3. Monotone Boolean networks

Remember, a Boolean function $f = (f_1, f_2, \ldots, f_m) \in B_{n,m}$ is <u>monotone</u> iff for all $a, b \in \{0,1\}^n$, $a \leq b$ implies $f_i(a) \leq f_i(b)$ for $1 \leq i \leq m$.

$M_{n,m}$ denote the set of monotone functions in $B_{n,m}$. We write M_n for $M_{n,1}$.

$\Omega_m := \{\wedge, \vee\}$ denotes the <u>monotone base</u>.

An Ω_m-network is also called <u>monotone network</u>.

First we shall mention some fundamental properties of monotone functions and monotone networks.

<u>Lemma 3.1</u>

Each prime implicant of a monotone function $f \in M_n$ contains only nonnegated variables.

<u>Proof</u>: exercise

<u>Theorem 3.1</u>

The set of monotone functions and the set of functions which can be computed by a monotone network are equal.

<u>Proof</u>: exercise

By an estimation of the number of n-ary monotone Boolean functions and an application of Shannon's counting argument one can also prove that nearly all monotone Boolean functions have exponential (monotone) network complexity. But with respect to single output monotone Boolean functions, no technique for counting a nonlinear number of gates has been developed before 1985. The best lower bound for the monotone network complexity of an explicit function in M_n before 1985 was of size $4n$.

Jürgen Tiekenheinrich, A $4n$-lower bound on the monotone network complexity of a one-output Boolean function, IPL 18 (1984), 201–202.

For functions in $M_{n,m}$ where $m = \Theta(n)$, nonlinear lower bounds have been proved since 1969. All these lower bound proofs use the following property of each monotone network β which computes a function $f = (f_1, f_2, \ldots, f_m) \in M_{n,m}$.

For all nodes u in β, the function $res_\beta(u)$ can be written as a polynomial; i.e.,

$$res_\beta(u) = \bigvee_{j=1}^{t} m_j ,$$

where each m_j is a monomial. Starting at the input nodes of the network, we can compute these polynomials in the obvious way by

applying the properties of the Boolean operations.

We call this representation of $res_\beta(u)$ the polynomial expansion of $res_\beta(u)$.

Let u_i, $1 \leq i \leq m$ be the output node of the network β which computes the function f_i. For the polynomial $res_\beta(u_i) = \bigvee_{j=1}^{t_i} m_j$ which is computed at u_i, the following hold:

1) For $1 \leq j \leq t_i$, the monomial m_j is an implicant of the function f_i.

2) For all prime implicants p of f_i there is a $j \in \{1, 2, \ldots, t_i\}$ with $m_j = p$.

If the first property is not fulfilled then there is an input $(a_1, a_2, \ldots, a_n) \in \{0,1\}^n$ such that

$$f_i(a_1, a_2, \ldots, a_n) = 0 \text{ but } res_\beta(u_i)(a_1, a_2, \ldots, a_n) = 1.$$

If the second property is not fulfilled then there is an input $(a_1, a_2, \ldots, a_n) \in \{0,1\}^n$ such that

$$f_i(a_1, a_2, \ldots, a_n) = 1 \text{ but } res_\beta(u_i)(a_1, a_2, \ldots, a_n) = 0.$$

Most functions considered for proving lower bounds are homogeneous. A Boolean function $f \in B_{n,m}$ is called k-homogeneous if all prime implicants of f are of length k.

Before we consider explicitly defined monotone functions in $M_{n,m}$, we shall develop some replacement rules. The idea is to replace a gate by another gate which computes a certain function without changing the function computed by the whole network

Reference

- Kurt Mehlhorn, Zvi Galil, Monotone switching circuits and Boolean matrix product, Computing 16 (1976), 99–111.

Theorem 3.2

Let $f, g \in M_n$ and $t \in PIM(g)$ where $tt' \notin PIM(f)$ for all monoms t' (including the empty monom). Let h be defined by $PIM(h) = PIM(g) \setminus \{t\}$. If there is a gate v computing the function g in a monotone network β which computes the function f and we replace in β the gate v by a gate v' which computes h then the resulting network β' still computes f.

Proof:

Let f' be the function computed by the network β'.

Since $h \leq g$ there holds $f' \leq f$.

Assume that $f' \neq f$.

Then there is an input $a \in \{0,1\}^n$ such that $f'(a) = 0$ but $f(a) = 1$.

Since we have obtained β' from β by the replacement of the gate v in β by the gate v', there holds

$$h(a) = 0 \quad \text{and} \quad g(a) = 1.$$

Since $g = h \vee t$ there holds $t(a) = 1$.

Consider $t^* \in PI(f)$ with $t^*(a) = 1$.

Claim $\exists$ monome t' with $t \cdot t' = t^*$.

Proof of claim

Consider any variable x_i in t.

$t(a) = 1 \Rightarrow a_i = 1$.

Consider the input $b \in \{0,1\}^n$, defined by

$$b_j := \begin{cases} a_j & \text{if } j \neq i \\ 0 & \text{if } j = i \end{cases}$$

Then by definition

$$b \leq a \quad \text{and} \quad t(b) = 0.$$

$$\Rightarrow \quad f(b) = f'(b) = 0.$$

$\Rightarrow$ $t^*(b) = 0$.

Since b and a only differ at position i, x_i is a variable in t^*.

Since x_i has been chosen to be any variable in t it follows

t is a submonom of t^*.

$\Rightarrow$ $\exists t'$ with $tt' = t^*$.

□

By the choice of t, t is not a submonome of a prime implicant of f, or contradiction.

■

Theorem 3.3

Let β be a monotone network computing f and let v be a gate in β with $res_\beta(v) = g$. Let t, t_1 and t_2 be monomes such that

i) $tt_1, tt_2 \in IM(g)$ and
ii) $\forall$ monoms t': $t't t_1, t't t_2 \in IM(f) \rightarrow t't \in IM(f)$.

If we replace in β the gate v by a gate v' which computes $h = g \vee t$ then the resulting network β' still computes f.

Proof:

Let f' be the function computed by β

Since $h \geq g$ there holds $f' \geq f$.

Suppose that $f' \neq f$.

Then there is an $a \in \{0,1\}^n$ with

$$f'(a) = 1 \text{ and } f(a) = 0.$$

$\Rightarrow$

$$h(a) = 1, \; g(a) = 0 \text{ and } t(a) = 1.$$

Consider $t^* \in PIM(f')$ with $t^*(a) = 1$.

$f(a) = 0 \Rightarrow t^* t \notin IM(f)$.

To get a contradiction, it suffices to prove

$$t^* t t_1 \,, \; t^* t t_2 \in IM(f).$$

Consider $b \in \{0,1\}^n$ with $t^* t t_j(b) = 1 \quad j = 1,2$.

$t^* \in PIM(f') \Rightarrow f'(b) = 1$

$t t_j \in IM(g) \Rightarrow g(b) = 1$

Hence, $h \geq g \Rightarrow h(b) = 1$

$\Rightarrow$

Since at the only gate where β and β' differ, the same value is computed at input b, both networks β and β' compute the same value at input b

$\Rightarrow f(b) = f(b') = 1. \Rightarrow t^* t t_j \in IM(f)$.

Now, we shall consider explicit functions in $\mathcal{M}_{n,m}$. (6)

3.1 Boolean sums

References

- E. I. Neciporuk, On a Boolean matrix, Systems Theory Res. 21 (1971), 236 - 239.
- Nicholas Pippenger, On another Boolean matrix, TCS 11 (1980), 49 - 56.
- Ingo Wegener, A new lower bound on the monotone network complexity of Boolean sums, Acta Informatica 13 (1980), 109 - 115.
- Kurt Mehlhorn, Some remarks on Boolean sums, Acta Informatica 12 (1979), 371 - 375.

A Boolean sum is the or of a set of variables. We shall consider the monotone network complexity of sets of Boolean sums

$$f = (f_1, f_2, \dots, f_m) : \{0,1\}^n \to \{0,1\}^m$$

where

$$f_i = \bigvee_{j \in F_i} x_j \quad \text{and} \quad F_i \subseteq \{1, 2, \dots, n\}.$$

A set of Boolean sums is called (h,k)-disjoint if for all pairwise distinct $i_0, i_1, i_2, \dots, i_h$:

$$|F_{i_0} \cap F_{i_1} \cap \dots \cap F_{i_h}| \leq k.$$

This means that any $h+1$ different Boolean sums have at most k variables in common.

Let $K_{s,t} = (A, B, E)$ denote the complete bipartite graph with $|A| = s$ and $|B| = t$.

We can represent a set $f: \{0,1\}^n \to \{0,1\}^m$ of Boolean sums by a bipartite graph

$$G_f = (X_n, F_m, E) \text{ where}$$

$X_n = \{x_1, x_2, \dots, x_n\}$, $F_m = \{f_1, f_2, \dots, f_m\}$ and $E = \{(x_j, f_i) \mid 1 \le j \le n, 1 \le i \le m \text{ and } j \in F_i\}$.

Then f is (h,k)-disjoint iff G_f does not contain $K_{k+1, h+1}$.

For an explicitly constructed set of Boolean sums in $M_{n,n}$, Neciporuk has proved an $\Omega(n^{3/2})$ lower bound for the monotone complexity in 1968. This was the first nonlinear lower bound for an explicitly defined function in $M_{n,m}$, $m \in O(n)$. The proof builds on the fact that $(1,1)$-disjoint Boolean sums have "nothing in common". Neciporuk uses a well known construction of a bipartite graph which contains $\Omega(n^{3/2})$ edges and no $K_{2,2}$.

T. Kővári, V. T. Sós, P. Turán, On a problem of K. Zarankiewicz, Colloq. Math 3 (1954), 50 – 57.

The well known problem of Zarankiewicz is the following. Let $G_2(n,n)$ denote a bipartite graph with n nodes in each colour class.

What is the maximal size $z(n,k)$ of a graph in $G_2(n,n)$ which does not contain a $K_{k,k}$?

Upper and lower bounds for $z(n,k)$ are known. For $k=2$ and $k=3$ these bounds are tight up a constant factor. For $k>3$ there is a non-constant gap between upper and lower bound such that the problem of Zarankiewicz is still open.

Some years later, Pippenger and Mehlhorn have generalized the approach of Neciporuk to Boolean sums which have "little in common" such that little can be gained by using conjunctions or overlap. We shall present Kurt Mehlhorns proof of the following theorem.

Theorem 3.4

Let $f: \{0,1\}^n \to \{0,1\}^m$ be a (n,k)-disjoint set of Boolean sums. Then

$$C_{\Omega_m}(f) \geq \sum_{i=1}^{m} \frac{\frac{|F_i|}{k} - 1}{n \cdot \max\{1, n-1, k-2\}}$$

This theorem can be used to show for an explicitly defined $(2,2)$-disjoint Boolean sum in $M_{n,n}$

an $\Omega(n^{5/3})$ lower bound.

The proof of Theorem 3.4 is based on two lemmas. The first lemma shows that using $\wedge$-gates can save at most the factor $\max\{h-2, k-2\}$.

Lemma 3.1

Let $f: \{0,1\}^n \to \{0,1\}^m$ be a (h,k)-disjoint set of Boolean sums. Then

$$C_\vee(f) \leq \max\{1, h-2, k-2\} \cdot C_{\wedge,\vee}(f).$$

Proof:

Let β be an optimal Ω_m-network for f. Suppose that β contains

$$s \text{ } \vee\text{-gates and } t \text{ } \wedge\text{-gates.}$$

$\Rightarrow$

$$C_{\Omega_m}(f) = s + t.$$

The idea is to eliminate successively the $\wedge$-gates using at most $k-1$ and $h-1$, respectively additional $\vee$-gates. Since the eliminated $\wedge$-gate is saved, we need for each elimination of an $\wedge$-gate at most $k-2$ and $h-2$, respectively additional gates.

More precisely, we construct a sequence $\beta_0, \beta_1, \ldots, \beta_t$ of monotone networks where β_i, $0 \leq i \leq t$ contains

$$t-i \text{ } \wedge\text{-gates and } \leq s + \max\{1, h-1, k-1\} \cdot i$$

Note that $\beta_0 = \beta$. Suppose that β_i, $0 \leq i < t$ is already constructed.

Let v be a last $\wedge$-gate in topological order; i.e., between v and the output nodes there is no other $\wedge$-gate. Since β_i is acyclic, the gate v exists.

Let $res_{B_i}(v) = g$ and let g_1 and g_2 be the functions computed at the ingoing edges of v. Then

$$g = s_1 \vee s_2 \vee \dots \vee s_p \vee t_1 \vee t_2 \vee \dots \vee t_q$$

where each s_j is a variable and each t_j is a monome of length at least two in the polynomial expansion of $res_{B_i}(v)$.

We distinguish two cases.

Case 1: $p \leq k$

By application of Theorem 3.2 with respect to $t_1, t_2, \dots, t_q$ it follows that g can be replaced by

$$s_1 \vee s_2 \vee \dots \vee s_p.$$

For doing this, we need

$$p - 1 \leq k - 1$$

additional $\vee$-gates, saving the $\wedge$-gate v.

The resulting network is β_{i+1}.

Case 2: $p > k$

Let $f_1, f_2, \ldots, f_\ell$ be the output functions which depend on $g = res_{B_1}(v)$.

Between v and f_j, $1 \le j \le \ell$ there are only $\vee$-gates. Hence, we can write for $1 \le j \le \ell$

$$f_j = g \vee u_j.$$

Since f_j is a Boolean sum, u_j is not the constant 1.

Furthermore, for $1 \le j \le \ell$

$$\{s_1, s_2, \ldots, s_p\} \subseteq F_j.$$

Since f is (n,k)-disjoint, $p > k$ implies

$$\ell \le n.$$

Claim:

For $1 \le j \le \ell$ either $f_j = g_1 \vee u_j$ or $f_j = g_2 \vee u_j$.

Proof of claim:

Note that

$$(g = g_1 \wedge g_2 \text{ and } f_j = g \vee u_j) \Rightarrow$$

$$f_j \le g_1 \vee u_j \text{ and } f_j \le g_2 \vee u_j.$$

Suppose that $f_j < g_1 \vee u_j$ and $f_j < g_2 \vee u_j$.

Then there are $\alpha_1, \alpha_2 \in \{0,1\}^n$ with

- $f_j(\alpha_1) = 0$ but $(g_1 \vee u_j)(\alpha_1) = 1$

and

- $f_j(\alpha_2) = 0$ but $(g_2 \vee u_j)(\alpha_2) = 1$

Let

$$\alpha_1 = (a_1, a_2, \dots, a_n) \text{ and } \alpha_2 = (b_1, b_2, \dots, b_n)$$

Consider

$$\alpha = (c_1, c_2, \dots, c_n) \text{ where}$$

$$c_i := \max\{a_i, b_i\}, \quad 1 \le i \le n.$$

Since f_j is a Boolean sum and $f_j(\alpha_1) = f_j(\alpha_2) = 0$ there holds

$$f_j(\alpha) = 0.$$

Since $g_1 \vee u_j$ and $g_2 \vee u_j$ are monotone and $(g_1 \vee u_j)(\alpha_1) = (g_2 \vee u_j)(\alpha_2) = 1$ there holds

$$(g_1 \vee u_j)(\alpha) = (g_2 \vee u_j)(\alpha_2) = 1.$$

$\Rightarrow$

$$u_j(\alpha) = 1 \quad \text{or} \quad \big(g_1(\alpha) = g_2(\alpha) = 1 \text{ and hence, } g(\alpha) = 1\big)$$

In both cases, we obtain

$$f_j(\alpha) = (g \vee u_j)(\alpha) = 1,$$

a contradiction.

□

B_{i+1} is constructed from B_i in the following way:

(1) Replace g by the constant 0.

This eliminates the $\wedge$-gate v and at least one $\vee$-gate. After this replacement, the output node computing f_j, $1 \le j \le \ell$ computes the function u_j.

(2) For each f_j, $1 \le j \le \ell$ use an $\vee$-gate to compute $u_j \vee g_k$, $k \in \{1,2\}$ where

$$f_j = g_k \vee u_j.$$

This adds $\ell \le h$ $\vee$ gates. Since one $\vee$-gate is saved, we need at most $\ell - 1 \le h - 1$ additional $\vee$-gates.

The resulting network is B_{i+1}.

This proves the lemma. □

Lemma 3.1 shows that for proving a lower bound for $C_{\Sigma,m}(f)$ for a set f of (h,k)-disjoint Boolean sums, we can restrict us to prove a lower bound for $C_\vee(f)$ in the case that h and k are constants.

All gates in an $\{\vee\}$-network computes the disjunction of some variables. We call such a gate

small if the number of these variables is $\leq k$ and large otherwise.

The following lemma derives a lower bound of the large gates in an $\{\vee\}$-network which computes a (n,k)-disjoint set of Boolean sums.

Lemma 3.2

Let $f: \{0,1\}^n \to \{0,1\}^m$ be a (n,k)-disjoint set of Boolean sums. Then

$$C_\vee(f) \geq \sum_{i=1}^{m} \frac{\frac{|F_i|}{k} - 1}{n}.$$

Proof:

Let β be an optimal $\{\vee\}$-network for f. Note that the input nodes of β are small.

Let v be a large gate in β. Since v computes the disjunction of $> k$ variables and f is (n,k)-disjoint, at most n outputs f_i depend on v.

The large gates of β connects the small gates of β to the output nodes. Since each small gate computes the disjunction of at most k variables, the output f_i is connected to at least

$$\left\lceil \frac{|F_i|}{k} \right\rceil$$

small gates.

For each gate v in β let $n(v)$ denote the number of outputs f_i which depend on v. If v is large then $n(v) \leq h$. Hence,

$$\sum_{\text{large } v \in \beta} n(v) \leq h \cdot |\{v \in \beta \mid v \text{ large}\}|.$$

Consider the set of all large gates H in β, which are connected with the output node which computes f_i, $1 \leq i \leq m$.

As observed above, H has to connect at least $\lceil \frac{|F_i|}{2} \rceil$ different small nodes with the output node f_i. Hence, this subnetwork must contain a binary tree with $\geq \frac{\lceil F_i \rceil}{2}$ leaves.

$\Rightarrow$

$$\sum_{\text{large } v \in \beta} n(v) = \sum_{i=1}^{m} \#\text{ large gates connected with } f_i$$

$$\geq \sum_{i=1}^{m} \left(\left\lceil \frac{|F_i|}{2} \right\rceil - 1 \right)$$

Hence, we obtain

$$|\{v \in \beta \mid v \text{ large}\}| \geq \frac{1}{h} \sum_{i=1}^{m} \left(\left\lceil \frac{|F_i|}{2} \right\rceil - 1 \right).$$

This implies

$$C_v(f) \geq \frac{1}{h} \sum_{i=1}^{m} \left(\left\lceil \frac{|F_i|}{2} \right\rceil - 1 \right).$$

□

Combining Lemma 3.1 and Lemma 3.2, Theorem 3.4 can be proved in the following way.

$$C_{\to 2m}(f) \underset{\text{Le. 3.1}}{\geq} \frac{1}{\max\{1, k-2, \ell-2\}} C_v(f)$$

$$\underset{\text{Le. 3.2}}{\geq} \sum_{i=1}^{k} \frac{\frac{|F_i|}{\ell} - 1}{k \cdot \max\{1, k-2, \ell-2\}}$$

W.G. Brown has constructed a graph in $G_2(n,n)$ which contains $\Omega(n^{5/3})$ edges and no $K_{3,3}$.

W. G. Brown, On graphs that do not contain a Thompson graph, Canad. Math. Bull. 9 (1966), 281 – 285.

Using this construction, we obtain an explicit $(2,2)$-disjoint set of Boolean sums $f: \{0,1\}^n \to \{0,1\}^n$ with $\sum_{i=1}^{n} |F_i| = \Omega(n^{5/3})$.

$\Rightarrow$

An $\Omega(n^{5/3})$ lower bound for the monotone network complexity of this function is proved.

3.2 Boolean matrix multiplication

References

- Vaughan R. Pratt, The power of negative thinking in multiplying Boolean matrices, SIAM J. Comput. 4 (1974), 326 - 330.
- Mike S. Paterson, Complexity of monotone networks for Boolean matrix product, TCS 1 (1975), 13 - 20
- Kurt Mehlhorn, Zvi Galil, Monotone switching circuits and Boolean matrix product, Computing 16 (1976), 99 - 111.

In 1974, Pratt has shown that each monotone network computing the product of two $n \times n$ Boolean matrices contains at least $\frac{1}{2} n^3$ $\wedge$-gates. Mehlhorn and Galil and Paterson have refined the method of Pratt and have proved that the school-method is the unique optimal monotone network for Boolean matrix multiplication. We shall present the proof of Mehlhorn and Galil and Paterson.

Let $r, p, q \in \mathbb{N}$, A be a $(r \times p)$- and B be a $(p \times q)$ Boolean matrix. Then

$$C := A \cdot B$$

is a $(r \times q)$ Boolean matrix where

$$c_{ik} := \bigvee_{j=1}^{p} a_{ij} \wedge b_{jk} .$$

Theorem 3.5

Each monotone network which computes the product of a $(r \times p)$ Boolean matrix with a $(p \times q)$ Boolean matrix contains at least $r \cdot p \cdot q$ $\wedge$-gates and at least $r \cdot q \cdot (p-1)$ $\vee$-gates.

Proof:

Let β be an optimal monotone network computing the product matrix $C = A \cdot B$.

Goal:
Isolation of an $\wedge$-gate computing $a_{i,j} \cdot b_{j,k}$ for all tripels (i,j,k).

General method:

Definition of predicates P on the nodes of β such that

- P holds for at least one output node of β and P does not hold for any input node of β.

$\Rightarrow$

There exists a gate v in β such that

- P does not hold for any ingoing edge of v but P holds for $res_\beta(v)$.

Let $I(P)$ denote the set of these gates.

$\rightsquigarrow$

1) Identification of $r \cdot q$ $\wedge$-gates for the products

$$a_{i_1} b_{1k} \quad 1 \leq i \leq r, \; 1 \leq k \leq p$$

and elimination of these gates by setting

a_{i_1} at 1, $1 \leq i \leq r$ and
b_{1k} at 0, $1 \leq k \leq p$.

The resulting monotone network computes the $(r, p-1, q)$ - matrix product C' where

$$c'_{ik} := \bigvee_{j=2}^{p} a_{ij} \wedge b_{jk}.$$

2) Application of induction.

If we consider a gate v then

- h denotes always $\mathrm{res}_\beta(v)$ and
- h_1, h_2 denote the input functions of v.

For $1 \leq i \leq r$, $1 \leq k \leq q$ we define the predicate P_{ik} by

$$P_{ik}(v) \Leftrightarrow a_{i_1} b_{1k} \leq h \wedge a_{i_1} \not\leq h \wedge b_{1k} \not\leq h.$$

This implies that $a_{i_1} b_{1k}$ is a prime implicant of h.

The input nodes of β does not fulfill P_{ik} but the output function c_{ik} fulfills P_{ik}.

$$\Rightarrow \; I(P_{ik}) \neq \emptyset.$$

Consider $v \in I(P_{ik})$. First, we shall show that v is an $\wedge$-gate.

For doing this assume that v is an $\vee$-gate.

From $a_{in} b_{1k}$ it follows that

$$a_{in} b_{1k} \leq h_1 \quad \text{or} \quad a_{in} b_{1k} \leq h_2.$$

W.l.o.g. we can assume

$$a_{in} b_{1k} \leq h_1.$$

Now $\neg P_{ik}(h_1) \Rightarrow$

$$a_{in} \leq h_1 \quad \text{or} \quad b_{1k} \leq h_1$$

$\Rightarrow \neg P_{ik}(h)$, a contradiction.

Hence, v is an $\wedge$-gate.

Since $a_{in} b_{1k} \leq h$ there hold

$$a_{in} b_{1k} \leq h_1 \quad \text{and} \quad a_{in} b_{1k} \leq h_2.$$

Furthermore, $\neg P_{ik}(h_1)$ and $\neg P_{ik}(h_2)$

$\Rightarrow$

Either

$$a_{in} \leq h_1 \quad \text{and} \quad b_{1k} \leq h_2$$

or

$$a_{in} \leq h_2 \quad \text{and} \quad b_{1k} \leq h_1.$$

Note that $a_{in} \leq h_1 \wedge a_{in} \leq h_2 \Rightarrow a_{in} \leq h$ and hence, $\neg P_{ik}(h)$. Analogously, we can exclude $b_{1k} \leq h_1 \wedge b_{1k} \leq h_2$.

Altogether, we have found for each pair (i,k) an $\wedge$-gate v with predicate $P_{ik}(v)$ holds.

Now we shall prove that these $\wedge$-gates are pairwise disjoint. For doing this, suppose

$$v \in I(P_{i_1k_1}) \cap I(P_{i_2k_2}) \text{ with } (i_1,k_1) \neq (i_2,k_2).$$

Up to symmetry, one of the two situations occurs.

1) $a_{i_1 1}, a_{i_2 1} \leq h_1$ and $b_{1k_1}, b_{1k_2} \leq h_2$

2) $a_{i_1 1}, b_{1k_2} \leq h_1$ and $b_{1k_1}, a_{i_2 1} \leq h_2$.

As we shall see, we can apply in both situations Theorem 3.3 to construct a better network than β. This contradicts the optimality of β.

Situation 1:

Suppose $i_1 \neq i_2$. Then for input function h_1 and $t = \wedge$, $t_1 = a_{i_1 1}$ and $t_2 = a_{i_2 1}$ the assumptions of Theorem 3.3 are fulfilled. Hence, h_1 can be replaced by $h_1 \vee 1$.

$\Rightarrow$

One input of v gets to be constant such that the gate v can be eliminated.

This contradicts the optimality of β.

If $i_1 = i_2$ then $k_1 \neq k_2$. Analgously, this case can be excluded as well.

Situation 2:

By application of Theorem 3.3, we can replace both inputs of $\vee$ by 1.

Exercise:

Show that in Situation 2, both inputs of $\vee$ can be fixed at 1 without changing the function computed by $\mathcal{B}$.

This contradicts the optimality of $\mathcal{B}$ as well.

Altogether, we have proved that the sets $I(P_{ik})$ are pairwise disjoint.

$\Rightarrow$

r·q different $\wedge$-gates are isolated.

Now we shall consider $\vee$-gates. Analogous to the consideration of $\wedge$-gates, we define for $1 \leq i \leq r$, $1 \leq k \leq q$ a predicate Q_{ik} in the following way:

$Q_{ik}(v) \Leftrightarrow a_{i1} b_{1k} \leq h \leq A_i \vee b_{1k}$ and $h \not\leq b_{1k}$

where $A_i := \bigvee_{j \neq 1} a_{ij}$.

The input nodes does not fulfill Q_{ik}. The output node c_{ik} fulfills Q_{ik} since

- $a_{i1} b_{1k} \in \mathrm{PIM}(c_{ik}) \Rightarrow a_{i1} b_{1k} \leq c_{ik}$.
- Each prime implicant of c_{ik} contains

the variable b_{1k} or a variable in $\{a_{i2}, a_{i3}, \ldots, a_{ir}\}$.
Hence,

$$c_{ik}(x) = 1 \Rightarrow (A_i \vee b_{1k})(x) = 1$$

$$\Rightarrow c_{ik} \leq A_i \vee b_{1k}.$$

$c_{ik} \not\leq b_{1k}$ is obvious.

Hence, $I(Q_{ik}) \neq \phi$ if $p \geq 2$.

Exercise

Prove the following two assertions.

a) If $v \in I(Q_{ik})$ then v is an $\vee$-gate and either $h_1 \leq b_{1k}$ or $h_2 \leq b_{1k}$.

b) The sets $I(Q_{ik})$ are pairwise disjoint.

Altogether, we have achieved the following:

1) For every pair (i,k) we have located an $\wedge$-gate in β with one of its input functions has prime implicant a_{in}. For different pairs we have identified different $\wedge$-gates.

$\Rightarrow$

Fixing a_{in} at 1 for $1 \leq i \leq r$ eliminates $r \cdot q$ $\wedge$-gates in β.

2) If $p > 1$ then we have located an $\vee$-gate in β for all pairs (i,k) where one of its input functions contains only prime

implicants containing the variable b_{1k}. For different pairs we have identified different v-gates.

$\Rightarrow$

Fixing b_{1k} at 0 for $1 \le k \le q$ eliminates $r \cdot q$ v-gates in β.

Finally, fixing a_{i1} at 1 and b_{1k} at 0 for $1 \le i \le r$ and $1 \le k \le q$ transforms β into a network for the functions

$$c'_{ik} := \bigvee_{j=2}^{p} a_{ij} b_{jk} .$$

Altogether, applying induction the assertion of the theorem is proved.

■

3.3 A generalized Boolean matrix product

References

- Ingo Wegener, Switching functions whose monotone complexity is nearly quadratic, TCS 9 (1979), 83–97.
- Ingo Wegener, Boolean functions whose monotone complexity is of size $\frac{n^2}{\log n}$, TCS 21 (1982), 213–224.

Let Y be the Boolean matrix product of the matrix X_1 with the transposed matrix X_2. Then we have $y_{ij} = 1$ iff the i-th row of X_1 and the j-th row of X_2 have a common one.

In 1978, Wegener has generalized this to the "direct product" of m $M \times N$-matrices $X_1, X_2, \ldots, X_m$. For each choice of one row of every matrix the corresponding output is one iff the chosen rows have a common one.

To formalize this let $x^i_{\mu\varepsilon}$ denote the element of matrix M_i at position (μ, ε). Then we say that $x^i_{\mu\varepsilon}$ is a variable of <u>type ε</u>. This means that the type of a variable is its position in the corresponding row.

For $1 \leq h_1, h_2, \ldots, h_m \leq M$ let

$$y_{h_1, h_2, \ldots, h_m} := \bigvee_{1 \leq \varepsilon \leq N} x^1_{h_1\varepsilon} x^2_{h_2\varepsilon} \cdot \ldots \cdot x^m_{h_m\varepsilon}.$$

We say that the prime implicant

$$(h_1, h_2, \ldots, h_m, \varepsilon) := x^1_{h_1\varepsilon} x^2_{h_2\varepsilon} \cdot \ldots \cdot x^m_{h_m\varepsilon}$$

is of <u>type ε</u>.

The <u>generalized Boolean matrix product</u> f^m_{MN}, $m, M \geq 2$ is defined as follows:

$$f^m_{MN} : \{0,1\}^{mMN} \longmapsto \{0,1\}^{M^m}$$

where

$$y_{h_1 h_2 \ldots h_m} := \bigvee_{1 \le \ell \le N} x^1_{h_1 \ell} \, x^2_{h_2 \ell} \cdot \ldots \cdot x^m_{h_m \ell}.$$

First, we shall prove an upper bound for the monotone network complexity of f^m_{MN}.

Theorem 3.6

$$C_{\Sigma_m}(f^m_{MN}) \le NM^m(2 + (m-1)^{-1}) \le 3NM^m.$$

Proof:

Assume that all monoms $(h_1, h_2, \ldots, h_i, \ell)$ are computed. Then NM^i additional $\wedge$-gates suffice to compute all monoms $(h_1, h_2, \ldots, h_{i+1}, \ell)$.

$\Rightarrow$

$$C_\wedge(f^m_{MN}) \le N \cdot \sum_{2 \le i \le m} M^i$$

$$\le N(M^{m+1} - 1)(M-1)^{-1}$$

$$\le N \cdot M^m (1 + (M-1)^{-1}).$$

Afterwards, each output can be computed using $N-1$ $\vee$-gates.

■

Our goal is to prove a $\frac{1}{2}NM^m$ lower bound for the number of $\wedge$-gates of a monotone network which computes f^m_{MN}.

First, we shall investigate the structure of optimal monotone networks computing f^m_{MN}.

Lemma 3.3

Let g be a function which is computed in a monotone network for f^m_{MN}. Let $(i_1, i_2, \ldots, i_m, \ell), (j_1, j_2, \ldots, j_m, \ell) \in PIM(g)$ for some $\ell \in \{1, 2, \ldots, N\}$ and $i_\ell, j_\ell \in \{1, 2, \ldots, M\}$, $1 \leq \ell \leq m$. Let $A := \{\ell \mid i_\ell = j_\ell\}$ and let $t := \bigwedge_{\ell \in A} x^\ell_{i_\ell \ell}$. Then g can be replaced by $h := g \vee t$ and the resulting network still computes f^m_{MN}.

Proof:

Let

$$t_1 := \bigwedge_{\ell \notin A} x^\ell_{i_\ell \ell} \quad \text{and} \quad t_2 := \bigwedge_{\ell \notin A} x^\ell_{j_\ell \ell}$$

Definition of $A \Rightarrow$

t_1 and t_2 have no common variable.

Now we shall prove that the assumptions of Theorem 3.3 are fulfilled with respect to the chosen t, t_1 and t_2.

By construction

$$tt_1, tt_2 \in PIM(g) \subseteq IM(g).$$

To prove the second assumption, we assume that this assumption does not hold, i.e.,

$\exists$ monome t' and output function $y_{u_1 u_2 \dots u_m}$ with

- $t'tt_1, t'tt_2 \in IM(y_{u_1 u_2 \dots u_m})$ but $t't \notin IM(y_{u_1 \dots u_m}$

Then

$t'tt_1$ contains a prime implicant $(u_1, u_2, \dots, u_m, e')$,

but

$t't$ does not contain the prime implicant $(u_1, u_2, \dots, u_m, e)$.

By construction, it holds

$$e' = e.$$

The same holds with respect to $t'tt_2$.

Altogether, we obtain

- $(u_1, u_2, \dots, u_m, e)$ is a submonomial of $t'tt_1$ and also a submonomial of $t'tt_2$ but not a submonomial of $t't$.

$\Rightarrow$

t_1 and t_2 must have a variable in common.

But this contradicts the definition of A.

□

A monomial u is <u>useful</u> for an output of f^m_{MN} if u is contained in a prime implicant of that output.

Lemma 3.3 $\Rightarrow$

If the function $g = res_\beta(u)$ includes several useful monomials of type ℓ then we can replace g in β by gvt where t is the common part of all useful monomials of type ℓ.

<u>Goal</u>:

Application of Lemma 3.3 without additional cost.

For doing this, we need the following properties:

1) $\vee$ gates are not counted.
2) All monomials of $< m$ variables are given for free; i.e., are given as additional inputs of the network.

A monotone network fulfilling both properties is called a $*$-network. C^*_{2m} is the associated complexity measure.

Given any $*$-network β for f^m_{MN}, we wish to transform β into a so-called <u>standard</u> $*$-network of the same complexity by applying Lemma 3.3.

For doing this, we consider the gates in β in any topological order. Let u be the current considered gate and $g := res_\beta(u)$.

For $1 \le \ell \le N$ let

$$t_\ell := \begin{cases} 0 & g \text{ contains at most one useful monomial of type } \ell \\ \text{common part of all useful monomials of type } \ell & \text{otherwise} \end{cases}$$

Without additional cost, we replace

$$g \quad \text{by} \quad g \vee t_1 \vee t_2 \vee \dots \vee t_N$$

$\rightsquigarrow$

Altogether, we obtain a $*$-network β' for f^m_{MN} such that

(*)
- all functions computed at the ingoing and outgoing edges of the $\wedge$-gates have at most one useful monomial of type ℓ as prime implicant.

<u>Theorem 3.7</u>

Let $m \ge 2$. Then

$$C_{\Omega_m}(f^m_{MN}) \ge C_\wedge(f^m_{MN}) \ge C^*_{\Omega_m}(f^m_{MN}) > \tfrac{1}{2} N \cdot M^m$$

Corollary 3.1

For $n \geq 4$ let $m(n) := \lfloor \log n \rfloor$, $M(n) := 2$, $N(n) := \lfloor \frac{n}{2 \log n} \rfloor$ and $h_n := f^{m(n)}_{M(n)N(n)}$. Then the function h_n depends on at most n variables and has at most n output functions. Furthermore, $C_{\Omega_m}(h_n) = \Omega\left(\frac{n^2}{\log n}\right)$.

Proof:

Exercise ∎

Proof of Theorem 3.7:

Let β be an optimal standard $*$-network for f^m_{MN}.

Goal.

For each $\wedge$-gate v in β, definition of a value function

$$c_v : PIM\left(f^m_{MN}\right) \longrightarrow [0,1]$$

such that

$$c(v) := \sum_{1 \leq h_1, h_2, \ldots, h_m \leq M} \; \sum_{1 \leq \ell \leq N} c_v(h_1, h_2, \ldots, h_m, \ell) \leq$$

Properties:

a) $c_v(h_1, h_2, \ldots, h_m, \ell)$ is an estimate of the

contribution of the $\wedge$-gate v to the computation of the prime implicant $(h_1, \dots, h_m, \ell)$.

b) Each gate contributes to all prime implicants at most the value one.

$\Rightarrow$

For an optimal $*$-network β there holds

$$c(\beta) := \sum_{v \text{ } \wedge\text{-gate in } \beta} c(v) \leq C_{\wedge}(\beta) = C^*_{\Omega_m}(f^m_{MN})$$

This means that $c(\beta)$ is a lower bound for $C^*_{\Omega_m}(f^m_{MN})$. The value function will have the following property:

Claim 1:

$$c(h_1, h_2, \dots, h_m, \ell) := \sum_{v \text{ } \wedge\text{-gate in } \beta} c_v(h_1, h_2, \dots, h_m, \ell) > \frac{1}{2}.$$

Before defining the value function and proving the claim we shall terminate the proof of the theorem.

Claim 1 $\Rightarrow$

$$C^*_{\Omega_m}(f^m_{MN}) \geq \sum_{1 \leq h_1, \dots, h_m \leq M} \sum_{1 \leq \ell \leq N} c(h_1, \dots, h_m, \ell)$$

$$> \frac{1}{2} \cdot N \cdot M^m.$$

Definition of the value function

Let v be an $\wedge$-gate in an optimal standard $*$-network β for f^m_{MN}. Let

g', g'' be the functions of the ingoing edges of v

$g := res_\beta(v)$.

Idea:

The value function c_v assigns to the $\wedge$-gate v a positive value for the prime implicant t in $PIM(f^m_{MN})$ if

- $t \in PIM(g)$ and $t \notin PIM(g')$ or
- $t \in PIM(g)$ and $t \notin PIM(g'')$.

Question:

In these cases, which value > 0 should be chosen?

To define these values let

$i_1, i_2, \ldots, i_{q'} \in \{1, 2, \ldots, N\}$ be those types such that

$\exists\, t_e \in PIM(f^m_{MN})$ of type i_e with

$t_e \in PIM(g)$ but $t_e \notin PIM(g')$.

Furthermore, let

$j_1, j_2, \ldots, j_{q''} \in \{1,2,\ldots,N\}$ be those types such that

$\exists\, t_e \in \mathrm{PIM}(f^m_{MN})$ of type j_e with

$t_e \in \mathrm{PIM}(g)$ but $t_e \notin \mathrm{PIM}(g'')$

Then we define for $t \in \mathrm{PIM}(f^m_{MN})$

$$c'_v(t) := \begin{cases} \frac{1}{2q'} & \text{if } t \in \mathrm{PIM}(g) \text{ and } t \notin \mathrm{PIM}(g'), \\ 0 & \text{otherwise} \end{cases}$$

$$c''_v(t) := \begin{cases} \frac{1}{2q''} & \text{if } t \in \mathrm{PIM}(g) \text{ and } t \notin \mathrm{PIM}(g''), \\ 0 & \text{otherwise} \end{cases}$$

Then, $c_v(t)$ is defined by

$$c_v(t) := c'_v(t) + c''_v(t).$$

Then we obtain because of property (*):

$$c'(v) = \sum_{1 \le h_1,\ldots,h_m \le M} \; \sum_{1 \le e \le N} c'_v(h_1,\ldots,h_m,e)$$

$$= q' \cdot \frac{1}{2 \cdot q'} = \frac{1}{2}$$

Note that in an optimal standard *-network, at an $\wedge$-gate at most one prime implicant of each type can have a value > 0.

and also

$$c''(v) = \sum_{1 \le l_1, \dots, l_m \le M} \sum_{1 \le \ell \le N} c''_v(l_1, \dots, l_m, \ell)$$

$$= q'' \cdot \frac{1}{2q''} = \frac{1}{2}.$$

$\Rightarrow$

$$c(v) = c'(v) + c''(v) = 1.$$

It remains to prove Claim 1.

Proof of Claim 1:

Consider the prime implicant $t = (l_1, l_2, \dots, l_m, \ell)$ and the corresponding output

$$y_t := y_{l_1 l_2 \dots l_m}.$$

Let $\mathcal{B}(t)$ be that subnetwork of $\mathcal{B}$ which contains the following gates and inputs.

gate v is contained in $\mathcal{B}(t)$ iff

there is a path P in $\mathcal{B}$ from v to the output y_t and t is a prime implicant of all functions computed on P (inclusive $\text{res}_{\mathcal{B}}(v)$).

Additionally, the inputs of the gates in $\mathcal{B}(t)$ are contained in $\mathcal{B}(t)$ as well.

Properties:

1) For each input function g of $\mathcal{B}(t)$ holds $t \notin \text{PIM}(g$

2) t is prime implicant of each function $res_{\beta}(v)$ where v is a gate in $\beta(t)$.

3) Let v be a gate in $\beta(t)$ with both direct predecessors of v are inputs of $\beta(t)$

$\Rightarrow$

v is an $\wedge$-gate (otherwise $t \notin PIM(res_{\beta}(v))$.)

4) If an input of $\beta(t)$ is input of an $\wedge$-gate of $\beta(t)$ then a proper shortening of t is a prime implicant of that input.

Let $s_1, s_2, \ldots, s_D$ be those inputs of $\beta(t)$ which are input of an $\wedge$-gate in $\beta(t)$.

Let $v(i)$ be an $\wedge$-gate in $\beta(t)$ with input s_i.
Let

$$c^*(v(i)) := \begin{cases} c'(v(i)) & \text{if } s_i \text{ is the first input of } v(i) \\ c''(v(i)) & \text{if } s_i \text{ is the second input of } v(i) \end{cases}$$

Properties 1 and 2 $\Rightarrow$ $c^*(v(i)) > 0$.

For $1 \leq i \leq D$ let $b_i := c^*(v(i))$.

W.l.o.g. we can assume $b_1 \geq b_2 \geq \ldots \geq b_D$.

Note $\sum_{i=1}^{D} b_i > \frac{1}{2} \Rightarrow$ Claim 1.

We shall prove $b_1 + b_2 + \ldots + b_D > \frac{1}{2}$.

Choose $w_i \in PIM(s_i)$ such that a proper prolongation w_i^* of w_i is contained in

$$PIM(res(v(i))) \cap PIM(f_{MN}^m)$$

and the type of w_i^* is different to the types of $w_1^*, w_2^*, \ldots, w_{i-1}^*$.

We can always choose $w_1^* = t$. We distinguish two cases

Case 1:

The choice of w_i according to the rules above is impossible for an $i \leq D$.

$\Rightarrow$

$c^*(v(i))$ is positiv for $\leq (i-1)$ prime implicants

Hence, by the definition of the value function

$$b_i \geq (2(i-1))^{-1}.$$

Because of $b_1 \geq b_2 \geq \ldots \geq b_D$, we obtain

$$\sum_{i=1}^{D} b_i \geq i\, b_i \geq i(2(i-1))^{-1} > \frac{1}{2}.$$

Case 2:

The choice of $w_1, w_2, \ldots, w_D$ according to the rules above is possible.

Construction $\Rightarrow$ $w_i \in PIM(s_i)$ for $1 \leq i \leq D$

$\Rightarrow \quad w_1 w_2 \cdots w_D \leq s_1 s_2 \cdots s_D.$

Construction of $\beta(+) \Rightarrow$

$\forall\, a$ with $s_i(a) = 1$ for $1 \leq i \leq a$
there holds $y_t(a) = 1$

(this can easily be shown by induction).

$\Rightarrow$

$$w_1 w_2 \cdots w_D \leq y_t.$$

All variables in w_i are of type ℓ_i and $\ell_1, \ell_2, \ldots, \ell_D$ are pairwise different.

Each w_i is a proper shortening of w_i^* and $c^{*(v(i))}(w_i^*) > 0$.

$\Rightarrow$

w_i contains $\leq m-1$ variables

$\Rightarrow$

$$w_1 w_2 \cdots w_D \notin IM(y_t)$$

a contradiction

Hence, Case 2 cannot occur.

■

3.4 The Boolean convolution

References

- Nicholas Pippenger, Leslie G. Valiant, Shifting graphs and their applications, JACM 23 (1976), 423 - 432.
- Edmund A. Lamagna, The complexity of monotone networks for certain bilinear forms, routing problems, sorting and merging, IEEE Trans. Comput. 28 (1979), 773 - 782.
- Norbert Blum, An $\Omega(n^{4/3})$ lower bound on the monotone network complexity of the n^{th} degree convolution, TCS 36 (1985), 59-69.
- Jürgen Weiß, An $n^{\frac{3}{2}}$ lower bound on the monotone network complexity of the Boolean convolution, Information and Control 59 (1983), 184 - 188.

Let

$$A := \{a_0, a_1, \dots, a_{n-1}\}, \quad B := \{b_0, b_1, \dots, b_{n-1}\}$$

be two disjoint sets of n variables. Then we define the n-th degree convolution C_n in the following way.

$$C_n = (c_0, c_1, \dots, c_{2n-2}) : \{0,1\}^{2n} \longrightarrow \{0,1\}^{2n-1}$$

where

$$c_k := \bigvee_{i+j=k} a_i b_j \qquad 0 \le k \le 2n-2.$$

All sets of monotone functions considered so far (Boolean sums, Boolean matrix multiplication, generalized Boolean matrix multiplication) have disjunctive properties that Boolean convolution does not have. To formalize this, we need some notations.

Let $A = \{a_0, a_1, \dots, a_p\}$ and $B = \{b_0, b_1, \dots, b_q\}$ be two disjoint sets of variables. A monotone function

$$f = (f_1, f_2, \dots, f_m) : A \cup B \to \{0,1\}^m$$

is <u>bilinear</u> if each prime implicant of f consists of one variable from A and one variable from B. Then, f is a set of <u>bilinear forms</u>.

<u>Example</u>:

Boolean matrix multiplication and also the Boolean convolution are bilinear.

◆

We can extend this definition to multilinear forms (i.e., we have more than two disjoint sets of variables) in the obvious way

f is a set of <u>semidisjoint</u> bilinear forms if f has the following properties:

1) $PIM(f_i) \cap PIM(f_j) = \emptyset$ for $1 \le i < j \le m$.

2) For $1 \le k \le m$, each variable $x_i \in X$ is contained in at most one prime implicant in $PIM(f_k)$ and also each variable $y_j \in Y$ is contained in at most one prime implicant in $PIM(f_k)$.

Note that Boolean matrix multiplication and also Boolean convolution are semidisjoint.

For $a_i b_j \in PIM(f)$, we define the following subset $PIM_{ij}(f)$ of $PIM(f)$ inductively:

a) $a_i b_j \in PIM_{ij}(f)$

b) $a_{i'} b_{j'} \in PIM_{ij}(f) \Rightarrow$

$a_{i'} b_{j''} \in PIM_{ij}(f) \quad \forall\, a_{i'} b_{j''} \in PIM(f)$
and also
$a_{i''} b_{j'} \in PIM_{ij}(f) \quad \forall\, a_{i''} b_{j'} \in PIM(f)$.

If we add the following third property to the properties above then we call the set of bilinear forms <u>disjoint</u>.

3) For $1 \le k \le m$ for each $PIM_{ij}(f)$:

$|\, PIM_{ij}(f) \cap PIM(f_k) \,| \le 1$.

Boolean matrix multiplication is disjoint.
Boolean convolution is not disjoint.

Exercise:

a) Give a formal definition of multilinear forms.
b) Show that Boolean matrix multiplication is a set of disjoint bilinear forms.
c) Show that Boolean convolution is not a set of disjoint bilinear forms.

Note that the generalized Boolean matrix multiplication is a set of disjoint multilinear forms.

Boolean sums are $(1,1)$-disjoint and $(2,2)$-disjoint, respectively. Boolean matrix multiplication is a set of disjoint bilinear forms. The generalized Boolean matrix multiplication is a set of disjoint multilinear forms. The Boolean convolution has not such disjointness property. Convolution is a set of semidisjoint bilinear forms. The sets of variables upon which two functions $f_z, f_{z'} \in C_n$ depend can be almost equal which is not the case for the sets of functions mentioned above.

As a consequence, the assumptions of Theorem 3.3 do not hold for the Boolean convolution such that this theorem cannot be applied.

Now, we shall prove a general lower bound for the monotone network complexity of semidisjoint bilinear forms.

Theorem 3.8

Let f be a semidisjoint bilinear form. Let r_i be the number of prime implicants which contain the variable i. Then

$$C_{\wedge,\vee}(f) \geq \sum_{i=1}^{p} r_i^{1/2}.$$

Corollary 3.1

The monotone network complexity of the Boolean n-th degree convolution is $\geq n^{3/2}$.

Proof:

This is a direct consequence of Theorem 3.8 since $r_i = n$ for $0 \leq i \leq n-1$.

∎

Proof of Theorem 3.8:

Let β be an optimal Ω_m-network for f.

Note that after replacing a_0 by 0, we obtain a subfunction of f which is semidisjoint as well. Moreover, the values r_i, $i>0$ do not change.

$\Rightarrow$

It suffices to prove that after setting a_0 to 0 at least r_0^2 gates have been eliminated.

Let

s_0 denote the number of functions f_i with $f_i = a_0 b_j$ for a j.

$\Rightarrow$

Setting a_0 to 0 eliminates those s_0 $\wedge$-gates where these outputs are computed.

Since $\mathcal{S}$ is semidisjoint, these gates cannot be used to compute other outputs.

Claim:

Setting a_0 to 0 eliminates at least $(r_0 - s_0)^{\frac{1}{2}}$ $\vee$-gates.

Note that this claim implies that fixing a_0 to 0 eliminates at least $r_0^{\frac{1}{2}}$ gates.

To prove the claim, we consider exactly those functions f_k which depend on a_0 and which contain at least two prime implicants.

Let

$$P = v_0, v_1, \ldots, v_m$$

be a path from a_0 to f_k.

$\Rightarrow$

The path P contains at least one v-gate v_k with

$$a_i b_j \leq res_\beta(v_k)$$

for an $i \neq 0$ and a j.

Otherwise, because of the semidisjointness of f, the function f_k could not contain two prime implicants.

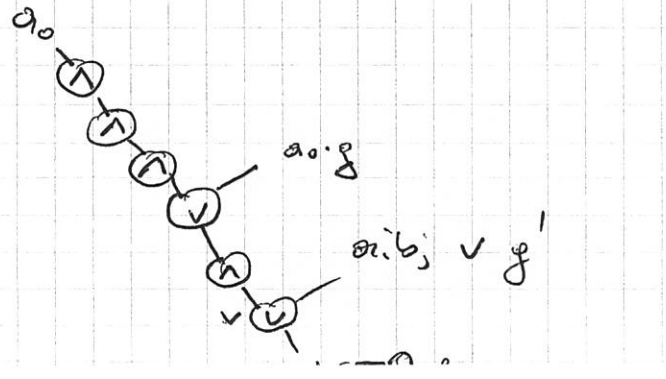
The first such an v-gate on P is called <u>suitable for P</u>. Let

$$\mathcal{P} := \{ P \mid P \text{ is a path from } a_0 \text{ to an output } f_k \text{ which depends on } a_0 \text{ and contains} \geq 2 \text{ prime implicants} \}.$$

Note that all implicants of the output of an v-gate between a_0 and such a gate contains the variable a_0.

Let $V^* := \{ \text{gate } v \mid v \text{ is suitable v-gate for a path } P \in \mathcal{P} \}$.

Consider any $v \in V^*$.



$\Rightarrow$

Fixing a_0 to 0 eliminates v.

Claim: $|V^*| \geq (r_0 - s_0)^{\frac{1}{2}}$.

Proof:

Let $V^* = \{v_1, v_2, \ldots, v_p\}$. Then there are $i_1, i_2, \ldots, i_p \neq 0$ and $j_1, j_2, \ldots, j_p$ with

$$a_{i_\ell} b_{j_\ell} \leq res_\beta(v_\ell), \quad 1 \leq \ell \leq p.$$

After setting

a_{i_ℓ}, b_{j_ℓ} to 1 for $1 \leq \ell \leq p$

on all gates in V^*, the constant 1 is computed.

$\Rightarrow$

No output f_k of f with ≥ 2 prime implicants depends on a_0.

Consider any such an output f_k and choose s such that

$$a_0 b_s \leq PIN(f_k).$$

$\Rightarrow$

After the assignment above for the function f'_k computed at that output node holds:

$f_k' = 1$ or $b_s \in PIM(f_k')$.

In the second case

$$a_{i_\ell} b_s \in PIM(f_k)$$

for an $\ell \in \{1, 2, \ldots, p\}$.

Since $i_\ell \neq 0$, this contradicts the semidisjointness of f. Hence,

$$f_k' = 1 \text{ and } a_{i_\ell} b_{j_m} \in PIM(f_k)$$

for some $1 \leq \ell, m \leq m$.

This holds for all $r_0 - s_0$ output nodes depending on a_0 and having ≥ 2 prime implicants.

Since f is semidisjoint, prime implicants of different output nodes are different.

At most p^2 distinct prime implicants can be constructed using p variables in A and p variables in B.

$\Rightarrow$

$$p^2 \geq r_0 - s_0$$

$\Rightarrow$

$$|V^*| \geq (r_0 - s_0)^{\frac{1}{2}}.$$

Now, the assertion can be proved using induction

Using Theorem 3.8, we have obtained an $\Omega(n^{\frac{3}{2}})$ lower bound for the number of $\vee$-gates in any monotone network which computes the n-th degree Boolean convolution. To get a lower bound for the number of $\wedge$-gates, we need some other techniques.

We start with an optimal monotone network β_0 computing C_n. We have no knowledge with respect to the structure of β_0. To get knowledge, we transform the network into a normal form network β_1 which computes a number of subfunctions of C_n. For doing this, we split each output of a gate into several parts. We do this in such a manner that after the transformation, the following normal form property holds.

- On every path P leading from a node e with $op(e) = b_r \in B$ to an output node there exists a node w such that:

  a) the direct successor of w on the path P is an $\wedge$-gate or the output node;

  b) $\exists\, b_s \in B$, $b_s \neq b_r$ and $\exists\, A_s \subseteq A$, $|A_s| \geq 2^q$ such that

$$b_s \wedge \Big(\bigvee_{a_j \in A_s} a_j \Big) \leq res_\beta(w).$$

The normal form transformation enlarges the number of $\wedge$-gates at most by the constant factor 4. During the transformation, we count some $\wedge$-gates.

After the termination of the normal form transformation, we have counted $\lfloor \frac{1}{2}(\frac{n^2}{9} - n) \rfloor$ $\wedge$-gates in B_0 and we are done or at least $n \cdot q$ products $a_i b_j$ are computed on the output gates in B_1 which compute the subfunction $\bar{c}_{i+j}$ of c_{i+j}.

$\Rightarrow$

$\exists$ a_i such that at least q products $a_i b_j$ are computed at those output gates.

Now we first set a_i to 1 and then we set successively all q b_e's to 1. We prove that after every fixing of a b_e at least $\frac{1}{2}$ of $\wedge$-gates are eliminated

$\Rightarrow$

$$\text{in total} \geq \lfloor \tfrac{1}{2} q^2 \rfloor \wedge \text{gates are eliminated}$$

To see this let us consider the computation graph which computes the product $a_i b_e$ at the output node for $\bar{c}_{i+e}$.

On every path P from the node e with $op(e) = b_e$ to the output node u computing $\bar{c}_{i+e}$ consider

the node w of the normal form property.

Assume there are less than $\frac{1}{2}q$ $\wedge$-gates.

$\Rightarrow$

$m < q$ pairwise distinct such nodes w are in the computation graph.

Normal form property $\Rightarrow$

$\exists\ b_{s_1}, b_{s_2}, \dots, b_{s_m} \in B,\ b_{s_j} \neq b_\ell \quad 1 \leq j \leq m$

and

$\exists\ A_1, A_2, \dots, A_m \subseteq A,\ |A_j| \geq 2q \quad 1 \leq j \leq m$

such that

$$a_i \bigwedge_{j=1}^{m} \left(b_{s_j} \wedge \left(\bigvee_{a_t \in A_j} a_t \right) \right) \leq res_{B_1}(u).$$

Definition of u-th degree convolution $\Rightarrow$

$\forall\, b_s \in B$ there exists at most one $a_t \in A$ with

$$a_t b_s \leq c_{i+\ell}.$$

Hence, for $1 \leq j \leq m$, it holds that

$\exists\ a_j' \in A_j$ such that $a_j' \cdot \bigwedge_{i=1}^{m} b_{s_i} \not\leq c_{i+\ell}$

and hence,

$$a_i \bigwedge_{j=1}^{m} a_j' \bigwedge_{i=1}^{m} b_{s_i} \not\leq c_{i+\ell}$$

but, by construction,

$$a_i \bigwedge_{j=1}^{m} a'_j \bigwedge_{i=1}^{n} b_{si} \leq res_\beta(u).$$

But this cannot happen since $\bar{c}_{i+e} = res_\beta(u)$ is a subfunction of c_{i+e}.

$\Rightarrow$ $\geq \frac{1}{2} q$ $\wedge$-gates exist in the computation graph.

By setting a_i and b_e to 1, all these $\wedge$-gates are eliminated.

Now, we shall give the lower bound proof.

The construction of β_1:

Let β_0 be a monotone network computing C_n with $L_\wedge(\beta_0) = \hat{C}_{2m}(C_n)$.

Let $0 < q < \frac{1}{2} n$.

We construct β_1 successively, beginning at the input nodes of β_0; i.e., in each construction step we take a node u in β_0 the direct predecessors of which were constructed in β_1 before.

$u \rightsquigarrow$ small network δ_u with output nodes u' and u''

The input nodes of β_u are the output nodes of δ_v and δ_w where v and w are the direct predecessors of u.

For $0 \leq k \leq 2n-2$, the node in $\mathcal{B}_0$ which computes c_k is denoted by c_k.

An $\wedge$-gate g with $pred(g) = \{v, w\}$ is called a (*)-type-gate if

$$op(v) \in B \text{ and } res_B(w) = \bigvee_{a_j \in A'} a_j$$

where $\emptyset \neq A_j \subseteq A$.

The network $\mathcal{B}_1$ is constructed such that the following holds:

i) $res_{B_1}(u') \vee res_{B_1}(u'') \leq res_{B_0}(u)$.

ii) If $\exists\, b_s \in B$, $A_s \subseteq A$ maximal, $A_s \neq \emptyset$ such that $b_s\left(\bigvee_{a_j \in A_s} a_j\right) \leq res_{B_1}(u')$ and $\bigvee_{a_j \in A_s} a_j \not\leq res_{B_1}(u')$ then $|A_s| \geq 2q$.

iii) On every path P leading from a node e with $op(e) = b_r \in B$ to an $\wedge$-gate g which is not a (*)-type-gate or to the node u'' there exists a node w with

$\exists\, b_s \in B$, $b_s \neq b_r$ and $\exists\, A_s \subseteq A$, $|A_s| \geq 2 \cdot q$ such that

$$b_s \wedge \left(\bigvee_{a_j \in A_s} a_j\right) \leq res_{B_1}(w).$$

Remark:

Property i) means that the output nodes of δ_u compute only subfunctions of $res_{B_0}(u)$.

Property iii) ensures that, after the construction of β_1, the normal form property introduced above holds.

Now we construct δ_u.

Case 1: u is an input node of β_0.

Then δ_u consists of the nodes

$$\begin{cases} u' \text{ with } op(u') = op(u) \\ u'' \text{ with } op(u'') = 0 \end{cases} \quad \text{if } op(u) \in B$$

$$\begin{cases} u' \text{ with } op(u') = 0 \\ u'' \text{ with } op(u'') = op(u) \end{cases} \quad \text{if } op(u) \in A.$$

Obviously, conditions i), ii), and iii) hold after this construction.

Case 2: u is an $\vee$-gate with $pred(u) = \{v, w\}$.

Then δ_u is constructed by

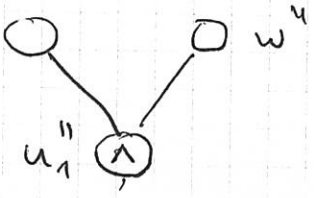
For this construction, we need no $\wedge$-gate. Since properties i), ii) and iii) hold for δ_v and δ_w, these properties also hold for δ_u.

Case 3: u is an $\wedge$-gate with $pred(u) = \{v, w\}$.

We have to realize $v'w'$, $v'w''$, $v''w'$ and $v''w''$!

Step 1: Realization of $v''w''$.

Construct



For the realization of the other three products, we have to take care that the properties ii) and iii) are not destroyed after the construction.

Hence, for v' and w', respectively, we distinguish two cases according to whether the following property is fulfilled or not.

We say for a node $g \in \{v', w'\}$ that g is bipotent if:

$$\exists\, b_s \in B,\ A_s \subseteq A \text{ and } \exists\, b_r \in B,\ b_r \neq b_s,\ A_r \subseteq A$$

such that

$$b_s \wedge \Big(\bigvee_{a_j \in A_s} a_j\Big) \vee b_r \wedge \Big(\bigvee_{a_j \in A_r} a_j\Big) \leq res_{B_1}(g).$$

Remark:

Since property ii) holds for δ_v and δ_w, $|A_s| \geq 2q$ and $|A_r| \geq 2q$.

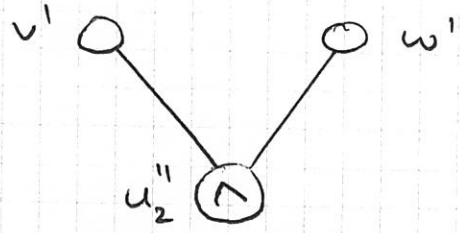
If a node g is not bipotent then either $res_{B_1}(g) = 0$ or $res_{B_1}(g) = b_r \wedge \Big(\bigvee_{a_j \in A_r}\Big)$ for

$b_r \in B$ and $A_r \subseteq A$.

Step 2: Realization of $v'w'$:

a) v' and w' are bipotent.

Construct



b) At least one of v' and w' is not bipotent.

$\Rightarrow$

There exists at most one $b_s \in B$ such that

$$b_s \wedge \left(\bigvee_{a_j \in A_s} a_j \right) \leq res_{\beta_1}(v'w') \quad \text{for } A_s \subseteq A.$$

If no such $b_s \in B$, $A_s \subseteq A$ exist then by the structure of C_u and Theorem 3.2, we can replace $v'w'$ by 0 without changing the functions which are computed and we need not realize the product $v'w'$.

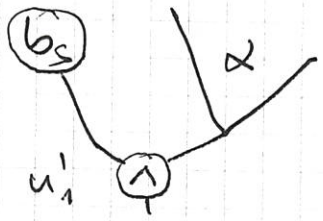
Let $b_s \in B$, $A_s \subseteq A$ maximal such that

$$b_s \wedge \left(\bigvee_{a_j \in A_s} a_j \right) \leq res_{\beta_1}(v'w').$$

We distinguish two cases.

i) $|A_S| \geq 2q$.

Then we construct



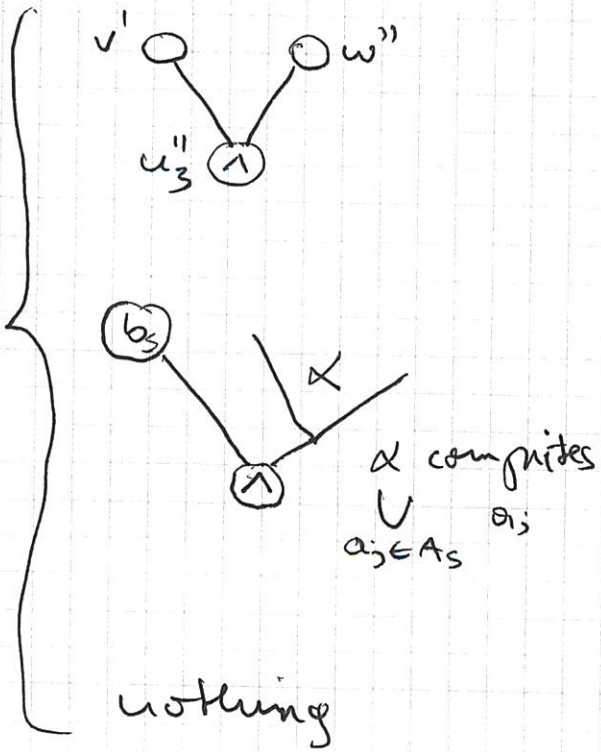
where α is a network which computes $\bigvee_{a_j \in A_S} a_j$ using only $(|A_S| - 1)$ $\vee$-gates.

ii) $|A_S| < 2q$.

Then we do not realize the product $v'w'$. By the structure of C_n, we destroy the computation of $< 2q$ prime implicants.

Step 3: Realization of $v'w''$.

Construct



if v' is bipotent

if v' is not bipotent and $\exists\, b_S \in \mathbb{B}$, $A_S \subseteq A$ maximal with $|A_S| \geq 2q$ such that $b_S \wedge (\bigvee_{a_j \in A_S} a_j) \leq res_{\mathbb{B}}(v'w'')$

otherwise

Step 4 Realization of $v''w'$.

Analogous to that for $v'w''$. Produce result in u''_4 or u'_3.

If in the construction above

$$u'_j \,,\ j \in \{1,2,3\} \quad (u''_j \,,\ j \in \{1,2,3,4\}, \text{ resp.})$$

do not exist then construct

$$u'_j \text{ with } op(u'_j) = 0 \quad (u''_j \text{ with } op(u''_j) = 0, \text{ resp.})$$

- Realization of u' and u'':

$$u' := \bigvee_{j \in \{1,2,3\}} u'_j \qquad u'' := \bigvee_{j \in \{1,2,3,4\}} u''_j .$$

Observe that for the construction of δ_u we need at most four $\wedge$-gates.

Since properties i) - iii) are fulfilled for δ_v and δ_w it follows directly by the construction that these properties are fulfilled for δ_u. exercise.

If we destroy the computation of some prime implicants of C_u then, by construction, for the number N of these prime implicants we have

$$N \leq t \cdot 2^q$$

where $(4-t)$ $\wedge$-gates are used for the realization of the four products $v'w'$, $v'w''$, $v''w'$ and $v''w''$.

The following lemma characterizes the network β_1.

Lemma 3.3

In β_1 the following properties are fulfilled:

(1) For all nodes $u \in \beta_0$ and the output nodes u', u'' of δ_u the following holds:

a) $res_{\beta_1}(u') \vee res_{\beta_1}(u'') \leq res_{\beta_0}(u)$.

b) If $\exists\, b_S \in B$, $A_S \subseteq A$ maximal $A_S \neq \emptyset$ such that $b_S \wedge \left(\bigvee_{a_j \in A_S} a_j \right) \leq res_{\beta_1}(u')$ then $|A_S| \geq 2q$.

(2) For $0 \leq k \leq 2n-2$ the output node c_k' computes 0.

(3) On every path P leading from a node e with $op(e) = b_r \in B$ to an $\wedge$-gate g which is not a $(*)$-type-gate or to c_k'', $k \in \{0, \ldots, 2n-2\}$ there exists a node w such that

$\exists\, b_s \in B$, $b_s \neq b_r$ and $\exists\, A_S \subseteq A$, $|A_S| \geq 2q$ such that

$$b_s \wedge \left(\bigvee_{a_j \in A_S} a_j \right) \leq res_{\beta_1}(w).$$

(4) $0 \leq L_\wedge(\beta_1) \leq 4 \cdot C^{\wedge}_{\Omega_m}(C_n) - m$ where at most $m \cdot 2q$ prime implicants of C_n have been destroyed.

Proof:

From the construction of β_1, (1) and (3) follow directly. Assertion (2) follows from (1) b) and the structure of c_k, $0 \le k \le 2n-2$. As observed above, for each $\wedge$-gate which is not used for the construction of δ_a, at most $2q$ prime implicants are destroyed. Hence, assertion (4) follows.

∎

Using the network β_1, we shall prove the following theorem

Theorem 3.9

$$\hat{C}_{\Omega_m}(C_n) \ge \left\lfloor \frac{1}{8} \min\left\{\left(\frac{n^2}{q} - n\right), q^2\right\} \right\rfloor$$

Setting $q := n^{\frac{2}{3}}$, we obtain the following Corollary.

Corollary 3.2

$$\hat{C}_{\Omega_m}(C_n) \ge \left\lfloor \frac{1}{8}\left(n^{\frac{4}{3}} - n\right) \right\rfloor$$

Proof:

If in (4) of Lemma 3.3 $m \ge \frac{1}{2}\left(\frac{n^2}{q} - n\right)$

then the lower bound is proved. Otherwise, at least $n \cdot q$ prime implicants of C_n remain.

$\Rightarrow$

$\exists\, a_i \in A, \exists\, \tilde{B} \subseteq B$ with $|\tilde{B}| = q$ such that

$$a_i b_e \leq res_{B_1}(C''_{i+e}) \quad \forall\, b_e \in \tilde{B}.$$

Let $\tilde{B} = \{b_{e_1}, b_{e_2}, \ldots, b_{e_q}\}$.

We first fix a_i to 1 and eliminate all superfluous gates.

Observation:

Fixing a_i to 1 does not destroy the normal form property since, if $a_i \in A_s$, the set A_s grows into the whole set A after fixing a_i to 1.

Then successively we set each $b_{e_j} \in \tilde{B}$ to 1 and also eliminate all superfluous gates.

Since fixing an input variable to 1 does not affect the property that one function implies another function, during this process the normal form property is not destroyed.

Now we prove that in each step in which we set a $b_{e_j} \in \tilde{B}$ to 1 at least $\frac{1}{2}q$ $\wedge$-gates are eliminated.

$\Rightarrow$

After the termination of this process we have eliminated at least $\frac{1}{2}q^2$ $\wedge$-gates and the theorem is proved.

Assume, we have constructed the monotone network β_2 from β_1 by setting

$$a_i, b_{e_1}, b_{e_2}, \dots, b_{e_{r-1}} \quad , 1 \le r < q$$

to 1.

Claim

After setting b_{e_r} to 1 at least $\frac{1}{2}q$ more $\wedge$-gates can be eliminated.

Proof of the claim:

Since $a_i b_{e_1} b_{e_2} \dots b_{e_{r-1}} \not\le res_{\beta_1}(c''_{i+e_r})$ and $a_i b_{e_r} \le res_{\beta_1}(c''_{i+e_r})$ the following hold:

i) $res_{\beta_2}(c''_{i+e_r}) \neq 1$,

ii) $b_{e_r} \le res_{\beta_2}(c''_{i+e_r})$.

Let e be the node in β_2 with $op(e) = b_e$
We consider all paths $P_1, P_2, \dots, P_s$ with

a) start node e and end node c''_{i+e_r} and

b) $b_{e_r} \le res_{\beta_2}(v)$ for all nodes v on P_j, $1 \le j \le s$.

Since $b_{e_r} \le res_{\beta_2}(c''_{i+e_r})$, at least one

such a path exists.

Normal form property $\Rightarrow$

On P_j, $1 \le j \le s$ there exists a node w such that

$\exists\, b_{t_j} \in B,\ b_{t_j} \neq b_{e_r},\ \exists\, A_j \subseteq A$ with $|A_j| \geq 2q$

such that $b_{t_j} \wedge \left(\bigvee_{a_p \in A_j} a_p \right) \le res_{\beta_2}(w)$.

It is clear that fixing b_{e_r} at 1 eliminates all $\wedge$-gates on the paths $P_1, P_2, \dots, P_s$.

If on the paths $P_1, P_2, \dots, P_s$ $\frac{1}{2}q$ $\wedge$-gates exist, we are done. Assume that on $P_1, P_2, \dots, P_s$ less than $\frac{1}{2}q$ $\wedge$-gates exist.

We consider the first $\wedge$-gates on the paths $P_1, P_2, \dots, P_s$.

Property (b) $\Rightarrow$

All $\wedge$-gates on the paths are not (*)-type-gates

Normal form property $\Rightarrow$

For every path leading from node e to the first $\wedge$-gate g on a path P_j, $1 \le j \le s$ (to c_{i+e_r} if no $\wedge$-gate on P_j exists)

there exists a node w such that

$\exists\, b_{t_j} \in B,\ b_{t_j} \neq b_{e_r},\ \exists\, A_j \subseteq A$ with $|A_j| \geq 2q$

such that

$$b_{t_j} \wedge \Big(\bigvee_{a_p \in A_j} a_p \Big) \leq res_{\beta_2}(w)$$

and $g \in suc(w)$ ($c''_{iter} \in suc(w)$, resp.)

$\Rightarrow$

$$\bigwedge_{j=1}^{s} \Big(b_{t_j} \wedge \Big(\bigvee_{a_p \in A_j} a_p \Big) \Big) \leq res_{\beta_2}(c''_{iter}).$$

and if no $\wedge$-gate on P_j exists then

$$b_{t_j} \wedge \Big(\bigvee_{a_p \in A_j} a_p \Big) \leq res_{\beta_2}(c''_{iter}).$$

Assume that for all P_j, $1 \leq j \leq s$, an $\wedge$-gate exists on P_j. (If there exists a P_j with no $\wedge$-gate on P_j then the same proof with $s = 1$ works).

Since less than $\frac{1}{2}q$ $\wedge$-gates exist on $P_1, P_2, \ldots, P_s$, less than q of the b_{t_j}, $1 \leq j \leq s$ can be pairwise distinct.

Let

$$B' := \{ b_{t_1}, b_{t_2}, \ldots, b_{t_s}, b_{e_1}, b_{e_2}, \ldots, b_{e_{r-1}} \}.$$

Note that $b_{e_r} \notin B'$. Then we have

$$a_i \bigwedge_{b_j \in B'} b_j \bigwedge_{j=1}^{s} \Big(\bigvee_{a_p \in A_j} a_p \Big) \leq res_{\beta_1}(c'_{iter})$$

Since $\forall b_j \in B'$ there exists at most one $a_d \in A$ with $a_d b_j \leq c_{iter}$

(namely $d = (i+\ell_r) - j$ if $(i+\ell_r) - j \geq 0$)

and since $|B'| < 2q$ and $|A_j| \geq 2q$, $1 \leq j \leq s$ the following holds:

$\exists a_{p_j} \in A_j$ such that $a_{p_j} \bigwedge_{b_j \in B'} b_j \not\leq c_{iter}$.

and hence

$$a_i \bigwedge_{b_j \in B'} b_j \bigwedge_{j=1}^{s} a_{p_j} \not\leq c_{iter}.$$

But by construction

$$a_i \bigwedge_{b_j \in B'} b_j \bigwedge_{j=1}^{s} a_{p_j} \leq res_{B_1}(c''_{iter})$$

and by Lemma 3.3, property (1) a), $res_{B_1}(c''_{iter})$ is a subfunction of c_{iter}, a contradiction.

$\Rightarrow$

On $P_1, P_2, \ldots, P_s$ at least $\frac{1}{2}q$ $\wedge$-gates exist.

Hence, the claim and therefore Theorem 3.9 is proved. ∎

Open problem

For n-th degree convolution reduce the gap between the upper bound n^2 and the lower bound $n^{3/2}$