数理解析研究所講究録
1054 巻 1998 年 54-65

On Tree-Shellable Boolean Functions

Yasuhiko TAKENAGA*, Kouji NAKAJIMA† and Shuzo YAJIMA‡
武永 康彦　中嶋廣二　矢島脩三
* Department of Computer Science and Information Mathematics,
University of Electro-Communications
takenaga@cs.uec.ac.jp
† Department of Information Science, Graduate School of Engineering, Kyoto University
‡ Faculty of Informatics, Kansai University

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 by adding negative literals to each term. The notion of shellablility was originally used in the theory of simplicial complexes and polytopes (for example, in [6, 7]). More recently, it is studied for its importance on reliability theory (for example, in [1, 2, 8]).

Various subclasses of shellable Boolean functions have been proposed, e.g. lexico-exchange function [2], aligned functions [4]. Both of them have some good properties when their product terms are ordered lexicographically.

In this paper, we define a tree-shellable function and an ordered tree-shellable function as restricted shellable Boolean functions. A tree-shellable function is a Boolean function such that the number of prime implicants equals the number of paths from the root to a leaf node in a binary decision tree (BDT) representation of it. An ordered tree-shellable function has the similar relation with an ordered BDT, which is a BDT such that, on all the 1-paths, variables appear according to a total order of variables.

A tree-shellable function has the following good properties. If a Boolean function is shellable, one can easily solve the following problem.
[Union of Product Problem] ([2])
Input: $Pr[x_i = 1](1 \leq i \leq n),\ f(x_1, ..., x_n)$
Output: $Pr[f(x_1, ..., x_n) = 1]$
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 tree-shellable, it is easy to compute the dual of f. The dual of a Boolean function $f(x_1, ..., x_n)$ is defined by $f^d = \overline{f(\overline{x_1}, ..., \overline{x_n})}$. The idea of the duality plays an important role in mathematical programming. In general, it seems to take exponential time to compute the DNF representation of the dual f^d from the DNF representation of f. However, if f is tree-shellable, it is possible to compute the irredundant DNF representation of f^d in polynomial time from a DNF representation of f. If 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 label 1 and label 0 for every leaf node.

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. 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 shelling variable order of ordered tree-shellabile quadratic functions.

2 Preliminaries

2.1 Basic Notations

Let $B = \{0, 1\}$, n be a natural number, and $[n] = \{1, 2, ..., n\}$. Let $[0] = \emptyset$. Let π be a permutation on $[n]$. π represents a total order of integers. Let $\pi(i)$ be the i-th 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.

$$\begin{cases} \min_\pi(S) = h & \text{if } h \in S \text{ and } h \prec_\pi i \quad \text{for all } i \in S \setminus \{h\} \\ \max_\pi(S) = h & \text{if } h \in S \text{ and } i \prec_\pi h \quad \text{for all } i \in S \setminus \{h\} \end{cases}$$

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 $I_s \cap \{\pi(1), ..., \pi(i-1)\} = I_t \cap \{\pi(1), ..., \pi(i-1)\}$, $\pi(i) \in I_s$ and $\pi(i) \notin I_t$ 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 $f(x_1, ..., x_n)$ be a Boolean function. We denote $f \geq g$ 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 $\bigwedge_{i \in I} x_i \bigwedge_{j \in J} x_j$ which satisfy $\bigwedge_{i \in I} x_i \bigwedge_{j \in J} x_j \leq f$, where $I, J \subseteq [n]$. An implicant which satisfies $\bigwedge_{i \in I - \{s\}} x_i \bigwedge_{j \in J} x_j \not\leq f$ for any $s \in I$ and $\bigwedge_{i \in I} x_i \bigwedge_{j \in J - \{t\}} x_j \not\leq f$ for any $t \in J$ is called a *prime implicant* of f.

An expression of the form $f = \bigvee_{k=1}^{m} \left(\bigwedge_{i \in I_k} x_i \bigwedge_{j \in J_k} \overline{x_j} \right)$ 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. If f is a positive Boolean function, f can be represented as a *positive* DNF (PDNF). A PDNF is a DNF such that $J_k = \emptyset$ for all k. 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 $(1 \leq k, l \leq m, k \neq l)$. For an irredundant PDNF, let $PI(f)$ be the set of all I_k. $PI(f)$ represents the prime implicants 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 the other nodes are labeled by variables. 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. A path P of an OBDT is represented by a sequence of literals. If the k-th 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 k-th 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 k-th element of P and the prefix of P with length k, respectively. Note that, on every 1-path, each variable appears at most once.

When the 0-edge and the 1-edge of node v point the same node, v is called to be a *redundant node*. An OBDD which has no redundant node is called a *reduced OBDT*. 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. An OBDT which satisfy $S(P) = [\max_\pi S(P)]$ 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 $J_1, ..., J_m (\subseteq [n])$ which satisfy the following conditions.

1. For any l $(1 \leq l \leq m)$, $\bigvee_{k=1}^{l} \bigwedge_{i \in I_{\pi_T(k)}} x_i = \bigvee_{k=1}^{l} \Big(\bigwedge_{i \in I_{\pi_T(k)}} x_i \bigwedge_{j \in J_{\pi_T(k)}} \overline{x_j} \Big)$.
2. For any s, t such that $1 \leq s < t \leq m$, $(I_s \cap J_t) \cup (I_t \cap J_s) \neq \emptyset$.

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 [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 $I_l \setminus I_j = \{h\}$, where $h = \min(I_i \setminus I_j)$. f is lexico-exchange if there exists π such that f is lexico-exchange with respect to π.

It is proved in [5] 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 satisfy the following conditions.

- T has m 1-paths $P_1, ..., P_m$.
- Each P_k corresponds to a term I_k by the rule that $i \in I_k$ iff $x_i \in P_k$.

Proof We have only to prove that the second condition holds. Let $P_1, ..., P_m$ 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. Then P_k represents the product term $\bigwedge_{i \in pos(P_k)} x_i \bigwedge_{i \in neg(P_k)} \overline{x_i}$.

If $pos(P_k) \subsetneq I_l$ for some k, l, minterm $\bigwedge_{i \in pos(P_k)} x_i \bigwedge_{i \notin pos(P_k)} \overline{x_i}$, which must be 0, becomes 1 by P_k. Thus, for any k, $pos(P_k) \supseteq I_l$ must hold for some l. We can assume w.l.o.g. that $pos(P_k) \supseteq I_k$. If $pos(P_k) \supsetneq I_k$, minterm $\bigwedge_{i \in I_k} x_i \bigwedge_{i \notin I_k} \overline{x_i}$, which is a minimum true point corresponding to I_k, cannot be 1 by P_k. Clearly, no other path makes it 1. Therefore, $pos(P_k) = I_k$ holds for any k $(1 \leq k \leq m)$. □

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

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 have 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.

Theorem 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 $\bigvee_{k=1}^{l} \bigwedge_{i \in I_{\pi_T(k)}} x_i = \bigvee_{k=1}^{l} (\bigwedge_{i \in pos(P_{\pi_T(k)})} x_i \bigwedge_{j \in neg(P_{\pi_T(k)})} \overline{x_j})$ for any l $(1 \leq l \leq m)$.

Proof For simplicity, we assume w.l.o.g. that $\pi_T(k) = k$ for any k.

Let $J_l = \{ i \mid \overline{x_i} \in P_l \}$. As $\bigwedge_{i \in I_l} x_i = (\bigwedge_{i \in I_l} x_i \bigwedge_{j \in J_l} \overline{x_j}) \bigvee (\bigvee_{j \in J_l} \bigwedge_{i \in I_l \cup \{j\}} x_i)$, we have to show that $\bigwedge_{i \in I_l \cup \{j\}} x_i \leq \bigvee_{k=1}^{l-1} (\bigwedge_{i \in pos(P_k)} x_i \bigwedge_{j \in neg(P_k)} \overline{x_j})$ for any $j \in J_l$.

Let x be an arbitrary assignment which makes $\bigwedge_{i \in I_l \cup \{j\}} x_i$ true. That is, $x_j = x_i = 1$ $(\forall i \in I_l)$ in x. As $f(x) = 1$, there must be a 1-path P_s that is traversed by x. As x satisfies $x_i = 1$ for any $i \in I_l$, $l \prec_{\pi_T} s$ never holds. That is, the assignment x makes $\bigvee_{k=1}^{l-1} (\bigwedge_{i \in pos(P_k)} x_i \bigwedge_{j \in neg(P_k)} \overline{x_j})$ true. Therefore, any assignment which makes $\bigvee_{j \in J_l} \bigwedge_{i \in I_l \cup \{j\}} x_i$ true also makes $\bigvee_{k=1}^{l-1} (\bigwedge_{i \in pos(P_k)} x_i \bigwedge_{j \in neg(P_k)} \overline{x_j})$ true. □

If a BDT T represents $f = \bigvee_{k=1}^{m} \bigwedge_{i \in I_k} x_i$ and has exactly m paths, T proves that f is tree-shellable. The next corollary holds for f and T.

Corollary 3 For any 1-path P_k of T and any $\overline{x_s} \in P_k$, there exists I_l which satisfy $P_k^{(t-1)} = P_l^{(t-1)}$, $P_k^t = \overline{x_s}$, $P_l^t = x_s$ and $I_k \cup \{s\} \supset I_l$ for some t.

Proof $z = \bigwedge_{i \in I_k \cup \{s\}} x_i \bigwedge_{j \notin I_k \cup \{s\}} \overline{x_j}$ is a true minterm of f. However, it is not an implicant of $\bigwedge_{i \in I_k} x_i \bigwedge_{j \in J_k} \overline{x_j}$ because P_k includes $\overline{x_s}$. Thus, there must exist a path P_z which make z true. P_z is on P_k until the node labeled by x_s. Therefore, $P_k^{(t-1)} = P_z^{(t-1)}$, $P_k^t = \overline{x_s}$ and $P_z^t = x_s$ hold.

Assume that $I_z \not\subset I_k \cup \{s\}$, where $I_z = \{i \mid x_i \in P_z\}$. Then, as P_z includes a positive literal which is not included in $I_k \cap \{s\}$, P_z cannot make z true. It is a contradiction. That is, $I_z \subset I_k \cup \{s\}$. □

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. Let π be the *shelling variable order* of f.

As an ordered tree-shellable function is tree-shellable, Proposition 1 and Theorem 2 also hold for ordered tree-shellable functions. It is easy to see that the shelling term order of an ordered tree-shellable function is equivalent to the lexicographical order based on the shelling variable order.

Theorem 4 A 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 $i \notin I_k$, $i \prec_\pi \max(I_k)$, either
i) there does not exist I_l such that $(I_k \cup \{i\}) \cap \{x_{\pi(1)}, x_{\pi(2)}, ..., i\} = I_l \cap \{x_{\pi(1)}, x_{\pi(2)}, ..., i\}$, or
ii) if such prime implicants exist, at least one of them satisfy $I_k \cup \{i\} \supset I_l$.

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 terms of 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: $P_k = \epsilon$ (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 start with $P_k x_i$, then $P_k = P_k \cdot \overline{x_i}$.

Next, we show that the OBDT constructed by this algorithm proves that f is ordered tree-shellable if f satisfies the condition of this theorem. Let F_l be the function represented by the OBDT with first l paths $P_1, ..., P_l$. To show that f is ordered tree-shellable, we have to prove that $F_l = \bigvee_{k=1}^{l} \bigwedge_{i \in I_k} x_i$ for any l by induction on l. When $l = 1$, F_1 clearly represents the first term of f. Assume that F_{k-1} represents the sum of first $k-1$ terms. Let $J_k = \{i \mid \overline{x_i} \in P_k\}$. When $s \in J_k$, there exists a term I_t which satisfy $(I_k \cup \{s\}) \cap [s] = I_t \cap [s]$. From condition ii), there exists I_t which satisfy $\bigwedge_{i \in I_k \cup \{s\}} x_i \leq \bigwedge_{j \in I_t} x_j$. From the induction assumption, $\bigwedge_{i \in I_k \cup \{s\}} x_i$ is an implicant of F_{k-1}. As $F_k = F_{k-1} \bigvee (\bigwedge_{i \in I_k} x_i \bigwedge_{j \in J_k} \overline{x_j}) = F_{k-1} \bigvee (\bigwedge_{i \in I_k} x_i \bigwedge_{j \in J_k} \overline{x_j}) \bigvee (\bigvee_{j \in J_k} \bigwedge_{i \in I_k \cup \{j\}} x_i) = F_{k-1} \bigvee (\bigwedge_{i \in I_k} x_i)$, F_k represents the sum of first k terms.

[only if] It is proved similarly to Corollary 3.

□

4.3 Aligned Function

Definition [4] 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 [4] can be written as follows if we use our terms.

Theorem 5 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 an aligned function is an ordered tree-shellable function. It can be seen as another definition of an aligned function. It is also shown in [4] 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 $\mathcal{S}$ be the class of all shellable functions, $\mathcal{LE}$ be the class of all lexico-exchange functions, $\mathcal{TS}$ be the class of all tree-shellable functions, $\mathcal{OTS}$ be the class of all ordered tree-shellable functions and $\mathcal{A}$ be the class of all aligned functions, respectively.

Theorem 6 $\mathcal{TS}$ is a proper subclass of $\mathcal{S}$.

Sketch of proof
$f = x_1x_2x_3+x_1x_2x_4+x_1x_2x_5+x_1x_3x_5+x_1x_3x_6+x_2x_4x_5+x_2x_4x_6+x_3x_4x_6+x_3x_5x_6+x_4x_5x_6$ is shellable but not tree-shellable. □

Tree-shellability of a Boolean function is checked by Algorithm TS (Fig.1). It checks whether $f = \bigvee_{k=1}^{m} \bigwedge_{i \in I_k} x_i$ is tree shellable and if f is tree shellable, it outputs m 1-paths which construct a BDT representation of f.

Note that $I_j \setminus P = \{s | s \in I_j, x_s \notin P\}$ and $s(P)$ is the sequence obtained by deleting the last literal from P. This algorithm checks the tree-shellability of a function based on Corollary 3. In this algorithm, we first give x_1 as the label of the root node. Then, check if the condition of Corollary 3 is satisfied at the root node, and examine recursively if two subfunctions $f|_{x_1=0}$ and $f|_{x_1=1}$ are both tree-shellable. If the above conditions hold, f is tree-shellable and there exists a BDT with the root node labeled by x_1 which proves 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 in general.

Theorem 7 $\mathcal{OTS}$ is a proper subclass of $\mathcal{TS}$.

Sketch of proof $f = x_1x_2x_3+x_1x_2x_4+x_1x_2x_5+x_1x_3x_6+x_2x_4x_5+x_2x_4x_6+x_3x_4x_6+x_3x_5x_6$ is tree-shellable but not ordered tree-shellable. □

It is possible to check the ordered tree-shellability by an algorithm similar to Algorithm TS. The only difference is that, in this case, the variable order must be the same for all subtrees.

Theorem 8 $\mathcal{A}$ is a proper subclass of $\mathcal{OTS}$.

Sketch of proof From Theorem 4, every aligned function is also ordered tree-shellable. $f = x_1x_2 + x_1x_3 + x_3x_4$ is ordered tree-shellable but not aligned. □

Theorem 9 $\mathcal{OTS}$ is equivalent to $\mathcal{LE}$.

Proof Let f be a positive Boolean function represented by a PDNF $f = \bigvee_{k=1}^{m} \bigwedge_{i \in I_k} x_i$. First, we assume that f is lexico-exchange with respect to a variable order π. We can assume w.l.o.g. that $\pi(k) = k$ for any k. If $h = \min(I_i \setminus I_j)$ for some I_i, I_j $(I_i \prec_L I_j)$, there exists I_l which satisfies $I_l \prec_L I_j$ and $I_l \setminus I_j = \{h\}$. It means that $I_j \cup \{h\} \supset I_l$. Assume $I_l \cap [h] \neq (I_j \cup \{h\}) \cap [h]$, then $I_l \cap [h-1] \neq I_j \cap [h-1]$. However, it contradicts with $I_l \setminus I_j = \{h\}$. Hence $I_l \cap [h] = (I_j \cup \{h\}) \cap [h]$. Therefore, for I_j and h, condition ii) of Theorem 4 is satisfied. For I_j and h' $(h' \notin I_j)$, if there exists no I_i which satisfy $h' = \min(I_i \setminus I_j)$, condition i) of Theorem 4 is satisfied. Hence, f is ordered tree-shellable with respect to π.

Algorithm TS

input $I_1, ..., I_m$

$I = \{1, 2, ..., m\}$, $P = \epsilon$ (empty sequence)

if $checkTS(1) = 1$ then output $P_1, ..., P_m$

  else f is not tree-shellable

checkTS(level)

```
for j = 1 to m do I'_j = I_j \ P
if I = {s} for some s and I'_s = ∅
    then P_s = P, return 1
for i = 1 to n do
    if x_i ∉ P then
        if i ∈ I'_s for every s ∈ I then
            P = P · x_i
            if checkTS(level + 1) = 1
                then P = s(P), return 1
                else P = s(P), return 0
        else if i ∈ I'_s and i ∉ I'_t for some s, t ∈ I then
            I = {s | s ∈ I, i ∉ I'_s}, P = P · x̄_i
            J(level + 1) = {t | t ∈ I, i ∈ I'_t}, Q(level + 1) = P · x_i
            if ∀s ∈ I ∃t ∈ J(level + 1) I'_s ∪ {i} ⊃ I'_t then
                if checkTS(level + 1) = 1 then
                    I = J(level + 1), P = Q(level + 1)
                    if checkTS(level + 1) = 1
                        then P = s(P), return 1
            P = s(P), I = I ∪ Q(level + 1)
return 0
```

Figure 1: Algorithm to check tree-shellability.

Second, we assume that f is ordered tree-shellable with respect to π. Let $I_i \prec_L I_j$ and $h = \min(I_i \setminus I_j)$. Then, $(I_j \cup \{h\}) \cap [h] = I_i \cap [h]$. From condition ii) of Theorem 4, there exists I_l such that $(I_j \cup \{h\}) \cap [h] = I_l \cap [h]$ and $I_j \cup \{h\} \supseteq I_l$. As $h \notin I_j$, $I_j \cup \{h\} \supseteq I_l$ 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 $V = \{x_1, x_2, ..., x_n\}$ and $(x_u, x_v) \in E$ iff $x_u x_v$ 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.* On the shelling variable order for quadratic functions, [3] shows that $x_{\pi(1)}, ..., x_{\pi(n)}$ is a shelling variable order of f if $x_{\pi(k)}$ is a cosimplicial node of the graph induced by $x_{\pi(k)}, ..., x_{\pi(n)}$ for any k $(1 \leq k \leq n-1)$.

We show the necessary and sufficient condition that π becomes a shelling variable order.

Theorem 10 For an ordered tree-shellable quadratic function, a variable order is a shelling variable order iff $x_{\pi(k)}$ is a cosimplicial node or an isolated node of the graph induced by $x_{\pi(k)}, ..., x_{\pi(n)}$ for any k $(1 \leq k \leq n-1)$.

Proof Consider $I \cup \{i\}$ for some $I \in PI(f), i \notin I$ and check when it satisfies the condition of Theorem 4. For simplicity, we call a term J which satisfy $i \in J, J \subset I \cup \{i\}$ as a term of type A, and one which satisfy $i \in J, J \not\subset I \cup \{i\}$ as a term of type B. Let $I = \{k, l\}$. We consider the following four cases.

Case1: The case when there exists at least one term of type A and no term of type B. When there is only one term of type A, let $x_i x_k$ be the term. If $l \prec i$, condition i) is satisfied. If $i \prec l$, condition ii) is satisfied. Therefore, the condition is always satisfied. It is easy to see that the condition is always satisfied even when there are two terms of type A.

Case2: The case when there exists at least one term of type B and no term of type A. When there is only one term of type B, let x_i, x_t $(t \neq k, t \neq l)$ be the term. If $\max(I) \prec i$, $I \cup \{i\}$ is out of consideration. If $i \prec \max(I)$, to satisfy condition i), either $k \prec i$, $l \prec i$ or $t \prec i$ must be satisfied. Obviously, condition ii) is never satisfied. Therefore, the variable order must satisfy $((k \prec i) \wedge (l \prec i)) \vee ((k \prec i) \vee (l \prec i) \vee (t \prec i)) = (k \prec i) \vee (l \prec i) \vee (t \prec i)$. If there are more than one terms of type B, the above condition must be satisfied for all of them.

Case3: The case when there exist prime implicants of type A and type B. Assume that $J = \{i,k\}$ be a term of type A and $K_j = \{i,t_j\}$ $(t_j \neq k, t_j \neq l, j = 1,2,...)$ be terms of type B. To satisfy condition i), it is necessary that $l \prec i$ for J, and that $k \prec i$, $l \prec i$ or $t_j \prec i$ for K_j. Condition ii) is satisfied by J when $i \prec l$. After all, the variable order must satisfy $((k \prec i) \wedge (l \prec i)) \vee ((l \prec i) \wedge ((k \prec i) \vee (l \prec i) \vee (\wedge_j t_j \prec i))) \vee (i \prec l)$.
The formula is always true. It is similar even when there exist two terms of type A. Thus, the condition is always satisfied.
Case4: The case when there exists neither a term of type A nor a term of type B. In this case, there exists no edge from x_i. Clearly, x_i does not affect the shelling variable order.

From the above discussion, we have only to consider Case 2 to decide the shelling variable order. Note that all the requirements for the shelling variable order have the form $* \prec i$. This means that x_i cannot be the first variable.

Next, we consider when a variable can be chosen as the first variable. When x_i is a cosimplicial node, for any edge (x_k, x_l), there exists either an edge (x_k, x_i) or (x_l, x_i). Then for any $I \in PI(f)$, $I \cup \{i\}$ is classified to Case 1 or Case 3. When there is no edge from x_i, it is classified to Case 4. Otherwise, there exists an edge $I = (x_k, x_l)$ such that there exists neither (x_k, x_i) nor (x_l, x_i). In this case, $I \cup \{i\}$ is classified to Case 2. Hence, x_i can be the first variable when either x_i is a cosimplicial node or there exists no edge from x_i.

When the first variable is fixed, some of the requirements for the variable order are satisfied. At last, we prove that the remaining requirements are equivalent to the requirements obtained by the subgraph induced by $V - \{x_i\}$ when x_i is the first variable. It is clear when there exists no edge from x_i. When x_i is a cosimplicial node, x_i affects the requirements if there exist nodes x_t, x_i such that $\{i,t\} \cup \{k\}$ is classified to Case 2. In this case, the requirements are written in the form $(i \prec *) \vee R$, where R is some requirement. Thus, all the requirements concerning x_i are removed. It is easy to check that no other requirements are removed.

As G is cotriangulated, the induced subgraph also has a cosimplicial node. Thus we can repeatedly choose an appropriate variable. □

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

Theorem 11 A tree-shellable quadratic function is ordered tree-shellable.

Proof Let T be the BDT which proves that a quadratic function f is tree-shellable. Let P_m be a 1-path which satisfy $i \prec_T m$ for any other 1-path P_i in T. We show that any varible order which is consisitent with the order of variables in P_m is a shelling variable order of f.

Let $P_{k_1}, ..., P_{k_t}$ be the paths that diverge from P_m at the node labeled by x_l. As $x_l \in P_{k_i}$ $(1 \leq i \leq t)$, after the 1-edge from the node labeled by x_l, P_{k_i} includes only one positive literal. 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. Therefore, it is possible to change the order of variables in these 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 its 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 shelalble functions. The notion of tree-shellability has also made it possible to characterize lexico-exchange and aligned functions in terms of tree-shellability.

References

[1] M. O. Ball and G. L. Nemhauser, "Matroids and a Reliability Analysis Problem", Math. Oper. Res., Vol.4, pp.132-143 (1979).

[2] M. O. Ball and J. S. Provan, "Disjoint Products and Efficient Computation of Reliability", Operations Research, Vol.36, No.5, pp.703-715 (1988).

[3] C. Benzaken, Y. Crama, P. Duchet, P. L. Hammer and F. Maffray, "More Characterizations of Triangulated Graphs", Journal of Graph Theory, Vol.14, No.4, pp.413-422 (1990).

[4] E. Boros, "Dualization of Aligned Boolean Function", RUTCOR Research Report 9-94 (1994).

[5] E. Boros, Y. Crama, O. Ekin, P. L. Hammer, T. Ibaraki and A. Kogan, "Boolean Normal Forms, Shellability and Reliability Computations", RUTCOR Resaerch Report 3-97 (1997).

[6] H. Brugesser and P. Mani, "Shellable Decompositions of cells and shperes", Math. Scand. Vol.29, pp.199-205 (1971).

[7] G. Danaraj and V. Klee, "Shellings of Sheres and Polytopes", Duke Math. J. Vol.41, pp.443-451 (1974).

[8] J. S. Provan and M. O. Ball, "Efficient Recognition of Matroids and 2-Monotonic Systems", in *Applications of Discrete Mathematics*, pp.122-134, R. Ringeisen and F. Roberts (eds.), SIAM, Philadelphia (1988).