# A boolean function requiring 3n network size

by

Norbert Blum

A 82/13
June 82

Fachbereich 1o
Universität des Saarlandes
D - 66oo Saarbrücken

Abstract: Paul [P] first proved a 2.5n-lower bound for the network complexity of an explicit boolean function. We modify the definition of Paul's function a little and prove a 3n-lower bound for the network complexity of that function.

## 1. Introduction

One of the most difficult problems in complexity theory is proving a nonlinear lower bound for the network complexity of an explicit boolean function. Although it is well known by a counting argument, that relative to the full basis most boolean functions need exponentially many operations, only linear lower bounds with small constant factor are known, for explicit boolean functions. Schnorr [S1] first proved a 2n-lower bound for a n-ary boolean function. Next Paul [P] proved a 2.5n-lower bound for another n-ary boolean function. Stockmeyer [St] proved, that the lower bound of Paul holds for a larger class of functions. In [S2] Schnorr gives a proof for a 3n-lower bound for the function defined by Paul. But Wegener [W] pointed out a gap in the proof of a lemma in Schnorr's proof. In [B] we use a weaker version of that lemma and prove a 2.75n-lower bound. Now we modify the definition of Paul's function a little and prove a 3n-lower bound.

## 2. Preliminaries

Let $K = \{0,1\}$ and $F_n = \{f \mid f : K^n \to K\}$. $F_2$ is the set of basic operations. $x_i : K^n \to K$ denotes the i-th variable. Let $V_n = \{x_i \mid 1 \le i \le n\}$.

A network $\beta$ is a directed, acyclic graph with:

(1) Each node has indegree 0 or 2.
(2) The nodes v with indegree 0 are the input nodes of $\beta$ and are labelled with a variable $op(v) \in V_n$.
(3) Each node u with indegree 2 is called a "gate" and is labelled with an $op(u) \in F_2$. The edges entering u are associated in a fixed ordered way with the arguments of $op(u) \in F_2$.

With each node v we associate a function $res_\beta(v) : K^n \to K$ with:

$$res_\beta(v) = \begin{cases} op(v) & \text{if } v \text{ is an input node} \\ res_\beta(u)\ op(v)\ res_\beta(w) & \text{otherwise} \\ \text{where } u,w \text{ are the predecessors of } v \text{ in that order.} & \end{cases}$$

The network $\beta$ computes all functions $f \in F_n$ such that there exists a node $v \in \beta$ with $res_\beta(v) = f$. $Res_\beta(v)$ depends on input variable $x_i$ if and only if there exists $(a_1,\dots,a_i,\dots,a_n)$ such that

$$res_\beta(v)(a_1,\dots a_i,\dots,a_n) \neq res_\beta(v)(a_1,\dots,\neg a_i,\dots,a_n)$$

$C(f)$ denotes the network complexity of the function f, i.e. $C(f)$ is the minimal number of gates, which are necessary for computing f.

For $f \in F_n$ and $a \in K$ let

$$f^a = \begin{cases} f & \text{if } a = 1 \\ \neg f & \text{if } a = 0 \end{cases}$$

We say: $f \in F_2$ is $\wedge$-type, if:

$$\exists\ a,b,c \in K : f(x,y) = (x^a \wedge y^b)^c$$

$f \in f_2$ is $\oplus$-type, if:

$$\exists\ a \in K : f(x,y) = (x \oplus y)^a$$

No $\wedge$-type function is $\oplus$-type and vice versa. A node $v \in \beta$ such that $res_\beta(v)$ is $\wedge$-type ($\oplus$-type) is called $\wedge$-type gate ($\oplus$-type gate).

The functions $f \in F_2$ can be classified in the following way: There exist:

(i) 2 constant functions
(ii) 4 functions depending on one variable
(iii) 1o functions depending on two variables. 8 of this functions are $\wedge$-type and 2 are $\oplus$ -type.

For a node v in $\beta$ let $suc(v) = \{u \mid v \rightarrow u \text{ is edge in } \beta\}$ and $pred(v) = \{u \mid u \rightarrow v \text{ is edge in } \beta\}$ be the set of direct successors and direct predecessors of v.

The functions, associated with the nodes in pred(v) are called input functions of v.

Throughout this paper, we use the following fact:

<u>Fact:</u> Let $\beta$ be a network computing $f \in F_n$. 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.

Let $U \subset V_n$ and $\alpha : U \rightarrow K$ be a mapping. Frequently we consider the restriction $f_\alpha$ of $f \in F_n$ under the assignment $\alpha$. More precisely, $f_\alpha$ is defined by:

$$f_\alpha(x_1,\ldots,x_n) = f(y_1,\ldots,y_n)$$

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

In a natural way an assignment $\alpha$ associates with a network $\beta$ a subnetwork $\beta_\alpha$, which is got by fixing input variables according to $\alpha$ and eliminating the unnecessary gates.

In the following, we write res(v) for $res_\beta(v)$ if $\beta$ is kept fixed. $\# S$ denotes the cardinality of the set S.

For proving the lower bound, we consider paths in a network. $(v \Rightarrow u)$ denotes a path from the node v to the node u.

## 3. The lower bound

For $\alpha = a_1 \ldots a_r \in K^*$ let $(\alpha)$ denote the binary number represented by $\alpha + 1$.

Let $\alpha_1 = a_1 \ldots a_{\log(n)}, \alpha_2 = a_{\log(n)+1} \ldots a_{2\log(n)}$ and $\alpha_3 = a_{2\log(n)+1} \ldots a_{3\log(n)}$ $(a_i \in K)$. Then we define:

$$f : K^{n+3\log(n)+3} \to K$$

$$f(a_1, \ldots, a_{3\log(n)}, p, q, r, x_1, \ldots x_n) :=$$

$$q((x_{(\alpha_1)} \wedge x_{(\alpha_2)}) \vee p(x_{(\alpha_2)} \wedge x^r_{(\alpha_3)})) \vee \neg q(x_{(\alpha_1)} \oplus x_{(\alpha_2)})$$

Remark:

If we set p := 0, then f is the function defined by Paul.

For $h := f_{p := 0}$, Paul has proved a 2.5n-lower bound. First he makes h independent of some inputs $x_i$, which allows to eliminate 3 gates each. After this, he knows quite exactly, how the "top" of the network looks. For the remaining s inputs, he proves without an inductive argument, the existence of 5/2s-2 gates.

Theorem:

For f defined above holds:

$$C(f) \geq 3n - 3$$

First we make f independent of some inputs $x_i$ which allows to eliminate 3 gates each. We use for this the entire proof of Paul and sketch this part only. For a more detailed analysis, see [P].

Define for $1 \le s \le n$ the statement $E_s$.
$E_s$ : For any function f: $K^{n+3\log(n)+3} \to K$ with the property:

$[\exists\, S \subseteq \{1,\ldots,n\}\ \#S = s$ such that for $a_1,a_2,a_3$ with

$(a_1),\ (a_2),\ (a_3) \in S :\ f(a_1,a_2,a_3,p,q,r,x_1,\ldots x_n) =$

$q((x_{(a_1)} \wedge x_{(a_2)}) \vee p(x_{(a_2)} \wedge x^r_{(a_3)})) \vee \neg q(x_{(a_1)} \oplus x_{(a_2)})]$

holds, $3s - 3 \le C(f)$.

$E_1$ is trivially true. Let $E_{s-1}$ be true. Now we prove that $E_{s-1}$ implies $E_s$. Let $\beta$ be any minimal size network for f. W.l.o.g. we assume that for each $i \in S$ there is a unique node $v \in \beta$ with $op(v) = x_i$.

<u>Case 1:</u> $\exists\, i \in S : \#suc(x_i) \ge 3$.

By fixing $x_i$ at 0 we can eliminate at least 3 gates of $\beta$. The reduced network computes the restriction $f_{x_i:=0}$ of f. From the induction hypothesis follows $C(f) \ge 3s-3$.

<u>Case 2:</u> $\exists\, i \in S : \#suc(x_i) = 2$ and $\exists\, v \in suc(x_i)$ such that v is an $\wedge$-type gate.

Choose $c \in K$ such that $res(v)_{x_i:=c}$ is constant. Then, by fixing $x_i$ at c, we can eliminate all nodes in $suc(x_i)$ and all nodes in $suc(v)$. Since $\beta$ is of minimal size, there are at least three different such nodes. The reduced network computes the restriction $f_{x_i:=c}$ of f. From the induction hypothesis follows $C(f) \geq 3s-3$.

Case 3: $\exists\ i \in S : \forall\ v \in suc(x_i) : v$ is a $\oplus$ -type gate. Then there exist nodes $u_1,\dots,u_r$ in $\beta$ with

(1) $u_1 \in suc(x_i)$

(2) $u_j$ is a $\oplus$ -type gate $\forall\ j \in \{1,\dots,r\}$

(3) $u_{j+1} \in suc(u_j)$ and $\#suc(u_j) = 1$ for $1 \leq j \leq r-1$.

(4) $\#suc(u_r) > 1$ or for $w \in suc(u_r)$ holds: w is an $\wedge$-type gate.

Let $x_i$, $g_1$ be the input functions of $u_1$ and $res_\beta(u_j)$, $g_{j+1}$ be the input functions of $u_{j+1}$, $1 \leq j < r$. Paul (case III in [P]) proves, that $u_1,\dots,u_r$ can be chosen such that $g_1,\dots,g_r$ do not depend on $x_i$. Then $res(u_r) = x_i \oplus g$ for some function g which does not depend on $x_i$.

Hence $res(u_r)_{x_i:=g}$ and $res(u_r)_{x_i:=\neg g}$ are constant. Therefore for each of the substitutions $x_i := g$ and $x_i := \neg g$, we can eliminate $u_r$ and all nodes in $suc(u_r)$. We distinguish two cases.

(i) $\#suc(u_r) \geq 2$. Then we eliminate at least 3 gates by fixing $x_i$ at g or at $\neg g$.

(ii) $\#suc(u_r) = 1$. Then $w \in suc(u_r)$ is an $\wedge$-type gate. Choose $\tilde{g} \in \{g,\neg g\}$ such that $res(w)_{x_i:=\tilde{g}}$ is constant. Then by fixing $x_i$ at $\tilde{g}$, we can eliminate at least 3 gates, namely $u_r$, w and all nodes in $suc(w)$.

The induction hypothesis implies now $C(f) \geq 3s-3$.

If none of the cases 1-3 apply, then $\forall\ i \in S$ holds:

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

(ii) $G_i$ is an $\wedge$-type gate.

Let be $G = \{G_i | i \in S\}$.

Paul proves the following lemma.

Lemma 1:

$\forall\ i, j \in S$ with $i \neq j : G_i \neq G_j$

Proof: Suppose $\exists\ i,j \in S$, $i \neq j$ with $G_i = G_j$. Then there exists $c \in K$ such that $res(G_j)_{x_i:=c}$ is constant. Hence $f_{x_i:=c}$ does not depend on $x_j$. But $f(a_1,a_2,a_3,0,0,r,x_1,\dots,x_n) = c \oplus x_j$ for $(a_1) = i$, $(a_2) = j$ and $x_i = c$ depends on $x_j$, a contradiction.

Case 4: $\exists\ i \in S : \#suc(G_i) \geq 2$.
Consider $c \in K$ with $res(G_i)_{x_i:=c}$ is constant. Then fixing $x_i$ at $c$ eliminates $G_i$ and all nodes in $suc(G_i)$. There are at least three different such gates. Hence from the induction hypothesis follows: $C(f) \geq 3s-3$.

It remains to consider:

Case 5: $\forall\ i \in S : \#suc(G_i) = 1$
We denote the unique direct successor of $G_i$ by $Q_i$.

Before analyzing case 5, we give some definitions:

A path in $\beta$ is called free, if no inner node is in $G$.

A node $w$ in $\beta$ is called a split, if the outdegree of $w$ is $\geq 2$.

A split w in $\beta$ is called <u>free split</u>, if $\exists\ u_1, u_2 \in \beta$, $u_1 \neq u_2$ such that:

a) $u_1$, $u_2 \in$ suc(w).

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

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

The next four lemmas are due to Paul, except the observation C is a $\oplus$-type gate in lemma 3, which is due to Schnorr.

<u>Lemma 2:</u>

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

<u>Proof:</u> Suppose that no free path $(G_i \Rightarrow t)$ exists. Then each path $(G_i \Rightarrow t)$ passes some $G_j$ with $j \neq i$. Construct the assignment $\alpha$ by fixing all variables except $x_i$, such that:

(i) $\mathrm{res}(G_\nu)_\alpha$ is constant $\forall\ \nu \in S \setminus \{i\}$

(ii) $f_\alpha = x_i$.

Since each path $(G_i \Rightarrow t)$ goes through some $G_\nu$ with $\nu \neq i$, $\mathrm{res}(t)_\alpha$ does not depend on $x_i$. But this is a contradiction to $f_\alpha = x_i$ and $\mathrm{res}(t)_\alpha = f_\alpha$. □

<u>Lemma 3:</u>

Let i, $j \in S$, $i \neq j$. Let C be a collector of a free path $(G_i \Rightarrow t)$ and a free path $(G_j \Rightarrow t)$. Then at least one of the following conditions is met:

(1) $\exists$ free split $\neq$ C on path $(G_i \Rightarrow C)$ or
$\exists$ free split $\neq$ C on path $(G_j \Rightarrow C)$.

(2) (i) $C$ is a $\oplus$-type gate.

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

Proof: Suppose, that (1) and (2) are not met. We distinguish two cases.

a) $C$ is an $\oplus$-type gate.
Construct assignment $\alpha$ by fixing all variables except $x_i$, $x_j$ such that

(i) $res(G_\nu)_\alpha$ is constant $\forall\ \nu \in S\setminus\{i,j\}$
(ii) $f_\alpha = x_i \wedge x_j$.

By assumption, all paths $(G_i \Rightarrow t)$ and $(G_j \Rightarrow t)$ goes through $C$ or a $G_\nu$, $\nu \in S\setminus\{i,j\}$. Since $res(C)_\alpha = (x_i \oplus x_j)^a$, $a \in K$ or $res(C)_\alpha$ depends on at most one variable, for $res(t)_\alpha$ the same holds.
Hence $f_\alpha \neq res(t)_\alpha$ a contradiction.

b) $C$ is an $\wedge$-type gate.
Construct assignment $\alpha$ by fixing all variables except $x_i$, $x_j$ such that

(i) $res(G_\nu)_\alpha$ is constant $\forall\ \nu \in S\setminus\{i,j\}$
(ii) $f_\alpha = x_i \oplus x_j$.

If $res(G_j)_\alpha$ depends on $x_i$, choose $c \in K$ 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 $f_{\alpha,x_i:=c}=c \oplus x_j$ depends on $x_j$, a contradiction. The case $res(G_i)_\alpha$ depends on $x_j$ is symmetric.

Now by assumption, all paths $(G_i \Rightarrow t)$ and $(G_j \Rightarrow t)$ goes through $C$ or a $G_\nu$, $\nu \in S\setminus\{i,j\}$. Since $res(C)_\alpha = (x_i^a \wedge x_j^b)^c$, $a,b,c \in K$ or $res(C)_\alpha$ depends on at most one variable , for $res(t)_\alpha$ the same holds.

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

From lemma 3 it follows immediately

Lemma 4:

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

Now we have isolate 2s gates, namely the gates $G_i$, $Q_i$ for $i \in S$. Let $Q = \{Q_i \mid i \in S\}$.

Lemma 5:

There are s-1 mutually distinct splits in $\beta$.

Proof: From lemma 3 follows: There are s-1 input nodes $x_i$, $i \in S$ with: any path $(Q_i \Rightarrow t)$ splits. And again from lemma 3 we derive, that these are mutually distinct. □

The rest of the proof is new.

Since we have at least s-1 splits, we have to connect at least 2(s-1) edges with the output node t. For edges, which correspond to a free path , no node in G can help to connect these with the output node t on the free paths by the definition of a free path .

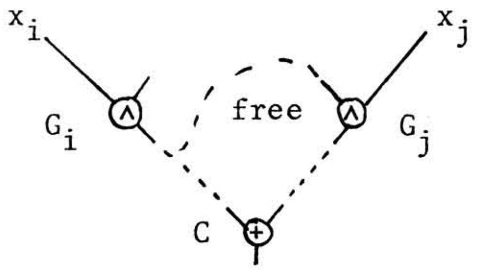
Next we prove, that all but one of the s-1 splits have to be free.

Assume, not all the s-1 splits are free. Then by lemma 3 there exist i,j, $i \neq j$ with the following properties:

(i) $\exists$ collector C of the free paths $(G_i \Rightarrow t)$ and $(G_j \Rightarrow t)$ with: $\nexists$ free split $\neq$ C on the path $(G_i \Rightarrow C)$ and $\nexists$ free split $\neq$ C on the path $(G_j \Rightarrow C)$

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

W.l.o.g. let $\exists$ free path $(G_i \Rightarrow G_j)$. Then we have the following situation:



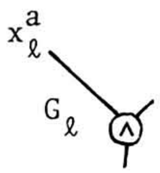

Lemma 6:

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

Proof: For i we know by assumption that $\exists$ free path $(G_i \Rightarrow G_j)$. Assume $\exists\ \ell \in S \setminus \{i,j\}$ with

$\nexists$ free path $(G_\ell \Rightarrow G_i)$ and $\nexists$ free path $(G_\ell \Rightarrow G_j)$.

Consider the node $G_\ell$




This node has input function $x_\ell^a$ with $a \in K$.

Now we construct assignment $\alpha$ by fixing all variables except $x_i, x_j, x_\ell, r$ such that:

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

b) $f_\alpha = (x_i \wedge x_j) \vee (x_j \wedge x_\ell^r)$.

We distinguish two cases:

Case 1: $res(G_j)_\alpha$ does not depend on $x_i$. Now fix $x_\ell$ at $\neg a$. Then $res(G_\ell)_{\alpha, x_\ell := \neg a}$ is constant and $f_{\alpha, x_\ell := \neg a} = x_i \wedge x_j$.

Now, as in the proof of lemma 3, we prove, that case 1 cannot happen.

Case 2: $res(G_j)_\alpha$ depends on $x_i$. Hence there is $b \in K$ such that $res(G_j)_{\alpha, r:=b}$ also depends on $x_i$.

Hence $res(G_j)_{\alpha, r:=b} = (x_i^c \wedge x_j^d)^\ell$, $c,d,\ell \in K$. Fix $x_i$ at $\neg c$. Then $res(G_j)_{\alpha, r:=b, x_i:=\neg c}$ is constant and hence $res(t)_{\alpha, r:=b, x_i:=\neg c}$ does not depend on $x_j$. But

$$f_{\alpha, r:=b, x_i:=\neg c} = (\neg c \wedge x_j) \vee (x_j \wedge x_\ell^b)$$

depends on $x_j$, a contradiction. □

Lemma 7:

$\forall\ \ell, \nu \in S \setminus \{j\}$, $\nu \neq \ell$ holds: If D is a collector of a free path $(G_\ell \Rightarrow t)$ and a free path $(G_\nu \Rightarrow t)$, then:

$\exists$ free split $\neq$ D on path $(G_\ell \Rightarrow D)$ or
$\exists$ free split $\neq$ D on path $(G_\nu \Rightarrow D)$.

Proof: Assume: $\nexists$ free split on path $(G_\ell \Rightarrow D)$ and $\nexists$ free split on path $(G_\nu \Rightarrow D)$. Then, by lemma 6, there exists a path $(G_j \Rightarrow G_\ell)$ or there exists a path $(G_j \Rightarrow G_\nu)$. But by construction, there exists paths $(G_\ell \Rightarrow G_j)$ and $(G_\nu \Rightarrow G_j)$ and hence, we have a cycle in the network. But this cannot happen by the definition of a network. □

From lemma 7, we can derive directly:

Lemma 8:

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

By lemma 8 and lemma 2 we have to connect at least $2(s-2)+2$ edges on free paths to the output node t. Since the nodes in G cannot help and for the nodes in Q only one input wire is free for connecting these edges, we need at least $2(s-2)+2-1-s$ nodes not in $G \cup Q$ on this paths.

Hence

$$\begin{aligned} C(f) &\geq \#G + \#Q + s-3 \\ &= 3s-3 \end{aligned}$$

This finishes the proof of the theorem.

Acknowledgement:

I thank Kurt Mehlhorn for valuable comments.

References:

[B] Blum, N.: A 2.75n-lower bound on the network complexity of boolean functions, Tech. Report Universität des Saarlandes, A 81/o5 (1981)

[P] Paul, W.J.: A 2.5n-lower bound on the combinational complexity of boolean functions, SIAM J. Comput. 6, 427-443 (1977)

[S1] Schnorr, C.P.: Zwei lineare untere Schranken für die Komplexität Boolescher Funktionen, Computing 13, 155-171 (1974)

[S2] Schnorr, C.P.: A 3n-lower bound on the network complexity of boolean functions, TCS 1o, 83-92 (198o)

[St] Stockmeyer, L.J.: On the combinational complexity of certain symmetric Boolean functions, Math. Systems Theory 1o, 323-336 (1977)

[W] Wegener, I.: Private communication (1981)