3. Monotone Boolean networks

Let $\mathcal{M}_n := \{\wedge, \vee\}$ be the monotone base. An $\mathcal{M}_n$-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 $f \in \mathcal{M}_n$ 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 Wegener 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 $4n$.


For functions in $M_{n,m}$ where $m = \Theta(n)$, super-linear lower bounds have been proved since 1968. All these lower bound proofs use the following property of each monotone network which computes a function $f = (f_1, f_2, \ldots, f_m) \in M_{n,m}$ which is based on the DNF-representation

$$r_{\beta}(g) = \bigvee_{j=1}^{t} m_j,$$

$m_j$ is a monomial of the function computed at the node $g$, $g \in \beta$.

**Exercise**

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

- Let $u_i$, $1 \leq i \leq m$ be the node in $\beta$ which computes the function $f_i$. For $r_{\beta}(u_i) = \bigvee_{j=1}^{t_i} m_j$, the following hold:
1) For \( 1 \leq j \leq t_i \), the monomial \( w_j \) is an implicant of the function \( f_i \).

2) For each prime implicant \( p \) of \( f_i \) there is a \( j \in \{1, 2, \ldots, t_i\} \) with \( w_j = p \).

If the first property is not fulfilled then there is an input \( \alpha = (a_1, a_2, \ldots, a_n) \in \{0, 1\}^n \) such that \( f_i(\alpha) = 0 \) but \( \text{res}_p(\alpha)(\alpha) = 1 \). If the second property is not fulfilled then there is an input \( \alpha = (a_1, a_2, \ldots, a_n) \in \{0, 1\}^n \) such that \( f_i(\alpha) = 1 \) but \( \text{res}_p(\alpha)(\alpha) = 0 \).

**Exercise**

Let \( \text{res}_p(\beta) = \bigvee_{j=1}^{t_i} w_j \) and \( f \in M_n \). Assume that there is a prime implicant \( p \) of \( f \) such that \( p + w_j \) for \( 1 \leq j \leq t_i \). Prove that there is an input \( \alpha = (a_1, a_2, \ldots, a_n) \in \{0, 1\}^n \) with \( f(\alpha) = 1 \) but \( \text{res}_p(\beta)(\alpha) = 0 \).

Most functions considered for proving lower bounds are homogeneous. A Boolean function \( f \in M_n \) is \( k \)-homogeneous if all prime implicants of \( f \) have length \( k \).

Before we consider explicitly defined monotone functions in \( M_n \), we develop some replacement rules. The idea is to replace a gate in \( \beta \) by another gate which computes a certain function without changing the function computed by the monotone network \( \beta \).
Reference

- Kurt Mehlhorn, Zvi Galil, Monotone switching circuits and Boolean matrix product, Computing 16 (1976), 89-111.

For \( f, g \in B_n \), we define

\[ f \leq g \iff f \land g = f. \]

Then \( f \) is called a subfunction of \( g \).

Theorem 3.2

Let \( f, g \in M_n \) and \( t' \in \text{PIM}(g) \) where \( t' \neq \text{PIM}(f) \) 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 \( \beta \) which computes the function \( f \) and we replace in \( \beta \) the gate \( v \) by a gate \( v' \) which computes \( h \) then the resulting network \( \beta' \) still computes \( f \).

Proof:

Let \( f' \) be the function computed by the network \( \beta' \). 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 \( \beta' \) from \( \beta \) by the replacement of the gate \( v \) in \( \beta \) by the gate \( v' \), there holds...
\( h(a) = 0 \) and \( g(a) = 1 \).

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

Consider \( t^* \in PIM(f) \) 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

\[
    b_j := \begin{cases} 
        a_i & \text{if } j \neq i \\
        0 & \text{if } j = i 
    \end{cases}
\]

Then, by definition

\( b < a \) and \( t(b) = 0 \).

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

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

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 \( t^* \).

\( \Rightarrow \quad \exists t' \) 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 \( \beta \) be a monotone network computing \( f \) and let \( v \) be a gate in \( \beta \) with \( \text{res}_\beta(v) = g \). Let \( t, t_1, \) and \( t_2 \) be monomials such that

i) \( tt_1, tt_2 \in IM(g) \) and

ii) \( \forall \) monomials \( t' \) : \( tt_1, tt_2 \in IM(f) \Rightarrow t't \in IM(f) \)

If we replace in \( \beta \) the gate \( v \) by a gate \( v' \) which computes \( h = g \circ v \) then the resulting network \( \beta' \) still computes \( f \).

Proof:

Let \( f' \) be the function computed by \( \beta' \):

\[ g \leq h \Rightarrow f \leq f'. \]

Suppose that \( f' \neq f \). Then

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

This implies

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

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

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

To get a contradiction, it suffices to prove

\[ t^*tt_1, t^*tt_2 \in IM(f). \]

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

\[ t^* \in PIM(f') \Rightarrow f'(b) = 1. \]
\[ \text{Hence, } g \preceq h \implies h(b) = 1. \]

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

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

Now we consider explicitly defined functions in \( M_{n,m} \).

### 3.1 Boolean Sums.

**References**

- E.I. 


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

\[ f = (f_1, f_2, \ldots, f_m): \{0,1\}^n \rightarrow \{0,1\}^m \]

where

\[ f_i = \bigvee_{j \in F_i} x_j \quad \text{and} \quad F_i \subseteq \{1,2,\ldots,n\} \]

A set of Boolean sums is called \((n,k)\)-disjoint if for all pairwise distinct \(i_0, i_1, \ldots, i_k\)

\[ |F_{i_0} \cap F_{i_1} \cap \ldots \cap F_{i_k}| = 1 \leq k. \]

This means that any \(k+1\) different Boolean sums have at most \(k\) variables in common.

Let \(K_{s,t} = (A \cup B, E)\) 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

\[ G_f = (X_u, F_m, E) \quad \text{where} \]

\[ X_u = \{x_1, x_2, \ldots, x_n\}, \quad F_m = \{f_1, f_2, \ldots, f_m\} \quad \text{and} \]

\[ E = \{(x_j, f_i) \mid 1 \leq j \leq n, 1 \leq i \leq m \quad \text{and} \quad j \in F_i\}. \]

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

For an explicitly constructed set of Boolean sums in \(H_{n, m, t}\), Neciporuk has proved on \(S_2(n^{3/2})\)
lower bound for its monotone complexity in 1968.
This was the first super-linear lower bound for an
explicitly defined function in $M_{m,n}$ 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 $K_{2,2}$.


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
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 follow-
ing theorem.
Theorem 3.4

Let \( f : \{0,1\}^n \rightarrow \{0,1\}^m \) be a \((n,k)\)-disjoint set of Boolean sums. Then
\[
C_{2^m}(f) \geq \sum_{i=1}^{\frac{F_{11}}{k^2}} \frac{1}{n \cdot \max \{1, n-2, k-2\}}.
\]

The proof of Theorem 3.4 is based on two lemmas. The first lemma shows that using \(\nu\)-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
\[
C_{\nu,3}(f) \leq \max \{ n, n-2, k-2\} \cdot C_{2^m}(f).
\]

Proof:

Let \( \beta \) be an optimal \(2^m\)-network for \( f \). Suppose that \( \beta \) contains \( s \) \(\nu\)-gates and \( t \) \(\lambda\)-gates; i.e.,
\[
C_{2^m}(f) = s + t.
\]

The idea is, to eliminate successively the \(\lambda\)-gates using at most \( k-1 \) and \( n-1 \), respectively additional \(\nu\)-gates. Since the eliminated \(\lambda\)-gate is saved, we need for each elimination of an \(\lambda\)-gate at most \( k-2 \) and \( n-2 \), respectively additional gates.

More precisely, we construct a sequence \( \beta_0, \beta_1, \ldots, \beta_t \) of monotone networks where \( \beta_i \), \( 0 \leq i \leq t \) contains \( t-i \) \(\lambda\)-gates and \( \leq s + \max \{ n, n-1, k-2\} \) \(\nu\)-gates.
Note that $\beta_0 = \beta$. Suppose that $\beta_i$, $0 \leq i < t$ is already constructed.

**Construction of $\beta_{i+1}$:**

Let $v$ be a last $n$-gate in topological order; i.e., between $v$ and the output nodes no other $n$-gate exist. Since $\beta_i$ 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

$$g = x_{d_1} v x_{d_2} v ... v x_{d_p} v t_1 v t_2 v ... v t_q$$

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 $t_1, t_2, ..., t_q$, $g$ can be replaced by

$$x_{d_1} v x_{d_2} v ... v x_{d_p} v t_1 v t_2 v ... v t_q$$

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

**Case 2:** $p > k$.

Let $f_{i_1}, f_{i_2}, ..., f_{i_k}$ be the output functions which depend on $g = \text{res}_{\beta_i}(v)$. 


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

\[ f_{r_j} = g \lor u_j. \]

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

\[ \{ d_1, d_2, \ldots, d_p \} \in F_{r_j} \]

Since \( f \) is \((d_1,k)\)-disjoint, \( p > k \) implies \( e \leq n \).

**Claim:**
For \( 1 \leq j \leq e \) either \( f_{r_j} = g_1 \lor u_j \) or \( f_{r_j} = g_2 \lor u_j \).

**Proof of claim:**
Note that \((g = g_1 \land g_2 \land f_{r_j} = g_1 \lor u_j)\).

\[ \Rightarrow f_{r_j} \leq g_1 \lor u_j \quad \text{and} \quad f_{r_j} \leq g_2 \lor u_j. \]

Suppose that

\[ f_{r_j} < g_1 \lor u_j \quad \text{and} \quad f_{r_j} < g_2 \lor u_j. \]

Then there are \( x_1, x_2 \in \{0,1\}^n \) with

\[ f_{r_j}(x_1) = 0 \quad \text{but} \quad (g_1 \lor u_j)(x_1) = 1 \quad \text{and} \]

\[ f_{r_j}(x_2) = 0 \quad \text{but} \quad (g_2 \lor u_j)(x_2) = 1. \]

Let \( x_1 = (a_1, a_2, \ldots, a_n) \) and \( x_2 = (b_1, b_2, \ldots, b_n) \).

Consider

\[ x = (c_1, c_2, \ldots, c_n) \] where \( c_j := \max \{ a_j, b_j \} \), \( 1 \leq j \leq n \).
Since $f_5$ is a Boolean sum and $f_5(x_1) = f_5(x_2) = 0$, there holds
\[ f_5(x) = 0. \]
Since $g_1 v u_j$ and $g_2 v u_j$ are monotone and
\[ (g_1 v u_j)(x_1) = (g_2 v u_j)(x_2) = 1 \]
there holds
\[ (g_1 v u_j)(x) = (g_2 v u_j)(x) = 1. \]
\[ \Rightarrow \]
\[ u_j(x) = 1 \text{ or } (g_1(x) = g_2(x) = 1 \text{ and hence, } g(x) = 1). \]
In both cases, we obtain
\[ f_{r_5}(x) = (g_1 v u_j)(x) = 1, \]
a contradiction.

$\beta_{i+1}$ is constructed from $\beta_i$ in the following way:

(1) Replace $g$ by the constant 0.

This eliminates the $v$-gate $v$ and at least one $v$-gate. After this replacement, the output node $f_{r_i}$, $1 \leq j \leq l$, computes the function $u_j$.

(2) For each $f_{r_5}$, $1 \leq j \leq l$, use one $v$-gate to compute $u_j v g_k$, $k \in \{1, 2, 3\}$ where
\[ f_{r_5} = g_k v u_j. \]
This adds \( \ell \leq n \) \( v \)-gates. Since one \( v \)-gate is saved, we need at most \( \ell - 1 \leq n - 1 \) additional \( v \)-gates. The resulting network is \( \beta_{i+n} \).

This proves the lemma.

Now we prove a lower bound for \( C_{Ev3}(f) \). All gates in an \( Ev3 \)-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 \( f : \{0,1\}^n \to \{0,1\}^m \) be a \((n,k)\)-disjoint set of Boolean sums. Then \( C_{Ev3}(f) \geq \sum_{i=n}^{m} \frac{1}{i} \left( \frac{1}{k} \right) - 1 \).

**Proof:**

Let \( \beta \) be an optimal \( Ev3 \)-network for \( f \). Note that the input nodes of \( \beta \) are small. Let \( v \) be a large gate in \( \beta \). Since \( v \) computes the disjunction of \( > k \) variables and \( f \) is \((n,k)\)-disjoint, at most \( k \) output nodes \( f_i \) depend on \( v \).

The large gates of \( \beta \) connect the small gates of \( \beta \) 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 \( v \in \beta \), let \( n(v) \) denote the number of output nodes \( f_i \) which depend on \( v \). If \( v \) is large, then \( n(v) \leq m \). Hence,

\[
\sum_{\text{large } v \in \beta} n(v) \leq m \cdot |\{ v \in \beta \mid v \text{ is large}\}|
\]

Consider the set \( H_i \) of all large gates \( v \in \beta \) 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.

\[
\Rightarrow \sum_{\text{large } v \in \beta} n(v) = \sum_{i=1}^{m} |H_i| \leq \sum_{i=1}^{m} \left( \lceil \frac{|F_i|}{k} \rceil - 1 \right).
\]

Hence we obtain

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

This implies

\[
C_{\text{sel}}(\ell) \geq \frac{1}{m} \sum_{i=1}^{m} \left( \lceil \frac{|F_i|}{k} \rceil - 1 \right).
\]

Proof of Theorem 3.4:

\[
C_{\text{sel}}(\ell) \geq \max_{1, m-2, k-2^3} \frac{1}{\ell} \sum_{i=1}^{m} \left( \lceil \frac{|F_i|}{k} \rceil - 1 \right) \geq \max_{1, m-2, k-2^3} \frac{C_{\text{sel}}(\ell)}{\ell}
\]
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 $K_{3,3}$.


Using this construction, we obtain an explicit $(2,2)$-disjoint set of Boolean sums $f: \{0,1\}^n \rightarrow \{0,1\}^n$ 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


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 $\geq n^{2-\frac{4}{5}}$ edges and no $K_{t,t+1}$ where $t \geq 2$ is an integer.

06.05.
3.2 Boolean matrix multiplication

References


In 1974, Pratt has shown that each monotone network computing the product of two n x n Boolean matrices contains at least $\frac{n}{2} n^3$ n-gates. Mehlhorn and Galil and Peterson 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 an $(p \times q)$ Boolean matrix. Then

$$C := A \cdot B$$

is an $r \times q$ Boolean matrix where

$$c_{ir} := \bigvee_{j=1}^{p} a_{ij} \land b_{jr}.$$
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 \(r \cdot p \cdot q\) \(\land\)-gates and at least \(r \cdot q \cdot (p - 1)\) \(\lor\)-gates.

Proof:
Let \(\beta\) be an optimal monotone network computing the product matrix \(C = A \cdot B\).

Goal: The isolation of an \(\land\)-gate which computes \(a_{ij} b_{jk}\) for each triple \((i, j, k)\).

General method:
Definition of predicates \(P\) on the nodes of \(\beta\) such that
- \(P\) holds for at least one output node of \(\beta\) and
- \(P\) does not hold for any input node of \(\beta\).

\(\Rightarrow\)
- \(\exists\) gate \(v\) in \(\beta\) such that
- \(P\) does not hold for any incoming edge of \(v\) but
- \(P\) holds for \(res_{\beta}(v)\).
Let \(IC(P)\) denote the set of these gates.

\(\Rightarrow\)
1) Identification of \(r \cdot q\) \(\land\)-gates for the products \(a_{ij} b_{jk}\), \(1 \leq i \leq r, 1 \leq k \leq q\)
and elimination of these gates by setting
\(a_{ij}\) to 1, \(1 \leq i \leq r\)
\(b_{jk}\) to 0, \(1 \leq k \leq q\).
The resulting monotone network computes the \((r, p^{-1}, q)\)-matrix product \(C\) where
\[
\hat{c}_{ik} = \bigvee_{j=2}^{p} a_{ij} \land b_{jk}.
\]

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
\[P_{ik}(v) \iff a_{ik} b_{1k} \leq h \land a_{ik} \land h \land b_{ik} \leq h.
\]
This implies that \(a_{ik} b_{1k}\) is a prime implicant of \(h\).

The input nodes of \(\beta\) does not fulfill \(P_{ik}\) but the output function \(c_{ik}\) fulfills \(P_{ik}\).

\[\Rightarrow\quad I(P_{ik}) \neq \emptyset.
\]

Consider \(v \in I(P_{ik})\). First we show that \(v\) is an \(x\)-gate. Suppose that \(v\) is an \(v\)-gate. Then
- \(a_{ik} b_{1k} \leq h\) implies \(a_{ik} b_{1k} \leq h_1\) or \(a_{ik} b_{1k} \leq h_2\).

W.l.o.g., we can assume \(a_{ik} b_{1k} \leq h_1\).

Now \(\neg P_{ik}(h_1)\) implies
\[
a_{ik} \leq h_1 \quad \text{or} \quad b_{1k} \leq h_1.
\]
and hence, \(a_{ik} \leq h\) or \(b_{1k} \leq h\).

\[\Rightarrow\quad \neg P_{ik}(h), \quad 	ext{a contradiction.}
\]
Hence, \( v \) is an \( \Lambda \)-gate.

Since \( a_{i_1} b_{i_2} \leq m \) there hold \( a_{i_1} b_{i_2} \leq m_1 \) and \( a_{i_1} b_{i_2} \leq m_2 \).

Furthermore, \( \neg P_{i_1}(m_1) \land \neg P_{i_2}(m_2) \Rightarrow \)

Either \( a_{i_1} \leq m_1 \land b_{i_2} \leq m_2 \) or
\[ a_{i_1} \leq m_2 \land b_{i_2} \leq m_1. \]

Note that \( a_{i_1} \leq m_1 \land a_{i_1} \leq m_2 \Rightarrow a_{i_1} \leq m \) and hence, \( \neg P_{i_1}(m) \). Analogously, we can exclude \( b_{i_2} \leq m_1 \land b_{i_2} \leq m_2. \)

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

Now we prove that these \( \Lambda \)-gates are pairwise distinct. For doing this, suppose

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

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

1) \( a_{i_1}, a_{i_2} \leq m_1 \) and \( b_{i_1}, b_{i_2} \leq m_2 \)

2) \( a_{i_1}, b_{i_2} \leq m_1 \) and \( b_{i_1}, a_{i_2} \leq m_2 \).

As we will see, we can apply in both situations Theorem 3.3 constructing a better network than \( \beta \).

This contradicts the optimality of \( \beta \).

Situation 1:

Suppose \( i_1 \neq i_2 \). Then for the input function \( m_1 \) and \( t = 1, t_1 = a_{i_1}, \) and \( t_2 = a_{i_2} \),

the assumptions of Theorem 3.3 are fulfilled.
Hence, \( n_i \) can be replaced by \( n_i \lor 1 \).

\[ \Rightarrow \]

One input of \( \lor \) gets to be constant such that the gate \( \lor \) can be eliminated.

This contradicts the optimality of \( \beta \).

If \( i_1 = i_2 \) then \( b_1 = b_2 \). This case can be excluded in the same way using \( h_2 \), \( t=1 \), \( t_1 = b_1 \), and \( t_2 = b_2 \).

**Situation 2:**

By application of Theorem 3.3, we can replace both inputs of \( \lor \) by 1.

**Exercise**

Show that in Situation 2, both inputs of \( \lor \) can be fixed at 1 without changing the function computed by \( \beta \).

This contradicts the optimality of \( \beta \) as well.

 Altogether, we have proved that the sets \( I(P_i) \) are pairwise disjoint.

\[ \Rightarrow \]

\( \therefore \) different \( \lor \)-gates are isolated.

Now we consider the \( \lor \)-gates in \( \beta \). Analogous to the consideration of the \( \land \)-gates, we define for \( 1 \leq i \leq r \), \( 1 \leq k \leq g \) a predicate \( Q_{ik} \) in the following way:

\[ Q_{ik}(\lor) \iff a_{ik} \lor b_{ik} \leq n \leq A_i \lor b_{ik} \quad \text{and} \quad n \neq b_{ik} \]

where

\[ A_i := \bigvee_{j \neq i} a_{ij} \]
The input nodes do not fulfill $Q_{i,k}$. The output node $c_{i,k}$ fulfills $Q_{i,k}$ since

- $a_{i_1} b_{i_1} \in \text{PIIM}(c_{i,k}) \Rightarrow c_{i_1} b_{i_1} \leq c_{i,k}$.
- Each prime implicant of $c_{i,k}$ contains the variable $b_{i_1}$ or a variable in $\{a_{i_2}, a_{i_3}, \ldots, a_{i_p}\}$. Hence, $c_{i,k}(x) = 1 \Rightarrow (A_i \lor b_{i_1})(x) = 1$.
  \[ \Rightarrow c_{i,k} \leq A_i \lor b_{i_1}. \]
- $c_{i,k} \neq b_{i_1}$ is obvious by the definition of $c_{i,k}$.

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

**Exercise**

Prove the following two assertions:

a) If $v \in I(Q_{i,k})$ then $v$ is an $v$-gate and either $b_1 \leq b_{i_1}$ or $b_2 \leq b_{i_1}$.

b) The sets $I(Q_{i,k})$ are pairwise disjoint.

Altogether, we have achieved the following:

1) For each pair $(i, k)$, we have located an $v$-gate in $\beta$ with one of its input functions has prime implicant $a_{i_1}$. For different pairs, we have identified different $v$-gates.

\[ \Rightarrow \text{fixing } a_{i_1} \text{ at } 1 \text{ for } 1 \leq i \leq r \text{ eliminates } \text{f.g } v\text{-gates in } \beta. \]

2) If $p > 1$ then we have located an $v$-gate for each pair $(i, k)$ where one of its input function
contains only prime implicants containing the variable $b_{ik}$. For different pairs, we have identified different $v$-gates.

$$
\Rightarrow
$$

Fixing $b_{ik} = 0$ for $1 \leq k \leq q$ eliminates $v$-gates in $\beta$.

**Exercise**

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

$$
c_{ik} = \bigvee_{j=2}^{n} a_{ij} b_{jk}.
$$

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

3.3 A generalized Boolean matrix product

References

- [Lutz Wegener](#), Switching functions whose monotone complexity is nearly quadratic, TCS 9 (1979), 83–87.

- [Lutz Wegener](#), Boolean functions whose monotone complexity is of size $\Omega(n^2 \log n)$, TCS 24 (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.
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 if the chosen rows have a common one.

To formalize this let $x_{me}^i$ denote the element of the matrix $M_i$ at position $(h, e)$. Then we say that $x_{me}^i$ is a variable of type $e$. This means that the type of a variable is its position in the corresponding row.

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

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

We say that the prime implicant

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

is of type $e$.

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

$$f_{MN}^m : \mathcal{B}^{m \times MN} \rightarrow \mathcal{B}^{m \times MN}$$

where

$$Y_{h_1 h_2 \ldots h_m} := \bigvee_{1 \leq e \leq N} x_{h_1 e}^1 x_{h_2 e}^2 \ldots x_{h_m e}^m$$

First we prove an upper bound for the monotone network complexity of $f_{MN}^m$. 
Theorem 3.6

\[ C^m_2 (f^{m}_{NM}) \leq N \cdot M^m (2 + (M-1)^{-1}) \leq 3 \cdot N \cdot M^m. \]

Proof:

Assume that all monomials \( (u_1, u_2, ..., u_{M'}, e) \) are computed. Then \( N \cdot M^i \) additional \( \lambda \) gates suffice to compute all monomials \( (u_1, u_2, ..., u_{i'}, e) \).

\[ \Rightarrow \]

\[ C^m_\lambda (f^{m}_{NM}) \leq N \cdot \sum_{2 \leq i \leq M} M_i \]

\[ \leq N \cdot (M^{m+1} - 1) (M-1)^{-1} \]

\[ \leq NM^m (1 + (M-1)^{-1}) \]

Afterwards, each output can be computed using \( N - 1 \) \( \lambda \) gates.

Our goal is to prove \( \frac{1}{2} NM^m \) lower bound for the number of \( \lambda \) gates in a monotone network which computes \( f^{m}_{NM} \). First we investigate the structure of optimal monotone networks computing \( f^{m}_{NM} \).

Lemma 3.3

Let \( g \) be a function which is computed on a gate in a monotone network for \( f^{m}_{NM} \). Let

\[ (i_1, i_2, ..., i_{M'}, e), (j_1, j_2, ..., j_{M'}, e) \in PIM (g) \]

for some \( e \in \{1, 2, ..., N^3 \} \) and \( i_k, j_k \in \{1, 2, ..., M^3 \} \)
Let $A := \{ x_i | i \in 3 \}$ and let
\[ t := \bigwedge_{k \in A} x_i^k. \]
Then $g$ can be replaced by
\[ h := g \cdot t \] and the resulting network still computes $f_N$.

**Proof:**
Let
\[ t_1 := \bigwedge_{k \in A} x_i^k \quad \text{and} \quad t_2 := \bigwedge_{k \in A} x_i^k. \]

**Definition of $A$**

$t_1$ and $t_2$ have no common variable.

Now we prove that the assumptions of Theorem 33 are fulfilled with respect to the chosen $t, t_1,$ and $t_2$.
By construction

\[ tt_1, tt_2 \in \mathcal{PM}(g) \subseteq \mathcal{IM}(g). \]

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

- a monomial $t'$ and an output function $y_{1,2,\ldots,m}$ with
  - $t't_1, t't_2 \in \mathcal{IM}(y_{1,2,\ldots,m})$ but $tt' \notin \mathcal{IM}(y_{1,2,\ldots,m})$

Then
\[ tt_1 \text{ contains a prime implicant } (h_1, h_2, \ldots, h_m, l') \]
but $tt'$ does not contain this prime implicant.

By construction, it holds $e' = e$. 
The same holds with respect to $t_1t_2$.

Altogether, we obtain

- $(h_1, h_2, \ldots, h_m, e)$ is a submonomial of $t_1t_2$
  and also a submonomial of $t_1t_1t_2$ but not
  a submonomial of $t_1$.

$\Rightarrow$

$t_1$ and $t_2$ have a variable in common.

But this contradicts the definition of $A$.

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

Lemma 3.3 $\Rightarrow$

- If the function $g = \text{res}_{p(e)}$ includes several useful
  monomials of type $e$ then we can replace $g$
in $f$ by $g\cdot t$ where $t$ is the common part of
all useful monomials of type $e$.

Goal:
Application of Lemma 3.3 without additional cost.

For doing this, we need the following properties:

1) $v$-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 an \( \ast \)-network. \( C_\ast \) is the associated complexity measure.

Given any \( \ast \)-network \( \beta \) for \( f^m_{\text{mon}} \), we wish to transform \( \beta \) into a so-called standard \( \ast \)-network of the same complexity by applying Lemma 3.3.

For doing this, we consider the gates in \( \beta \) in any topological order. Let \( u \) be the current con = sided gate and \( g := \text{res}_\beta (u) \).

For \( 1 \leq l \leq N \) let

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

Without additional cost, we replace:

\[ g \]

by \( g \cup t_1 \cup t_2 \cup \ldots \cup t_N \).

\( \Rightarrow \) Altogether, we obtain a \( \ast \)-network \( \beta' \) for \( f^m_{\text{mon}} \) 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.
Theorem 3.7
Let $m \geq 2$. Then

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

Corollary 3.1
For $n \geq 4$ let $m(n) := \lceil \log n \rceil$, $M(n) := 2$, $N(n) := \lceil \frac{n}{2 \log n} \rceil$, and $N_n := f_{N(n)}(n(n))$. Then the function $M_n$ depends on at most $n$ variables and has at most $n$ output functions. Furthermore,

$$C_{2m}^*(M_n) = \Omega \left( \frac{n^2}{\log n} \right).$$

Proof:
Exercise

Proof of Theorem 3.7:
Let $\beta$ be an optimal standard $*$-network for $f_{MN}^m$.

Goal:
For each $\land$-gate $v$ in $\beta$, the definition of a value function

$$c(v) : \Pi M (f_{MN}^m) \rightarrow \{0,1\}$$

Such that

$$c(v) := \sum_{1 \leq i_1, i_2, \ldots, i_m \leq N} \sum_{1 \leq \ell \leq N} c_v(h_{i_1}, h_{i_2}, \ldots, h_{i_m}, \ell) \leq 1.$$
Properties:

a) \( c_v (h_1, h_2, \ldots, h_m, e) \) is an estimation of the contribution of the \( v \)-gate to the computation of the prime implicant \( (h_1, h_2, \ldots, h_m, e) \).

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

\[ \Rightarrow \]

For an optimal \( \ast \)-network \( \beta \) there holds

\[ c(\beta) := \sum_{v \text{ gate in } \beta} c(v) \leq C^*_{m_m} (f_{HN}). \]

This means that \( c(\beta) \) is a lower bound for \( C^*_{m_m} (f_{HN}) \).

Claim 1.

\[ c(h_1, h_2, \ldots, h_m, e) := \sum_{v \text{ gate in } \beta} c_v (h_1, h_2, \ldots, h_m, e) \geq \frac{1}{2} \]

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

Claim 1 \( \Rightarrow \)

\[ C^*_{m_m} (f_{HN}) \geq \sum_{1 \leq h_1, h_2, \ldots, h_m \leq M} \sum_{1 \leq e \leq N} c(h_1, \ldots, h_m, e) \]

\[ \geq \frac{1}{2} \cdot N \cdot M^m \]
Definition of the value function

Let \( v \) be an \( \gamma \)-gate in an optimal standard \( \ast \)-network \( \beta \) for \( \text{f}_{\text{MN}} \). Let \( g := \text{res}_\beta (v) \) and \( g', g'' \) be the functions of the ingoing edges of \( v \).

Idea

The value function \( c_v \) assigns to the \( \gamma \)-gate \( v \) a positive value for the prime implicant \( t \in \text{PIM} (\text{f}_{\text{MN}}) \) if

\[ t \in \text{PIM}(g) \text{ and } (t \notin \text{PIM}(g') \text{ or } t \notin \text{PIM}(g'')). \]

Question: Which value \( > 0 \) should be chosen?

To define these values let

\[ i_1, i_2, ..., i_q \in \{1, 2, ..., N\} \]

be those types such that

\[ \exists t \in \text{PIM}(\text{f}_{\text{MN}}) \text{ of type } i_e \text{ with } t \in \text{PIM}(g) \text{ but } t \notin \text{PIM}(g'). \]

Furthermore, let

\[ j_1, j_2, ..., j_q \in \{1, 2, ..., N\} \]

be those types such that

\[ \exists t \in \text{PIM}(\text{f}_{\text{MN}}) \text{ of type } j_e \text{ with } t \in \text{PIM}(g) \text{ but } t \notin \text{PIM}(g''). \]

Then we define for \( t \in \text{PIM}(\text{f}_{\text{MN}}) \)

\[ c_v(t) := \begin{cases} \frac{1}{2q'} & \text{if } t \in \text{PIM}(g) \text{ and } t \notin \text{PIM}(g') \\ 0 & \text{otherwise} \end{cases} \]
\[ c_{uv}(t) := \begin{cases} \frac{1}{2q^n} & \text{if } t \in \text{PM}(q) \text{ and } t \notin \text{PM}(q^{1}) \\ 0 & \text{otherwise} \end{cases} \]

Then \( c_{v}(t) \) is defined by

\[ c_{v}(t) := c_{v}^{1}(t) + c_{v}^{2}(t). \]

Property (x) \( \Rightarrow \)

\[ c_{v}(v) = \sum_{1 \leq h_{1}, \ldots, h_{m} \leq M} \sum_{1 \leq l \leq N} c_{v}^{1}(h_{1}, h_{2}, \ldots, h_{m}, l) \]

\[ = q^{n} \cdot \frac{1}{2q^{n}} = \frac{1}{2}. \]

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

Analogously,

\[ c_{v}^{2}(v) = \sum_{1 \leq h_{1}, \ldots, h_{m} \leq M} \sum_{1 \leq l \leq N} c_{v}^{2}(h_{1}, h_{2}, \ldots, h_{m}, l) \]

\[ = q^{n} \cdot \frac{1}{2q^{n}} = \frac{1}{2}. \]

\( \Rightarrow \)

\[ c(v) = c^{1}(v) + c^{2}(v) = 1. \]

It remains to prove Claim 1.

Proof of Claim 1:

Consider the prime implicant \( t := (h_{1}, h_{2}, \ldots, h_{m}, l) \) and the corresponding output \( y_{t} := y_{h_{1}, h_{2}, \ldots, h_{m}} \). 

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

- gate $v$ is in $\beta(t)$ iff there is a path $P$ in $\beta$ from $v$ to $y_t$ and $t$ is a prime implicant of all functions computed on $P$ (inclusive res$_{\beta(cu)}$).
- 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 $t \in PIM(g)$
2) $t \in PIM(\text{res}_{\beta(cu)})$ where $u$ is a gate in $\beta(t)$.
3) Let $u$ be a gate in $\beta(t)$ with both direct predeces- sors of $u$ are inputs of $\beta(t)$. Then, $u$ is an $\wedge$-gate (otherwise $t \in PIM(\text{res}_{\beta(cu)})$).
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 $s_1, s_2, \ldots, s_d$ be those inputs of $\beta(t)$ which are input of an $\wedge$-gate in $\beta(t)$.

Let $u_{ci}$ be an $\wedge$-gate in $\beta(t)$ with input $s_i$. Let

$$
c^*(u_{ci}) := \begin{cases} 
c'(u_{ci}) & \text{if } s_i \text{ is the first input of } u_{ci} \\
c''(u_{ci}) & \text{if } s_i \text{ is the second input of } u_{ci} 
\end{cases}
$$
Properties 1) and 2) $\Rightarrow c^*(v_c(i)) > 0$

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

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

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

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

- a proper prolongation $w_i^*$ of $w_i$ is contained in $PIM(res_{B}(v_c(i))) \cap PIM(f_{MN})$ and
- the type of $w_i^*$ is different to the types of $w_1^*, w_2^*, \ldots, w_{i-1}^*$.

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

**Case 1:** The choice of $w_i$ according to the rules above is impossible for an $i \leq D$.

$\Rightarrow$

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

Hence, by the definition of the value function $b_i > (2(i-1)^{-1}$.

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

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

Case 2: 7 Case 1.

Construction: \( \Rightarrow w_i \in \Pi_1 M (s_i) \) for \( 1 \leq i \leq D \).

\( \Rightarrow \ w_1 w_2 \ldots w_D \in s_1 s_2 \ldots s_D \).

Construction of \( P(c) \): \( \Rightarrow \)

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

\( y_t(a) = 1 \).

(This can easily be shown by induction.)

\( \Rightarrow \ w_1 w_2 \ldots w_D \in y_t \).

All variables in \( w_i \) are of type \( \ell_i \) and \( \ell_1, \ell_2, \ldots, \ell_0 \)
are pairwise different.

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

\( \Rightarrow \ w_i \) contains \( \leq \m - 1 \) variables

\( \Rightarrow \)

\( w_1 w_2 \ldots w_D \in \Pi_1 (y_t) \)

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, Sliding graphs and their applications, JACM 23 (1976), 423 - 432.


- Norbert Blum, An $n^{2/3}$ lower bound on the monotone network complexity of the $n^{th}$ degree convolution, TCS 36 (1985), 59 - 68.


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

$$C_n(c_0, c_1, \ldots, c_{2n-2}) : \{0,1\}^{2n} \rightarrow \{0,1\}^{2^{n-1}}$$

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

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 \( A := \{a_0, a_1, \ldots, a_{p-1}\} \) and \( B = \{b_0, b_1, \ldots, b_{q-1}\} \) be two disjoint sets of variables. A monotone function 
\[ f = (f_1, f_2, \ldots, f_m) : A \cup B \rightarrow \{0,1\}^m \]
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.

If \( f \) is a set of semidisjoint bilinear forms if \( f \) has the following properties:
1) \( \text{PIM}(f_i) \cap \text{PIM}(f_j) = \emptyset \) for \( 1 \leq i < j \leq m \).
2) For $1 \leq k \leq n$, each variable in $A \cup B$ is contained in at most one prime implicant in $PIM(f_k)$.

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

For $a; b; j \in PIM(f)$, we define inductively the following subset $PIM_{i,j}(f)$ of $PIM(f)$:

a) $a; b; j \in PIM_{i,j}(f)$,

b) $a; b; j \in PIM_{i,j}(f) \Rightarrow a; b; j \in PIM_{i,j}(f)$ for all $a; b; j \in PIM(f)$

and also $a; b; j \in PIM_{i,j}(f) \Rightarrow a; b; j \in PIM(f)$.

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

3) For $1 \leq k \leq n$, $0 \leq i, j \leq n-1$ there holds

$$|PIM_{i,j}(f_k) \cap PIM(f_k)| \leq 1.$$ 

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 \((m,k)\)-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 \(c_x, c_y \in C_n\) 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 \(c\)-seminoidisjoint bilinear forms given 1982 by Jürgen Weiss in his diploma thesis.

**Theorem 3.8**

Let \(f\) be a set of seminoidisjoint bilinear forms. Let \(r_i\) be the number of prime implicants which contain the variable \(a_i\). Then

\[
C_{\chi}(f) \geq \sum_{i=0}^{2^{-n}} r_i^{1/2}
\]

**Corollary 3.2**

The monotone network complexity of the Boolean \(n\)-\(m\) degree convolution is \(\geq n^{3/2}\).
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 $F$ be an optimal $2^m$-network for $f$. Note that after replacing $a_0$ by 0, we obtain a sub-function of $f$ which is also semi-disjoint. Moreover, the values $r_i$, $i > 0$ do not change.

$\Rightarrow$
It suffices to prove that after setting $a_0$ to 0, at least $\frac{r_0}{2}$ gates have been eliminated.

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

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

Since $f$ is semi-disjoint, these gates cannot be used for the computation of the other outputs in an optimal $2^m$-network for $f$.

Claim
Setting $a_0$ to 0 eliminates at least $(r_0 - s_0) \cdot \frac{1}{2}$ $n$-gates

Note that this claim implies that setting $a_0$ to 0 eliminates at least $r_0 \cdot \frac{1}{2}$ gates.
We identify the output node and the function which is computed at the output node. Let

\[ F := \{ f_k | \exists d : a_b, d \in PIM(f_k) \land |PIM(f_k)| > 2 \}. \]

Consider any path \( P = v_0, v_1, \ldots, v_m \) from \( a_0 \) to \( f_k \) where \( f_k \in F \). Then, the semi-disjointness of \( f \) and \( |PIM(f_k)| \geq 2 \) imply that \( P \) contains at least one \( v \)-gate \( v_2 \) with \( a_i b_j \leq res \_\beta (v_2) \) for \( a_i \neq 0 \) and any \( j \in \{0, 1, \ldots, q-1\} \).

**Exercise**

Show that the property "\( P \) contains no \( v \)-gate \( v_2 \) with \( a_i b_j \leq res \_\beta (v_2) \) for \( a_i \neq 0 \) and some \( j \)" implies that \( f_k \) does not contain two prime implicants.

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

\[ P := \{ P | P \text{ is a path from } a_0 \text{ to } a_k \in F \}. \]

Let

\[ V^* := \{ v \text{-gate } v | v \text{ is suitable for a } P \in P \}. \]

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

- \( p \) contains at least one of the following monomials as a submonomial:
  - \( a_0 \),
  - the conjunction of two variables in \( A \),
  - \( \wedge_2 \).
- the conjunction of two variables in \( T \).

Exercise

Give a formal proof that for each gate \( w \) between \( a_o \) and \( v \) (not \( v \)) each prime implicant \( p \) of \( \text{res}_p(w) \) has property \((*)\).

\[ \Rightarrow \]

After setting \( a_o \) to 0 and eventual applications of Theorem 3.2, \( \text{res}_p(cw) \) can be replaced by 0.

\[ \Rightarrow \]

After setting \( a_o \) to 0, each gate \( v \in V^* \) can be eliminated.

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

To prove this claim let

\[ V^* = \{ v_1, v_2, \ldots, v_D \} \]

Then there are \( i_1, i_2, \ldots, i_D, i_e = 0 \) for \( 1 \leq e \leq D \) and \( j_1, j_2, \ldots, j_D \) with

\[ \alpha_{i_e} \beta_{i_e} \leq \text{res}_p(cv_2), \quad 1 \leq e \leq D \]

Consider any \( f_\epsilon \in F \). Let \( a_0 b_\epsilon c_\epsilon d_\epsilon \in \text{PIM}(f_\epsilon) \). Let \( \beta' \) be the subnetwork of \( \beta \) which constructs the prime implicant \( a_0 b_\epsilon c_\epsilon d_\epsilon \) at the output node \( f_\epsilon \).

On each path \( P \) from \( a_0 \) to \( f_\epsilon \) in \( \beta' \), there is a node \( v_e \in V^* \). Instead of using the input function
if we on $P$ one can use the other input function.

$\Rightarrow$

$b_{dx}a_{i_1}a_{i_2}\cdot\cdot\cdot a_{i_D}b_{j_1}b_{j_2}\cdot\cdot\cdot b_{j_0} \leq f_x$.

$\Rightarrow\exists x, x' \in \{1, 2, \ldots, D\}$ with $a_{i_x}b_{j_x}, \in \text{PM}(f_x)$.

Since $f$ is semi-disjoint each $a_{i_x}b_{j_x}, 1 \leq x, x' \leq D$ can be a prime implicant of at most one function in $F$. Note that $|F| = \overline{r_0} - s_0$.

$\Rightarrow$

$D^2 \geq \overline{r_0} - s_0$

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

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

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

Exercise:

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

To get a lower bound for the number of $v$-gates, we need some other techniques.
We start with an optimal monotone network $\beta_0$ which computes $C_n$. We have no knowledge about the structure of $\beta_0$.

Idea:

Transform the network $\beta_0$ into a monotone network $\beta_1$ such that:

1) $\beta_1$ computes a number of subfunctions of $C_n$ and
2) we know enough about the structure of $\beta_1$ such that we can show that either $\beta_1$ contains a large number of $n$-gates or we have eliminated a large number of $n$-gates during the transformation of $\beta_0$ to $\beta_1$.

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 $\text{out}(u) = b_r \in B$ to an output node there is a node $w$ such that:
  a) the direct successor of $w$ on $P$ is an $n$-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 $b_s \land \left( \bigvee_{a_j \in A_s} a_j \right) \leq \text{res}_{\beta_1}(w)$.

The normal form transformation encircles the number
of \( \land \)-gates at most by the factor 4. During the transformation, some \( \land \)-gates are eliminated. We count these eliminated gates. After the termination of the normal form transformation, we have counted \( \frac{1}{2} \left( \frac{n^2}{q} - n \right) \frac{1}{2} \) \( \land \)-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 \( B_1 \), which compute the subfunction \( \overline{c_{i+j}} \) of \( c_{i+j} \).

\[ \Rightarrow \]

\( \exists q_i \in A \) 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 \( q \) \( b \)'s to 1. We prove that after every fixing of a \( b \), at least \( \frac{q}{2} \) \( \land \)-gates are eliminated.

\[ \Rightarrow \]

In total, \( \geq \frac{1}{2} q^2 \frac{1}{2} \) \( \land \)-gates are eliminated.

To see this let us consider the subgraph \( B'_i \) of \( B_1 \) which computes the product \( a_i b_e \) at the output node \( \overline{c_{i+e}} \).

On every path \( P \) from the input node \( u \) with open \( u \) = \( b_e \) to the output node computing \( \overline{c_{i+e}} \) considers the node \( w \) of the normal form property.

Assume that \( B'_i \) contains less than \( \frac{q}{2} \) \( \land \)-gates.
\[ m < q \text{ pairwise distinct such nodes } u \text{ of the normal form property are contained in } \beta_1. \]

**Normal form property \( \Rightarrow \)**

\[ \exists b_{s_1}, b_{s_2}, \ldots, b_{s_m} \in B, b_{s_j} \neq b_x \quad 1 \leq j \leq m \quad \text{and} \]

\[ \exists A_1, A_2, \ldots, A_m \subseteq A \quad |A_j| \geq 2q \quad 1 \leq j \leq m \]

such that

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

**Definition of the \( n \)-th degree convolution \( \Rightarrow \)**

\[ \forall b_s \in B \text{ there exists at most one } a_t \in A \text{ with} \]

\[ a_t b_s \leq c_{i+e}. \]

Hence, for \( 1 \leq j \leq m \) it holds

\[ \exists a_{j_i} \in A_j \text{ such that } a_{j_i} \bigwedge_{r=1}^m b_{s_r} \neq c_{i+e} \]

and therefore

\[ a_i \bigwedge_{j=1}^m a_{j_i} \bigwedge_{r=1}^m b_{s_r} \neq c_{i+e}. \]

**By construction,**

\[ a_i \bigwedge_{j=1}^m a_{j_i} \bigwedge_{r=1}^m b_{s_r} \leq \text{res}_{\beta_1}(g). \]

This contradicts that \( c_{i+e} = \text{res}_{\beta_1}(g) \) is an subfunction of \( c_{i+e} \).

\[ \Rightarrow \]
\( \beta' \) contains \( \geq \frac{1}{2} q \) \( \heartsuit \)-gates.

By setting \( a_i \) and \( b_2 \) to 1, all these \( \heartsuit \)-gates are eliminated.

Now we carry out the idea.

The construction of \( \beta_0 \)

Let \( \beta_0 \) be a monotone network computing \( C_n \) with minimal number of \( \heartsuit \)-gates. Let \( 0 < q < \frac{1}{2} n \).

Beginning at the input nodes of \( \beta_0 \), we construct \( \beta_1 \) successively. In each step, we consider a node \( u \) in \( \beta_0 \), the direct predecessors of which were constructed in \( \beta_1 \) before.

\[ u \rightarrow \text{small network } \delta_u \text{ with output nodes } u', u'' \]

The input nodes of \( \delta_u \) are the output nodes of \( \delta_v \) and \( \delta_w \), where \( v \) and \( w \) are the direct predecessors of \( u \) in \( \beta_0 \).

For \( 0 \leq k \leq 2n - 2 \), the node in \( \beta_0 \) which computes \( c_k \) is denoted by \( c_k \).

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

\[ \text{op}(g) \subseteq B \text{ and } \text{resp}_0(g_2) = \bigvee_{a_j \in A'} a_j \]

where \( \emptyset \neq A' \subseteq A \).
The network \( \beta_1 \) is constructed such that the following hold:

i) \( \text{res}_{\beta_n}(u') \cup \text{res}_{\beta_1}(u'') \leq \text{res}_{\beta_0}(u) \).

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 \( \hat{u} \) with \( \text{op}(\hat{u}) = b_r \in B \) to an \( n \)-gate \( G \) which is not an \((*)\)-type-gate or to the node \( u'' \), there exists a node \( u \) with

\( \exists b_s \in B, b_s \neq b_r \) and \( \exists A_s \subseteq A, |A_s| \geq 2q \)

such that

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

Remark:

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

Now we shall construct \( \delta_n \). We distinguish three cases.

Case 1: \( u \) is an input node of \( \beta_0 \).

Then \( \delta_n \) consists of the nodes
\[
\begin{aligned}
\begin{cases}
    u' \text{ with } \text{op}(u') = \text{op}(u) \\
    u'' \text{ with } \text{op}(u'') = 0
\end{cases} & \text{ if } \text{op}(u) \in B \\
\begin{cases}
    u' \text{ with } \text{op}(u') = 0 \\
    u'' \text{ with } \text{op}(u'') = \text{op}(u)
\end{cases} & \text{ if } \text{op}(u) \in A
\end{aligned}
\]

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

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

Then \( \delta_u \) is constructed by

```
     v'        w'        v^n
    / \        / \        / \\
   u' \quad \quad u'' \quad u''
```

For this construction, we need no \( u \)-gate. Since properties i), ii) and iii) hold for \( \delta_v \) and \( \delta_w \), these properties also hold for \( \delta_u \).

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

We have to realize \( v'w', v'w'', v''w' \) and \( v''w'' \).

**Step 1:** Realization of \( v''w'' \)

Construct

```
     v''
  /    \
 u''   w''
```

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

\[ \exists b_s \in B, A_s \subseteq A \text{ and } b_r \in B, b_r \neq b_s, A_r \subseteq A \quad \text{such that} \quad
b_s \wedge (\bigvee_{a_j \in A_s} a_j) \vee b_r \wedge (\bigvee_{a_j \in A_r} a_j) \leq \text{res}_{B_1}(g). \]

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

If a node $g \in \{v', w'\}$ is not bipotent then either

\[ \text{res}_{B_1}(g) = 0 \quad \text{or} \quad \text{res}_{B_1}(g) \leq b_r \wedge (\bigvee_{a_j \in A_r} a_j) \]

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

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

a) $v'$ and $v''$ are bipotent.

Construct

```
        v'
       /|
      / \
     /   \v
    w'    u''
```

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

\[ \Rightarrow \]

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

\[ b_s \wedge (\bigvee_{a_j \in A_s} a_j) \leq \text{res}_{B_1}(v'w') \quad \text{for } A_s \subseteq A. \]

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

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

We distinguish two cases:

i) $|A_s| > 2q$.

Then we construct

$$\begin{array}{c}
\bullet \ b_s \\
\downarrow \\
\bullet \ a \\
\downarrow \\
\bullet \ u_2
\end{array}$$

where $\alpha$ is a network which computes $V$ using only $(|A_s| - 1) v$-gates.

ii) $|A_s| \leq 2q$.

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 $v''$, $A_s = A$ maximal
with $|A_s| \geq 2q$ and
such that
$\exists \lambda \left( V \mathbf{a}_j \right) \leq \text{res} \beta_n \left( v'w' \right)$
otherwise

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

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

If in the construction above
$u''_j$, $j \in \{1,2,3,4\}$
do not exist then we construct
$u''$ with $\text{op}(u''_j) = 0$ or $u''_j$ with $\text{op}(u''_j) = 0$.

Realization of $u'$ and $u''$:

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

For the construction of $\delta_u$, we need at most
four $\wedge$-gates.
Exercise

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

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

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

where $(4 - t)$ $\land$-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 $\beta_1$.

Lemma 3.3

In $\beta_1$, the following properties are fulfilled:

1. For all nodes $u \in \beta_0$ and the output nodes $u', u''$ of $\delta_n$, the following holds:
   a) $\text{res}_{\beta_1}(u') \lor \text{res}_{\beta_1}(u'') \leq \text{res}_{\beta_0}(u)$.
   b) If $\exists b_s \in B$, $A_s \subseteq A$ maximal, $A_s \neq \emptyset$ such that $b_s \land (\lor_{a_j \in A_s} a_j) \leq \text{res}_{\beta}(w')$ then $1A_s1 \geq 2q$.

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

3. On every path $P$ leading from a node $u$ with $\text{op}(u) = b_r \in B$ to an $\land$-gate $g$ which is not an $\land$-type $g$-gate or to $c_k$, $k \in \{0, 1, \ldots, 2n - 2\}$, there exists a node $w$ such that $\exists b_s \in B$, $b_s = b_r$ and $\exists A_s \subseteq A$, $1A_s1 \geq 2q$ such that $b_s \land (\lor_{a_j \in A_s} a_j) \leq \text{res}_{\beta}(w)$.
(4) \(0 \leq L^*_{n}(\beta_1) \leq 4 C^\wedge_{2m}(C_n) - m\) where \(L^*_{n}(\beta_1)\) is the number of \(n\)-gates in \(\beta_1\) and at most \(m\cdot 2q\) prime implicants of \(C_n\) have been destroyed.

**Proof:**
From the construction of \(\beta_1\), (4) and (3) follow directly. Property (2) follows from (1) (4) and the structure of \(C_k\), \(0 \leq k \leq 2n - 2\). As observed above, for each \(n\)-gate which is not used for the construction of \(\delta_n\), at most \(2q\) prime implicants have been destroyed. Hence, assertion (4) follows.

Using the network \(\beta_1\), we prove the following theorem.

**Theorem 3.9**
Let \(C^\wedge_{2m}(C_n)\) denote the minimal number of \(n\)-gates in a monotone network which computes \(C_n\). Then

\[C^\wedge_{2m}(C_n) \geq \left\lfloor \frac{1}{16} \min \left\{ \left( \frac{n^2}{q} - n \right), q^2 \right\} \right\rfloor\]

Setting \(q = n^{4/3}\), we obtain the following corollary.

**Corollary 3.2**

\[C^\wedge_{2m}(C_n) \geq \left\lfloor \frac{1}{16} \left( n^{4/3} - n \right) \right\rfloor\]

**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
\[ n^2 - \frac{1}{4} \left( \frac{n^2}{q} - n \right) 2q \geq \frac{n^2}{2} \]

prime implicants of \( C_n \) remain.

\[ \exists a_i \in A, \exists b_i \in B \text{ with } 1 \leq l \leq 9 \text{ such that } a_i b_i \leq \text{res}_{B_i} (C_{i+1}) \land b_i \in \hat{B} \]

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 \( a_i \in A \), the set \( A_i \) grows into the whole set \( A \) after fixing \( a_i \) to 1.

Then successively, we set each \( b_j \in \hat{B} \) 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 \( a_i b_j \in \hat{B} \) to 1, at least \( \frac{1}{2} q^2 \) \( \wedge \)-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 \( \beta_2 \) from \( \beta_1 \) by setting \( a_i, b_i, b_2, \ldots, b_{r-1}, 1 \leq r < q \) 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 $a: b_{e_1} b_{e_2} \ldots b_{e_{r-1}} \neq \text{res}_{p_1}(c_i + e_r)$ and $a: b_{e_r} \leq \text{res}_{p_1}(c_i + e_r)$ the following hold:

i) $\text{res}_{p_2}(c_i + e_r) \neq 1$ and
ii) $b_{e_r} \leq \text{res}_{p_2}(c_i + e_r)$.

Let $h$ be the node in $p_2$ with $\text{opt}(h) = b_{e_r}$. We consider all paths $P_1, P_2, \ldots, P_5$ with

a) Start node $h$ and end node $c_i + e_r$, and
b) $b_{e_r} \leq \text{res}_{p_2}(v)$ for all nodes $v$ on $P_3, P_4$.

Since $b_{e_r} \leq \text{res}_{p_2}(c_i + e_r)$ at least one such a path exists.

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

If on these paths $\frac{1}{4} q$ $\wedge$-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 (*)-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
If no $y$-gate on $P_j$ exists, there is a node $w$ such that

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

such that

$$b_{t_j} \wedge \left( \bigvee_{a_d \in A_j} a_d \right) \leq \res_{B_2} (c_{i+r})$$

and $q \in \text{succ}(w)$ (if $c_{i+r} \in \text{succ}(w)$, resp.)

$$\Rightarrow$$

$$\forall j = 1, \ldots, s \left( b_{t_j} \wedge \left( \bigvee_{a_d \in A_j} a_d \right) \right) \leq \res_{B_2} (c_{i+r})$$

and if no $y$-gate on $P_j$ exists then

$$b_{t_j} \wedge \left( \bigvee_{a_d \in A_j} a_d \right) \leq \res_{B_2} (c_{i+r})$$

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

Since less than $\frac{1}{2} q \wedge$-gates exist on $P_1, P_2, \ldots$, 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

$$\forall j = 1, \ldots, s \left( \bigvee_{a_d \in A_j} a_d \right) \leq \res_{B_2} (c_{i+r})$$
Since \( A, B \subseteq B' \) there exists at most one \( a_d \in A \) with \( a_d, b_j \in C_{iter} \)
(namely \( d = (i+e_r)-j \) if \( (i+e_r)-j \geq 0 \)), \( |B'| \leq 2q \) and \( |A_p| \geq 2q \), \( 1 \leq p \leq s \),
the following holds:

\[ \exists a_{dp} \in A_p \text{ such that } a_{dp} \land b_j \not\in C_{iter} \]

and hence,

\[ a_i \land_{b_j \in B'} b_j \supseteq \bigwedge_{p=1}^s a_{dp} \not\in C_{iter} \quad \text{and hence,} \]

But by construction

\[ a_i \land_{b_j \in B'} b_j \supseteq \bigwedge_{p=1}^s a_{dp} \not\in \text{res}_{P_1} (C_{iter}) \]

and by Lemma 3.3 property (a) \( a_{dp} \), \( \text{res}_{P_1} (C_{iter}) \)

is a subfunction of \( C_{iter} \), a contradiction.

\[ \Rightarrow \quad \text{On } P_1, P_2, \ldots, P_s \text{ at least } \frac{1}{2}q \text{ n-gates exist.} \]

Hence, the claim and therefore Theorem 3.8 is proved.

Grinchuk and Sergeev have improved the lower bound
for the number of n-gates to \( \Omega(n^2 \log \log^2 n) \). 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 circular matrices (see also Jukna, pp. 386 - 390).