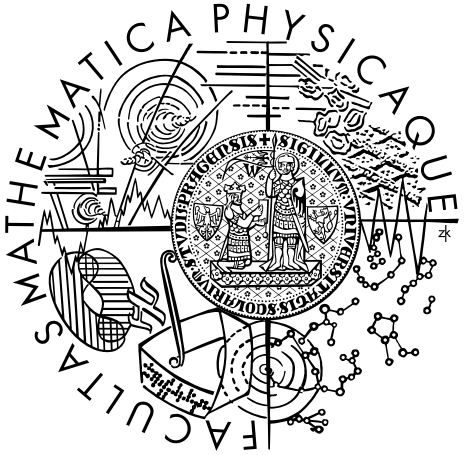
Make sure citations are typeset properly.

¿ Use a different citation style.

¿ Change reference names to adhere to reference guidelines at `http://en.wikibooks.org/wiki/LaTeX/Labels_and_Cross-referencing`.

Charles University in Prague
Faculty of Mathematics and Physics

MASTER THESIS



Filip Bártek

Minimum representations of Boolean functions defined by multiple intervals

Department of Theoretical Computer Science and Mathematical Logic

Supervisor of the master thesis: RNDr. Petr Kučera, Ph.D.

Study programme: Informatics

Study branch: Theoretical Computer Science

Prague 2015

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University in Prague has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date ..... signature

Title: Minimum representations of Boolean functions defined by multiple intervals

Author: Filip Bártek

Department: Department of Theoretical Computer Science and Mathematical Logic

Supervisor: RNDr. Petr Kučera, Ph.D., Department of Theoretical Computer Science and Mathematical Logic

Abstract:

When we interpret the input vector of a Boolean function as a binary number, we define interval Boolean function $f^n_{[a,b]}$ so that $f^n_{[a,b]}(x) = 1$ if and only if $a \leq x \leq b$. Disjunctive normal form is a common way of representing Boolean functions. Minimizing DNF representation of an interval Boolean function can be performed in linear time. [4] The natural generalization to k-interval functions seems to be significantly harder to tackle. In this thesis, I discuss the difficulties with finding an optimal solution and introduce a $2k$-approximation algorithm.

¿ Remove citation from abstract.

Keywords: Boolean minimization, disjunctive normal form, interval functions

Contents

Introduction

Boolean functions defined by intervals were introduced by Schieber et al. [4]. A pair of n-bit numbers a, b defines a 1-interval function $f^n_{[a,b]}$:

$$f^n_{[a,b]}(x) = 1 \iff a \leq x \leq b$$

Schieber et al. showed an efficient method to find minimum DNF representation of any 1-interval Boolean function given by a pair of endpoints.

Does Umans show this for term-wise minimization as well? Note that Čepek et al. uses Σ with CNF representations. Shouldn't one of the problems be Π_2-hard instead?

Since the problem of Boolean minimization is in general Σ^p_2-complete [5], there is an interesting question of how difficult it is to minimize DNF representation of a function defined by a set of intervals. Dubovský investigated the problem for 2-interval functions.

In this thesis we review the existing results regarding minimization of DNF representations of single- and multi-interval functions, and then present 3 new results:

- a simplified algorithm for minimizing DNF representations of 2-switch 2-interval functions (section 3.1),
- show that the technique used for proving the optimality of the algorithms that minimize DNF representation of 1- and 2-switch (that is 1-interval and certain class of 2-interval) functions can not be used in functions with 3 or more switches (section 3.2) and
- introduce an algorithm for k-interval functions which gives a solution which is at worst $2k$ times larger than an optimal one (chapter 4).

¿ Mention the "Better approximation is hard" result.

1. Definitions

The basic terminology used in this thesis was introduced by Schieber et al. [4], Dubovský [1] and Hušek [2, 3].

1.1 Vector operations

Throughout the thesis we will use vectors over small finite domains extensively. We shall write a vector as a sequence of symbols and vectors. For example, if v is a vector of length n, then $00v11$ denotes a vector of length $n+4$ whose leading two components have the value 0, trailing two components have the value 1 and i-th component for $i \in \{3, \ldots, n+2\}$ has the value of the $(i-2)$-th component of v.

Definition 1.1.1 (Component extraction $v^{[i]}$)**.** Let v be a vector of length n and $1 \leq i \leq n$. Then $v^{[i]}$ is the i-th component of v.

Definition 1.1.2 (Subsequence extraction $v^{[a,b]}$)**.** Let v be a vector of length n and $1 \leq a \leq b \leq n$. Then $v^{[a,b]}$ is the subvector of v that starts at a-th position and ends at b-th position ($v^{[a,b]} = v^{[a]}v^{[a+1]} \ldots v^{[b-1]}v^{[b]}$).

¿ Change ... to ⋯.

Notably, $v = v^{[1]} \ldots v^{[n]} = v^{[1,n]}$.

Definition 1.1.3 (Symbol repetition $\alpha^{\{n\}}$)**.** Let α be a symbol (for example 0 or 1) and $n \in \{0, 1, \ldots\}$. Then $\alpha^{\{n\}}$ is the vector of length n each component of which is equal to α.

For example $0^{\{2\}} = 00$.

1.2 Binary vectors

Definition 1.2.1 (Binary vector)**.** We will use the term *binary vector* to denote a vector over the Boolean domain. That is, x is a binary vector if and only if $x \in \{0,1\}^n$ for some $n \in \{0, 1, \ldots\}$. We call such x an *n-bit binary vector*.

Note that an n-bit binary vector x represents the natural number $\sum_{i=1}^{n} x^{[i]} 2^{n-i} = \sum_{i|x^{[i]}=1} 2^{n-i}$. An n-bit vector corresponds to a number between 0 and $2^n - 1$. We will not differentiate between a number and its binary vector representation. Specifically, we will use a lexicographic linear ordering on binary vectors – for n-bit vectors a and b, $a \leq b$ if and only if $a = b$ or $a^{[i]} = 0$ and $b^{[i]} = 1$ where i is the smallest index such that $a^{[i]} \neq b^{[i]}$. Clearly, this ordering corresponds to the natural ordering on the represented numbers.

Definition 1.2.2 (Most significant bit)**.** The *most significant bit (MSB)* of a binary vector x is its first bit, that is $x^{[1]}$.

The n *most significant bits* of a vector form its prefix of length n, that is $x^{[1,n]}$.

1.3 Boolean functions

Definition 1.3.1 (Boolean function)**.** $f : \{0,1\}^n \to \{0,1\}$ is an *n-ary Boolean function.*

Since we will not be dealing with any non-Boolean functions, in the remainder of the text, we will use the terms "function" and "Boolean function" interchangeably.

Definition 1.3.2 (True point)**.** Let f be an n-ary Boolean function. An n-bit binary vector x is a *true point* of f if and only if $f(x) = 1$.

Equivalently, we define a *false point* of f as any point x such that $f(x) = 0$.

We'll denote the set of all true points of f as $TP(f)$ and the set of all false points as $FP(f)$.

1.4 DNF representations of Boolean functions

Every n-ary Boolean function f can be expressed as a propositional formula $\mathcal{F}$ on n variables $x_1, \ldots, x_n$. There's a natural bijection between binary vectors of length n and valuations of n variables (1-bits in the binary vector correspond exactly to 1-valued variables in the valuation).

Definition 1.4.1 (Disjunctive normal form)**.** A *literal* is an occurrence of a variable (x_i) or its negation ($\overline{x_i}$) in a propositional formula.

A *term* is an elementary conjunction of literals, that is a conjunction in which every variable appears at most once.

A *disjunctive normal form (DNF)* propositional formula is a disjunction of terms.

We will often view terms as sets of literals and DNF formulas as sets of terms.

A DNF formula $\mathcal{F}$ is a *DNF representation* of a Boolean function f if and only if $(\forall x \in \{0,1\}^n)[f(x) = 1 \iff \mathcal{F}(x) \equiv 1]$, where $\mathcal{F}(x)$ is the formula given by substituting each occurrence of a variable x_i in $\mathcal{F}$ with the value $x^{[i]}$ for every $i \in \{1, \ldots, n\}$.

Definition 1.4.2 (Minimum DNF representation)**.** A DNF representation of function f is *minimum* if and only if there is no DNF representation of f with fewer terms.

We will denote the size of the minimum DNF representation of f by $dnf(f)$.

The focus of this thesis is minimizing the number of terms of DNF representations of certain Boolean functions.

Note that a function may have more than one minimum DNF representation.

1.5 Ternary vectors

Definition 1.5.1 (Ternary vector)**.** A *ternary vector* is a vector over the set $\{0, 1, \phi\}$.

There's a natural bijection between ternary vectors and terms (elementary conjunctions of literals). Each of the n components of a ternary vector T corresponds to an occurrence (or its absence) of one of the n variables of a term C:

¿ Wrap in table environment and add a caption.

$T^{[i]}$	$C \cap \{x_i, \overline{x_i}\}$
ϕ	$\emptyset$
1	$\{x_i\}$
0	$\{\overline{x_i}\}$

Note that since C is elementary, it can not contain both x_i and $\overline{x_i}$ for any i.

Namely, a *binary* vector (a ternary vector that does not contain any ϕ) corresponds to a full term (that is one that contains every variable), and the ternary vector $\phi^{\{n\}}$ corresponds to the empty term (tautology).

It's easy to see that a set of ternary vectors, each of length n, corresponds to a DNF *formula* on n variables.

Definition 1.5.2 (Spanning)**.** A ternary vector T of length n *spans* a binary vector x of length n if and only if $(\forall i \in \{1, \ldots, n\})[T^{[i]} = \phi \text{ or } T^{[i]} = x^{[i]}]$.

The i-th symbol of a ternary vector constrains the i-th symbol of the spanned binary vector (or, equivalently, the valuation of the i-th variable). If $T^{[i]}$ is a "fixed bit" (that is $T^{[i]} \in \{0, 1\}$), the spanned binary vector x must have the same value in its i-th position, that is $x^{[i]} = T^{[i]}$. If $T^{[i]}$ is the "don't care symbol" ϕ, the spanned binary vector may have any value in its i-th position.

Definition 1.5.3 (Spanned set)**.** Let T be a ternary vector of length n. We denote the set of points spanned by T as $span(T)$:

$$span(T) = \{x \in \{0, 1\}^n | T \text{ spans } x\}$$

Note that a ternary vector with m ϕ-positions spans 2^m binary vectors.

We'll also use a natural extension of spanning to sets of ternary vectors – if $\mathcal{T}$ is a set of ternary vectors, then

$$span(\mathcal{T}) = \bigcup_{T \in \mathcal{T}} span(T)$$

Definition 1.5.4 (Exact spanning)**.** A ternary vector T of length n *spans exactly* a set of n-bit binary vectors S if and only if $span(T) = S$.

T *spans exactly* an n-ary Boolean function f if and only if $span(T) = TP(f)$.

Again, we will generalize *exact spanning* to sets of ternary vectors – the set of ternary vectors $\mathcal{T}$ *spans exactly* S if and only if $span(\mathcal{T}) = S$.

Definition 1.5.5 (Spanning set)**.** Let f be an n-ary Boolean function. The set $\mathcal{T}$ of n-bit ternary vectors is a *spanning set* of f if and only if it $\mathcal{T}$ spans exactly f, that is $span(\mathcal{T}) = TP(f)$.

In other words, a spanning set of a function spans all of its true points and none of its false points.

Note that spanning sets of a function correspond to DNF representations of the function. The correspondence preserves size (number of ternary vectors and terms, respectively), so a minimum spanning set corresponds to a minimum DNF representation.

Definition 1.5.6 (Disjoint spanning set)**.** A spanning set $\mathcal{T}$ is *disjoint* if and only if the spanned sets of its vectors do not overlap, that is

$$(\forall T_i, T_j \in \mathcal{T})[T_i = T_j \text{ or } span(T_i) \cap span(T_j) = \emptyset]$$

In terms of DNF, disjoint spanning sets correspond to DNF formulas such that every valuation satisfies at most one term.

We have introduced two equivalent representations of Boolean functions – one based on ternary vectors and the other being the DNF representation. Table 1.1 shows the corresponding terms side by side.

ternary vector	term
ternary vector set	DNF formula
binary vector	variable valuation
ternary vector *spans* a binary vector	valuation *satisfies* a term
ternary vector set *spans* a binary vector	valuation *satisfies* a DNF formula

Table 1.1: Corresponding terms used in Boolean function representations

Definition 1.5.7 (Complement)**.** The complement of a binary symbol α ($\alpha \in \{0, 1\}$), denoted $\overline{\alpha}$, is $1 - \alpha$.

We get the complement of a binary vector x by flipping all of its bits, that is $\overline{x} = (\overline{x_1}, \ldots, \overline{x_n}) = 1^{\{n\}} - x$.

The complement of the ϕ symbol is ϕ ($\overline{\phi} = \phi$).

It follows that we obtain the complement of a ternary vector by flipping all of its fixed bits.

1.6 Orthogonal sets and coverable functions

Definition 1.6.1 (Orthogonality)**.** Let x, y be true points of Boolean function f ($f(x) = f(y) = 1$). x and y are *orthogonal* (with respect to f) if and only if every ternary vector that spans both x and y necessarily also spans a false point of f.

A set of true points S is *orthogonal* if and only if all the vectors in S are pairwise orthogonal.

This notion of orthogonality was introduced by Dubovský [1].

¿ Relate orthogonality to disjointness of essential sets. Don't forget to deal with DNF/CNF difference.

For a given f, we will call the size of its maximum orthogonal set $ortho(f)$.

Theorem 1.6.0.1 ($ortho(f) \leq dnf(f)$)**.** *For any Boolean function, the size of its maximum orthogonal set is at most as great as the size of its minimum DNF representation.*

Proof. We will prove the statement by contradiction. Let f be a Boolean function such that $ortho(f) > dnf(f)$. Let V be an orthogonal set of size $ortho(f)$ and $\mathcal{T}$ be any spanning set of size $dnf(f)$ (such spanning set exists because of the correspondence between DNF representations and spanning sets). Since $|\mathcal{T}| < |V|$ and $\mathcal{T}$ spans $TP(f) \supseteq S$, there must be a ternary vector $T \in \mathcal{T}$ that spans two distinct vectors $x, y \in V$. However, since V is orthogonal, x is orthogonal to y, so T necessarily spans a false point, which contradicts the premise that $\mathcal{T}$ is a feasible spanning set of f. □

We conclude that the size of any orthogonal set is a lower bound on the size of minimum DNF representation. Specifically, if we show a DNF representation of a function and an orthogonal set of the same size, we conclude that the representation is minimum. The orthogonal set certifies optimality of the DNF representation.

Definition 1.6.2 (Coverability)**.** Let f be a Boolean function. f is *coverable* if and only if $ortho(f) = dnf(f)$.

Informally speaking, a coverable function is one for which it may be easy to prove optimality of a spanning set. We will see in Chapters 2 and 3 that all 1-interval and a concise class of 2-interval functions are coverable. We will show examples of functions which are not coverable in section 3.2.

¿ Reword.

The concept of coverability was originally introduced by Čepek et al. [6]. They use a different yet equivalent definition. For brevity, we will omit proof of the equivalence and simply use Definition 1.6.2, which is sufficient for our purpose.

Is it actually equivalent? Is "equivalent" a correct word, esp. since Cepek's coverability is in CNF (dual)?

¿ Write the proof. May require many definitions.

1.7 k-interval Boolean functions

Definition 1.7.1 (k-interval Boolean function)**.** Let $a_1, b_1, \ldots, a_k, b_k$ be n-bit binary vectors such that $0^{\{n\}} \leq a_1 \leq b_1 < a_2 \leq \cdots < a_i \leq b_i < a_{i+1} \leq \cdots < a_k \leq b_k \leq 1^{\{n\}}$. Then $f^n_{[a_1,b_1],\ldots,[a_k,b_k]} : \{0,1\}^n \to \{0,1\}$ is a function defined as follows:

$$f^n_{[a_1,b_1],\ldots,[a_k,b_k]}(x) = \begin{cases} 1 & \text{if } a_i \leq x \leq b_i \text{ for some } i \\ 0 & \text{otherwise} \end{cases}$$

We call $f^n_{[a_1,b_1],\ldots,[a_k,b_k]}$ a *k-interval Boolean function* and the vectors $a_1, b_1, \ldots, a_k, b_k$ its *endpoints*.

Note that the inequalities ensure that the intervals $[a_i, b_i]$ are non-empty and don't intersect.

Definition 1.7.2 (Proper k-interval Boolean function)**.** Let $f^n_{[a_1,b_1],\ldots,[a_k,b_k]}$ be a k-interval Boolean function. If the endpoints satisfy the inequalities $b_1 < a_2 - 1$, $\ldots$, $b_i < a_{i+1} - 1$, $\ldots$, $b_{k-1} < a_k - 1$ (in other words, adjacent intervals are separated by at least one false point), we call $f^n_{[a_1,b_1],\ldots,[a_k,b_k]}$ a *proper k-interval Boolean function*.

Especially, $f^n_{[a,b]}$ is a function whose true points form the interval $[a, b]$.

¿ Show that complementing bits preserves number of switches.

Table 1.2 shows examples of interval functions.

Interval function	Propositional formula	Description
$f^n_{[1\{n\},1\{n\}]}$	$\bigwedge_i x_i$	Conjunction
$f^n_{[0\{n-1\}1,1\{n\}]}$	$\bigvee_i x_i$	Disjunction
$f^n_{[0\{n\},1\{n\}]}$	1 (an empty term)	Tautology
$f^n_{[0\{n-1\}1,1\{n-1\}0]}$	$\overline{x_1 \dots x_n} \wedge \overline{\overline{x_1} \dots \overline{x_n}} \equiv \bigvee_{i \neq j} x_i \overline{x_j}$	Non-equivalence
$f^n_{[0\{n\},0\{n\}],[1\{n\},1\{n\}]}$	$x_1 \dots x_n \vee \overline{x_1} \dots \overline{x_n}$	Equivalence

Table 1.2: Examples of interval functions

1.8 l-switch Boolean functions

When we discuss 2-interval functions in chapter 3, we will find another distinction of Boolean functions useful. It is based on the concept of switch as introduced by Hušek [2, 3].[1]

Definition 1.8.1 (l-switch function)**.** A Boolean function f is *l-switch* if and only if there are exactly l input vectors x such that $f(x) \neq f(x+1)$.

We call such vectors *switches* of f.

[1] Hušek first used the Czech term "zlom" in his thesis [2, p. 13] and later translated the term as "switch" in personal communication [3].

2. 1-interval functions

In this chapter, we will show an efficient way to compute a minimum spanning set of any 1-interval Boolean function defined by the interval endpoints. The algorithm was originally shown by Schieber et al. [4].

In the chapters that follow, we will use this algorithm as a procedure in order to span multi-interval functions.

In the remainder of this chapter, a and b ($a \leq b$) will denote the endpoints of the interval in question and n the number of input bits; we'll be looking for a minimum representation of function $f^n_{[a,b]}$.

Input n-bit numbers a, b such that $a \leq b$

Output A set of ternary vectors

The algorithm differentiates various cases of input values and solves some of the cases by recursive calls.

Kučera

Buď rozvést nebo odstranit.

Bártek

Rozvedeno; stačí takhle?

2.1 Trivial cases

We will first deal with the trivial intervals.

2.1.1 $a = b$

We span the interval $[a, a]$ with the single ternary vector a.

From now on, let $a < b$.

2.1.2 $a = 0^{\{n\}}$ and $b = 1^{\{n\}}$

We span the interval $[0^{\{n\}}, 1^{\{n\}}]$ with the single ternary vector $\phi^{\{n\}}$, which corresponds to an empty term.

From now on, let $a > 0^{\{n\}}$ or $b < 1^{\{n\}}$.

2.2 Prefix and suffix case

Since we have dealt with the trivial cases, we are now left with the situation $0^{\{n\}} \leq a < b \leq 1^{\{n\}}$ and $a > 0^{\{n\}}$ or $b < 1^{\{n\}}$.

In this section we shall consider the prefix case, that is $[0^{\{n\}}, b]$ ($a = 0^{\{n\}}$), and the suffix case, that is $[a, 1^{\{n\}}]$ ($b = 1^{\{n\}}$).

2.2.1 Suffix case

¿ Typeset "1-1" properly (correct dash etc.).

The prefix and suffix cases are complementary. There is a 1-1 correspondence between the spanning sets of suffix intervals and the spanning sets of prefix intervals. The correspondence spans the suffix interval $[a, 1^{\{n\}}]$ by complementing each of the vectors in the spanning set of the prefix interval $[0^{\{n\}}, \overline{a}]$:

$$\mathcal{T} \text{ spans } [a, 1^{\{n\}}] \iff \{\overline{T} | T \in \mathcal{T}\} \text{ spans } [0^{\{n\}}, \overline{a}]$$

¿ Prove correctness.

Since the correspondence preserves size of the spanning sets, it also preserves optimality. Thus we can use the correspondence to optimally span suffix intervals using a procedure that spans prefix intervals. We will show such procedure in the following section.

2.2.2 Prefix case

Let's proceed to span a prefix interval. Let $a = 0^{\{n\}}$ and $b < 1^{\{n\}}$. Let c be the n-bit number $b+1$. Since $b < 1^{\{n\}}$, we do not need more than n bits to encode c.

The algorithm produces one ternary vector for each bit that is set to 1 in c. If o is a position of a 1 in c ($c^{[o]} = 1$), then the corresponding ternary vector is $c^{[1,o-1]}0\phi^{\{n-o\}}$. Thus we get the following spanning set:

$$\mathcal{T} = \{c^{[1,o-1]}0\phi^{\{n-o\}} | c^{[o]} = 1\}$$

Theorem 2.2.2.1 (Feasibility)**.** *$\mathcal{T}$ spans exactly the interval $[0^{\{n\}}, b]$.*

Proof. To see that every number spanned by $\mathcal{T}$ is in $[0^{\{n\}}, b]$, note that given an index o of a bit which is set to 1 in c, the biggest number spanned by $c^{[1,o-1]}0\phi^{\{n-o\}}$ is $c^{[1,o-1]}01^{\{n-o\}}$ which is still strictly smaller than c.

On the other hand, consider a number x smaller than c and let o be the most significant bit in which x and c differ. It follows that $x^{[1,o-1]} = c^{[1,o-1]}$ and $0 = x^{[o]} < c^{[o]} = 1$. Then $c^{[1,o-1]}0\phi^{\{n-o\}} \in \mathcal{T}$ spans x. □

Theorem 2.2.2.2 (Optimality)**.** *$\mathcal{T}$ is the minimum spanning set of $f^n_{[0^{\{n\}},b]}$.*

Proof. We will construct an orthogonal set V of size $|\mathcal{T}|$. With the aid of Theorem 1.6.0.1, this proves that $\mathcal{T}$ is a minimum spanning set.

Similarly to the spanning vectors, the orthogonal true points correspond to 1-bits of c:

$$V = \{c^{[1,o-1]}0c^{[o+1,n]} | c^{[o]} = 1\}$$

¿ Show in detail that vectors created from different 1 bits in c must differ.

Clearly $|V| = |\mathcal{T}|$. Also note that all points in V are smaller than c, so they are true points.

We need to prove that V is orthogonal. Let $x, y \in V$, $x \neq y$. Let o_x and o_y be the positions of the symbol 1 in c that were used to construct x and y ($x = c^{[1,o_x-1]}0c^{[o_x+1,n]}$, $y = c^{[1,o_y-1]}0c^{[o_y+1,n]}$). Since $x \neq y$, necessarily $o_x \neq o_y$.

Let T be any ternary vector that spans both x and y. To see that T must also span the false point c, note that every component i of c matches the respective component in at least one of x, y (for every i, $c^{[i]} = x^{[i]}$ or $c^{[i]} = y^{[i]}$).

Table 2.1 shows a more detailed comparison of the corresponding subvectors of x, y and c in case $o_x < o_y$.

We conclude that V is orthogonal and since $|\mathcal{T}| = |V|$, $\mathcal{T}$ is a minimum spanning set by Theorem 1.6.0.1. □

It is easy to see that the spanning set $\mathcal{T}$ produced by the algorithm is disjoint. Since it is minimum in general, it is a minimum disjoint spanning set too.

	$[1, o_x - 1]$	o_x	$[o_x + 1, o_y - 1]$	o_y	$[o_y + 1, n]$
x	$\underline{c^{[1,o_x-1]}}$	0	$\underline{c^{[o_x+1,o_y-1]}}$	$\underline{c^{[o_y]}}$	$\underline{c^{[o_y+1,n]}}$
y	$\underline{c^{[1,o_x-1]}}$	$\underline{c^{[o_x]}}$	$\underline{c^{[o_x+1,o_y-1]}}$	0	$\underline{c^{[o_y+1,n]}}$
c	$c^{[1,o_x-1]}$	$c^{[o_x]}$	$c^{[o_x+1,o_y-1]}$	$c^{[o_y]}$	$c^{[o_y+1,n]}$

Table 2.1: Subvectors of x, y and c in case $o_x < o_y$. The underlined subvectors of x and y match their counterparts in c.

2.3 General case

Having solved the trivial and prefix and suffix cases, we are left with the situation $0^{\{n\}} < a < b < 1^{\{n\}}$.

If a and b have the same MSB, we recursively span $[a^{[2,n]}, b^{[2,n]}]$ and prepend the common MSB to the solution. This clearly preserves both feasibility and optimality.

Let us now consider the case when $a^{[1]} \neq b^{[1]}$. Since $a \leq b$, necessarily $a^{[1]} = 0$ and $b^{[1]} = 1$. This restriction leaves us with four possible combinations of pairs of leading bits of a and b:

1. $a^{[1,2]} = 01$, $b^{[1,2]} = 10$
2. $a^{[1,2]} = 00$, $b^{[1,2]} = 10$
3. $a^{[1,2]} = 01$, $b^{[1,2]} = 11$
4. $a^{[1,2]} = 00$, $b^{[1,2]} = 11$

Following Schieber et al. [4], we will deal with each of these cases separately.

Schieber et al.'s proofs of feasibility and optimality of the following algorithm are rather technical and out of scope of this text. Note, however, that the proof of optimality is implicitly based on building an orthogonal set of the same size as the computed spanning set.

We will denote $a^{[3,n]}$ with $\hat{a}$ ($a = a^{[1,2]}\hat{a}$) and $b^{[3,n]}$ with $\hat{b}$ ($b = b^{[1,2]}\hat{b}$).

2.3.1 $a^{[1,2]} = 01$ and $b^{[1,2]} = 10$

In this case we span the two sub-intervals $[01\hat{a}, 011^{\{n-2\}}]$ and $[100^{\{n-2\}}, 10\hat{b}]$ separately.

The endpoints of the sub-interval $[01\hat{a}, 011^{\{n-2\}}]$ share the leading bit 0. This means that this sub-interval is immediately reduced to the suffix instance $[1\hat{a}, 11^{\{n-2\}}]$, which can be spanned using the suffix algorithm show in section 2.2.

The remaining sub-interval $[100^{\{n-2\}}, 10\hat{b}]$ is reduced to a prefix interval $[00^{\{n-2\}}, 0\hat{b}]$ in a similar fashion.

2.3.2 $a^{[1,2]} = 00$ and $b^{[1,2]} = 10$

In this case, we divide the interval into three sub-intervals:

- $[00\hat{a}, 001^{\{n-2\}}]$
- $[010^{\{n-2\}}, 011^{\{n-2\}}]$

- $[100^{\{n-2\}}, 10\hat{b}]$

The sub-interval $[010^{\{n-2\}}, 011^{\{n-2\}}]$ is spanned by the single ternary vector $01\phi^{\{n-2\}}$.

The sub-intervals $[00\hat{a}, 001^{\{n-2\}}]$ and $[100^{\{n-2\}}, 10\hat{b}]$ are spanned together as follows:

1. Recursively solve the $(n-1)$-bit instance $[0\hat{a}, 1\hat{b}] = [a^{[1]}a^{[3,n]}, b^{[1]}b^{[3,n]}]$
2. Insert a 0-bit in the second position of the vectors from the resulting spanning set

Note that $a^{[2]} = b^{[2]} = 0$, so the n-bit vectors produced this way exactly span the union of the intervals $[00\hat{a}, 001^{\{n-2\}}]$ and $[100^{\{n-2\}}, 10\hat{b}]$.

2.3.3 $a^{[1,2]} = 01$ and $b^{[1,2]} = 11$

This case is complementary to case 2.3.2. As in suffix case, note that flipping all the bits in the endpoints transforms this case to case 2.3.2 and flipping the fixed bits in the resulting spanning set preserves correctness.

Does it?

2.3.4 $a^{[1,2]} = 00$ and $b^{[1,2]} = 11$

Let j be maximal such that $a^{[1,j]} = 0^{\{j\}}$ and $b^{[1,j]} = 1^{\{j\}}$. Since $a > 0^{\{n\}}$ (or $b < 1^{\{n\}}$), necessarily $j < n$.

The number j gives us three sub-intervals to span:

- $[a, 0^{\{j\}}1^{\{n-j\}}]$
- $[0^{\{j-1\}}10^{\{n-j\}}, 1^{\{j-1\}}01^{\{n-j\}}]$
- $[1^{\{j\}}0^{\{n-j\}}, b]$

Note that the second sub-interval contains exactly the numbers that don't start with either j zeros or j ones, and as such can be spanned by j ternary vectors. We construct $T_1, \ldots, T_j$ that span the second sub-interval by appending $\phi^{\{n-j\}}$ to each of the j cyclic shifts of $01\phi^{\{j-2\}}$. Thus, $T_1 = 01\phi^{\{j-2\}}\phi^{\{n-j\}}$, …, $T_i = \phi^{\{i-1\}}01\phi^{\{j-1-i\}}\phi^{\{n-j\}}$ (for $i \in \{1, \ldots, j-1\}$), …, $T_{j-1} = \phi^{\{j-2\}}01\phi^{\{n-j\}}$, $T_j = 1\phi^{\{j-2\}}0\phi^{\{n-j\}}$.

The other two sub-intervals are spanned recursively. Let $a'' = a^{[j,n]}$ and $b'' = b^{[j,n]}$. Note that $a''^{[1]} = 0$ and $b''^{[1]} = 1$. Note that $|a''| = |b''| = n - j + 1$. Since $j \geq 2$, $n - j + 1 < n$. Let $\mathcal{T}''$ be the spanning set of $[a'', b'']$ computed recursively.

The spanning set of $[a, b]$ will be computed from the vectors $T_1, \ldots, T_j$ and $\mathcal{T}''$ based on the relation between $(n-j)$-bit suffixes of a and b. Let $a' = a^{[j+1,n]}$ and $b' = b^{[j+1,n]}$.

$b' < a' - 1$

In this case, the spanning set of $[a, b]$ consists of the vectors $T_1, \ldots, T_j$ and vectors obtained by prepending each vector from $\mathcal{T}''$ with $\phi^{\{j-1\}}$.

$b' \leq a' - 1$

In this case, the spanning set of $[a, b]$ consists of the vectors $T_1, \ldots, T_{j-1}$ (omitting T_j) and a set $\mathcal{T}'$ which is derived from $\mathcal{T}''$ in the following way:

$$\begin{aligned}\mathcal{T}' = &\{\phi^{\{j-1\}}T | T \in \mathcal{T}'' \text{ and } T^{[1]} \in \{0, \phi\}\} \\ &\cup \{1\phi^{\{j-1\}}T^{[2,n-j+1]} | T \in \mathcal{T}'' \text{ and } T^{[1]} = 1\}\end{aligned}$$

3. 2-interval functions

In this chapter we will revise the results specific to 2-interval functions. Dubovský [1] provides more detailed overview.

Reword "seem to require".

Dubovský identified several classes of 2-interval functions [1, p. 5] which seem to require different approaches to spanning.It turns out that it is useful to consider the number of switches to differentiate between these classes. Recall that an l-switch function has exactly l input vectors x such that $f(x) \neq f(x+1)$.

It is easy to see that prefix and suffix interval functions are 1-switch.

Depending on $f(0)$, an l-switch Boolean function f is proper $\lceil \frac{l}{2} \rceil$-interval in case $f(0) = 0$, or proper $\lceil \frac{l+1}{2} \rceil$-interval in case $f(0) = 1$.

Equivalently, a proper k-interval n-ary function is:

- $(2k)$-switch in case $f(0^{\{n\}}) = f(1^{\{n\}}) = 0$
- $(2k-1)$-switch in case $f(0^{\{n\}}) \neq f(1^{\{n\}})$
- $(2k-2)$-switch in case $f(0^{\{n\}}) = f(1^{\{n\}}) = 1$

Namely, a 2-switch Boolean function such that $f(0) = 0$ is 1-interval, and a 2-switch function such that $f(0) = 1$ is 2-interval.

3.1 2-interval 2-switch functions

Let f be a 2-interval 2-switch Boolean function (necessarily $f(0^{\{n\}}) = f(1^{\{n\}}) = 1$).[1] We will show that f can be optimally spanned by reduction to a 2-switch 1-interval Boolean function f' of at most the same arity, which can be spanned using the algorithm introduced in chapter 2.

Note that Dubovský introduced an optimization algorithm for this class of functions [1, section 3.2], following an argument similar to the one used by Schieber et al. to span 1-interval functions [4]. However, the reduction approach gives an interesting insight in the problem and appears easier to describe.

From the definition of f we know that it is 2-interval with the outer interval endpoints being $0^{\{n\}}$ and $1^{\{n\}}$:

$$f = f^n_{[0^{\{n\}}, b_1], [a_2, 1^{\{n\}}]}$$

Let's call the inner interval endpoints b_1 and a_2. $0^{\{n\}} \leq b_1$, $b_1 < a_2 - 1$ (f is proper 2-interval – otherwise it would be a tautology and thus trivial to span), $a_2 \leq 1^{\{n\}}$.

3.1.1 $b_1^{[1]} = a_2^{[1]}$

In case b_1 and a_2 have the same MSBs, we recursively solve the $(n-1)$-bit case $[0^{\{n-1\}}, b_1^{[2,n]}], [a_2^{[2,n]}, 1^{\{n-1\}}]$, and add an extra ternary vector to span the remainder of the true points.

[1] This definition corresponds to class A_0 in Dubovský's classification [1, p. 5].

Without loss of generality, let $b_1^{[1]} = a_2^{[1]} = 0$. The complementary case is symmetric.

Let $b'_1 = b_1^{[2,n]}$ and $a'_2 = a_2^{[2,n]}$. Let $\mathcal{T}'_0$ be an optimal spanning set of $f' = f^{n-1}_{[0^{\{n-1\}},b'_1],[a'_2,1^{\{n-1\}}]}$. Note that f' is of the same type (that is 2-interval 2-switch) and has a lower arity, so we can span it by recursion. Let $\mathcal{T}_0$ be the n-bit set we get by prepending the symbol 0 to each vector from $\mathcal{T}'_0$ ($\mathcal{T}_0 = \{0T | T \in \mathcal{T}'_0\}$). Let $\mathcal{T}_1 = \{1\phi^{\{n-1\}}\}$. Let $\mathcal{T} = \mathcal{T}_0 \cup \mathcal{T}_1$. We claim that $\mathcal{T}$ spans exactly $f^n_{[0^{\{n\}},b_1],[a_2,1^{\{n\}}]}$ and that $\mathcal{T}$ is minimum such.

Theorem 3.1.1.1. *$\mathcal{T}$ spans exactly $f^n_{[0^{\{n\}},b_1],[a_2,1^{\{n\}}]}$ (feasibility).*

Proof.

1. $span(\mathcal{T}) \subseteq TP(f)$

   Let $c \in span(T)$, $T \in \mathcal{T}$. We need to show that $c \leq b_1$ or $c \geq a_2$.

   (a) $c^{[1]} = 1$: Since $a_2^{[1]} = 0$, necessarily $c \geq a_2$.

   (b) $c^{[1]} = 0$: Necessarily $T \in \mathcal{T}_0$. Let $T' = T^{[2,n]}$ and $c' = c^{[2,n]}$. Since T spans c, T' spans c'. By construction of $\mathcal{T}_0$ necessarily $T' \in \mathcal{T}'_0$, so T' does not span any number in $[b'_1 + 1, a'_2 - 1]$. Thus $c' \leq b'_1$ or $c' \geq a'_2$. Observe that prepending a 0 bit to c', b'_1 and a'_2 preserves the inequalities:

   $$c' \leq b'_1 \text{ or } c' \geq a'_2 \Rightarrow 0c' \leq 0b'_1 \text{ or } 0c' \geq 0a'_2$$

   Since $0c' = c$, $0a' = a$ and $0b' = b$, we get $c \leq b_1$ or $c \geq a_2$.

2. $TP(f) \subseteq span(\mathcal{T})$

   Let $c \leq b_1$ or $c \geq a_2$. We need a $T \in \mathcal{T}$ that spans c.

   (a) $c^{[1]} = 1$: $1\phi^{n-1} \in \mathcal{T}_1 \subseteq \mathcal{T}$ spans c.

   (b) $c^{[1]} = 0$: Similarly to b'_1 and a'_2, let $c' = c^{[2,n]}$. Since b_1, a_2 and c all start with a 0 bit, $c' \leq b'_1$ or $c' \geq a'_2$. Since $\mathcal{T}'_0$ spans $f^{n-1}_{[0^{\{n-1\}},b'_1],[a'_2,1^{\{n-1\}}]}$, there is $T \in \mathcal{T}'_0$ that spans c'. Prepending the symbol 0 to both T and c' preserves spanning, so $0T \in \mathcal{T}_0 \subseteq \mathcal{T}$ spans c.

□

Theorem 3.1.1.2. *$\mathcal{T}$ is a minimum spanning set of $f^n_{[0^{\{n\}},b_1],[a_2,1^{\{n\}}]}$ (optimality).*

Proof. Let $\mathcal{T}_{opt}$ be an optimal spanning set of $f^n_{[0^{\{n\}},b_1],[a_2,1^{\{n\}}]}$. We'll show that $|\mathcal{T}_{opt}| \geq |\mathcal{T}|$.

First recall that $\mathcal{T}'_0$ is an optimal spanning set of $f^{n-1}_{[0^{\{n-1\}},b'_1],[a'_2,1^{\{n-1\}}]}$ and $|\mathcal{T}| = |\mathcal{T}'_0| + 1$. Thus we need to show $|\mathcal{T}_{opt}| \geq |\mathcal{T}'_0| + 1$

$\mathcal{T}_{opt}$ must span the true point $\bar{b}_1 = 1(b_1 + 1)^{[2,n]}$ (since a_2 starts with a 0, all numbers that start with a 1 must be true points). We don't need more than n bits for $\bar{b}_1$ because $b_1 < a_2 \leq 01^{\{n-1\}}$. Let $T_{\bar{b}_1} \in \mathcal{T}_{opt}$ be a ternary vector that spans $\bar{b}_1$. If $T_{\bar{b}_1}$ spanned any vector that starts with a 0, then $T_{\bar{b}_1}$ would also span

the vector $b_1 + 1$. However, since f is *proper* 2-interval, $b_1 + 1$ is a false point ($b_1 < b_1 + 1 < a_2$). So $T_{\bar{b}_1}$ can't span any vector that starts with a 0.

To get a contradiction, let's suppose that $|\mathcal{T}_{opt}| \leq |\mathcal{T}'_0|$. Since we need a dedicated vector $T_{\bar{b}_1} \in \mathcal{T}_{opt}$ to span $\bar{b}_1$, there are at most $|\mathcal{T}_{opt}| - 1 \leq |\mathcal{T}'_0| - 1$ vectors in $\mathcal{T}_{opt}$ that span a number that starts with 0 (so these vectors start with 0 or ϕ). Removing the first symbol of these vectors, we get a spanning set of $f^{n-1}_{[0^{\{n-1\}},b'_1],[a'_2,1^{\{n-1\}}]}$ of size at most $|\mathcal{T}'_0| - 1$, which contradicts $\mathcal{T}'_0$'s optimality.

We conclude that we need at least $|\mathcal{T}'_0| + 1$ vectors to span $f^n_{[0^{\{n\}},b_1],[a_2,1^{\{n\}}]}$, proving the lower bound of $|\mathcal{T}_{opt}|$ and the optimality of $\mathcal{T}$. □

¿ Note that the proof could be rephrased using orthogonal sets explicitly.

We have shown that in case $b_1^{[1]} = a_2^{[1]}$, our algorithm produces a feasible and optimal spanning set.

3.1.2 $b_1^{[1]} \neq a_2^{[1]}$

Let $b_1^{[1]} \neq a_2^{[1]}$. Since $b_1 \leq a_2$, necessarily $b_1^{[1]} = 0$ and $a_2^{[1]} = 1$.

By flipping the leading bit in each of the endpoints b_1 and a_2, we will reduce this case to an instance of spanning a single true point interval on n bits.

Let $b'_1 = 1b_1^{[2,n]}$ and $a'_2 = 0a_2^{[2,n]}$ (flipping the leading bits of b_1 and a_2). Note that $a'_2 \leq b'_1$.

Let $\mathcal{T}'$ be a minimum spanning set of $f^n_{[a'_2,b'_1]}$. This set can be computed efficiently using the method shown in chapter 2.

Let's flip all the non-ϕ leading symbols of vectors in $\mathcal{T}'$, forming $\mathcal{T}$. Note that $|\mathcal{T}| = |\mathcal{T}'|$ and also that all the vectors in $\mathcal{T}'$ and $\mathcal{T}$ have the length n.

We claim that $\mathcal{T}$ is a minimum spanning set of $f^n_{[0^{\{n\}},b_1],[a_2,1^{\{n\}}]}$.

The argument is based on a pair of lemmas.

Lemma 3.1.2.1. *Let c be an n-bit binary vector and a, b be $(n-1)$-bit binary vectors. Let $c' = \overline{c^{[1]}}c^{[2,n]}$ (flipping the leading bit of c). Then ($c \leq 0b$ or $c \geq 1a$) if and only if ($c' \geq 0a$ and $c' \leq 1b$).*

Proof.

1. $c^{[1]} = 0$:

   Let $c = 0c^{[2,n]}$ and $c' = 1c^{[2,n]}$.

   Since $c^{[1]} = 0$, $c \not\geq 1a$. Thus ($c \leq 0b$ or $c \geq 1a$) if and only if $c \leq 0b$.

   Since $c^{[1]} = 0$, $c \leq 0b$ if and only if $c^{[2,n]} \leq b$.

   Since $c'^{[1]} = 1$ and $c'^{[2,n]} = c^{[2,n]}$, $c^{[2,n]} \leq b$ if and only if $c' \leq 1b$.

   From $c'^{[1]} = 1$ we immediately get $c' \geq 0a$.

   Chaining the equivalences together:

$$\begin{aligned} c \leq 0b \text{ or } c \geq 1a &\iff c \leq 0b \\ &\iff c^{[2,n]} \leq b \\ &\iff c' \leq 1b \\ &\iff c' \leq 1b \text{ and } c' \geq 0a \end{aligned}$$

2. $c^{[1]} = 1$:

   This case is symmetric to the previous one.

□

Lemma 3.1.2.2. *Let $\mathcal{T}$ be a set of ternary vectors and $\mathcal{T}' = \{\overline{T^{[1]}}T^{[2,n]} | T \in \mathcal{T}\}$. Let a, b be any $(n-1)$-bit binary vectors. Then $\mathcal{T}$ spans $f^n_{[0^{\{n\}},0b],[1a,1^{\{n\}}]}$ if and only if $\mathcal{T}'$ spans $f^n_{[0a,1b]}$.*[2]

Proof.

1. If $\mathcal{T}$ spans $f^n_{[0^{\{n\}},0b],[1a,1^{\{n\}}]}$, then $\mathcal{T}'$ spans $f^n_{[0a,1b]}$:

   Let $\mathcal{T}$ span $f^n_{[0^{\{n\}},0b],[1a,1^{\{n\}}]}$.

   Let c' be an n-bit binary vector. We need to show that $\mathcal{T}'$ spans c' if and only if $c' \geq 0a$ and $c' \leq 1b$.

   Let $c = \overline{c'^{[1]}}c'^{[2,n]}$. By definition of $\mathcal{T}'$, $\mathcal{T}'$ spans c' if and only if $\mathcal{T}$ spans c.

   Since $\mathcal{T}$ spans $f^n_{[0^{\{n\}},0b],[1a,1^{\{n\}}]}$, $\mathcal{T}$ spans c if and only if $c \leq 0b$ or $c \geq 1a$.

   From Lemma 3.1.2.1 we get ($c \leq 0b$ or $c \geq 1a$) if and only if ($c' \geq 0a$ and $c' \leq 1b$).

   Chaining the equivalences together:

$$\begin{aligned} \mathcal{T}' \text{ spans } c' &\iff \mathcal{T} \text{ spans } c \\ &\iff c \leq 0b \text{ or } c \geq 1a \\ &\iff c' \geq 0a \text{ and } c' \leq 1b \\ &\iff f^n_{[0a,1b]}(c') = 1 \end{aligned}$$

2. If $\mathcal{T}'$ spans $f^n_{[0a,1b]}$, then $\mathcal{T}$ spans $f^n_{[0^{\{n\}},0b],[1a,1^{\{n\}}]}$:

   Let $\mathcal{T}'$ span $f^n_{[0a,1b]}$.

   Let c be an n-bit binary vector. We need to show that $\mathcal{T}$ spans c if and only if $c \leq 0b$ or $c \geq 1a$.

   Let $c' = \overline{c^{[1]}}c^{[2,n]}$. By definition of $\mathcal{T}'$, $\mathcal{T}$ spans c if and only if $\mathcal{T}'$ spans c'.

   Since $\mathcal{T}'$ spans $f^n_{[0a,1b]}$, $\mathcal{T}'$ spans c' if and only if $c' \geq 0a$ and $c' \leq 1b$.

   Using Lemma 3.1.2.1, we get ($c' \geq 0a$ and $c' \leq 1b$) if and only if ($c \leq 0b$ or $c \geq 1a$).

   Chaining the equivalences together:

$$\begin{aligned} \mathcal{T} \text{ spans } c &\iff \mathcal{T}' \text{ spans } c' \\ &\iff c' \geq 0a \text{ and } c' \leq 1b \\ &\iff c \leq 0b \text{ or } c \geq 1a \\ &\iff f^n_{[0^{\{n\}},0b],[1a,1^{\{n\}}]}(c) = 1 \end{aligned}$$

□

[2]Note that the statement holds even in the single case that the function is not proper 2-interval, that is $a = 1^{\{n-1\}}$ and $b = 0^{\{n-1\}}$.

Lemma 3.1.2.2 shows a size preserving bijection between the spanning sets of functions of form $f^n_{[0^{\{n\}},0b],[1a,1^{\{n\}}]}$ and $f^n_{[0a,1b]}$ for general a and b. Since b_1 starts with a 0 ($b_1 = 0b$), a_2 starts with a 1 ($a_2 = 1a$) and the set $\mathcal{T}$ was constructed from an optimal spanning set $\mathcal{T}'$ of $f^n_{[0a_2^{[2,n]},1b_1^{[2,n]}]}$ using the bijection shown in 3.1.2.2, that is flipping the leading symbols of all the ternary vectors, both feasibility and optimality of $\mathcal{T}$ with respect to $f^n_{[0^{\{n\}},b_1],[a_2,1^{\{n\}}]}$ follow from the feasibility and optimality of $\mathcal{T}'$ with respect to $f^n_{[0a_2^{[2,n]},1b_1^{[2,n]}]}$.

3.2 Functions with 3 and more switches

The situation changes in 3-switch functions. In 1-switch (prefix and suffix) and 2-switch (1-interval and 2-interval with extreme outer endpoints) functions, we managed to efficiently find minimum spanning sets. The proofs of optimality of the spanning sets depend on construction of sufficiently large orthogonal sets.[3] Doing so, we depended on *coverability* of the functions in question.

Dubovský has identified a 3-switch function whose maximum orthogonal set is strictly smaller than its minimum spanning set. The function is $f^4_{[0,4],[9,14]}$. The size of its maximum orthogonal set is 4 and the size of its minimum spanning set is 5.[1, p. 32] Dubovský proved the bounds on sizes of orthogonal sets and spanning sets by exhaustion using software tools.

This result shows that 3-switch functions are not coverable in general.

We will generalize the result for functions with larger number of switches.

Lemma 3.2.0.3. *If there is an l-switch function that is not coverable, then there is also an $(l+1)$-switch function that is not coverable.*

Proof. Let f be an n-ary l-switch function that is not coverable.

1. $f(1^{\{n\}}) = 1$:

   Let f' be an $(n+1)$-ary function defined in the following way:

   $$f'(x') = \begin{cases} f(x'^{[2,n+1]}) & \text{if } x'^{[1]} = 0 \\ 0 & \text{if } x'^{[1]} = 1 \end{cases}$$

   Since f is l-switch, there are exactly l vectors x of length n ($x < 1^{\{n\}}$) such that $f(x) \neq f(x+1)$. Prepending a 0 to each of these vectors, we get l vectors x' of length $n+1$ such that $f'(x') \neq f'(x'+1)$. The $(l+1)$-st "switch" vector of f' is $01^{\{n\}}$, since $f'(01^{\{n\}}) = 1$ and $f'(10^{\{n\}}) = 0$. Thus we have shown that f' is $(l+1)$-switch.

   Prove that f' doesn't have more switch vectors.

   It is easy to see that there is a size preserving 1-1 correspondence between the spanning sets of f and f'. The same holds for orthogonal sets. This proves that if f is not coverable, neither is f'.

   ¿ Elaborate.

2. $f(1^{\{n\}}) = 0$:

   Let f' be an $(n+1)$-ary function defined in the following way:

[3] While we have only shown an orthogonal set explicitly in the case of prefix interval (subsection 2.2.2), the other cases use orthogonal sets implicitly, both in [4] and section 3.1.

$$f'(x') = \begin{cases} f(x'^{[2,n+1]}) & \text{if } x'^{[1]} = 0 \\ 0 & \text{if } x' \in [10^{\{n\}}, 1^{\{n\}}0] \\ 1 & \text{if } x' = 1^{\{n+1\}} \end{cases}$$

Note that with Kucera on 15 April 2015, we found it necessary for f to also have the false point $1^{\{n-1\}}0$ for the procedure to work. I believe that the proof also works in the other case as well.

Again, the l switches of f translate to l switches of f' by prepending a 0. The $(l+1)$-st switch is $1^{\{n\}}0$ in this case.

$dnf(f') \geq dnf(f) + 1$ We need a special ternary vector to span the true point $1^{\{n+1\}}$ in f'. If a ternary vector spanned both $1^{\{n+1\}}$ and $0x$, it would necessarily also span the false point $01^{\{n\}}$.

If we could span f' with less than $dnf(f) + 1$ vectors, we could span f with less than $dnf(f)$ vectors, since given a spanning set of f', leaving out the vector that spans the true point $1^{\{n+1\}}$ and removing the leading symbol of the rest (necessarily a 0) gives a spanning set of f.

$ortho(f') \leq ortho(f) + 1$ To get a contradiction, let V' be an orthogonal set of f' of size $ortho(f) + 2$. Let $V = \{v'^{[2,n+1]} | v' \in V' \text{ and } v'^{[1]} = 0\}$. Since V' only consists of true points of f', there can be at most one vector in V' that starts with a 1. It follows that $|V| \geq |V'| - 1 = ortho(f) + 1$. V is an orthogonal set of f – if $0u$ and $0v$ are orthogonal in f', every ternary vector that spans them must span a false point $0x$. Leaving out the leading symbol preserves the relation, and thus orthogonality as well. We have shown an orthogonal set of f of size at least $ortho(f) + 1$, which contradicts the premise that $ortho(f)$ is the size of a maximum orthogonal set of f.

Putting the inequalities together:

$$ortho(f') \leq ortho(f) + 1 < dnf(f) + 1 = dnf(f')$$

We conclude that f' is an $(l+1)$-switch non-coverable function.

□

We conclude with the following theorem:

Theorem 3.2.0.1. *For any $l \geq 3$, there is an l-switch non-coverable function. Moreover, the arity of such function can be as small as $l + 1$.*

This result indicates that minimizing the DNF representation of functions that have at least 3 switches requires a different approach for proving optimality.

Review.

¿ Also differentiate the functions based on their values in the points $0^{\{n\}}$ and $1^{\{n\}}$ and show we can find an $(l+1)$-switch function for all the relevant cases.

4. $2k$-approximation algorithm for k-interval functions

Is "relatively" a good word to use?

In this chapter, we will show an algorithm that computes a relatively small spanning set of any k-interval Boolean function. The input intervals are represented by pairs of endpoints (n-bit numbers). An approximation ratio of $2k$ will be proved.

Check whether it really is straightforward.

The algorithm is a straightforward extension of Schieber et al.'s suffix-prefix approximation algorithm for disjoint spanning sets of 1-interval functions [4, section 6].

¿ Rephrase here or in Conclusion (almost identical sentence).

We will only consider *proper* k-interval Boolean functions, that is those whose adjacent intervals are separated by at least one false point (see definition 1.7.2). A k-interval Boolean function which is not proper k-interval can be efficiently reduced to a proper l-interval where $l < k$ by joining the adjoint intervals.

4.1 Description

Is "adjoint" a good word for this quality?

Input Numbers $a_1, b_1, \ldots, a_k, b_k$ that satisfy the inequalities $0^{\{n\}} \leq a_1$, $a_1 \leq b_1$, $b_1 < a_2 - 1$, $\ldots$, $a_i \leq b_i$, $b_i < a_{i+1} - 1$ (for $i \in \{1, \ldots, k-1\}$), $\ldots$, $b_{k-1} < a_k - 1$, $a_k \leq b_k$, $b_k \leq 1^{\{n\}}$.

Such numbers are endpoints of intervals of a proper k-interval Boolean function.

Output A set of ternary vectors.

Procedure The algorithm spans each of the intervals $[a_i, b_i]$ separately. The set that spans $[a_i, b_i]$ will be denoted $\mathcal{T}_i$, and the output set will be denoted $\mathcal{T} = \Cup_{i=1}^{k} \mathcal{T}_i$.

Typeset union as Bigcup.

Let's consider the interval $[a_i, b_i]$. Let c be the longest common prefix of a_i and b_i. Let j be its length ($|c| = j$).

¿ Index c and j with i.

If $j = n$, that is $c = a_i = b_i$, $\mathcal{T}_i$ consists of the single vector $c = a_i = b_i$.

If $j = n-1$, that is $c = a_i^{[1,n-1]} = b_i^{[1,n-1]}$, $a_i^{[n]} = 0$ and $b_i^{[n]} = 1$, $\mathcal{T}_i$ consists of the single vector $c\phi$.

We are left with the situation $j < n-1$.

Note that $a_i^{[j+1]} = 0$ and $b_i^{[j+1]} = 1$, since $a_i \leq b_i$ and $a_i \neq b_i$. Now let $a' = a_i^{[j+2,n]}$ and $b' = b_i^{[j+2,n]}$ ($a_i = c0a'$ and $b_i = c1b'$). Optimally span the suffix interval $[a', 1^{\{n-j-1\}}]$ and the prefix interval $[0^{\{n-j-1\}}, b']$ using the (linear time) algorithm introduced in section 2.2 and producing spanning sets $\mathcal{T}'_{i,0}$ and $\mathcal{T}'_{i,1}$ respectively. Now simply prepend $a_i^{[1,j+1]} = c0$ to all vectors in $\mathcal{T}'_{i,0}$ and $b_i^{[1,j+1]} = c1$ to all vectors in $\mathcal{T}'_{i,1}$, producing $\mathcal{T}_{i,0}$ and $\mathcal{T}_{i,1}$ respectively. The spanning set of $[a_i, b_i]$ is then $\mathcal{T}_i = \mathcal{T}_{i,0} \cup \mathcal{T}_{i,1}$.

¿ Leave out "(linear time)".

¿ Treat more efficiently the cases $a_i^{[j+1,n]} = 0^{\{n-j\}}$ (prefix) and $b_i^{[j+1,n]} = 1^{\{n-j\}}$ (suffix), especially $a_1 = 0^{\{n\}}$ and $b_k = 1^{\{n\}}$. This doesn't improve the approximation ratio though and may complicate the proof.

4.2 Feasibility

Theorem 4.2.0.2. *The algorithm spans exactly* $f^n_{[a_1,b_1],\ldots,[a_k,b_k]}$.

Proof. We need to show that for each i, $\mathcal{T}_i$ spans exactly $[a_i, b_i]$. This is obvious in cases $j = n$ and $j = n-1$.

In the remaining case $j < n-1$, observe that $0\mathcal{T}'_{i,0} = \{0T | T \in \mathcal{T}'_{i,0}\}$ spans exactly the interval $[0a', 01^{\{n-j-1\}}]$ and $1\mathcal{T}'_{i,1}$ spans exactly the interval $[10^{\{n-j-1\}}, 1b']$. Since $10^{\{n-j-1\}} = 01^{\{n-j-1\}} + 1$, the union of these sets spans exactly the interval $[0a', 1b'] = [a_i^{[j+1,n]}, b_i^{[j+1,n]}]$. Prepending the common prefix c to $0\mathcal{T}'_{i,0} \cup 1\mathcal{T}'_{i,1}$ preserves the spanning to $[a_i, b_i]$ and matches the operation performed in the algorithm to produce $\mathcal{T}_i$.

The union of such $\mathcal{T}_i$s clearly spans the union of the intervals. □

4.3 Approximation ratio

Theorem 4.3.0.3. *Let* $\mathcal{T}_{opt}$ *be an optimal spanning set of* $f^n_{[a_1,b_1],\ldots,[a_k,b_k]}$ *and let* $\mathcal{T}$ *be the spanning set returned by the algorithm. We claim that:*

$$|\mathcal{T}| \leq 2k|\mathcal{T}_{opt}|$$

Proof. To prove the statement, we will show an orthogonal set V of the function $f^n_{[a_1,b_1],\ldots,[a_k,b_k]}$ of size $|V| \geq \frac{|\mathcal{T}|}{2k}$. By Theorem 1.6.0.1, the existence of such orthogonal set gives a lower bound $|V|$ on $|\mathcal{T}_{opt}|$.

First, let's consider the case that $|\mathcal{T}_i| = 1$ for every i. Note that this can only happen if $a_i^{[1,n-1]} = b_i^{[1,n-1]}$ for every i. In this case, the orthogonal set V consists of the single vector a_1 (in fact, we could use any true point as every set of size 1 is trivially orthogonal). Since $|\mathcal{T}_i| = 1$ for every $i \in \{1, \ldots, k\}$, $|\mathcal{T}| = k$. It is now easy to see that $|V| = 1 \geq \frac{1}{2} = \frac{k}{2k} = \frac{|\mathcal{T}|}{2k}$.

We are now left with the case that $|\mathcal{T}_i| > 1$ for some i. Note that this can only happen if the corresponding $j < n-1$.

¿ Index j with i.

Let $\mathcal{T}_{i,\alpha}$ be the largest of the partial spanning sets of this type produced during the course of the algorithm. Note that the corresponding $\mathcal{T}_i = \mathcal{T}_{i,\alpha} \cup \mathcal{T}_{i,\overline{\alpha}}$ need not be the largest of the interval spanning sets in case $\mathcal{T}_{i,\overline{\alpha}}$ is small.

¿ Clarify.

Without loss of generality, let $\alpha = 1$. The set $\mathcal{T}'_{i,1}$ that exactly spans $[0^{\{n-j-1\}}, b' = b_i^{[j+2,n]}]$ was in this case obtained using the prefix algorithm introduced in subsection 2.2.2. In the proof of optimality of this algorithm (Theorem 2.2.2.2), we showed an orthogonal set that matches the size of the spanning set produced by the algorithm. Let $V'_{i,1}$ be this orthogonal set of $[0^{\{n-j-1\}}, b']$ ($|V'_{i,1}| = |\mathcal{T}'_{i,1}|$). Recall that the orthogonality of points in $V'_{i,1}$ only depends on the single false point $b' + 1$.

¿ Discuss soundness.

Let $V = \{c1v | v \in V'_{i,1}\}$. We claim that V is an orthogonal set of $f^n_{[a_1,b_1],\ldots,[a_k,b_k]}$.

What if $b_i = 1^{\{n\}}$?

By prepending $c1$ to $(b'+1)$ we get b_i+1. Since $b_i < a_{i+1}-1$, b_i+1 is a false point of $f^n_{[a_1,b_1],\ldots,[a_k,b_k]}$. It follows that the points in V are orthogonal, since their orthogonality only depends on the false point $c1(b'+1) = b_i+1$. From Theorem 1.6.0.1 we get $|\mathcal{T}_{opt}| \geq |V|$.

Clarify.

Since $\mathcal{T}_{i,\alpha}$ is the largest of the at most $2k$ spanning sub-sets of this type and the spanning sets of the other types are singular and less numerous, necessarily $|T| \leq 2k|\mathcal{T}_{i,\alpha}| = 2k|V| \leq 2k|\mathcal{T}_{opt}|$. □

Note that the spanning set returned by the algorithm is disjoint. This makes the algorithm $2k$-approximation for the disjoint case as well:

Theorem 4.3.0.4. *Let $\mathcal{T}_{disjoint}$ be an optimal* disjoint *spanning set of $f^n_{[a_1,b_1],\ldots,[a_k,b_k]}$ and let $\mathcal{T}$ be the (disjoint) spanning set returned by our algorithm. We claim that:*

$$|\mathcal{T}| \leq 2k|\mathcal{T}_{disjoint}|$$

Proof. Since every disjoint spanning set is a spanning set in general, $|\mathcal{T}_{opt}| \leq |\mathcal{T}_{disjoint}|$, where $\mathcal{T}_{opt}$ is an optimal (not necessarily disjoint) spanning set of $f^n_{[a_1,b_1],\ldots,[a_k,b_k]}$.

Using Theorem 4.3.0.3 we get:

$$|\mathcal{T}| \leq 2k|\mathcal{T}_{opt}| \leq 2k|\mathcal{T}_{disjoint}|$$

□

Theorem 4.3.0.5. *The approximation ratio of $2k$ is tight.*

Proof. For every k that is a power of 2 we will show a k-interval function such that $|\mathcal{T}| = 2k|\mathcal{T}_{opt}|$ (following the notation from Theorem 4.3.0.3).

Let $k = 2^{n_k}$. Let P be all the n_k-bit numbers, that is $P = \{0,1\}^{n_k}$. Note that $|P| = 2^{n_k} = k$. Let $n = n_k + 3$.

For each $p \in P$, we define the interval $[a_p, b_p]$ by appending certain 3-bit suffixes to p:

- $a_p = p000$
- $b_p = p011$

The first appended bit (0 for a_p and 0 for b_p) ensures that there is at least one false point between any pair of intervals defined this way (all the points that have a 1 in their (n_k+1)-st position are false points).

The remaining pair of appended bits (00 for a_p and 11 for b_p) ensures that $a_p^{[1,n-1]} \neq b_p^{[1,n-1]}$, so the algorithm will span the sub-intervals $[a_p = p000, p001]$ and $[p010, b_p = p011]$ separately, producing at least 2 vectors to span $[a_p, b_p]$. The configuration also enables the interval $[a_p, b_p]$ to be spanned by the single ternary vector $p0\phi^{\{2\}}$.

¿ Rephrase.

Thus we have k intervals none of which intersect or are adjoint .

Since $P = \{0,1\}^{n_k}$, we can span all the intervals by the single ternary vector $\phi^{\{n_k\}}0\phi^{\{2\}}$. Clearly $|\mathcal{T}_{opt}| = 1$.

However, the approximation algorithm uses $2k$ vectors to span the intervals, since it spans each of the intervals separately and uses the two vectors $p00\phi, p01\phi$ for each interval.

We get $|\mathcal{T}| = 2k = 2k|\mathcal{T}_{opt}|$.

Note that the optimal spanning set is trivially disjoint, so the ratio is tight in disjoint case as well. □

5. Better approximation is hard

¿ Leave the chapter out, since the result is not very interesting and the proof is quite complicated. Or give just a sketch of the proof.

¿ Use a better chapter title.

Finish the chapter.

A simple improvement can be performed on the $2k$-approximation algorithm introduced in the previous chapter by spanning each of the intervals $[a_i, b_i]$ optimally using Schieber et al.'s algorithm. Dubovský has shown that such approach actually improves the approximation ratio to 2 (as opposed to 4) in 2-interval functions. However, we will show in this chapter that the improvement diminishes for larger numbers of intervals.

Put the preceding introduction paragraph in context of the chapter.

Dubovský has shown a 2-approximation algorithm for spanning general 2-interval functions (including 3-switch and 4-switch).[1, p. 33] The algorithm simply spans each of the two intervals separately optimally and then returns the union of these partial spanning sets. A natural generalization of this algorithm to k-interval functions spans each of the k intervals separately optimally. It's easy to see that this algorithm performs at least as well as the $2k$-approximation algorithm shown in chapter 4. We'll show, however, that for a large k, the approximation ratio of the improved algorithm converges to $2k$.

Consider rewording.

In order to do this, we will construct a class of "hard" multi-interval Boolean functions.

5.1 Approximation-hard functions

Let $l \geq 1$ denote the length of the interval prefix. Then $k = 2^{l-1} + 1$ is the number of intervals of the multi-interval function we construct. Let $n \geq l + 2$ be the target function arity. We will construct a proper k-interval n-ary function that consists of 2 short intervals and $2^{l-1} - 1$ long intervals.

The short intervals reside in the subtrees associated with the prefixes $0^{\{l\}}$ and $1^{\{l\}}$:

- $[a_1, b_1] = [0^{\{l\}}\ 0^{\{n-l-1\}}1, 0^{\{l\}}\ 1^{\{n-l-1\}}0]$
- $[a_k, b_k] = [1^{\{l\}}\ 0^{\{n-l-1\}}1, 1^{\{l\}}\ 1^{\{n-l-1\}}0]$

The long intervals reside in the remaining subtrees and each of them intersects two of them:

- $[a_2, b_2] = [0^{\{l-2\}}01\ 0^{\{n-l-1\}}1, 0^{\{l-2\}}10\ 1^{\{n-l-1\}}0]$
- $[a_3, b_3] = [0^{\{l-3\}}011\ 0^{\{n-l-1\}}1, 0^{\{l-3\}}100\ 1^{\{n-l-1\}}0]$

$\vdots$

-

Typeset better; esp. don't show the bullet here.

- $[a_{k-1}, b_{k-1}] = [1^{\{l-2\}}01\ 0^{\{n-l-1\}}1, 1^{\{l-2\}}10\ 1^{\{n-l-1\}}0]$

In other words, for $i \in \{2, \ldots, k-1 = 2^{l-1}\}$, the prefix of the endpoint a_i, denoted p_i, is the l-bit number $2(i-2)+1$ (note that p_is exhaust all the odd numbers between $1 = 0^{\{l-2\}}\ 01$ and $2^l - 3 = 1^{\{l-2\}}\ 01$). Using p_i we get the endpoints of the i-th interval:

- $a_i = (p_i\ 0^{\{n-l-1\}}1)$
- $b_i = ((p_i + 1)\ 1^{\{n-l-1\}}0)$

Note that a_i for every i ends with the suffix $0^{\{n-l-1\}}1$ ($a_i^{[l+1,n]} = 0^{\{n-l-1\}}1$) and b_i always ends with the suffix $1^{\{n-l-1\}}0$. The l-bit prefixes of a_i and b_i are either equal in case $i \in \{1, k\}$, or successive numbers in case $i \in \{2, \ldots, k-1\}$.

Really?

We claim that $f = f^n_{[a_1,b_1],\ldots,[a_k,b_k]}$ can be spanned with as few as n ternary vectors, while the approximation algorithm produces a spanning set of size [?].

Insert the number here.

Lemma 5.1.0.4 ($dnf(f) \leq n$)**.** *The function $f^n_{[a_1,b_1],\ldots,[a_k,b_k]}$ defined above can be spanned with as few as n vectors.*

Proof. First observe that f has a limited number of false points, as each pair of successive intervals is separated by just two false points and there are just two false points which do not lie between the intervals (they are the two extreme outer points $0^{\{n\}}$ and $1^{\{n\}}$). There are $2k$ false points in total, each of them being either $a_i - 1$ or $b_i + 1$ for some i.

To span f, we first span the intervals $[j\ 0^{\{n-l-1\}}1, j\ 1^{\{n-l-1\}}0]$ for every $j \in \{0,1\}^l$. The set $\mathcal{T}_{in} = \{\phi^l\ s | s \text{ is a cyclic shift of } 01\phi^{n-l-2}\}$ exactly spans the union of the intervals $[j\ 0^{\{n-l-1\}}1, j\ 1^{\{n-l-1\}}0]$. To see that it actually does, observe that for every j, we want to span all the vectors with prefix j except the two vectors $(j\ 0^{\{n-l\}})$ and $(j\ 1^{\{n-l\}})$. The cyclic shifts of $01\phi^{n-l-2}$ span exactly all the numbers with the exception of $0^{\{n-l\}}$ and $1^{\{n-l\}}$, since every number that contains both a 0 and a 1 has some position i such that $x^{[i]} = 0$ and $x^{[(i+1) \mod (n-l)]} = 1$. The size of $\mathcal{T}_{in}$ is clearly $n - l$.

We are left with two true points to span in each of the long intervals. There are $k - 2 = 2^{l-1} - 1$ long intervals. Spanning each of the remaining true points separately, we get $\mathcal{T}_{out}$ of size $2(k-2) = 2(2^{l-1} - 1) = 2^l - 2$.

Use a better algorithm – one that produces merely $n + l - 2$ vectors.

□

l	$k = 2^{l-1} + 1$	$\lvert\mathcal{T}_{approx}\rvert = 2^l(n-l)$	$\lvert\mathcal{T}\rvert = n + l - 2$	$\lim_{n\to\infty} \frac{\lvert\mathcal{T}_{approx}\rvert}{\lvert\mathcal{T}\rvert}$
1	2	$2n - 4$	$n - 1$	2
2	3	$4n - 8$	n	4
3	5	$8n - 24$	$n + 1$	8
4	9	$16n - 64$	$n + 2$	16

$$\lim_{n\to\infty} \frac{|\mathcal{T}_{approx}|}{|\mathcal{T}|} = 2^l$$

$$\lim_{l\to\infty} \frac{\lim_{n\to\infty} \frac{|\mathcal{T}_{approx}|}{|\mathcal{T}|}}{k} = 2$$

Finish.

Conclusion

We have shown that with respect to finding a minimum DNF representation, 2-switch 2-interval functions can be reduced to 1-interval functions.

Then we showed that for every $l \geq 3$, there is an l-switch function that is not coverable, extending Dubovský's result about 3-switch functions. This suggests that possible future algorithms that optimally span multi-interval functions are going to require a different approach for proving optimality.

To remedy the difficulty with spanning multi-interval functions, we introduced a $2k$-approximation algorithm for minimizing DNF representations of k-interval functions. The algorithm naturally extends Schieber et al.'s suffix-prefix approximation algorithm for disjoint spanning sets of 1-interval functions [4, section 6]. Our algorithm produces disjoint spanning sets and has the approximation ratio of $2k$ for the problem of finding a minimum disjoint representation as well.

Some interesting questions are still left to be answered regarding multi-interval Boolean functions. Since general Boolean minimization is Σ_2^p-hard [5], there must be some k such that minimization of k-interval functions is Σ_2^p-hard, and a k' for which minimization is NP-hard. Schieber et al. showed that such $k' > 1$, which is the only bound shown so far. There is a substantial difference between 1- and 2-interval functions since the former are coverable in general, while the latter are not. This prevents us from using a similar approach to minimizing 2-interval functions. I suspect this difference could correspond to a difference in computational complexity of the problem.

The $2k$-approximation algorithm presented in this thesis is constructed so that we can use the orthogonal sets for proving the approximation ratio. There may be a better approximation algorithm for k-interval functions and searching for it could provide an interesting insight in proving lower bounds for minimum DNF representation size of the functions in question.

Really? Probably yes; note that we can translate a k-interval function in a spanning set of size $2kn$.

Is it a good idea to "suspect" in a thesis?

¿ Suspect more explicitly, e.g. 3-switch functions are NP-hard.

We may have shown a better one in `betterapprox`.

Bibliography

[1] Jakub Dubovský. A construction of minimum DNF representations of 2-interval functions. Master's thesis, Charles University in Prague, 2012. URL `https://is.cuni.cz/webapps/zzp/detail/116613/`.

[2] Radek Hušek. Properties of interval boolean functions. Master's thesis, Charles University in Prague, 2014. URL `https://is.cuni.cz/webapps/zzp/detail/145114/`. [In Czech].

[3] Radek Hušek. personal communication, April 2015.

[4] Baruch Schieber, Daniel Geist, and Ayal Zaks. Computing the minimum DNF representation of boolean functions defined by intervals. *Discrete Applied Mathematics*, 149(1–3):154 – 173, 2005. ISSN 0166-218X. doi: http://dx.doi.org/10.1016/j.dam.2004.08.009. URL `http://www.sciencedirect.com/science/article/pii/S0166218X05000752`. Boolean and Pseudo-Boolean Functions.

[5] Christopher Umans. The minimum equivalent DNF problem and shortest implicants. In *Foundations of Computer Science, 1998. Proceedings. 39th Annual Symposium on*, pages 556–563, Nov 1998. doi: 10.1109/SFCS.1998.743506. URL `http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=743506`.

[6] Ondřej Čepek, Petr Kučera, and Petr Savický. Boolean functions with a simple certificate for CNF complexity. *Discrete Applied Mathematics*, 160(4–5):365 – 382, 2012. ISSN 0166-218X. doi: http://dx.doi.org/10.1016/j.dam.2011.05.013. URL `http://www.sciencedirect.com/science/article/pii/S0166218X11001958`.

Capitalize "Boolean" in bibliography properly.

List of Tables

List of Abbreviations

DNF disjunctive normal form. ii, 1, 3, 5–8, 20, 27

MSB most significant bit. 4, 12, 15

Todo list