

3. Monotone Boolean networks

Let $\Omega_m := \{\wedge, \vee\}$ be the monotone base. An Ω_m -network is also called a monotone network. First we mention some fundamental properties of monotone Boolean functions and monotone networks.

Lemma 3.1

Each prime implicant of a monotone Boolean function $f \in M_n$ contains only non-negated variables.

Proof:

exercise

■

Exercise

Show that both constant functions are monotone.

Theorem 3.1

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

Proof

exercise

■

By an estimation of the number of n -ary monotone Boolean functions and an application of Shannon's counting argument, it can also be proved that nearly all monotone Boolean functions have exponential (monotone) network complexity (see Wegner pp. 98-105).

But with respect to single output monotone Boolean functions, no technique for counting a super-linear number of gates in a monotone network 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 $4 \cdot n$.

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

For functions in $M_{n,m}$ where $m = \Theta(n)$, super-linear lower bounds have been proved since 1969. All these lower bound proofs use the following property of each monotone network which computes a function $f = (f_1, f_2, \dots, f_m) \in M_{n,m}$ which is based on the DNF-representation

$$\text{res}_\beta(g) = \bigvee_{j=1}^t m_j, \quad m_j \text{ is a monomial}$$

of the function computed at the node g , $g \in \beta$.

Exercise

Let β be a monotone network. Give an algorithm which computes for all nodes g in β the DNF-representation of $\text{res}_\beta(g)$. What is the time used by your algorithm?

- Let u_i , $1 \leq i \leq m$ be the node in β which computes the function f_i . For $\text{res}_\beta(u_i) = \bigvee_{j=1}^{t_i} m_j$, 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 each prime implicant p of f_i there is a $j \in \{1, 2, \dots, t_i\}$ with $m_j = p$.

If the first property is not fulfilled then there is an input $a = (a_1, a_2, \dots, a_n) \in \{0, 1\}^n$ such that $f_i(a) = 0$ but $\text{res}_\beta(u_i)(a) = 1$. If the second property is not fulfilled then there is an input $a \in \{0, 1\}^n$ such that $f_i(a) = 1$ but $\text{res}_\beta(u_i)(a) = 0$.

Exercise

Let $\text{res}_\beta(g) = \bigvee_{j=1}^t m_j$ and $f \in M_n$. Assume that there is a prime implicant p of f such that $p \neq m_j$ for $1 \leq j \leq t$. Prove that there is an input $a = (a_1, a_2, \dots, a_n) \in \{0, 1\}^n$ with $f(a) = 1$ but $\text{res}_\beta(g)(a) = 0$.

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

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

Reference

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

For $f, g \in B_n$, we define

$$f \leq g \Leftrightarrow f \wedge g = f.$$

Then f is called a subfunction of g .

Theorem 3.2

Let $f, g \in M_n$ and $t \in \text{PIM}(g)$ where $tt' \notin \text{PIM}(f)$ for all monomials t' (including the empty monomial). Let h be defined by $\text{PIM}(h) = \text{PIM}(g) \setminus \{t\}$. 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 $a \in \{0, 1\}^n$ 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

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

By construction, $g = h \vee t$. Hence, $t(a) = 1$.

Consider $t^* \in \text{PIM}(f)$ with $t^*(a) = 1$. Since $f(a) = 1$, such a prime implicant of f exists.

Claim: $\exists$ monomial t' with $tt' = t^*$.

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

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

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

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

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

Since x_i has been chosen to be any variable in t it follows that t is a submonomial of t^* .

$$\Rightarrow \exists t' \text{ with } tt' = t^*$$

□

By the choice of t , t is not a submonomial of any prime implicant of f , a contradiction.

■

Theorem 3.3

Let β be a monotone network computing f and let v be a gate in β with $\text{res}_\beta(v) = g$. Let t, t_1 and t_2 be monomials such that

- i) $tt_1, tt_2 \in \text{IM}(g)$ and
- ii) $\forall$ monomials t' : $t'tt_1, t'tt_2 \in \text{IM}(f) \Rightarrow t't \in \text{IM}(f)$

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' .

$$g \leq h \Rightarrow f \leq f'$$

Suppose that $f' \neq f \Rightarrow$

$$\exists a \in \{0,1\}^n: f'(a) = 1 \text{ and } f(a) = 0.$$

$\Rightarrow$

$$h(a) = 1, g(a) = 0 \Rightarrow t(a) = 1.$$

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

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

To get a contradiction, it suffices to prove

$$t^*tt_1, t^*tt_2 \in \text{IM}(f).$$

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

$$t^* \in \text{PIM}(f') \Rightarrow f'(b) = 1.$$

$$t t_j \in \text{IM}(g) \Rightarrow g(b) = 1.$$

$$\text{Hence, } g \leq h \Rightarrow h(b) = 1.$$

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

$$\Rightarrow f(b) = f'(b) = 1 \Rightarrow t^* t t_j \in \text{IM}(g), j=1,2.$$

■

Now we consider explicitly defined functions in $M_{n,m}$.

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.
- Hugo 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 disjunction of a set of variables. We consider the monotone network complexity of sets of Boolean sums

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

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 $i_0, i_1, \dots, i_h$

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

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 $f : \{0, 1\}^n \rightarrow \{0, 1\}^m$ of Boolean sums by a bipartite graph

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

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

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

28.04.

For an explicitly constructed set of Boolean sums in $M_{n,m}$, Neciporuk has proved an $\Omega(n^{3/2})$

lower bound for its monotone complexity in 1969. This was the first super-linear lower bound for an explicitly defined function in $M_{n,m}$ where $m = O(n)$. 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 the set of bipartite graphs 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 to 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 present Mehlhorn's proof of the following theorem.

Theorem 3.4

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

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

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 $\max\{n-2, k-2\}$.

Lemma 3.1

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

$$C_{\Sigma \vee}(f) \leq \max\{1, n-2, k-2\} \cdot C_{\Sigma_m}(f).$$

Proof:

Let β be an optimal Σ_m -network for f . Suppose that β contains s $\vee$ -gates and t $\wedge$ -gates; i.e., $C_{\Sigma_m}(f) = s+t$.

The idea is, to eliminate successively the $\wedge$ -gates using at most $k-1$ and $n-1$, 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 $n-2$, respectively additional gates.

More precisely, we construct a sequence $\beta_0, \beta_1, \dots, \beta_t$ of monotone networks where β_i , $0 \leq i \leq t$ contains $t-i$ $\wedge$ -gates and $\leq s + \max\{1, n-1, k-1\} \cdot i$ $\vee$ -gates.

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

Construction of β_{i+1} :

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

Let $\text{res}_{\beta_i}(v) = g$ and let g_1 and g_2 be the functions computed at the ingoing edges of v . Let

$$g = x_{d_1} \vee x_{d_2} \vee \dots \vee x_{d_p} \vee t_1 \vee t_2 \vee \dots \vee t_q$$

where each x_{d_j} is a variable and each t_j is a monomial of length at least two be the DNF-representation of $\text{res}_{\beta_i}(v)$.

We distinguish two cases.

Case 1: $p \leq k$.

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

$$x_{d_1} \vee x_{d_2} \vee \dots \vee x_{d_p}$$

using $p-1 \leq k-1$ additional $\vee$ -gates and saving the $\wedge$ -gate v . The resulting network is β_{i+1} .

Case 2: $p > k$

Let $f_{r_1}, f_{r_2}, \dots, f_{r_k}$ be the output functions which depend on $g = \text{res}_{\beta_i}(v)$.

By construction, between v and f_{r_j} , $1 \leq j \leq \ell$ there are only v -gates. Hence, we can write for $1 \leq j \leq \ell$

$$f_{r_j} = g \vee u_j.$$

Since f_{r_j} is a Boolean sum, u_j is not the constant 1. Furthermore, for $1 \leq j \leq \ell$

$$\{d_1, d_2, \dots, d_p\} \subseteq F_{r_j}$$

Since f is (h, k) -disjoint, $p > k$ implies $\ell \leq h$.

Claim:

For $1 \leq j \leq \ell$ either $f_{r_j} = g_1 \vee u_j$ or $f_{r_j} = g_2 \vee u_j$.

Proof of claim

Note that ($g = g_1 \wedge g_2$ and $f_{r_j} = g \vee u_j$).

$$\Rightarrow f_{r_j} \leq g_1 \vee u_j \quad \text{and} \quad f_{r_j} \leq g_2 \vee u_j.$$

Suppose that

$$f_{r_j} < g_1 \vee u_j \quad \text{and} \quad f_{r_j} < g_2 \vee u_j.$$

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

$$\bullet f_{r_j}(\alpha_1) = 0 \quad \text{but} \quad (g_1 \vee u_j)(\alpha_1) = 1 \quad \text{and}$$

$$\bullet f_{r_j}(\alpha_2) = 0 \quad \text{but} \quad (g_2 \vee u_j)(\alpha_2) = 1.$$

Let $\alpha_1 = (a_1, a_2, \dots, a_n)$ and $\alpha_2 = (b_1, b_2, \dots, b_n)$.

Consider

$$\alpha = (c_1, c_2, \dots, c_n) \quad \text{where} \quad c_j := \max\{a_j, b_j\}, \quad 1 \leq j \leq n.$$

Since f_{r_j} is a Boolean sum and $f_{r_j}(x_1) = f_{r_j}(x_2) = 0$ (67)
there holds

$$f_{r_j}(x) = 0.$$

Since $g_1 \vee u_j$ and $g_2 \vee u_j$ are monotone and

$$(g_1 \vee u_j)(x_1) = (g_2 \vee u_j)(x_2) = 1$$

there holds

$$(g_1 \vee u_j)(x) = (g_2 \vee u_j)(x) = 1.$$

$\Rightarrow$

$$u_j(x) = 1 \text{ or } (g_1(x) = g_2(x) = 1 \text{ and hence, } g(x) = 1).$$

In both cases, we obtain

$$f_{r_j}(x) = (g \vee u_j)(x) = 1,$$

a contradiction.

□

β_{i+1} is constructed from β_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 f_{r_j} , $1 \leq j \leq l$ computes the function u_j .

(2) For each f_{r_j} , $1 \leq j \leq l$ use one $\vee$ -gate to compute $u_j \vee g_k$, $k \in \{1, 2\}$ where

$$f_{r_j} = g_k \vee u_j.$$

This adds $\ell \leq h$ v -gates. Since one v -gate is saved, we need at most $\ell-1 \leq h-1$ additional v -gates. The resulting network is β_{i+1} .

This proves the lemma. ■

Now we prove a lower bound for $C_{\{v\}}(f)$. All gates in an $\{v\}$ -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.

Lemma 3.2

Let $f: \{0,1\}^n \rightarrow \{0,1\}^m$ be a (h,k) -disjoint set of Boolean sums. Then
$$C_{\{v\}}(f) \geq \sum_{i=1}^m \frac{1}{h} \left(\left\lceil \frac{|F_i|}{k} \right\rceil - 1 \right).$$

Proof:

Let β be an optimal $\{v\}$ -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 (h,k) -disjoint, at most h output nodes f_i depend on v .

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

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

small gates.

For each gate v in β let $nc(v)$ denote the number of output nodes f_i which depend on v . If v is large then $nc(v) \leq k$. Hence,

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

Consider the set H_i of all large gates in β which are connected with the output node which computes f_i , $1 \leq i \leq m$. As observed above, H_i has to connect at least $\lceil \frac{|F_i|}{k} \rceil$ different small nodes with the output node f_i . Hence, this subnetwork must contain a binary tree with $\lceil \frac{|F_i|}{k} \rceil$ leaves.

$\Rightarrow$

$$\begin{aligned} \sum_{\text{large } v \in \beta} nc(v) &= \sum_{i=1}^m |H_i| \\ &\geq \sum_{i=1}^m \left(\lceil \frac{|F_i|}{k} \rceil - 1 \right). \end{aligned}$$

Hence we obtain

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

This implies

$$C_{\{v\}}(f) \geq \frac{1}{k} \cdot \sum_{i=1}^m \left(\lceil \frac{|F_i|}{k} \rceil - 1 \right).$$

Proof of Theorem 3.4:

$$\begin{aligned} C_{\Sigma_m}(f) &\stackrel{\text{le. 3.1}}{\geq} \frac{1}{\max\{1, k-2, k-2\}} C_{\{v\}}(f) \\ &\stackrel{\text{le. 3.2}}{\geq} \sum \frac{\frac{|F_i|}{k} - 1}{k \cdot \max\{1, k-2, k-2\}}. \end{aligned}$$

Nick Pippenger has given another proof for Theorem 3.4. It would be interesting to read Pippenger's proof and to compare both proofs with respect to the used properties of the function f .

Exercise

Read Pippenger's paper and compare both proofs.

In 1966, 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 $f: \{0,1\}^n \rightarrow \{0,1\}^m$ with $\sum_{i=1}^n |F_i| = \Omega(n^{5/3})$. Theorem 3.4 implies an $\Omega(n^{5/3})$ lower bound for the monotone network complexity of this function.

Further constructions

- A.E. Andreev, On a family of Boolean matrices, Moscow Univ. Math. Bull. 41 (1986), 78-82.
- J. Kollár, L. Rónyai, T. Szabó, Norm-graphs and bipartite Turán numbers, Combinatorica 16 (1996), 399-406.

give sets of Boolean sums $f: \{0,1\}^n \rightarrow \{0,1\}$ with $\Omega(n^{2-\varepsilon})$ monotone network complexity for arbitrarily small $\varepsilon > 0$. Especially, Kollár et al. have constructed graphs in $G_2(n, n)$ which contain $\geq n^{2-\frac{1}{t}}$ edges and no $K_{t, (t+1)}$ where $t \geq 2$ is an integer.

06.05.

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 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 proved that the school-method is the unique optimal monotone network for Boolean matrix multiplication. We present their proof.

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

$$C := A \cdot B$$

is an $r \times q$ 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 an $(r \times p)$ Boolean matrix and 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: The isolation of an $\wedge$ -gate which computes $a_{ij} b_{jk}$ for each triple (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 .

$\Rightarrow$

$\exists$ gate v in β such that

- P does not hold for any ingoing edge of v but P holds for $\text{res}_\beta(v)$.

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

$\leadsto$

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

$$a_{ik} b_{jk}, \quad 1 \leq i \leq r, \quad 1 \leq k \leq q$$

and elimination of these gates by setting

$$\begin{aligned} a_{ik} &\text{ to } 1, & 1 \leq i \leq r \\ b_{jk} &\text{ to } 0, & 1 \leq k \leq q. \end{aligned}$$

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 $\text{res}_\beta(v)$ 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

$$P_{ik}(v) \Leftrightarrow a_{i1} b_{1k} \leq h \wedge a_{i1} \not\leq h \wedge b_{1k} \not\leq h.$$

This implies that $a_{i1} b_{1k}$ 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 show that v is an $\wedge$ -gate. Suppose that v is an v -gate. Then

- $a_{i1} b_{1k} \leq h$ implies $a_{i1} b_{1k} \leq h_1$ or $a_{i1} b_{1k} \leq h_2$.

W.l.o.g., we can assume $a_{i1} b_{1k} \leq h_1$.

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

$$a_{i1} \leq h_1 \text{ or } b_{1k} \leq h_1$$

and hence, $a_{i1} \leq h$ or $b_{1k} \leq h$

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

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

Since $a_{i_1} b_{1k_2} \leq n$ there hold $a_{i_1} b_{1k_2} \leq h_1$ and $a_{i_1} b_{1k_2} \leq h_2$.
Furthermore, $\neg P_{i_1 k_2}(h_1) \wedge \neg P_{i_1 k_2}(h_2) \Rightarrow$

Either $a_{i_1} \leq h_1 \wedge b_{1k_2} \leq h_2$ or
 $a_{i_1} \leq h_2 \wedge b_{1k_2} \leq h_1$.

Note that $a_{i_1} \leq h_1 \wedge a_{i_1} \leq h_2 \Rightarrow a_{i_1} \leq h$ and
hence, $\neg P_{i_1 k_2}(h)$. Analogously, we can exclude
 $b_{1k_2} \leq h_1 \wedge b_{1k_2} \leq h_2$.

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

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

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

Up to symmetry, one of the following two situations
occurs:

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

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

Situation 1:

Suppose $i_1 \neq i_2$. Then for the input function h_1 and
 $t = 1$, $t_1 = a_{i_1}$ and $t_2 = a_{i_2}$,
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 $k_1 \neq k_2$. This case can be excluded in the same way using h_2 , $t=1$, $t_1 = b_{1k_1}$ and $t_2 = b_{1k_2}$.

Situation 2:

By application of Theorem 3.3, we can replace both inputs of v by 1.

Exercise

Show that in Situation 2, both inputs of v can be fixed at 1 without changing the function computed by β .

This contradicts the optimality of β as well.

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

$\Rightarrow$

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

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

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

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

- $a_{i1}b_{1k} \in \text{PIM}(c_{ik}) \Rightarrow a_{i1}b_{1k} \leq c_{ik}$.
- Each prime implicant of c_{ik} contains the variable b_{1k} or a variable in $\{a_{i2}, a_{i3}, \dots, a_{ip}\}$.
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 by the definition of c_{ik} .

Hence, $I(Q_{i2}) \neq \emptyset$ if $p \geq 2$.

Exercise

Prove the following two assertions:

- If $v \in I(Q_{ik})$ then v is an v -gate and either $a_{i1} \leq b_{1k}$ or $a_{i2} \leq b_{1k}$.
- The sets $I(Q_{ik})$ are pairwise disjoint.

Altogether, we have achieved the following:

- For each pair (i, k) , we have located an n -gate in β with one of its input functions has prime implicant a_{i1} . For different pairs, we have identified different n -gates.

$\Rightarrow$

fixing a_{i1} at 1 for $1 \leq i \leq r$ eliminates r.g n -gates in β .

- If $p > 1$ then we have located an v -gate for each pair (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 $1 \leq k \leq q$ eliminates $r \cdot q$ v -gates in β .

Exercise

Show that fixing a_{ij} at 1 for $1 \leq i \leq r$ and b_{1k} at 0 for $1 \leq k \leq q$ transforms β into an optimal monotone network for the functions

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

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 and 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.

(45)

In 1978, Wegener has generalized this to the "direct product" of m $M \times N$ -matrices $X_1, X_2, \dots, X_m$. 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_{he}^i denote the element of the matrix M_i at position (h, e) . Then we say that x_{he}^i is a variable of type e . This means that the type of a variable is its position in the corresponding row.

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

$$y_{h_1 h_2 \dots h_m} := \bigvee_{1 \leq e \leq N} x_{h_1 e}^1 x_{h_2 e}^2 \dots x_{h_m e}^m.$$

We say that the prime implicant

$$(h_1, h_2, \dots, h_m, e) := x_{h_1 e}^1 x_{h_2 e}^2 \dots x_{h_m e}^m$$

is of type e .

The generalized Boolean matrix product f_{MN}^m , $m, M \geq 2$ is defined as follows:

$$f_{MN}^m : \{0, 1\}^{mMN} \rightarrow \{0, 1\}^{M^m}$$

where

$$y_{h_1 h_2 \dots h_m} := \bigvee_{1 \leq e \leq N} x_{h_1 e}^1 x_{h_2 e}^2 \dots x_{h_m e}^m.$$

First we prove an upper bound for the monotone network complexity of f_{MN}^m .

Theorem 3.6

$$C_{\wedge_m}(f_{NM}^m) \leq N \cdot M^m (2 + (M-1)^{-1}) \leq 3 \cdot N \cdot M^m$$

Proof:

Assume that all monomials $(h_1, h_2, \dots, h_{i-1}, e)$ are computed. Then $N \cdot M^i$ additional $\wedge$ -gates suffice to compute all monomials $(h_1, h_2, \dots, h_i, e)$.

$\Rightarrow$

$$\begin{aligned} C_{\wedge}(f_{NM}^m) &\leq N \cdot \sum_{2 \leq i \leq m} M^i \\ &\leq N \cdot (M^{m+1} - 1) (M-1)^{-1} \\ &\leq NM^m (1 + (M-1)^{-1}) \end{aligned}$$

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 in a monotone network which computes f_{NM}^m . First we investigate the structure of optimal monotone networks computing f_{NM}^m .

Lemma 3.3

Let g be a function which is computed at a gate in a monotone network for f_{NM}^m . Let

$$(i_1, i_2, \dots, i_m, e), (j_1, j_2, \dots, j_m, e) \in \text{PIM}(g)$$

for some $e \in \{1, 2, \dots, N\}$ and $i_k, j_k \in \{1, 2, \dots, M\}$,

$1 \leq k \leq m$. Let $A := \{k \mid i_k = j_k\}$ and let

$t := \bigwedge_{k \in A} x_{i_k j_k}^k$. Then g can be replaced by

$h := g \vee t$ and the resulting network still computes f_m .

Proof:

Let

$$t_1 := \bigwedge_{k \notin A} x_{i_k j_k}^k \quad \text{and} \quad t_2 := \bigwedge_{k \notin A} x_{j_k i_k}^k.$$

Definition of $A \Rightarrow$

t_1 and t_2 have no common variable.

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

By construction

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

To prove the second assumption, suppose that this assumption does not hold; i.e.,

There is a monomial t' and a output function $y_{h_1, h_2, \dots, h_m}$ with

- $t'tt_1, t'tt_2 \in \text{IM}(y_{h_1, h_2, \dots, h_m})$ but $t't \notin \text{IM}(y_{h_1, \dots, h_m})$

Then

$t'tt_1$ contains a prime implicant $(h_1, h_2, \dots, h_m, l')$ but $t't$ does not contain this prime implicant.

By construction, it holds $l' = l$.

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

Altogether, we obtain

- $(h_1, h_2, \dots, h_m, \ell)$ 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 have a variable in common.

But this contradicts the definition of A . ■

A monomial m is useful for an output of f^m if m is a submonomial of a prime implicant of that output.

Lemma 3.3 $\Rightarrow$

- If the function $g = \text{res}_\beta(u)$ includes several useful monomials of type ℓ then we can replace g in β by $g \vee t$ where t is the common part of all useful monomials of type ℓ .

Goal:

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 $\ast$ -network. $C_{\Omega_m}^*$ is the associated complexity measure.

Given any $\ast$ -network β for f_{MN}^m , we wish to transform β into a so-called standard $\ast$ -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 $g := \text{res}_\beta(u)$.

For $1 \leq l \leq N$ let

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

Without additional cost, we replace

g by $g \vee t_1 \vee t_2 \vee \dots \vee t_N$.

$\leadsto$

Altogether, we obtain a $\ast$ -network β' for f_{MN}^m such that

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

Theorem 3.7

Let $m \geq 2$. Then

$$C_{\Sigma_m}(f_{MN}^m) \geq C_{\Sigma_m}^*(f_{MN}^m) > \frac{1}{2} N \cdot M^m.$$

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 $h_n := f_{M(n)N(n)}^{m(n)}$. Then the function h_n depends on at most n variables and has at most n output functions. Furthermore,

$$C_{\Sigma_m}(h_n) = \Omega\left(\frac{n^2}{\log n}\right).$$

Proof:

Exercise

Proof of Theorem 3.7:

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

Goal:

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

$$c(v) : \text{PIM}(f_{MN}^m) \rightarrow [0, 1]$$

Such that

$$c(v) := \sum_{1 \leq h_1, h_2, \dots, h_m \leq M} \sum_{1 \leq e \leq N} c_v(h_1, h_2, \dots, h_m, e) \leq 1.$$

Properties:

- a) $c_v(h_1, h_2, \dots, h_m, l)$ is an estimation of the contribution of the $\wedge$ -gate v to the computation of the prime implicant $(h_1, h_2, \dots, h_m, l)$.
- b) Each gate contributes to all prime implicants at most the value one.

$\Rightarrow$

For an optimal $\wedge$ -network β there holds

$$c(\beta) := \sum_{v \wedge\text{-gate in } \beta} c(v) \leq C_{\Omega_m}^*(f_{MN}^m).$$

This means that $c(\beta)$ is a lower bound for $C_{\Omega_m}^*(f_{MN}^m)$.

Claim 1:

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

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

Claim 1 $\Rightarrow$

$$\begin{aligned} C_{\Omega_m}^*(f_{MN}^m) &\geq \sum_{1 \leq h_1, h_2, \dots, h_m \leq M} \sum_{1 \leq l \leq N} c(h_1, \dots, h_m, l) \\ &> \frac{1}{2} N \cdot M^m. \end{aligned}$$

Definition of the value function

Let v be an $\wedge$ -gate in an optimal standard $\ast$ -network β for f_{MN}^m . Let $g := \text{res}_\beta(v)$ and g', g'' be the functions of the incoming edges of v .

Idea

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

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

Question: Which value > 0 should be chosen?

To define these values let

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

$\exists t_e \in \text{PIM}(f_{MN}^m)$ of type i_e with $t_e \in \text{PIM}(g)$ but $t_e \notin \text{PIM}(g')$.

Furthermore, let

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

$\exists t_e \in \text{PIM}(f_{MN}^m)$ of type j_e with $t_e \in \text{PIM}(g)$ but $t_e \notin \text{PIM}(g'')$.

Then we define for $t \in \text{PIM}(f_{MN}^m)$

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

$$c_v''(t) := \begin{cases} \frac{1}{2q''} & \text{if } t \in \text{PIM}(g) \text{ and } t \notin \text{PIM}(g'') \\ 0 & \text{otherwise} \end{cases} \quad (26)$$

Then $c_v(t)$ is defined by

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

Property (*) $\Rightarrow$

$$\begin{aligned} c'(v) &= \sum_{1 \leq h_1, \dots, h_m \leq M} \sum_{1 \leq l \leq N} c_v'(h_1, h_2, \dots, h_m, l) \\ &= q' \cdot \frac{1}{2 \cdot q'} = \frac{1}{2}. \end{aligned}$$

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

Analogously,

$$\begin{aligned} c''(v) &= \sum_{1 \leq h_1, \dots, h_m \leq M} \sum_{1 \leq l \leq N} c_v''(h_1, h_2, \dots, h_m, l) \\ &= q'' \cdot \frac{1}{2q''} = \frac{1}{2}. \end{aligned}$$

$\Rightarrow$

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

It remains to prove Claim 1.

Proof of Claim 1:

Consider the prime implicant $t := (h_1, h_2, \dots, h_m, l)$ and the corresponding output $y_t := y_{h_1 h_2 \dots h_m}$

Let $\beta(t)$ be that subnetwork of β which contains the following gates and inputs:

- gate v is in $\beta(t)$ iff
there is a path P in β from v to y_t and t is a prime implicant of all functions computed on P (inclusive $\text{res}_\beta(v)$).
- the inputs of the gates in $\beta(t)$ are contained in $\beta(t)$ as well.

Properties:

- 1) For each input function g of $\beta(t)$ holds $t \notin \text{PIM}(g)$
- 2) $t \in \text{PIM}(\text{res}_\beta(v))$ 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 $\beta(t)$. Then, v is an $\wedge$ -gate (otherwise $t \notin \text{PIM}(\text{res}_\beta(v))$).
- 4) If an input of $\beta(t)$ is input of an $\wedge$ -gate $\beta(t)$ then a proper shortening of t is a prime implicant of that input.

Let $s_1, s_2, \dots, s_D$ 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) $\Rightarrow c^*(vci) > 0$

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

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

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

To prove $\sum_{i=1}^D b_i > 1/2$ choose $w_i \in \text{PIM}(s_i)$ such that

- a proper prolongation w_i^* of w_i is contained in $\text{PIM}(\text{res}_B(vci)) \cap \text{PIM}(f_{\text{fin}}^u)$ and
- the type of w_i^* is different to the types of $w_1^*, w_2^*, \dots, w_{i-1}^*$.

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 $i \leq D$.

$\Rightarrow$

$c^*(vci)$ is positive for $\leq (i-1)$ prime implicants

Hence, by the definition of the value function

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

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

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

Case 2: $\neg$ Case 1.

Construction $\Rightarrow w_i \in \text{PIM}(s_i)$ for $1 \leq i \leq D$.

$$\Rightarrow w_1 w_2 \dots w_D \leq s_1 s_2 \dots s_D.$$

Construction of $\beta(t) \Rightarrow$

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

$$y_t(a) = 1.$$

(This can easily be shown by induction.)

$$\Rightarrow w_1 w_2 \dots w_D \leq y_t.$$

All variables in w_i are of type l_i and $l_1, l_2, \dots, l_D$ are pairwise different.

Each w_i is a proper shortening of w_i^* and $c^*(c)$.

$\Rightarrow w_i$ contains $\leq m-1$ variables

$$\Rightarrow w_1 w_2 \dots w_D \in \text{IM}(y_t)$$

a contradiction

Hence, Case 2 cannot occur.

This finishes the proof of the theorem.

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.
- M.I. Grinchuk, I. S. Sergeev, Twin circulant matrices and lower bounds on the complexity of some Boolean operators, arXiv: 1701.08557 v1 [cs.CC] 30. Jan 2017.

Let $A = \{a_0, a_1, \dots, a_{n-1}\}$, $B = \{b_0, b_1, \dots, b_{n-1}\}$ be two disjoint sets of n variables. Then the n -th degree convolution C_n is defined by

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

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

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 $A := \{a_0, a_1, \dots, a_{p-1}\}$ and $B = \{b_0, b_1, \dots, b_{q-1}\}$ be two disjoint sets of variables. A monotone function

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

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

Example:

Boolean matrix multiplication and also the Boolean convolution are sets of bilinear forms.



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 semidisjoint bilinear forms if f has the following properties:

$$1) \text{ PIM}(f_i) \cap \text{PIM}(f_j) = \emptyset \text{ for } 1 \leq i < j \leq m.$$

2) For $1 \leq k \leq m$, each variable in $A \cup B$ is contained in at most one prime implicant in $\text{PIM}(f_k)$.

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

For $a, b \in \text{PIM}(f)$, we define inductively the following subset $\text{PIM}_{ij}(f)$ of $\text{PIM}(f)$:

- a) $a, b \in \text{PIM}_{ij}(f)$,
- b) $a, b \in \text{PIM}_{ij}(f) \Rightarrow a, b \in \text{PIM}_{ij}(f)$ for all $a, b \in \text{PIM}(f)$

and also $a, b \in \text{PIM}_{ij}(f) \wedge a, b \in \text{PIM}(f)$.

A semidisjoint set of bilinear forms is disjoint if it fulfills also the following third property:

- 3) For $1 \leq k \leq m$, $0 \leq i, j \leq n-1$ there holds $|\text{PIM}_{ij}(f_k) \cap \text{PIM}(f_k)| \leq 1$.

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

Exercise:

- a) Give a formal definition of multilinear forms. Extend the definitions of semidisjointness and disjointness to multilinear forms.
- b) Show that the generalized Boolean matrix multiplication is a set of disjoint multilinear forms.
- c) Show that the convolution is not a set of disjoint bilinear forms.

Boolean sums for which we have proved a lower bound are (n, k) -disjoint. 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 a disjointness property. The set of variables upon which two functions $c_1, c_2 \in C_n$ 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 prove a general lower bound for the monotone network complexity of ^(sets of) semidisjoint bilinear forms given 1982 by Jürgen Weiß in his diploma thesis.

Theorem 3.8

Let f be a set of semidisjoint bilinear forms. Let r_i be the number of prime implicants which contain the variable a_i . Then

$$C_{\Sigma_m}(f) \geq \sum_{i=0}^{p-1} r_i^{1/2}.$$

Corollary 3.2

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

(94)

Proof:

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

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 also semidisjoint. Moreover, the values r_i , $i > 0$ do not change.

$\Rightarrow$

It suffices to prove that after setting a_0 to 0, at least $r_0^{1/2}$ gates have been eliminated.

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

$\Rightarrow$

Setting a_0 to 0 eliminates these s_0 n -gates where these outputs are computed.

Since f is semidisjoint, these gates cannot be used for the computation of the other outputs in an optimal Ω_m -network for f .

Claim

Setting a_0 to 0 eliminates at least $(r_0 - s_0)^{1/2}$ v -gates

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

We identify the output node and the function which is computed at the output node. Let

$$\mathcal{F} := \{ f_z \mid \exists d: a_0 b_d \in \text{PIM}(f_z) \wedge |\text{PIM}(f_z)| \geq 2 \}.$$

Consider any path $P = v_0, v_1, \dots, v_m$ from a_0 to f_z where $f_z \in \mathcal{F}$. Then, the semidisjointness of f and $|\text{PIM}(f_z)| \geq 2$ imply that P contains at least one v -gate v_e with $a_i b_j \leq \text{res}_B(v_e)$ for an $i \neq 0$ and any $j \in \{0, 1, \dots, q-1\}$.

Exercise

Show that the property " P contains no v -gate v_e with $a_i b_j \leq \text{res}_B(v_e)$ for an $i \neq 0$ and some j " implies that f_z does not contain two prime implicants.

The first such an v -gate on P is called suitable for P . Let

$$\mathcal{P} := \{ P \mid P \text{ is a path from } a_0 \text{ to an } f_z \in \mathcal{F} \}.$$

Let

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

Consider any $v \in V^*$. By construction, for each gate w between a_0 and v , each prime implicant p of $\text{res}_B(w)$ has the following property:

- p contains at least one of the following monomials as a submonomial

- (*) - a_0 ,
- the conjunction of two variables in A ,

- the conjunction of two variables in $\mathcal{B}$.

Exercise

Give a formal proof that for each gate w between a_0 and v ($\neq v$) each prime implicant p of $\text{res}_{\beta}(w)$ has property (*).

$\Rightarrow$

After setting a_0 to 0 and eventual applications of Theorem 3.2, $\text{res}_{\beta}(w)$ can be replaced by 0.

$\Rightarrow$

After setting a_0 to 0, each gate $v \in V^*$ can be eliminated.

Claim: $|V^*| \geq (\tau_0 - s_0)^{\frac{1}{2}}$

To prove this claim let

$$V^* = \{v_1, v_2, \dots, v_D\}.$$

Then there are $i_1, i_2, \dots, i_D$, $i_\ell \neq 0$ for $1 \leq \ell \leq D$ and $j_1, j_2, \dots, j_D$ with

$$a_{i_\ell} b_{j_\ell} \leq \text{res}_{\beta}(v_\ell), \quad 1 \leq \ell \leq D$$

Consider any $f_k \in \mathcal{F}$. Let $a_0 b_{d(k)} \in \text{PIM}(f_k)$. Let β' be the subnetwork of β which constructs the prime implicant $a_0 b_{d(k)}$ at the output node f_k .

On each path P from a_0 to f_k in β' there is a node $v_\ell \in V^*$ instead of using the input function

of v_k on P one can use the other input function.

$\Rightarrow$

$$b_{d(k)}, a_{i_1} a_{i_2} \dots a_{i_D} b_{j_1} b_{j_2} \dots b_{j_D} \leq f_k.$$

$$\Rightarrow \exists \ell, \ell' \in \{1, 2, \dots, D\} \text{ with } a_{i_\ell} b_{j_{\ell'}} \in \text{PIM}(f_k).$$

Since f is semidisjoint each $a_{i_\ell} b_{j_{\ell'}}$, $1 \leq \ell, \ell' \leq D$ can be a prime implicant of at most one function in $\mathcal{F}$. Note that $|\mathcal{F}| = r_0 - s_0$.

$\Rightarrow$

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

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

This proves the claim. Now, the assertion can be proved using induction. □

We can use the proof of Theorem 3.8 to prove an $\Omega(n^{3/2})$ lower bound for the number of v -gates in any monotone network which computes the n -th degree convolution.

Exercise:

Show that each monotone Boolean network computing the n -th degree convolution contains $\Omega(n^{3/2})$ v -gates.

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 which computes C_n . We have no knowledge about the structure of β_0 .

Idea:

Transform the network β_0 into a monotone network β_1 such that

- i) β_1 computes a number of subfunctions of C_n and
- ii) we know enough about the structure of β_1 such that we can show that either β_1 contains a large number of $\wedge$ -gates or we have eliminated a large number of $\wedge$ -gates during the transformation of β_0 to β_1 .

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 an input node u with $op(u) = b_r \in B$ to an output node there is a node w such that
 - a) the direct successor of w on P is an $\wedge$ -gate or the output node and
 - b) $\exists b_s \in B, b_s \neq b_r$ and $\exists A_s \subseteq A, |A_s| \geq 2$ such that

$$b_s \wedge \left(\bigvee_{a_j \in A_s} a_j \right) \leq \text{res}_{\beta_1}(w).$$

The normal form transformation enlarges the number

of $\wedge$ -gates at most by the factor 4. During the transformation, some $\wedge$ -gates are eliminated. We count these eliminated gates. After the termination of the normal form transformation, we have counted $\lfloor \frac{1}{2}(\frac{n^2}{q} - n) \rfloor$ $\wedge$ -gates and we are done or at least $\frac{n^2}{2}$ products $a_i b_j$ are still computed at those output nodes in β_1 which compute the subfunction $\bar{c}_{i+j}$ of c_{i+j} .

$\Rightarrow$

$\exists a_i \in A$ such that at least q products $a_i b_j$ are computed at these output nodes.

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 $\frac{1}{2}q$ $\wedge$ -gates are eliminated.

$\Rightarrow$

In total, $\geq \lfloor \frac{1}{2} q^2 \rfloor$ $\wedge$ -gates are eliminated.

To see this let us consider the subgraph β'_1 of β_1 which computes the product $a_i b_e$ at the output node $\bar{c}_{i+e}$.

On every path P from the input node u with $\text{op}(u) = b_e$ to the output node q computing $\bar{c}_{i+e}$ consider the node w of the normal form property.

Assume that β'_1 contains less than $\frac{1}{2}q$ $\wedge$ -gates.

$\Rightarrow$

$m < q$ pairwise distinct such nodes w of the normal form property are contained in β_1 .

Normal form property $\Rightarrow$

$\exists b_{s_1}, b_{s_2}, \dots, b_{s_m} \in B, b_{s_j} \neq b_x \ 1 \leq j \leq m$ and
 $\exists A_1, A_2, \dots, A_m \subseteq A, |A_j| \geq 2q \ 1 \leq j \leq m$
such that

$$a_i \bigwedge_{j=1}^m \left(b_{s_j} \wedge \left(\bigvee_{a_t \in A_j} a_t \right) \right) \leq \text{res}_{\beta_1}(g).$$

Definition of the n -th degree convolution $\Rightarrow$

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

$$a_t b_s \leq c_{ite}.$$

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

$$\exists a_j \in A_j \text{ such that } a_j, \bigwedge_{r=1}^m b_{s_r} \not\leq c_{ite}$$

and therefore

$$a_i \bigwedge_{j=1}^m a_j, \bigwedge_{r=1}^m b_{s_r} \not\leq c_{ite}.$$

By construction,

$$a_i \bigwedge_{j=1}^m a_j, \bigwedge_{r=1}^m b_{s_r} \leq \text{res}_{\beta_1}(g).$$

This contradicts that $\bar{c}_{ite} = \text{res}_{\beta_1}(g)$ is a subfunction of c_{ite} .

$\Rightarrow$

β_1 contains $\geq \frac{1}{2}q$ $\wedge$ -gates.

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

Now we carry out the idea.

The construction of β_1

Let β_0 be a monotone network computing C_n with minimal number of $\wedge$ -gates. Let $0 < q < \frac{1}{2}n$.

Beginning at the input nodes of β_0 , we construct β_1 successively. In each step, we consider 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 δ_u are the output nodes of δ_v and δ_w where v and w are the direct predecessors of u in β_0 .

For $0 \leq k \leq 2n-2$, the node in β_0 which computes c_k is denoted by c_k .

An $\wedge$ -gate g with $\text{pred}(g) = \{g_1, g_2\}$ is called a (*)-type-gate if

$$\text{op}(g_1) \in B \text{ and } \text{res}_{\beta_0}(g_2) = \bigvee_{a_j \in A'} a_j$$

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

The network β_1 is constructed such that the following hold:

i) $\text{res}_{\beta_1}(u') \vee \text{res}_{\beta_1}(u'') \leq \text{res}_{\beta_0}(u).$

ii) If $\exists b_s \in B$, $A_s \subseteq A$ maximal, $A_s \neq \emptyset$ such that $b_s \left(\bigvee_{a_j \in A_s} a_j \right) \leq \text{res}_{\beta_1}(u')$ then $|A_s| \geq 2q.$

iii) On every path P leading from a node h with $\text{op}(h) = b_r \in B$ 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 $\exists A_s \subseteq A$, $|A_s| \geq 2q$ such that

$$b_s \wedge \left(\bigvee_{a_j \in A_s} a_j \right) \leq \text{res}_{\beta_1}(w).$$

Remark:

Property i) means that the output nodes of δ_u compute only subfunctions of $\text{res}_{\beta_0}(u)$. Property iii) ensures that, after the construction of β_1 , the normal form property introduced above holds.

Now we shall construct δ_u . We distinguish three cases.

Case 1: u is an input node of β_0 .

Then δ_u consists of the nodes

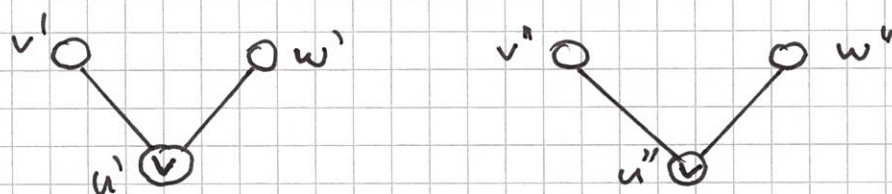
$$\begin{cases} u' \text{ with } op(u') = op(u) \\ u'' \text{ with } op(u'') = 0 \end{cases} \quad \text{if } op(u) \in B$$

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

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



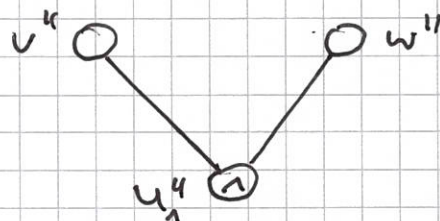
For this construction, we need no $\wedge$ -gate. Since properties i), ii) and iii) hold for δ_v and δ_w , these properties also hold for δ_u .

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. First, we introduce a notation.

A node $g \in \{v', w'\}$ is bipotent if
 $\exists b_s \in B, A_s \subseteq A$ and $b_r \in B, b_r \neq b_s, A_r \subseteq A$
 Such that

$$b_s \wedge \left(\bigvee_{a_j \in A_s} a_j \right) \vee b_r \wedge \left(\bigvee_{a_j \in A_r} a_j \right) \leq \text{res}_{\beta_1}(g).$$

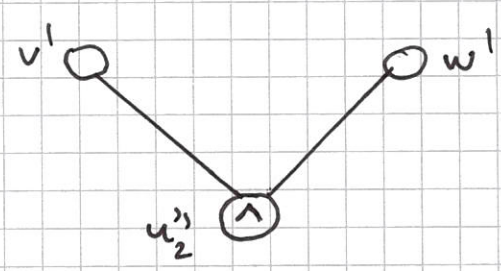
Since property ii) holds for δ_v and δ_w , there hold $|A_s| \geq 2q$ and $|A_r| \geq 2q$.

If a node $g \in \{v', w'\}$ is not bipotent then either $\text{res}_{\beta_1}(g) = 0$ or $\text{res}_{\beta_1}(g) \leq b_r \wedge \left(\bigvee_{a_j \in A_r} a_j \right)$ 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 is at most one $b_s \in B$ such that

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

If no such $b_s \in B, A_s \subseteq A$ exist then by the

structure of C_n and Theorem 3.2, we can replace $v'w'$ by 0 without changing the functions which are computed at the output nodes. Hence, we need not realize the product $v'w'$.

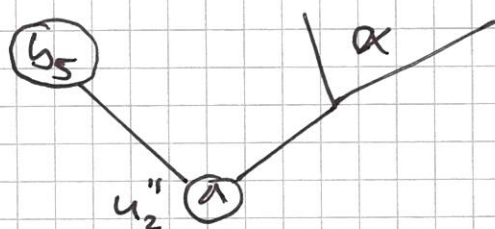
Let $b_s \in B$, $A_s \subseteq A$ maximal such that

$$b_s \wedge \left(\bigvee_{a_j \in A_s} a_j \right) \leq \text{res}_{B_1}(v'w').$$

We distinguish two cases:

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

Then we construct



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

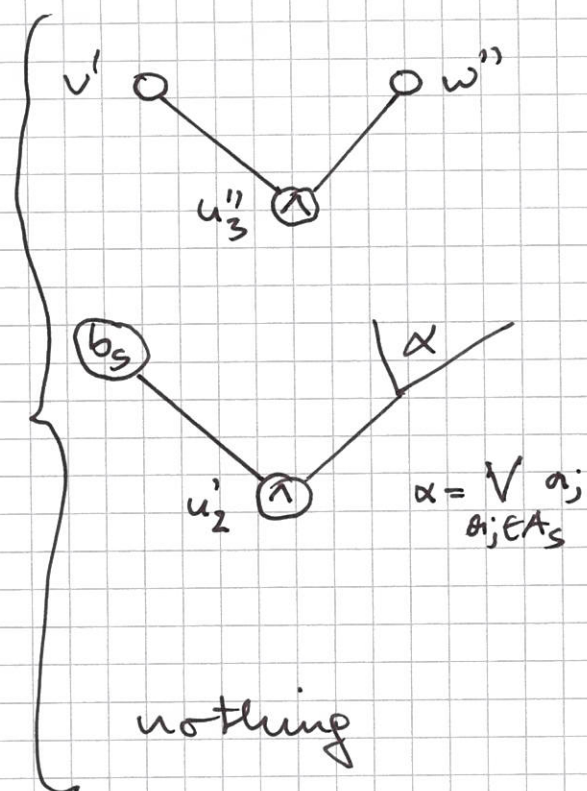
ii) $|A_s| < 2q$.

Then we do not realize the product $v'w'$.

By the structure of C_n , we destroy the computation of $< 2q$ prime implicants of C_n .

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

We construct the following network



if v' is bipotent

if v' is not bipotent and $\exists b_s \in B, A_s \subseteq A$ maximal with $|A_s| \geq 2q$ such that $b_s \wedge (\bigvee_{a_j \in A_s} a_j) \leq \text{res}_{\beta_1}(v'w'')$

otherwise

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

Analogous to that for $v'w''$. Produce the result in u_4'' or u_3' .

If in the construction above

$u_j', j \in \{1, 2, 3\}$ or $u_j'', j \in \{1, 2, 3, 4\}$

do not exist then we construct

u_j' with $\text{op}(u_j') = 0$ or u_j'' with $\text{op}(u_j'') = 0$.

Realization of u' and u'' :

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

For the construction of δ_u , we need at most four 1-gates.

Exercise

Prove that after the construction of δ_n , the properties i), ii) and iii) are fulfilled for δ_n .

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

$$N < t \cdot 2q$$

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_n , the following holds:

a) $\text{res}_{\beta_1}(u') \vee \text{res}_{\beta_1}(u'') \leq \text{res}_{\beta_0}(u)$.

b) If $\exists b_s \in B$, $A_s \subseteq A$ maximal, $A_s \neq \emptyset$ such that $b_s \wedge (\bigvee_{a_j \in A_s} a_j) \leq \text{res}_{\beta_1}(u')$ 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 h with $\text{op}(h) = b_r \in B$ to an $\wedge$ -gate g which is not a $(*)$ -type-gate or to c_k'' , $k \in \{0, 1, \dots, 2n-2\}$, 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

$$b_s \wedge (\bigvee_{a_j \in A_s} a_j) \leq \text{res}_{\beta_1}(w).$$

(4) $0 \leq L_{\wedge}(\beta_1) \leq 4 C_{\Sigma_m}^{\wedge}(C_n) - m$ where $L_{\wedge}(\beta_1)$ is the number of $\wedge$ -gates in β_1 and 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. Property (2) follows from (1) b) and the structure of c_k , $0 \leq k \leq 2n-2$. As observed above, for each $\wedge$ -gate which is not used for the construction of δ_n , at most $2q$ prime implicants have been destroyed. Hence, assertion (4) follows. ■

Using the network β_1 , we prove the following theorem.

Theorem 3.1

Let $C_{\Sigma_m}^{\wedge}(C_n)$ denote the minimal number of $\wedge$ -gates in a monotone network which computes C_n . Then

$$C_{\Sigma_m}^{\wedge}(C_n) \geq \left\lfloor \frac{1}{16} \min \left\{ \left(\frac{n^2}{q} - n \right), q^2 \right\} \right\rfloor$$

Setting $q := n^{2/3}$, we obtain the following corollary.

Corollary 3.2

$$C_{\Sigma_m}^{\wedge}(C_n) \geq \left\lfloor \frac{1}{16} (n^{4/3} - n) \right\rfloor.$$

Proof:

If in (4) of Lemma 3.3 $m \geq \frac{1}{4} \left(\frac{n^2}{q} - n \right)$ then the lower bound is proved. Otherwise, at least

$$n^2 - \frac{1}{4} \left(\frac{n^2}{q} - n \right) 2q > \frac{n^2}{2}$$

(109)

prime implicants of C_n remain.

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

$$a_i b_e \leq \text{res}_{\beta_1}(C_{i+e}) \quad \forall b_e \in \tilde{B}$$

First we 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 eliminate all superfluous gates.

Since fixing an input variable to 1 does not affect the property that a 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 $a_i, b_{e_1}, b_{e_2}, \dots, b_{e_{r-1}}, 1 \leq r < q$ to 1.

Claim

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

Proof of claim

Since $a; b_{e_1} b_{e_2} \dots b_{e_{r-1}} \notin \text{res}_{\beta_1}(C_{i+e_r}^{\wedge})$ and $a; b_{e_r} \leq \text{res}_{\beta_1}(C_{i+e_r}^{\wedge})$ the following hold:

- i) $\text{res}_{\beta_2}(C_{i+e_r}^{\wedge}) \neq 1$ and
- ii) $b_{e_r} \leq \text{res}_{\beta_2}(C_{i+e_r}^{\wedge})$.

Let h be the node in β_2 with $\text{op}(h) = b_{e_r}$. We consider all paths $P_1, P_2, \dots, P_5$ with

- a) start node h and end node $C_{i+e_r}^{\wedge}$ and
- b) $b_{e_r} \leq \text{res}_{\beta_2}(v)$ for all nodes v on $P_j, 1 \leq j \leq 5$.

Since $b_{e_r} \leq \text{res}_{\beta_2}(C_{i+e_r}^{\wedge})$ at least one such a path exists.

Obviously, setting b_{e_r} to 1 eliminates all $\wedge$ -gates on the paths $P_1, P_2, \dots, P_5$.

If on these paths $\frac{1}{2}q$ $\wedge$ -gates exist then we are done. Assume that less than $\frac{1}{2}q$ $\wedge$ -gates exist.

Property b) $\Rightarrow$

All $\wedge$ -gates on these paths are not($\ast$)-type-gates.

Normal form property $\Rightarrow$

For every path leading from the node h to the first $\wedge$ -gate g on a path $P_j, 1 \leq j \leq 5$ (or to

C_{iter}'' if no $\wedge$ -gate on P_j exists) there is 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

$$b_{t_j} \wedge \left(\bigvee_{a_d \in A_j} a_d \right) \leq \text{res}_{\beta_2}(w).$$

and $g \in \text{succ}(w)$ ($C_{i+e_r}'' \in \text{succ}(w)$, resp.)

$\Rightarrow$

$$\bigwedge_{j=1}^s \left(b_{t_j} \wedge \left(\bigvee_{a_d \in A_j} a_d \right) \right) \leq \text{res}_{\beta_2}(C_{i+e_r}'')$$

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

$$b_{t_j} \wedge \left(\bigvee_{a_d \in A_j} a_d \right) \leq \text{res}_{\beta_2}(C_{i+e_r}'').$$

Assume that for all P_j , $1 \leq j \leq s$, an $\wedge$ -gate on P_j exists. (Otherwise, the same proof with $s = 1$ works.)

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

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

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

$$a_i \wedge \bigwedge_{b_j \in B'} \bigwedge_{p=1}^s \left(\bigvee_{a_d \in A_p} a_d \right) \leq \text{res}_{\beta_2}(C_{i+e_r}'').$$

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

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

$|B'| < 2q$ and $|A_p| \geq 2q$, $1 \leq p \leq s$,
the following holds:

$\exists a_{d_p} \in A_p$ such that $a_{d_p} \bigwedge_{b_j \in B'} b_j \notin C_{iter}$
and hence,

$$a_i \bigwedge_{b_j \in B'} b_j \bigwedge_{p=1}^s a_{d_p} \notin C_{iter}$$

But by construction

$$a_i \bigwedge_{b_j \in B'} b_j \bigwedge_{p=1}^s a_{d_p} \leq \text{res}_{P_1}(C_{iter}^{\prime\prime})$$

and by Lemma 3.3 property (1) a), $\text{res}_{P_1}(C_{iter}^{\prime\prime})$ is a subfunction of C_{iter} , a contradiction.

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

Hence, the claim and therefore Theorem 3.2 is proved. $\blacksquare$

Grinchuk and Sergeev have improved the lower bound for the number of $\vee$ -gates to $\Omega(\frac{n^2}{\log n})$. They use the fact that the Boolean convolution can be reduced to Boolean cyclic convolution which can be reduced to certain Boolean sums which are related to circulant matrices (see also Jukna, pp. 386 - 390).