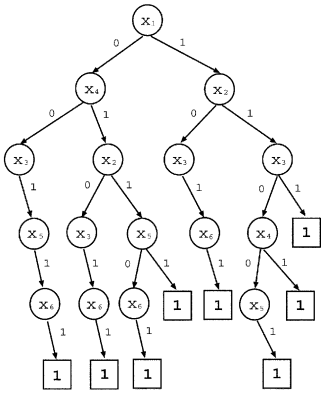


Fig. 2. A BDT representing f_2.

also shown in [5] that it is possible to decide if a positive function is aligned or not and find a shelling variable order π if it is shellable in polynomial time.

5. Relations among shellable functions

In this section, we show relations among various shellable Boolean functions. Let $\mathscr{S}$ be the class of all shellable functions, $\mathscr{LE}$ be the class of all lexico-exchange functions, $\mathscr{TS}$ be the class of all tree-shellable functions, $\mathscr{OTS}$ be the class of all ordered tree-shellable functions and $\mathscr{A}$ be the class of all aligned functions, respectively.

Theorem 7. *$\mathscr{TS}$ is a proper subclass of $\mathscr{S}$.*

Proof. Consider $f_1 = x_1x_2x_3 + x_1x_2x_4 + x_1x_2x_5 + x_1x_3x_5 + x_1x_3x_6 + x_2x_4x_5 + x_2x_4x_6 + x_3x_4x_6 + x_3x_5x_6 + x_4x_5x_6$. f_1 is shellable because $f_1 = x_1x_2x_3 + x_1x_2\overline{x_3}x_4 + x_1x_2\overline{x_3}\,\overline{x_4}x_5 + x_1\overline{x_2}x_3x_5 + x_1\overline{x_2}x_3\overline{x_5}x_6 + \overline{x_1}x_2x_4x_5 + \overline{x_1}x_2x_4\overline{x_5}x_6 + \overline{x_1}x_3x_4x_6 + \overline{x_1}x_3\overline{x_4}x_5x_6 + \overline{x_2}\,\overline{x_3}x_4x_5x_6$. We can see that f is not tree-shellable by checking all the possible trees representing f_1. □

Theorem 8. *$\mathscr{OTS}$ is a proper subclass of $\mathscr{TS}$.*

Proof. Consider $f_2 = x_1x_2x_3 + x_1x_2x_4 + x_1x_2x_5 + x_1x_3x_6 + x_2x_4x_5 + x_2x_4x_6 + x_3x_4x_6 + x_3x_5x_6$. The BDT shown in Fig. 2 witnesses that f_2 is tree-shellable. We can see that f is not ordered tree-shellable by checking all variable orders. □

Theorem 9. *$\mathscr{A}$ is a proper subclass of $\mathscr{OTS}$.*

Proof. From Theorem 6, every aligned function is also ordered tree-shellable. Consider $f_3 = x_1x_2 + x_1x_3 + x_3x_4$. As shown in Fig. 1(a), f_3 is ordered tree-shellable with respect

to variable order $x_1x_2x_3x_4$. We can see that f is not aligned by checking all variable orders. □

Theorem 10. *$\mathcal{OTS}$ is equivalent to $\mathcal{LE}$.*

Proof. Let f be a positive Boolean function represented by a PDNF $f = \bigvee_{k=1}^{m} \bigwedge_{i\in I_k} x_i$. We assume w.l.o.g. that $\pi(k) = k$ for any k.

First, assume that f is lexico-exchange with respect to a variable order π. Then we prove that f satisfies the condition of Theorem 5 with the same variable order. That is, we prove that, for any prime implicant I_k and $s \notin I_k$, $s \prec_\pi \max(I_k)$, if there exist prime implicants I_l that satisfy $(I_k \cup \{s\}) \cap [s] = I_l \cap [s]$, at least one of them satisfies $I_l \subsetneq I_k \cup \{s\}$.

Assume that there exists $I_{l'}$ that satisfies $(I_k \cup \{s\}) \cap [s] = I_{l'} \cap [s]$. Then, $s = \min(I_{l'} \setminus I_k)$ and $I_{l'} \prec_{\mathrm{L}} I_k$ hold. Thus, from the definition of a lexico-exchange function, there exists I_l which satisfies $I_l \prec_{\mathrm{L}} I_k$ and $I_l \setminus I_k = \{s\}$. If $(I_k \cup \{s\}) \cap [s] \subsetneq I_l \cap [s]$, it contradicts $I_l \setminus I_k = \{s\}$. If $(I_k \cup \{s\}) \cap [s] \supsetneq I_l \cap [s]$, there exists $t \leqslant s$ which satisfies $t \in I_k$ and $t \notin I_l$. It contradicts $I_l \prec_{\mathrm{L}} I_k$. Thus, $(I_k \cup \{s\}) \cap [s] = I_l \cap [s]$ holds. Also, from the equality $I_l \setminus I_k = \{s\}$, $I_l \subsetneq I_k \cup \{s\}$ holds.

Next, we assume that f is ordered tree-shellable with respect to π. We prove that f is lexico-exchange using the condition of Theorem 5. Let I_i and I_j be any prime implicants satisfying $I_i \prec_{\mathrm{L}} I_j$. Let $h = \min(I_i \setminus I_j)$. Then, $(I_j \cup \{h\}) \cap [h] = I_i \cap [h]$ holds. Therefore, condition (ii) of Theorem 5 is satisfied. That is, there exists I_l such that $(I_j \cup \{h\}) \cap [h] = I_l \cap [h]$ and $I_l \subsetneq I_j \cup \{h\}$. As $h \notin I_j$, $(I_j \cup \{h\}) \cap [h] = I_l \cap [h]$ means that $I_l \prec_{\mathrm{L}} I_j$ and $I_l \subsetneq I_j \cup \{h\}$ means that $I_l \setminus I_j = \{h\}$. Hence, f is lexico-exchange with respect to π. □

6. Tree-shellability of quadratic functions

In this section, we consider the case when a positive Boolean function is quadratic. A Boolean function is called *quadratic* if all the terms consist of exactly two literals. A positive quadratic function f is represented by a graph $G = (V, E)$, where $V = \{x_1, x_2, \ldots, x_n\}$ and $(x_u, x_v) \in E$ iff x_ux_v is a term of f.

It is shown in [3] that a quadratic function is lexico-exchange (or ordered tree-shellable) iff it is represented by a cotriangulated graph. A graph is called *cotriangulated* if any induced subgraph contains a vertex whose nonneighbors form an independent set. We call such a vertex *cosimplicial* and we call a vertex which has no edges *isolated*. On the shelling variable order for quadratic functions, [3] shows that $x_{\pi(1)}, \ldots, x_{\pi(n)}$ is a shelling variable order of f if $x_{\pi(k)}$ is a cosimplicial node or an isolated node of the graph induced by $x_{\pi(k)}, \ldots, x_{\pi(n)}$ for any k $(1 \leqslant k \leqslant n-1)$.

We show that it is also the necessary condition for π to be a shelling variable order. Let G denote the graph representing f and $G_\pi(k)$ denote the graph induced by $x_{\pi(k)}, \ldots, x_{\pi(n)}$. Here $G = G_\pi(1)$.

Theorem 11. *For an ordered tree-shellable quadratic function, if π is a shelling variable order, $x_{\pi(k)}$ is a cosimplicial node or an isolated node of $G_\pi(k)$ for any k $(1 \leqslant k \leqslant n-1)$.*

Proof. The outline of this proof is as follows. In the first step, we list all the requirements for π to be a shelling variable order using Theorem 5. Let us consider any $I \cup \{i\}$ $(I \in PI(f),\ i \notin I)$. Here, $I \cup \{i\}$ corresponds to $I_k \cup \{s\}$ in Theorem 5. Then $I \cup \{i\}$ should satisfy the condition of Theorem 5 if $i \prec \max(I)$. In the second step, we show that π satisfies Theorem 11 when all the requirements are satisfied.

Now we show the detail of this proof. In the first step, we consider the following four cases depending on terms other than I. Let $I = \{k, l\}$. For simplicity, we call a term J which satisfies $i \in J$ and $J \subsetneq I \cup \{i\}$ a term of type A, and one which satisfies $i \in J$, $J \nsubseteq I \cup \{i\}$ a term of type B. Note that there exist at most two terms of type A, $x_i x_k$ and $x_i x_l$.

Case 1: The case when there exists at least one term of type A and no term of type B. There are six possible orders of x_i, x_k, x_l in π, that is, $i \prec k \prec l$, $i \prec l \prec k$, $k \prec i \prec l$, $k \prec l \prec i$, $l \prec k \prec i$ and $l \prec i \prec k$. We consider which of them can be allowed using Theorem 5. If $(k \prec i) \wedge (l \prec i)$ (that is, if $k \prec l \prec i$ or $l \prec k \prec i$), $i \prec \max(I)$ does not hold. Then the condition of Theorem 5 need not be considered in these two cases. Assume that there is only one term of type A and let $x_i x_k$ be the term. If $l \prec i$, condition (i) of Theorem 5 is satisfied because there exists no term $I_{l'}$ which satisfies $(I \cup \{i\}) \cap \{\pi(1), \pi(2), \ldots, i\} = I_{l'} \cap \{\pi(1), \pi(2), \ldots, i\}$. In other cases, condition (ii) is satisfied because there exists a term $x_i x_k$. Therefore, the condition is always satisfied and no requirement is obtained. In a similar manner, we can see that the condition is always satisfied even when there are two terms of type A.

Case 2: The case when there exists at least one term of type B and no term of type A. Assume there is only one term of type B. Let the term be $x_i x_t$ $(t \neq k, t \neq l)$. Among all the orders of x_i, x_k, x_l, x_t in π, if $(k \prec i) \wedge (l \prec i)$, $i \prec \max(I)$ does not hold. We can see that condition (i) is satisfied if $(k \prec i) \vee (l \prec i) \vee (t \prec i)$. In other cases, that is, if $(i \prec k) \wedge (i \prec l) \wedge (i \prec t)$, condition (i) is not satisfied because there exists a term $x_i x_t$. Moreover, condition (ii) is not satisfied either because there exists no term other than I that is a subset of $\{i, k, l\}$. From the above discussions, the variable order must satisfy

$$((k \prec i) \wedge (l \prec i)) \vee ((k \prec i) \vee (l \prec i) \vee (t \prec i)) = (k \prec i) \vee (l \prec i) \vee (t \prec i).$$

Therefore, a requirement $(k \prec i) \vee (l \prec i) \vee (t \prec i)$ is obtained. If there are more than one terms of type B, the above condition must be satisfied for all of them.

Case 3: The case when there exist both prime implicants of types A and B. Assume that there is only one term of type A, $x_i x_k$, and $x_i x_{t_j}$ $(t_j \neq k, t_j \neq l, j = 1, 2, \ldots)$ be terms of type B. From similar discussions as previous cases, we can obtain the following. Condition (i) is satisfied iff $l \prec i$ and $(k \prec i) \vee (l \prec i) \vee (t_j \prec i)$ are satisfied. Condition (ii) is satisfied by $x_i x_k$ iff $i \prec l$. After all, the variable order must

satisfy

$$((k \prec i) \wedge (l \prec i)) \vee ((l \prec i) \wedge ((k \prec i) \vee (l \prec i) \vee (\wedge_j t_j \prec i))) \vee (i \prec l).$$

As the formula is always true, the condition is always satisfied. Therefore no requirement is obtained. It is similar even when there exist two terms of type A.

Case 4: The case when there exists neither a term of type A nor a term of type B. That is, the case when x_i is an isolated node. In this case, condition (i) is always satisfied and no requirement is obtained.

From the above discussion, requirements for variable order π are obtained only in Case 2. The requirements are produced for all I and i classified to Case 2 and all of them must be satisfied. Let $R(G)$ denote the set of all the requirements produced by a function whose corresponding graph is G. Now, we should note that the requirements produced in Case 2 are the conjunction of the form $* \prec i$. This means that x_i cannot be the first variable.

In the second step, we prove using induction on k that π is not a shelling variable order if $x_{\pi(k)}$ is neither a cosimplicial node nor an isolated node of $G_\pi(k)$. For $k=1$, assume that $x_{\pi(1)}$ is not a cosimplicial node. Then there exists an edge (x_k, x_l) such that neither x_k nor x_l is a neighbor of $x_{\pi(1)}$. Let $I = x_k x_l$ and $i = x_{\pi(1)}$. Then $I \cup \{i\}$ is classified to Case 2 above. In this case, the obtained requirements indicates that x_i cannot be the first variable so as to make π a shelling variable order.

If $\pi(1), \ldots, \pi(j)$ $(1 \leqslant j \leqslant n-1)$ are chosen appropriately, some of the requirements for π are already satisfied and removed. Thus we prove the following claim.

Claim. *Assume that π satisfies that $x_{\pi(j)}$ is a cosimplicial node or an isolated node of $G_\pi(j)$ for any j $(1 \leqslant j \leqslant k-1)$. Then the remaining requirements are equivalent to $R(G_\pi(k))$.*

If it holds, similarly to the case of $k=1$, $x_{\pi(k)}$ should be a cosimplicial node or an isolated node of $G_\pi(k)$.

Proof. It is obvious for $k=1$ where $R(G_\pi(k)) = R(G)$. Assume that this claim holds for $k=m$. Then the remaining requirements are equivalent to $R(G_\pi(k))$. When $x_{\pi(k)}$ is removed from $G_\pi(k)$, some more requirements are removed.

If $x_{\pi(k)}$ is an isolated node, it does not appear in the requirements. Thus $R(G_\pi(k)) = R(G_\pi(k-1))$ and the claim holds. If $x_{\pi(k)}$ is a cosimplicial node, it appears only in the form $\pi(k) \prec s$ for some s. Thus, when it is chosen as the kth variable, all the sum terms in the formula that include $\pi(k) \prec s$ are removed. Clearly no other requirements are removed. It is not difficult to see that a term is removed iff it is not generated from a graph without $x_{\pi(k)}$. Then the remaining requirements are equivalent to $R(G_\pi(k))$. □

At last, we show an example. The graph representation of $f = x_1x_2 + x_2x_3 + x_3x_4$ (that is, $G_\pi(1)$) is shown in Fig. 3(a). x_2 and x_3 are cosimplicial nodes in $G_\pi(1)$. For $I = \{1,2\}$ and $i=4$, $J = \{3,4\}$ is a term of type B and there is no term of type A.

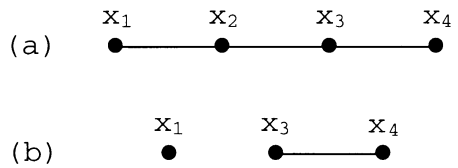


Fig. 3. An example for Theorem 11. ($f = x_1x_2 + x_2x_3 + x_3x_4$).

Thus, we obtain a requirement $(1 \prec 4) \vee (2 \prec 4) \vee (3 \prec 4)$. Similarly, for $I = \{3,4\}$ and $i = 1$, we obtain $(2 \prec 1) \vee (3 \prec 1) \vee (4 \prec 1)$. No other requirement is obtained from this graph. Thus we can see that neither x_1 nor x_4 can be the first variable. If we choose x_2 as the first variable, the remaining graph $G_\pi(2)$ is as shown in Fig. 3(b). In $G_\pi(2)$, x_3 and x_4 are cosimplicial nodes and x_1 is an isolated node. Because both of the above requirements are satisfied by choosing x_2 as the first variable, any variable can be chosen as the next variable. □

The next theorem shows that tree-shellability is equivalent to ordered tree-shellability on quadratic functions.

Theorem 12. *A tree-shellable quadratic function is ordered tree-shellable.*

Proof. Let T be a BDT which witnesses that a quadratic function f is tree-shellable. Let P_m be the leftmost 1-path in T. We show that any variable order which is consistent with the order of variables in P_m is a shelling variable order of f.

Let $P_{k_1}, \ldots, P_{k_t}$ be the paths that diverge from P_m at the node labeled by x_l. As $x_l \in P_{k_i}$ $(1 \leqslant i \leqslant t)$, after the 1-edge from the node labeled by x_l, P_{k_i} includes at most one positive literal. For example, in Fig. 1(a), two paths x_1x_2 and $x_1\overline{x_2}x_3$ diverge from the leftmost path at the node labeled by x_1. Clearly, both of the paths include one 1-edge after the 1-edge from x_1.

It is easy to see that a positive function all of whose terms have only one literal is ordered tree-shellable with respect to any variable order. In Fig. 1(a), the variable order of the subtree whose root is the node labeled by x_2 can be changed arbitrarily. Therefore, it is possible to change the order of variables in the 1-paths so that it is consistent with that of P_m. □

7. Conclusion

In this paper, we have defined tree-shellable and ordered tree-shellable Boolean functions, which have the property that every 1-path of a BDT representation has a one-to-one correspondence to a prime implicant. We have shown some basic properties of tree-shellable and ordered tree-shellable functions, and clarified relations among classes of several shellable functions. The notion of tree-shellability has also made it possible to characterize several subclasses of shellable functions in terms of tree-shellability. That is, tree-shellable, lexico-exchange (equivalently ordered tree-shellable) and aligned functions correspond to BDT, OBDT and leveled OBDT, respectively.

References

[1] M.O. Ball, G.L. Nemhauser, Matroids and a reliability analysis problem, Math. Oper. Res. 4 (1979) 132–143.
[2] M.O. Ball, J.S. Provan, Disjoint products and efficient computation of reliability, Oper. Res. 36 (1988) 703–715.
[3] C. Benzaken, Y. Crama, P. Duchet, P.L. Hammer, F. Maffray, More characterizations of triangulated graphs, J. Graph Theory 14 (1990) 413–422.
[4] J.C. Bioch, T. Ibaraki, Complexity of identification and dualization of positive Boolean functions, Inform. Comput. 123 (1995) 50–63.
[5] E. Boros, Dualization of aligned Boolean function, RUTCOR Research Report 9–94 1994.
[6] E. Boros, Y. Crama, O. Ekin, P. L. Hammer, T. Ibaraki, A. Kogan, Boolean normal forms, shellability and reliability computations, SIAM J. Discrete Math. 13 (2000) 212–226.
[7] E. Boros, P.L. Hammer, T. Ibaraki, K. Kawakami, Polynomial time recognition of 2-monotonic positive Boolean functions given by an oracle, SIAM J. Comput. 26 (1997) 93–109.
[8] H. Brugesser, P. Mani, Shellable decompositions of cells and spheres, Math. Scand. 29 (1971) 199–205.
[9] Y. Crama, Dualization of regular Boolean functions, Discrete Appl. Math. 16 (1987) 79–85.
[10] G. Danaraj, V. Klee, Shellings of spheres and polytopes, Duke Math. J. 41 (1974) 443–451.
[11] C. Domingo, Polynomial time algorithm for some self-duality problems, Proc. CIAC'97, Lecture Notes in Computer Science, vol. 1203, Springer, Berlin, 1997, pp. 171–180.
[12] T. Eiter, G. Gottlob, Identifying the minimal transversals of a hypergraph and related problems, SIAM J. Comput. 24 (1995) 1278–1304.
[13] M.L. Fredman, L. Khachiyan, On the complexity of dualization of monotone disjunctive normal forms, J. Algorithms 21 (1996) 618–628.
[14] K. Makino, T. Ibaraki, The maximum latency and identification of positive Boolean functions, SIAM J. Comput. 26 (1997) 1363–1383.
[15] U.N. Peled, B. Simeone, Polynomial-time algorithm for regular set-covering and threshold synthesis, Discrete Appl. Math. 12 (1985) 57–69.
[16] J.S. Provan, M.O. Ball, Efficient Recognition of matroids and 2-monotonic systems, in: R. Ringeisen, F. Roberts (Eds.), Applications of Discrete Mathematics, SIAM, Philadelphia, 1988,, pp. 122–134.