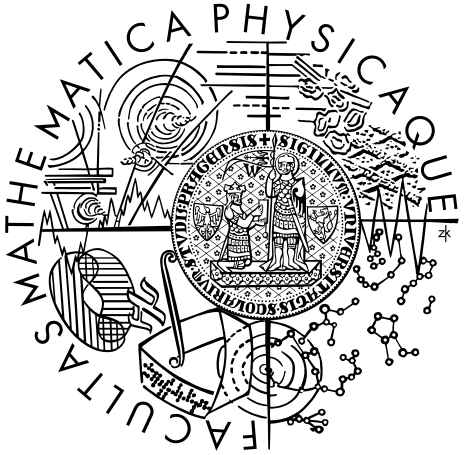
Charles University in Prague
Faculty of Mathematics and Physics

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.

Keywords: Boolean minimization, disjunctive normal form, interval functions

Contents

Introduction

Write a proper introduction.

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.

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 ($v^{[a,b]} = (v^{[a]}, v^{[a+1]}, \ldots, v^{[b-1]}, v^{[b]})$).

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

Definition 1.1.3 (Concatenation uv)**.** Let u, v be vectors of lengths m, n respectively. Then $uv = (u_1, \ldots, u_m, v_1, \ldots, v_n)$ is the concatenation of the vectors u, v.

Note that we will use symbols and vectors of length 1 interchangeably, for example $v = v^{[1]}v^{[2]} \ldots v^{[n]}$.

Definition 1.1.4 (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.

Mention arithmetic operations on binary vectors and numbers.

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 propositional logic formula $\mathcal{F}$ is in *disjunctive normal form (DNF)* if and only if $\mathcal{F}$ is a disjunction of terms, where *terms* are elementary[1] conjunctions of literals.

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 it expresses the same function, that is $(\forall x \in \{0,1\}^n)[f(x) = 1 \iff \mathcal{F}(x) = 1]$.

Clarify what $\mathcal{F}(x)$ means?

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.

Rewrite; should be more natural or use the bijection explicitly.

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:

[1] In an elementary conjunction of literals, no variable appears in both polarities. A trivial example of a non-elementary conjunction is $x_1\overline{x_1}$.

$T^{[i]}$	$C \cap \{x_i, \overline{x_i}\}$
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.

Is there a better, more established term?

We have introduced two equivalent representations of Boolean functions – one based on ternary vectors and the other being the DNF representation. Table 1.5 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 $1 - \alpha$.

Kučera

Pokud neřeknete, jak jsou reprezentována záporná čísla, tak tento zápis nedává smysl.

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 Interval Boolean functions

Definition 1.6.1 (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 1.6.2 (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 $[a, b]$.

Define l-switch function?

Table 1.6 shows examples of interval functions.

Typeset the table more pretty?

Interval function	Propositional formula	Description
$f^2_{[11,11]}$	$x_1 \wedge x_2$	Conjunction
$f^2_{[01,11]}$	$x_1 \vee x_2$	Disjunction
$f^n_{[0^{\{n\}},1^{\{n\}}]}$	1	Tautology
$f^n_{[0^{\{n-1\}}1,1^{\{n-1\}}0]}$	$\overline{x_1 \dots x_n} \wedge \overline{\overline{x_1} \dots \overline{x_n}}$	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

2. 1-interval functions

In this chapter, we will show and 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

Kučera

Buď rozvést nebo odstranit.

The algorithm is recursive in input length.

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 $[a, 1^{\{n\}}]$ ($b = 1^{\{n\}}$).

Note that prefix and suffix cases are complementary. We may transform a suffix instance $[a, 1^{\{n\}}]$ to a prefix instance $[0^{\{n\}}, \overline{a}]$ (where $\overline{a}$ is the complementary vector of a) and flip the polarity of the resulting ternary vectors in all their components. Since the instances $[a, 1^{\{n\}}]$ and $[0^{\{n\}}, \overline{a}]$ are symmetric in general, using an optimal algorithm to span the prefix instance $[0^{\{n\}}, \overline{a}]$ ensures we get an optimal spanning set of the original suffix instance as well.

Let's proceed to span a prefix instance. 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 2.2.0.1 (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 2.2.0.2 (Optimality). *$\mathcal{T}$ is the minimum spanning set of $f^n_{[0^{\{n\}},b]}$.*

Define orthogonality in chapter Definitions.

Relate to essential sets.

Refer to a statement that demonstrates properties of orthogonal sets.

Explain in detail.

Proof. We'll construct a set V of $|\mathcal{T}|$ true points no pair of which can be spanned by a single ternary vector. Dubovský[1] calls such sets "orthogonal". Doing so, we'll show a lower bound $|\mathcal{T}|$ on the size of feasible solutions, proving optimality of $\mathcal{T}$.

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

$$V = \{c^{[1,o-1]}0c^{[o+1,n]} | c^{[o]} = 1\} \tag{2.1}$$

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

It's easy to see that any ternary vector that spans two different points in V must also span the false point c, so it can't be a part of the solution.

If there was a feasible spanning set of size smaller than $|V|$, at least one of its vectors would need to span at least two points in V. As we have shown, such ternary vector would necessarily also span the false point c, leading to contradiction with the set's feasibility (Theorem 2.2.0.1). □

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 MSB to the solution.

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 MSBs:

Avoid using MSBs.

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

Kučera
pokud chcete mít v citacích jméno, tak použijte odpovídající BiBTeXový styl (autoři rok)

Following Schieber et al.,[4] we'll deal with each of these cases separately. The proofs of feasibility and optimality of the following algorithm are rather technical and they are out of scope of this text. Please refer to the original article for the proofs. Note that the proof of optimality is implicitly based on showing orthogonal sets.

Consider spelling the leading bits of a and b explicitly, for example $00a'$ in place of a.

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

In this case, we'll span the two sub-intervals $[a, 01^{\{n-1\}}]$ and $[10^{\{n-1\}}, b]$ separately. Note that after leaving out the initial shared bit, they are prefix and suffix interval, respectively, so the algorithm from section 2.2 can be used to span each of the sub-intervals.

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

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

- $[a, 001^{\{n-2\}}]$
- $[010^{\{n-2\}}, 011^{\{n-2\}}]$
- $[100^{\{n-2\}}, b]$

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

The subintervals $[a, 001^{\{n-2\}}]$ and $[100^{\{n-2\}}, b]$ are spanned together as follows:

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

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 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.

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 $a > 0^{\{n\}}$ (and $b < 1^{\{n\}}$), necessarily $j < n$.

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's thesis [1] provides more detailed information.

Dubovský identified several classes of 2-interval functions [1, p. 5] which seem to require different approaches for spanning. I find Hušek's concept of "switch" ("zlom" in Czech) [2, 3] useful to differentiate between these classes.

Reword "seem to require".

Definition 3.0.1 (l-switch function)**.** A Boolean function f is *l-switch* if and only if there are exactly l input vectors x such that $f(x) \neq f(x+1)$.

Move definition to Definitions?

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 $\lceil \frac{l}{2} \rceil$-interval in case $f(0) = 0$, or $\lceil \frac{l+1}{2} \rceil$-interval in case $f(0) = 1$.

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

3.1 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 seems 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.

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

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

$\mathcal{T}_1 = \{1\phi^{\{n-1\}}\}$. 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 $\bar{b}_1 = 1(b_1 + 1)^{[2,n]}$. 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 $\mathcal{T}$. □

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.

[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 $b = 0^{\{n-1\}}$.

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:

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

□

Consider rewording, e.g. "size-preserving".

Consider rewording, e.g. "one-to-one correspondence" or "correspondence".

Lemma 3.1.2.2 shows a size-conservative 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 3-switch functions

Rephrase.

Draw connection to "essential sets".

Cite Dubovsky.

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 have implicitly used so-called orthogonal sets for proving optimality of the spanning sets produced by our algorithms.

However, 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]

Try to generalize for larger number of switches.

This result indicates that minimizing the DNF representation of 3-switch functions requires a different approach for proving optimality.

4. $2k$-approximation algorithm for minimizing DNF representation of k-interval Boolean functions

In this chapter, an algorithm will be shown that computes a 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.

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 1.6.2). 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 touching adjacent intervals.

4.1 Algorithm

4.1.1 Description

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

Output A set of ternary vectors

Procedure The algorithm goes through all the intervals $[a_i, b_i]$. For each i, the longest common prefix of a_i and b_i is computed. Let j be its length. Note that $a^{[j+1]} = 0$ and $b^{[j+1]} = 1$. Now let $a'' = a_i^{[j+2,n]}$ and $b'' = b_i^{[j+2,n]}$. 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. Prepend $a_i^{[1,j+1]}$ and $b_i^{[1,j+1]}$ to the respective ternary vectors and add them to the output spanning set.

Treat (explicitly) the degenerate cases: $a_1 = 0$, $b_k = 2^n - 1$, $j = n$, $j = n - 1$.

4.1.2 Feasibility

Theorem 4.1.2.1. *The algorithm spans exactly* $f^n_{[a_1,b_1],\ldots,[a_k,b_k]}$.

Proof. This is easy to see from the fact that the intervals $[a_i, b_i]$ form a partition of the true point set and that each of them is spanned exactly by the suffix and prefix procedure. □

Go into more detail.

4.1.3 Approximation ratio

Theorem 4.1.3.1. *Let $\mathcal{T}_{opt}$ be an optimal spanning set of $f^n_{[a_1,b_1],\ldots,[a_k,b_k]}$ and let $\mathcal{T}_{approx}$ be the spanning set returned by the algorithm. We claim that:*

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

Proof. Let $\mathcal{T}_x$ be the largest (n-bit) spanning set of a "suffix" or "prefix" subinterval added in the algorithm. Without loss of generality, let the respective subinterval be "prefix" $[b_i^{[1,j+1]}0^{\{n-j-1\}}, b_i]$. From Section 2.2 we know that there is an orthogonal set of $[b_i^{[1,j+1]}0^{\{n-j-1\}}, b_i]$ of size $|\mathcal{T}_x|$, and moreover that its orthogonality only depends on the false point $b_i + 1$. Note, however, that $b_i + 1$ is also a false point in $f^n_{[a_1,b_1],\ldots,[a_k,b_k]}$ since $b_i < a_{i+1} - 1$. Thus we obtain an orthogonal set of size $|\mathcal{T}_x|$ for the k-interval function, limiting the size of its optimal spanning set $|\mathcal{T}_{opt}| \geq |\mathcal{T}_x|$.

Clarify: the interval is prefix or suffix on $n - j - 1$ bits, but not in general on n bits.

Since $\mathcal{T}_x$ is the largest of the $2k$ partial sets used to span the function in the approximation algorithm, we know that $|\mathcal{T}_{approx}| \leq 2k|\mathcal{T}_x|$.

Reference a more general statement.

Joining the inequalities together we conclude:

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

□

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

Theorem 4.1.3.2. *Let $\mathcal{T}_{disjoint}$ be an optimal* disjoint *spanning set of $f^n_{[a_1,b_1],\ldots,[a_k,b_k]}$ and let $\mathcal{T}_{approx}$ be the (disjoint) spanning set returned by our algorithm. We claim that:*

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

Proof. Since every disjoint spanning set is a spanning set in general, $|\mathcal{T}_{disjoint}| \geq |\mathcal{T}_{opt}|$, 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 4.1.3.1 we get:

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

□

Theorem 4.1.3.3. *The approximation ratio of $2k$ is tight.*

Proof. For every k that is a power of 2 we'll show a k-interval function such that $|\mathcal{T}_{approx}| = 2k|\mathcal{T}_{opt}|$ (following the notation from Theorem 4.1.3.1).

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

For each $p \in P$, we define the interval $[a_p, b_p]$ by appending 2-bit suffixes to p:

- $a_p = p00$
- $b_p = p01$

The first appended bit (0 for both a_p and b_p) ensures that there is at least one false point between any pair of intervals defined this way (all the points that have 1 in their $(n_k + 1)$-st component are false points). The second appended bit (0 for a_p and 1 for b_p) ensures that the interval has two points, so the approximation algorithm will use two vectors to span it. Thus we have k intervals on (n_k+2)-bit numbers, none of which intersect or touch.

Since $P = \{0, 1\}^{n_k}$, we can span all the intervals by the single ternary vector $\phi^{\{n_k\}}0\phi$. 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 two vectors for each interval.

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

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

5. Better approximation

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 enhanced algorithm converges to $2k$.

Consider re-wording.

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

Finish.

Conclusion

Finish.

Bibliography

[1] Jakub Dubovský. A construction of minimum DNF representations of 2-interval functions. Master's thesis, Charles University in Prague, 2012.

[2] Radek Hušek. Properties of interval boolean functions. Master's thesis, Charles University in Prague, 2014. [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. Boolean and Pseudo-Boolean Functions.

List of Tables

List of Abbreviations

DNF disjunctive normal form. ii, 1, 4–6, 16, 17

MSB most significant bit. 9, 12

Todo list