2. Lower bounds on the network complexity of Boolean functions

References

- Claus-Peter Schnorr, Zwei lineare untere Schranken für die Komplexität Boolescher Funktionen, Computing 13 (1974), 155 - 171.
- Wolfgang Paul, A 2.5n-lower bound on the combinational complexity of boolean functions, SIAM J. Comput. 6 (1977), 427 - 443.
- Larry Stockmeyer, On the combinational complexity of certain symmetric Boolean functions, Math. Systems Theory 10 (1977), 323 - 336.
- Norbert Blum, A Boolean function requiring 3n network size, TCS 28 (1984), 337 - 345.
- Uri Zwick, A 4n lower on the combinational complexity of certain symmetric Boolean functions over the basis of unate dyadic Boolean functions, SIAM J. Comput. 20 (1991), 499 - 505.

Although the network complexity of almost all Boolean functions is exponential, no nonlinear lower bound of an explicit defined Boolean function in NP is known. We shall present some of the few known linear lower bound

Proofs. First we consider the base B_2.

A function $f \in B_n$ is called <u>degenerated</u> if f does not depend on all variables $x_1, x_2, \ldots, x_n$. This means that there is a variable x_i such that the subfunctions of f after setting $x_i := 0$ and $x_i := 1$ are equal.

<u>Theorem 2.1</u>

For each nondegenerated function $f \in B_n$,

$C_{B_2}(f) \geq n-1$.

<u>Proof</u>:

Let β be an optimal B_2-network for f; i.e., f is computed at the output node of β.

<u>Observation</u>:

- The outdegree of each input node x_i in β is at least one. (Otherwise, f would not depend on x_i.)
- Each gate which is not the output node has outdegree at least one. (Otherwise, we could eliminate this gate without changing the function computed at the output node. This cannot happen in an optimal network.

Both observations imply that the algorithm has to construct a path from n nodes to the output node. A gate with indegree one cannot reduce the number of nodes for which a path to the output node have to be constructed. A gate with indegree

two reduces the number of nodes for which a path to the output node have to be constructed at most by

$$\begin{cases} \text{one} & \text{if } g \text{ is not the output node} \\ \text{two} & \text{if } g \text{ is the output node.} \end{cases}$$

$\Rightarrow$

The number of gates of indegree two is at least $n-1$.

■

The proof of Theorem 2.1 is only based on the graph structure of the network β. There are nondegenerated functions in B_n which have network size $n-1$ (e.g., $f: \{0,1\}^n \to \{0,1\}$ with $f(x_1, x_2, \ldots, x_n) = \bigwedge_{i=1}^{n} x_i$). Hence, we cannot obtain a better lower bound in this way such that for proving larger lower bounds further techniques are needed.

All known proofs of larger lower bounds use the so-called <u>gate-elimination method</u>. This method uses induction. By an assignment of some variables with values from $\{0,1\}$, a specific number of gates is eliminated in each step and the resulting function is of the same type as the function before the assignment. We will demonstrate the gate-elimination method by proving some lower bounds. First, we need a notation.

A Boolean function $f \in B_n$ belongs to the class $Q^n_{2,3}$ if for all different $i,j \in \{1,2,\dots,n\}$ the following is fulfilled:

i) After the replacement of x_i and x_j by constants, we obtain at least three different subfunctions.

ii) If $n \geq 4$ then there is a constant c_i such that after the replacement of x_i by c_i a subfunction in $Q^{n-1}_{2,3}$ is obtained.

Schnorr has defined the class $Q^n_{2,3}$ in a way such that for each function in this class a lower bound $2n-3$ can be proved easily

Theorem 2.2

For all $f \in Q^n_{2,3}$ there holds $C_{B_2}(f) \geq 2n-3$.

Proof:

Observe that each function $f \in Q^n_{2,3}$ is nondegenerated. If f would not depend on x_i then f could have at most two different subfunctions with respect to x_i and any $x_j \neq x_i$.

Let β be an optimal B_2-network for f. Consider a gate v in β such that both direct predecessors of v are input or negated input nodes. Obviously, such a gate exists.

Optimality of $\beta \Rightarrow$

Both input nodes correspond to different variables x_i and x_j.

If all nodes on the paths from both input nodes to v have outdegree one then f depends on x_i and x_j only via the gate v. v can only compute 0 or 1.

$\Rightarrow$

f can have at most two different subfunctions with respect to x_i and x_j. This contradicts the definition of $Q^n_{2,3}$.

$\Rightarrow$

At least one of the two paths from the input node to v splits.

W.l.o.g. we can assume that the path from x_i to v splits. We distinguish two cases.

Case 1: $n = 3$.

Then then the network has to construct paths from at least four nodes to the output node. Analogous to the proof of Theorem 2.1, we can show the existence of at least three gates in β.

exercise

Case 2: $n > 3$

Assume that the assertion is proved for $\ell < n$. Replace x_i by c_i.

$\Rightarrow$

- At least two successors of the input node x_i can be eliminated.

- The resulting network β' computes a function in $Q_{2,3}^{n-1}$

$\Rightarrow$

assumption β' contains at least $2(n-1)-3$ gates.

$\Rightarrow$

β contains at least $2n-3$ gates. ∎

Example:

Counting-function

$C_{k,m}^{n} : \{0,1\}^n \to \{0,1\}$ where

$$C_{k,m}^{n}(x_1, x_2, \ldots, x_n) := \begin{cases} 1 & \text{if } \sum_{i=1}^{n} x_i = k \bmod m \\ 0 & \text{otherwise} \end{cases}$$

Let us consides $C_{k,3}^{n}$. After the replacement of x_i and x_j by constants, we obtain the following different subfunctions:

$$\begin{cases} C_{k,3}^{n-2} & \text{if } x_i + x_j = 0 \\ C_{k-1,3}^{n-2} & \text{if } x_i + x_j = 1 \\ C_{k-2,3}^{n-2} & \text{if } x_i + x_j = 2 \end{cases}$$

For $n \geq 4$, we define $c_i := 0$ for all i. After the replacement of x_i by c_i, we obtain the subfunction $C_{k,3}^{n-1}$ which is in $Q_{2,3}^{n-1}$.

◆

The so-called storage access function $SA_n \in B_{n+k}$ where $n = 2^k$ is defined on a k-bit number $a = a_1 a_2 \dots a_k$ and n variables $x_1, x_2, \dots, x_n$ by

$$SA_n(a, x) := x_{(a)} \quad \text{where}$$

(a) is the binary number represented by $a + 1$.

Wolfgang Paul has proved a $2n - 2$ lower bound for the B_2-complexity of SA_n.

Theorem 2.3

$$C_{B_2}(SA_n) \geq 2n - 2.$$

Proof:

Our goal is to give an inductive proof. But the replacement of x-variables does not lead to storage access functions. Hence, we define a larger class of functions.

Let $s \in \{1, 2, \dots, n\}$. F_s contains all functions $f \in B_{n+k}$ such that there exists $S \subseteq \{1, 2, \dots, n\}$ where $|S| = s$ such that for all a with $(a) \in S$

$$f(a, x_1, x_2, \dots, x_n) = x_{(a)}.$$

Since $SA_n \in F_n$, it suffices to prove a $2s-2$ lower bound for all functions in F_s.

$s = 1$: trivial since $2s - 2 = 0$.

Let $s \geq 2$. Assume that the lower bound holds for $\ell < s$; i.e., $C_{B_2}(f) \geq 2\ell - 2 \quad \forall f \in F_\ell$.

Consider any function $f \in F_s$. Let β be an optimal B_2-network for f. We prove that we obtain a network for a subfunction in F_{s-1} after the elimination of at least two gates. Three cases are possible.

Case 1: $\exists\, i \in S$: input node x_i has outdegree ≥ 2.

Setting x_i to a constant eliminates at least two gates. Replace S by $S \setminus \{i\}$. The resulting network computes a function $f' \in F_{s-1}$. By assumption

$$C_{B_2}(f') \geq 2(s-1) - 2.$$

$$\Rightarrow \quad C_{B_2}(f) \geq 2s - 2.$$

Case 2: $\exists\, i \in S$: input node x_i has outdegree one and the edge from x_i goes into an $\wedge$-type gate v; i.e., $res_\beta(v) = (x_i^b \wedge g^c)^d$ with $b, c, d \in \{0,1\}$ and g is a function of the other input variables.

If we replace x_i by $\bar{b}$ then the output of v is replaced by the constant 0^d. Replacing S by $S \setminus \{i\}$, we see that the resulting network computes a function $f' \in F_{s-1}$. Hence, i cannot be the output gate.

At least the gate v and its successor have been eliminated. By assumption

$$C_{B_2}(f') \geq 2(s-1) - 2.$$

Altogether we obtain

$$C_{B_2}(f) \geq 2s-2.$$

Case 3: $\exists i \in S$: input node x_i has outdegree one and the edge from x_i goes into a $\oplus$-type gate v.

Then $res_B(v) = x_i \oplus g \oplus b$ for some function g of the other variables and $b \in \{0,1\}$.

Furthermore, v cannot be the output node. To see this consider $j \in S \setminus \{i\}$ and $(a) = j$. The network has to compute x_j independently from the value of x_i. A change of the value of x_i changes the result of the gate v.

The value of x_i has no influence on the output for all $(a) \in S \setminus \{i\}$.

$\Rightarrow$

After the replacement of x_i by an arbitrary function we obtain a network for a function $f' \in F_{s-1}$.

$\rightsquigarrow$

Choose the function g to replace x_i. Then the gate v computes the constant b.

$\Rightarrow$

v and its successors are eliminated.

As in Case 1, we obtain

$$C_{B_2}(f) \geq 2s-2.$$

■

Wolfgang Paul has also proved a 2.5 n lower bound for the B_2-complexity of the function

$$f: \{0,1\}^{n+2\log n+1} \longrightarrow \{0,1\}$$

defined in the following way:

Let

$$\sigma_1 := a_1, a_2, \dots, a_{\log n}$$

$$\sigma_2 := a_{\log n+1}, \dots, a_{2\log n}$$

and

$$f(a_1, a_2, \dots, a_{2\log n}, q, x_1, x_2, \dots, x_n) = \begin{cases} x_{(\sigma_1)} \wedge x_{(\sigma_2)} & \text{if } q = 1 \\ x_{(\sigma_1)} \oplus x_{(\sigma_2)} & \text{if } q = 0. \end{cases}$$

Larry Stockmeyer has applied the ideas of Paul to prove a 2.5 n lower bound for several fundamental symmetric functions.

A function $f \in B_{n,m}$ is <u>symmetric</u> if

$$f(x_1, x_2, \dots, x_n) = f(x_{\pi(1)}, x_{\pi(2)}, \dots, x_{\pi(n)})$$

for all permutations π of $1, 2, \dots, n$.

I have considered the function

$$f: \{0,1\}^{n+3\log n+1} \longrightarrow \{0,1\}$$

where

$$f(a_1, a_2, \dots, a_{3\log n}, r, x_1, x_2, \dots, x_n) = x_{(\sigma_1)}^{r} \wedge (x_{(\sigma_2)} \oplus x_{(\sigma_3)})$$

with $\sigma_3 := a_{2\log n+1}, a_{2\log n+2}, \ldots, a_{3\log n}$.

We have extended the method of Paul and proved

$$C_{B_2}(f) \geq 3n-3.$$

Note that the function f is similar to Paul's function.

- If $r=1$ and $x_{(\sigma_1)}=1$ then

$$f(\sigma_1,\sigma_2,\sigma_3,r,x_1,x_2,\ldots,x_n) = x_{(\sigma_2)} \oplus x_{(\sigma_3)}.$$

- If $r=1$ and $x_{(\sigma_3)}=0$ then

$$f(\sigma_1,\sigma_2,\sigma_3,r,x_1,x_2,\ldots,x_n) = x_{(\sigma_1)} \wedge x_{(\sigma_2)}.$$

We will present the proof of this lower bound. Throughout the proof, we will use the following lemma.

Lemma 2.1

Let β be a network computing $f \in B_n$. Let $v \in \beta$ be an $\wedge$-type gate or a $\oplus$-type gate. If one input function of v is constant then we can eliminate the gate v and the reduced network still computes f.

Proof:

exercise

■

Let $U \subset V_n$ and $\alpha: U \to \{0,1\}$ be a mapping. Frequently, we will consider the restriction f_α of $f \in B_n$

under the assignment α. More precisely, f_α is defined by

$$f_\alpha(x_1, x_2, \ldots, x_n) = f(y_1, y_2, \ldots, y_n) \text{ with}$$

$$y_i := \begin{cases} \alpha(x_i) & \text{if } x_i \in U \\ x_i & \text{if } x_i \notin U \end{cases}$$

In a natural way, an assignment α associates with a network $\mathcal{B}$ a subnetwork $\mathcal{B}_\alpha$ which is obtained by fixing the input variables according to α and eliminating the unnecessary gates.

In the sequel, we write $\text{res}(v)$ for $\text{res}_{\mathcal{B}}(v)$ if $\mathcal{B}$ is kept fixed. $\#S$ denotes the cardinality of the set S. For proving the lower bound we will consider paths in a network. $(v \Rightarrow u)$ denotes a path from node v to node u. $(v \Rightarrow u[$ denotes a path from node v to a node in $\text{pred}(u)$.

Structure of the proof:

Step 1: Make f independent of input variables which allow the elimination of three gates.

Step 2: Prove for the remaining network without an inductive argument the existence of enough gates.

First we describe Step 1.

Define for $1 \leq s \leq n$ the following statement E_s:

E_s: $C_{B_2}(f) \geq 3s-3$ for all functions

$$f: \{0,1\}^{n+3\log n+1} \to \{0,1\}$$

with the property

[$\exists\, S \subseteq \{1,2,\dots,n\}$, $\#S = s$ such that for $\sigma_1, \sigma_2, \sigma_3$ with $(\sigma_1), (\sigma_2), (\sigma_3) \in S$:

$$f(\sigma_1, \sigma_2, \sigma_3, r, x_1, x_2, \dots, x_n) = x_{(\sigma_1)}^{r} \wedge (x_{(\sigma_2)} \oplus x_{(\sigma_3)})$$

Since $3\cdot 1 - 3 = 0$, E_1 is trivially true.

Let $s \geq 2$ and assume that E_{s-1} is true. Now we prove $E_{s-1} \Rightarrow E_s$.

Let β be any optimal B_2-network for f. W.l.o.g., we can assume that for each $i \in S$ there is an unique node $u \in \beta$ with $op(u) = x_i$.

Case 1: $\exists\, i \in S$ such that $\#\,suc(x_i) \geq 3$.

By fixing x_i at 0, we can eliminate at least three gates of β. The reduced network computes the restriction $f_{x_i:=0}$ of f.

Inductive assumption $\Rightarrow C_{B_2}(f_{x_i:=0}) \geq 3(s-1)-3$

$$\Rightarrow \quad C_{B_2}(f) \geq 3s-3.$$

Case 2: $\exists\, i \in S$ such that $\#\,suc(x_i) = 2$ and $\exists\, v \in suc(x_i)$ such that v is an n-type gate

Choose $c \in \{0,1\}$ such that $res(v)_{x_i:=c}$ is constant.

Fixing x_i at c eliminates all nodes in $suc(x_i)$ and all nodes in $suc(v)$.

Optimality of β $\Rightarrow$ These are at least three gates.

The reduced network computes the restriction $f_{x_i:=c}$ of f. Hence, by the inductive hypothesis

$$C_{B_2}(f) \geq 3s-3.$$

Case 3: $\neg$ (Case 1 or Case 2) and
$\exists i \in S$ such that $\forall v \in suc(x_i)$ v is an $\oplus$-type gate.

We distinguish two subcases.

3.1 $\#\, suc(x_i) = 1$.

Then there exist nodes $u_1, u_2, \ldots, u_r$ in β with

1) $u_1 \in suc(x_i)$,
2) u_j is a $\oplus$-type gate for $1 \leq j \leq r$,
3) $u_{j+1} \in suc(u_j)$ and $\#\, suc(u_j) = 1$ for $1 \leq j < r$ and
4) $\#\, suc(u_r) > 1$ or for $w \in suc(u_r)$ there holds: w is an $\wedge$-type gate.

Let

x_i, g_1 input functions of gate u_1
$res(u_j), g_{j+1}$ input functions of gate u_{j+1}, $1 \leq j < r$

Then there holds

$res(u_r) = x_i \oplus g$ where

$g = g_1 \oplus g_2 \oplus \dots \oplus g_r$ does not depend on x_i.
Note that β is acyclic.

$\Rightarrow$

$res(u_r)_{x_i := g}$ and $res(u_r)_{x_i := \neg g}$ are constant.

$\Rightarrow$

After the substitution of x_i by g or by $\neg g$, we can eliminate the following gates:

- $u_1, u_2, \dots, u_r$ and all gates in $suc(u_r)$.

g and $\neg g$, respectively can be computed using $r-1$ additional gates.

Two subcases are possible.

3.1.1 $\# suc(u_r) \geq 2$

Then we eliminate at least $r+2$ gates by fixing x_i at g or at $\neg g$.

3.1.2 $\# suc(u_r) = 1$.

Then the unique $w \in suc(u_r)$ is an $\wedge$-type gate.
Choose $\tilde{g} \in \{g, \neg g\}$ such that

$res(w)_{x_i := \tilde{g}}$ is constant.

$\Rightarrow$

After fixing x_i at $\tilde{g}$, we can eliminate at least $r+2$ gates, namely $u_1, u_2, \dots, u_r, w$ and all nodes in $suc(w)$.

In both subcase, an application of the induction hypothesis gets

$$C_{B_2}(f) \geq 3s - 3.$$

3.2 $\#\, suc(x_i) = 2$.

Then both successors u_1 and v_1 of x_i are $\oplus$-type gates.

Then there exist nodes $u_1, u_2, \ldots, u_r, v_1, v_2, \ldots, v_e \in B$ with

1) $u_1 \in suc(x_i)$,
2) u_j is a $\oplus$-type gate for $1 \leq j < r$,
3) $u_{j+1} \in suc(u_j)$ and $\#\, suc(u_j) = 1$ for $1 \leq j < r$ and
4) $\#\, suc(u_r) > 1$ or $w \in suc(u_r)$ is an $\wedge$-type gate

and

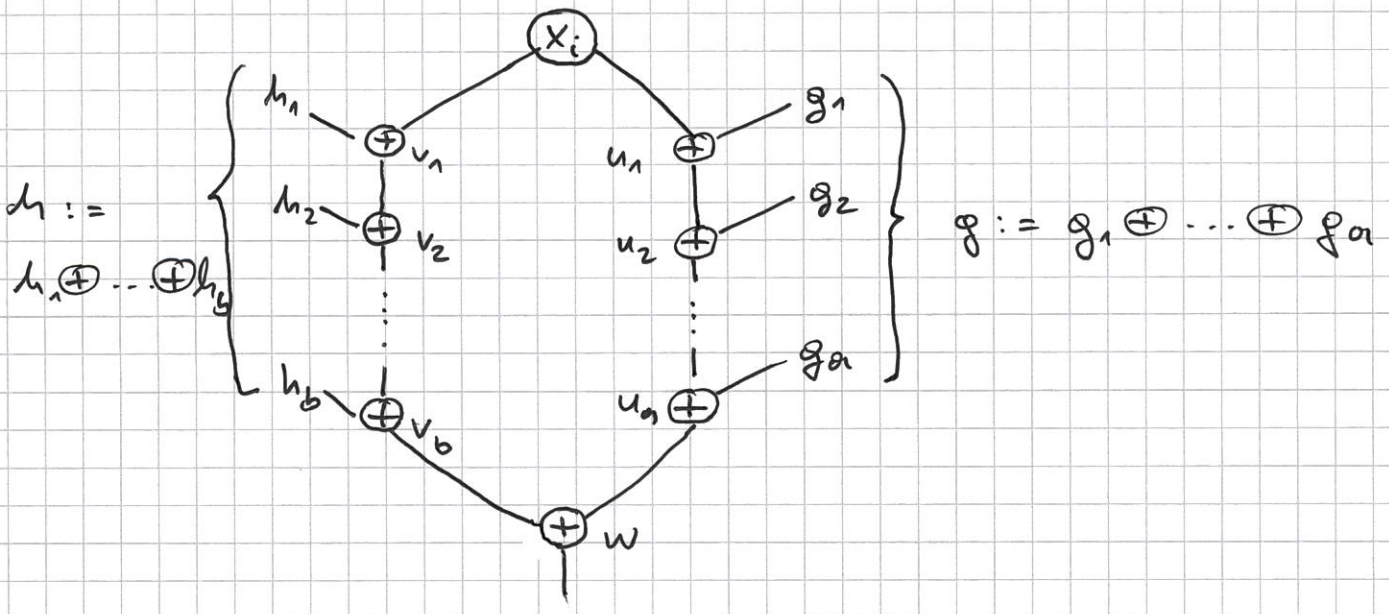
1) $v_1 \in suc(x_i) \setminus \{u_1\}$,
2) v_j is a $\oplus$-type gate for $1 \leq j < e$,
3) $v_{j+1} \in suc(v_j)$ and $\#\, suc(v_j) = 1$ for $1 \leq j < e$ and
4) $\#\, suc(v_e) > 1$ or $w \in suc(v_e)$ is an $\wedge$-type gate.

Claim: $\{u_1, u_2, \ldots, u_r\} \cap \{v_1, v_2, \ldots, v_e\} = \emptyset$.

Proof:

Assume that $\{u_1, u_2, \ldots, u_r\} \cap \{v_1, v_2, \ldots, v_e\} \neq \emptyset$.

Then we have the following situation:



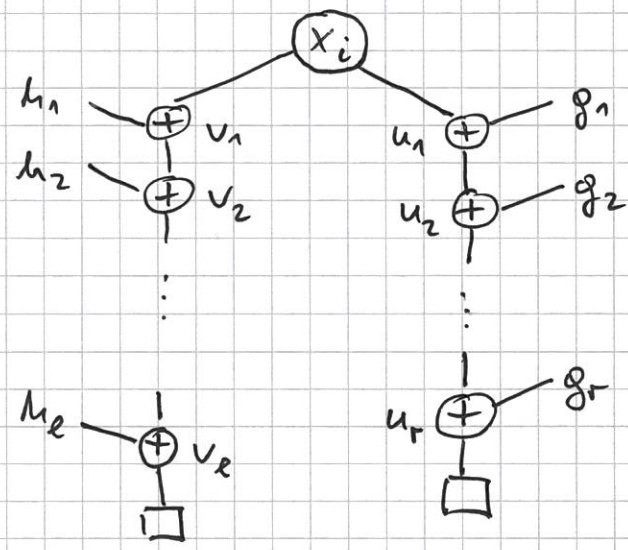
Then $res(w) = h \oplus g \oplus x_i \oplus x_i$

$= h \oplus g \oplus 0$

$\Rightarrow$ f does not depend on x_i, a contradiction.

□

Therefore, we have the following situation:



β acyclic $\Rightarrow$ $g_1, g_2, \dots, g_r$ do not depend on x_i

or

$h_1, h_2, \dots, h_e$ do not depend on x_i.

Exercise

Prove the following statement:

$\exists p, q$ such that h_p and g_q depend on x_i

$\Rightarrow \beta$ contains a cycle.

W.l.o.g., we can assume that $g_1, g_2, \ldots, g_r$ do not depend on x_i.

Now we can prove $C_{B_2}(f) \geq 3s-3$ as in Case 3.1.

Exercise

Finish the proof of Subcase 3.2.

Assume that none of the cases 1-3 arises. Then for all $i \in S$ the following holds:

a) $\# \, suc(x_i) = 1$. We denote the unique node in $suc(x_i)$ by G_i.

b) G_i is an $\wedge$-type gate.

Let

$$G := \{ G_i \mid i \in S \}.$$

Lemma 2.2

$\forall \, i, j \in S$ with $i \neq j$ there hold $G_i \neq G_j$.

Proof:

Suppose $\exists \, i, j \in S$, $i \neq j$ with $G_i = G_j$.

Then there exists $c \in \{0,1\}$ such that

$$res(G_j)_{x_i := c} \text{ is constant.}$$

$\Rightarrow$ $f_{x_i := c}$ does not depend on x_j.

But $f(\sigma_1, \sigma_2, \sigma_3, \tau, x_1, \dots, x_n) = c \oplus x_j$ for

$(\sigma_1) \neq i, j$, $\tau = 1$, $x_{(\sigma_1)} = 1$, $(\sigma_2) = i$,
$x_i = c$ and $(\sigma_3) = j$

depends on x_j, a contradiction.

□

Case 4: $\exists\, i \in S$ such that $\#\,\mathrm{suc}(G_i) \geq 2$.

Consider $c \in \{0,1\}$ with $\mathrm{res}(G_i)_{x_i := c}$ is constant. Then fixing x_i at c eliminates G_i and all gates in $\mathrm{suc}(G_i)$. Hence, by the induction hypothesis

$$C_{B_2}(f) \geq 3s - 3.$$

It remains to consider the following case.

Case 5: $\forall\, i \in S$ there holds $\#\,\mathrm{suc}(G_i) = 1$.

We denote the unique direct successor of G_i by Q_i. Let

$$Q := \{ Q_i \mid i \in S \}.$$

Notations:

A path $(v \Rightarrow w)$ in β is called free if no node $\neq v, w$ is in $\mathcal{G}$. A node w in β is called split if $\mathrm{outdegree}(w) \geq 2$. Let t be the node in β with $\mathrm{res}(t) = f$. A split w in β is called a free split

if there are nodes u_1, u_2 in β, $u_1 \neq u_2$ such that

a) $u_1, u_2 \in Suc(w)$,

b) $\exists$ free paths $(u_1 \Rightarrow t)$ and $(u_2 \Rightarrow t)$ in β.

A node w is called the <u>collector</u> of the free paths $(G_i \Rightarrow t)$ and $(G_j \Rightarrow t)$, $i \neq j$ if w is the first node which is on both paths.

<u>Lemma 2.3</u>

$\forall i \in S$ there exists a free path $(G_i \Rightarrow t)$.

<u>Proof:</u>

Suppose that no free path $(G_i \Rightarrow t)$ exists.

$\Rightarrow$

Each path $(G_i \Rightarrow t)$ passes some G_j, $j \neq i$.

Construct the assignment α by fixing all variables except x_i such that

i) $res(G_v)_\alpha$ is constant $\forall v \in S \setminus \{i\}$.

ii) $f_\alpha = x_i^a$, $a \in \{0,1\}$.

Since each path $(G_i \Rightarrow t)$ goes through some G_v with $v \neq i$, $res(t)$ does not depend on x_i.

But this is a contradiction to

$$f_\alpha = x_i^a \quad \text{and} \quad res(t)_\alpha = f_\alpha.$$

□

Note that Lemma 2.3 implies $G \cap Q = \emptyset$.

Lemma 2.4

Let $i, j \in S$, $i \neq j$. Let C be the collector of a free path $(G_i \Rightarrow t)$ and a free path $(G_j \Rightarrow t)$. If $\nexists$ free split $\neq C$ on the path $(G_i \Rightarrow C)$ and $\nexists$ free split $\neq C$ on the path $(G_j \Rightarrow C)$ then the following hold:

i) C is a $\oplus$-type gate, and
ii) $\exists$ free path $(G_i \Rightarrow G_j)$ or $\exists$ free path $(G_j \Rightarrow G_i)$

Proof:

Suppose that the assertion does not hold. We distinguish two cases.

Case 1: C is a $\oplus$-type gate.

Then by assumption, all paths $(G_i \Rightarrow G_j)$ and $(G_j \Rightarrow G_i)$ are not free.

Construct an assignment α by fixing all variables except x_i, x_j such that

a) $\text{res}(G_v)_\alpha$ is constant $\forall\, v \in S \setminus \{i, j\}$.
b) $f_\alpha = x_i \wedge x_j^a$, $a \in \{0, 1\}$

By assumption, all paths $(G_i \Rightarrow t)$ and $(G_j \Rightarrow t)$ go through C or a G_v, $v \in S \setminus \{i, j\}$.

Since there is no free path $(G_i \Rightarrow G_j)$ and no free path $(G_j \Rightarrow G_i)$, $\text{res}(G_i)_\alpha$ does not depend on x_j and $\text{res}(G_j)_\alpha$ does not depend on x_i.

Hence, $res(C)_\alpha = (x_i \oplus x_j)^b$, $b \in \{0,1\}$ or $res(C)_\alpha$ depends on at most one variable, and so the same holds for $res(t)_\alpha$.

$\Rightarrow$ $f_\alpha \neq res(t)_\alpha$, a contradiction.

Case 2: C is an $\wedge$-type gate.

Construct an assignment α by fixing all variables except x_i, x_j such that

a) $res(G_v)_\alpha$ is constant $\forall\, v \in S \setminus \{i,j\}$.

b) $f_\alpha = x_i \oplus x_j$

(Here, we need the control variable r for forcing that such an assignment α exists.)

If $res(G_j)_\alpha$ depends on x_i, choose $c \in \{0,1\}$ such that $res(G_j)_{\alpha, x_i := c}$ is constant.
Hence, $res(t)_{\alpha, x_i := c}$ does not depend on x_j.
But $f_{\alpha, x_i := c}$ depends on x_j, a contradiction.

The case $res(G_i)_\alpha$ depends on x_j is symmetric.

$\Rightarrow$ $res(C)_\alpha = (x_i^a \wedge x_j^b)^c$, $a,b,c \in \{0,1\}$

or

$res(C)_\alpha$ depends on at most one variable.

By assumption, all paths $(G_i \Rightarrow t)$ and $(G_j \Rightarrow t)$ go through C or a G_v, $v \in S \setminus \{i,j\}$
Hence, the same holds for $res(t)_\alpha$ as for $res(C)_\alpha$.

$\Rightarrow$ $f_\alpha \neq res(t)_\alpha$, a contradiction. □

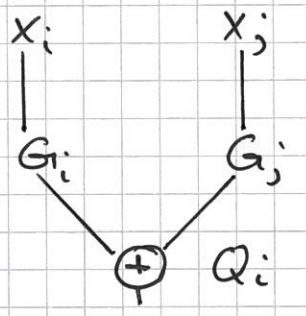
Lemma 2.5

$\forall\, i,j \in S$ holds: $i \neq j \Rightarrow Q_i \neq Q_j$.

Proof:

Suppose that $\exists\, i,j$, $i \neq j$ but $Q_i = Q_j$. Then Q_i is the collector of the free paths $(G_i \Rightarrow t)$ and $(G_j \Rightarrow t)$.

Lemma 2.4 $\Rightarrow$ Q_i is a $\oplus$-type gate.
Hence, we have the following situation:



Lemma 2.4 $\Rightarrow$

$\exists$ free path $(G_i \Rightarrow G_j)$ or $\exists$ free path $(G_j \Rightarrow G_i)$.

This is impossible since both nodes G_i and G_j have outdegree one. □

Now we have isolated 2s gates, namely the gates G_i, Q_i for $i \in S$. Our goal is to isolate s-3 further gates.

Lemma 2.6

There are at least $s-1$ mutually distinct splits in β.

Proof:

Let $S' := S$. Choose $i, j \in S'$, $i \neq j$ with free paths $(G_i \Rightarrow t)$ and $(G_j \Rightarrow t)$ such that

- $\forall\, \ell \in S' \setminus \{i, j\}$, no free path $(G_\ell \Rightarrow t)$ goes through the collector C of the free paths $(G_i \Rightarrow t)$ and $(G_j \Rightarrow t)$.

Exercise:

Show that such a choice is possible.

Lemma 2.4 $\Rightarrow$

One of the paths $(G_i \Rightarrow C[$ and $(G_j \Rightarrow C[$, respectively splits into a free path to the output node t or splits into a free path to the node G_j and G_i, respectively.

W.l.o.g., let the path $(G_i \Rightarrow C[$ split. Set

$$S' := S' \setminus \{i\}.$$

Construction $\Rightarrow$

No free path $(G_\ell \Rightarrow t)$, $\ell \in S'$ has a common node with the path $(G_i \Rightarrow C[$.

Repeating the arguments above $s-1$ times proves the lemma.

□

Since we have at least $s-1$ splits, we have to connect at least $2(s-1)$ edges with the output node t.

Definition of a free path $\Rightarrow$

No node in G can be used to connect edges on free paths.

Claim:

All but one of the $s-1$ splits from Lemma 2.6 have to be free.

Proof:

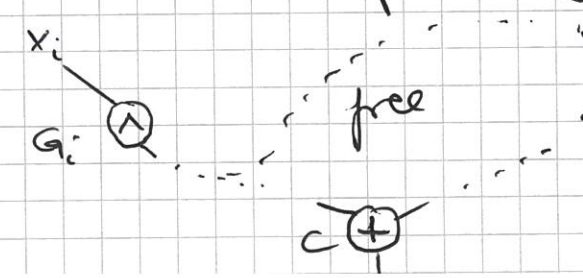
Assume that less than $s-1$ splits isolated in the proof of Lemma 2.6 are free.

Lemma 2.4 $\Rightarrow$

$\exists\, i,j \in S,\ i \neq j$ with the following property:

- Let C be the collector of the free paths $(G_i \Rightarrow t)$ and $(G_j \Rightarrow t)$.
  Then there is no free split on the path $(G_i \Rightarrow C[$ and no free split on the path $(G_j \Rightarrow C[$.
  There is a free path $(G_i \Rightarrow G_j)$ and C is a $\oplus$-type gate.

Then we have the following situation:



Lemma 2.7

$\forall u \in S \setminus \{j\}$ $\exists$ free path $(G_u \Rightarrow G_i)$ or $\exists$ free path $(G_u \Rightarrow G_j)$.

Proof:

For i we know by assumption that $\exists$ free path $(G_i \Rightarrow G_j)$.

Assume that $\exists\, \ell \in S \setminus \{i,j\}$ with
$\#$ free path $(G_\ell \Rightarrow G_i)$ and $\#$ free path $(G_\ell \Rightarrow G_j)$.

Now we construct an assignment α by fixing all variables except x_i, x_j, x_ℓ such that

a) $\operatorname{res}(G_u)_\alpha$ is constant $\forall\, u \in S \setminus \{i,j,\ell\}$

b) $f_\alpha = x_j \wedge (x_i \oplus x_\ell)$

We distinguish two cases.

Case 1: $\operatorname{res}(G_j)_\alpha$ does not depend on x_i.

Now we fix x_ℓ at $a \in \{0,1\}$ such that

$$\operatorname{res}(G_\ell)_{\alpha,\, x_\ell := a} \quad \text{is constant.}$$

$\Rightarrow$

$$f_{\alpha,\, x_\ell := a} = x_j \wedge x_i^b \,, \quad b \in \{0,1\}.$$

Now, as in the proof of Lemma 2.4 we show that Case 1 cannot happen.

Case 2: $\operatorname{res}(G_j)_\alpha$ depends on x_i.

Then
$$\operatorname{res}(G_j)_\alpha = (x_i^c \wedge x_j^d)^e \,, \quad c,d,e \in \{0,1\}.$$

Fix x_i at $\neg c$. Then $res(G_j)_{\alpha, x_i := \neg c}$ is constant.

$\Rightarrow$

$res(t)_{\alpha, x_i := \neg c}$ does not depend on x_j.

But

$$f_{\alpha, x_i := \neg c} = x_j \wedge (\neg c \oplus x_e) = x_j \wedge x_e^c$$

depends on x_j, a contradiction.

□

Now we prove that all other splits isolated in the proof of Lemma 2.6 have to be free.

Lemma 2.8

For all $\ell, v \in S \setminus \{j\}$, $v \neq \ell$ the following holds:
If D is the collector of a free path $(G_\ell \Rightarrow t)$ and a free path $(G_v \Rightarrow t)$ then
$\exists$ free split $\neq D$ on path $(G_\ell \Rightarrow \cdot)$ or
$\exists$ free split $\neq D$ on path $(G_v \Rightarrow D)$.

Proof:
Suppose that
$\nexists$ free split $\neq D$ on path $(G_\ell \Rightarrow D)$ and
$\nexists$ free split $\neq D$ on path $(G_v \Rightarrow D)$.

Lemma 2.4 $\Rightarrow$
There is a free path $(G_\ell \Rightarrow G_v)$ or there is a free path $(G_v \Rightarrow G_\ell)$ and D is a $\oplus$-type gate.
Hence, we can apply Lemma 2.7 to ℓ and v.

$\Rightarrow$

$\exists$ a path $(G_j \Rightarrow G_e)$ or $\exists$ a path $(G_j \Rightarrow G_v)$.

But by construction there exists paths $(G_e \Rightarrow G_j)$ and $(G_v \Rightarrow G_j)$ and, hence, we have a cycle in the network. But this cannot happen by the definition of a network.

□

From Lemma 2.8, we can directly derive the following lemma.

Lemma 2.9

There are at least $s-2$ mutually distinct free splits in β.

Exercise

Prove Lemma 2.9.

Now we can count the gates in β.

Lemma 2.9 and Lemma 2.3 $\Rightarrow$

We have to connect at least $2(s-2)+2$ edges on free paths to the output node t. Since the paths are free, the nodes in G cannot help for doing this. For the nodes in Q, only one input wire is free for connecting these edges. There are s nodes in Q. Hence, at least $s-2$ edges have to be connected to the output node using new nodes.

One new node (except the output node) can

decrease the number of edges only by one. The output node can decrease the number of edges only by two.

$\Rightarrow$

We need at least $s-3$ new gates.

Altogether, we have obtained

$$C_{B_2}(f) \geq \#G + \#Q + s - 3 = 3s - 3.$$

This concludes the proof of the following theorem.

Theorem 2.4

Let $f: \{0,1\}^{n+3\log n+1} \to \{0,1\}$ be defined by

$$f(a_1, a_2, \dots, a_{3\log n}, r, x_1, x_2, \dots, x_n) = x_{(a_1)}^{r} \wedge (x_{(a_2)} \oplus x_{(a_3)}).$$

Then $C_{B_2}(f) \geq 3n - 3$.

Magnus Gausdal Find, Alexander Golovnev, Edward A. Hirsch, Alexander S. Kulikov,
A better-than-3n lower bound for the circuit complexity of an explicit function, ECCC, Revision of Report No. 166 (2015), February 16, 2016 (37 pages)

have shown a $(3 + \frac{1}{86})n - o(n)$ lower bound for another function by a much more complicated proof.

Note that $x \oplus y = x \wedge \bar{y} \vee \bar{x} \wedge y$. Hence, each $\oplus$-type gate can be realized using three gates

from $\{\wedge, \vee\}$ and some negations. Therefore, if we restrict us to the base $\Omega_0 = \{\wedge, \vee, \neg\}$, we can realize a B_2-network by an Ω_0-network increasing the size of the network at most by the constant factor three. Hence, for proving a super-linear lower bound for the network complexity of a Boolean function, it suffices to prove this with respect to the base Ω_0. Such a super-linear lower bound would imply a super-linear lower bound for the B_2-complexity of the same function as well.

But also with respect to the base Ω_0, only linear lower bounds with small constant factor are proved for explicitely defined Boolean functions. The largest lower bound with complete proof is $4n$ by Uri Zwick. m

Kazuo Iwama, Oded Lachish, Hiroki Morizumi, Ran Raz, An explicit lower bound of $5n - o(n)$ for Boolean circuits, preprint 2005. (can be obtained via the homepage of Ran Raz).

a $5n$ lower bound is claimed. The proof ideas look fine. But too much is left for the reader such that the proof cannot be considered to be complete.

The inability to prove lower bounds for the Ω_0-complexity of explicit Boolean functions has

led to the consideration of restricted models of Boolean networks like monotone or bounded-depth Boolean networks. For both restricted models, exponential lower bounds for the complexity of explicit Boolean functions are known. The hope is that methods, developed for a restricted model, could be extended to get a proof of a super-linear lower bound for the Ω_0-complexity of an explicit Boolean function.

It is my opinion that methods developed with respect to bounded-depth networks do not help to get a solution of the general problem. In contrast to this, I believe that methods developed with respect to monotone networks could help. Hence, we shall consider monotone Boolean networks extensively.