BRICS LS-96-4 S. Skyum: A Non-Linear Lower Bound for Monotone Circuit Size

BRICS

Basic Research in Computer Science

A Non-Linear Lower Bound for Monotone Circuit Size

Sven Skyum

BRICS Lecture Series LS-96-4

ISSN 1395-2048 December 1996

See back inner page for a list of recent publications in the BRICS Lecture Series. Copies may be obtained by contacting:

BRICS
Department of Computer Science
University of Aarhus
Ny Munkegade, building 540
DK - 8000 Aarhus C
Denmark

Telephone: +45 8942 3360
Telefax: +45 8942 3255
Internet: BRICS@brics.dk

BRICS publications are in general accessible through World Wide Web and anonymous FTP:

`http://www.brics.dk/`
`ftp://ftp.brics.dk/pub/BRICS`

A Non-Linear Lower Bound for Monotone Circuit Size

Sven Skyum

Sven Skyum
BRICS[1]
Department of Computer Science
University of Aarhus
Ny Munkegade
DK-8000 Aarhus C, Denmark

[1]**B**asic **R**esearch **I**n **C**omputer **S**cience,
Centre of the Danish National Research Foundation.

Preface

In complexity theory we are faced with the frustrating situation that we are able to prove very few non trivial lower bounds. In the area of combinatorial complexity theory which this note is about, the situation can be described quite precisely in the sense that although almost all functions are very complex (have exponential circuit size), no one has been able to prove any non-linear lower bounds for explicitly given functions.

One alley which is often taken is to restrict the models to make larger lower bounds more likely. Until 1985 no non-linear lower bounds were known for monotone circuit size either (the best lower bound was $4n$). In 1985, Razborov [1] proved a *super polynomial* lower bound for a specific family of Boolean functions.

Razborov proved the following:

> Any family of monotone Boolean circuits accepting graphs containing cliques of size $\lfloor n/2 \rfloor$ has super polynomial size when the graphs are represented by their incidence matrices.

The main purpose of this note is to give a "simple" version of Razborov's proof. This leads to a weaker result but the proof contains all the ingredients of Razborov's proof.

We will prove:

> Any family of monotone Boolean circuits accepting graphs containing cliques of size 3 (*triangles*) has size $\Omega(n^3/\log^4 n)$.

In Section 1 we introduce Boolean functions and the complexity model we are going to use, namely *Boolean circuits*. In Section 2 the appropriate definitions of graphs are given and some combinatorial properties about them are proven. Section 3 contains the proof of the main result.

References

[1] A.A. Razborov: Lower bounds on the monotone complexity of some Boolean functions, *Dokl. Akad. Nauk SSSR 281(4) (1985) 798 - 801 (In Russian); English translation in: Soviet Math. Dokl. 31 (1985) 354-357.*

[2] J.van Leeuwen (Ed.): *Handbook of Theoretical Computer Science, Volume A, Elsevier Science Publishers B.V. (1990) pp. 781-785.*

Contents

1 Boolean functions and circuits **1**

2 Graphs and Combinatorial lemmas **5**

3 Putting things together **8**

4 Problems **14**

1 Boolean functions and circuits

In this note 1 stands for **true** and 0 stands for **false**. All functions are Boolean functions so we will generally omit *Boolean*.

With B_n we denote the set of functions $f : \{0,1\}^n \to \{0,1\}$.

We say that f *accepts* $\vec{x}$ if $f(\vec{x}) = 1$ and *rejects* $\vec{x}$ if $f(\vec{x}) = 0$.

A simple way to represent a function is to give a table for it. That is, for each of the 2^n different inputs to list the value 0 or 1 of the function on the input. Note that this immediately implies the following fact.

Fact 1.1 $|\mathrm{B}_n| = 2^{2^n}$.

x	y	0	$\wedge$		x		y	$\oplus$	$\vee$		$\equiv$	$\overline{y}$	$\Leftarrow$	$\overline{x}$	$\Rightarrow$		1
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Figure 1: The $2^{2^2} = 16$ functions in B_2.

We will deal mostly with *monotone* functions, to be defined next.

Definition 1.1

- *For* $\vec{a} = (a_1, a_2, \cdots, a_n)$ *and* $\vec{b} = (b_1, b_2, \cdots, b_n)$, $\vec{b}$ dominates $\vec{a}$, *written as* $\vec{a} \leq \vec{b}$ *if* $a_i \leq b_i$ *for* $i = 1, 2, \ldots, n$.
- $f \in \mathrm{B}_n$ *is* monotone *if* $\vec{a} \leq \vec{b} \Rightarrow f(\vec{a}) \leq f(\vec{b})$.
- *For* $f, g \in \mathrm{B}_n$, g dominates f *written as* $f \leq g$ *if* $f(\vec{x}) \leq g(\vec{x})$ *for all* $\vec{x} \in \{0,1\}^n$.

Exercise 1.1 *List the six monotone functions in* B_2.

Exercise 1.2 *List the functions in* B_2 *which are dominated by* $\oplus$.

Exercise 1.3 *Let* $f \in \mathrm{B}_n$ *be an arbitrary monotone function and* $f_b(\vec{x}) = f(b, \vec{x})$ *for* $b = 0, 1$. *Prove that* f_1 *dominates* f_0.

The size of representing a function in B_n by its table is 2^n.

We can also represent a function by an expression for it. In this note we restrict ourselves to expressions over the operations $\neg, \vee$, and $\wedge$. $\neg$ has priority over $\wedge$ which again has priority over $\vee$. We usually write $\neg x$ as $\overline{x}$ and omit $\wedge$ together with redundant parantheses.

Example: $(\neg(x \wedge y)) \vee (y \vee z)$ is written as $\overline{xy} \vee y \vee z$.

Exercise 1.4 *Which of the functions in Figure 1 is given by* $\overline{x} \vee y$*?*

Exercise 1.5 *Give a table for the function* $\overline{xy} \vee y \vee z$ *in* B_3 *and give a simpler/shorter expression for it.*

A third, more compact, way of representing a function is to give the *circuit* or *network* that computes it.

Definition 1.2

- *A* circuit C *with* input $\vec{x} = (x_1, x_2, \cdots, x_n)$ *is a directed acyclic labelled graph. The nodes have in-degree 0,1 or 2.*
  The nodes of in-degree 0 are called inputs *and are labelled with a variable* x_i *or a constant 0 or 1. The remaining nodes* $\{v_1, v_2, \ldots, v_s\}$ *are called* gates *and labelled* $\neg$, $\wedge$ *or* $\vee$. *A gate labelled* τ *is called a* τ*-gate.*
- *A* $\neg$*-gate has one predecessor while* $\wedge$*-gates and* $\vee$*-gates have two.*
  A predecessor for v_i *is either an input or belongs to* $\{v_1, v_2, \ldots, v_{i-1}\}$.
- *We associate a function* $\mathrm{val}(v)$ *with each gate and input.* val *is defined recursively:*

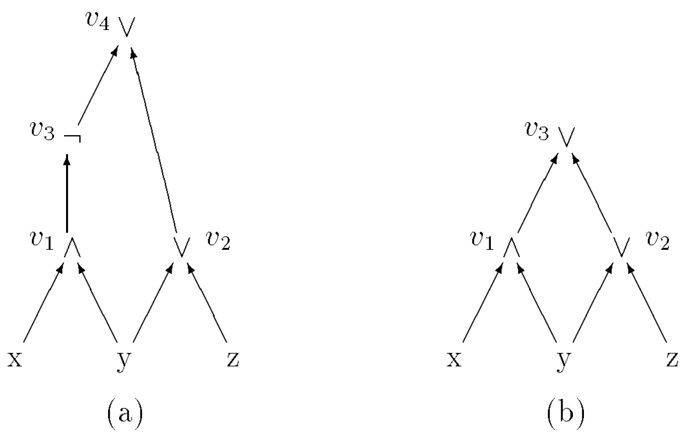

Figure 2: (a) A circuit that computes $\overline{xy} \vee y \vee z$. (b) A (monotone) circuit that computes $xy \vee y \vee z$ $(= y \vee z)$.

- $\mathrm{val}(0) = 0$, $\mathrm{val}(1) = 1$, *and for all* $x \in \{x_1, x_2, \cdots, x_n\}$, $\mathrm{val}(x) = x$.
- *If* v *is a* $\neg$*-gate with predecessor* u *then* $val(v) = \overline{val(u)}$.
- *If* v *is an* $\wedge$*-gate (* $\vee$*-gate) with predecessors* u_1 *and* u_2 *then* $val(v) = val(u_1) \wedge val(u_2)$ $(val(u_1) \vee val(u_2))$.

The function *computed by* C is the function $\mathrm{val}(v_s)$ if $s > 0$. If the circuit contains no gates, then it computes one of the coordinate functions x_i $(i = 1, 2, \ldots, n)$ or one of the constant functions 0 or 1.

A circuit is *monotone* if it contains no $\neg$-gates.

Figure 2 shows examples of circuits.

Circuit *size* (s in the definition) models computation time. It is the number of Boolean operations involved in computing $val(v_s)$. The *depth* of a circuit models parallel time. $val(v_s)$ can be computed (in parallel) by computing in turn all gates of depth $1, 2, \ldots, d$, where d is the depth of v_s. For the circuit in Figure 2(a), the size is 4 and the depth is 3. $val(v_1)$ and $val(v_2)$ can be computed in parallel (they both have depth 1).

Note that an expression can be considered as a special form of circuit.

The following lemma shows that all functions are computed by circuits and more importantly (in this note) that the set of monotone functions coincides with the set of functions computed by monotone circuits.

Lemma 1.1 (a) *All functions in* B_n *can be computed by circuits.*

(b) *All functions computed by monotone circuits are monotone.*

(c) *All monotone functions are computed by monotone circuits.*

Proof (a) follows by observing that for an arbitrary function $f \in \mathrm{B}_n$ with $n \geq 1$,

$$f(x_1, x_2, \cdots, x_n) = \overline{x_1} f(0, x_2, x_3, \ldots, x_n) \vee x_1 f(1, x_2, x_3, \ldots, x_n).$$

(b) is proved by induction in the circuit size and (c) follows from the identity

$$f(x_1, x_2, \cdots, x_n) = f(0, x_2, x_3, \ldots, x_n) \vee x_1 f(1, x_2, x_3, \ldots, x_n)$$

which is valid for monotone functions. □

Exercise 1.6 *Prove the identities in the proof above.*

Definition 1.3 *The (monotone) circuit size of a function is the size of the smallest (monotone) circuit computing it.*

Although a circuit seems to be a far more compact representation than a table, a counting argument shows that for almost all functions this is not so.

We have already seen (Fact 1.1) that $|\mathrm{B}_n| = 2^{2^n}$. The following Lemma shows that the number of different circuits of size s grows much slower.

Lemma 1.2 *There are at most* $3^s(s+n+1)^{2s}$ *different circuits of size* s *over* $\{\vee, \wedge, \neg\}$.

Proof There are 3^s different labellings of s gates. Each of the predecessors for gate v_i can be chosen from $i-1$ gates, n inputs and two constants. Since $i-1+n+2 \leq s+n+1$, the Lemma follows. □

Combining Fact 1.1 and Lemma 1.2 we get for instance that at most half of all functions in B_n can be computed by a circuit of size $2^n/2n$. Despite of this no one has been able to prove a large lower bound (on circuit size) for any specific function. If we restrict ourselves to monotone functions and circuits, the matter changes.

The remaining definitions in this section introduce the notion of *separation* which is the "key" notion in the lower bound proof by Razborov to be presented in Section 3.

Definition 1.4 *Let $A, B \subset \{0,1\}^n$. $f \in \mathrm{B}_n$ is said to* separate *A from B if it satisfies:*

$$A \subseteq f^{-1}(1) \text{ and } B \subseteq f^{-1}(0)$$

Exercise 1.7 *Prove that if $f \in \mathrm{B}_n$ separates A from B, then f separates A' from B' for any $A' \subseteq A$ and $B' \subseteq B$.*

(a,b)-separation introduced next is a generalisation of separation. It captures the notion of "almost" separation.

Definition 1.5 *Let $A, B \subset \{0,1\}^n$. $f \in \mathrm{B}_n$ is said to (a,b)-separate A from B if it satisfies: $|A \cap f^{-1}(0)| \leq a$ and $|B \cap f^{-1}(1)| \leq b$.*

Thus $(0,0)$-separation and separation is the same.

2 Graphs and Combinatorial lemmas

All graphs $G = (V, E)$ in this note are undirected graphs over the node set $V = \{1, 2, \ldots, n\}$. We allow no self loops (edges $\{i, i\}$). Thus $|E| \leq \frac{n(n-1)}{2}$.

We define various classes of graphs to be used later and prove some combinatorial properties about them which we are also going to use in the next section.

Definition 2.1

- *A* triangle *is a graph where* $E = \{\{i,j\},\{i,k\},\{j,k\}\}$ *for some distinct nodes* i*,* j*, and* k*. It is denoted* $\triangle_{ijk}$*. The set of all triangles over* n *nodes is denoted* Tr_n*.*
- *A* complete bipartite graph *is a graph where* $E = A \times (V \setminus A)$ *for some* $A \subseteq V$*. It is denoted* Bip_A*. The set of all complete bipartite graphs over* n *nodes is denoted* CB_n*.*
- *A* star *is a graph where* $E \subset \{\{i,j\} \mid j \neq i\}$ *for some* i*. All edges in a star share a common node as end-point. This node (*i*) is called the* centre *of the star.*
- *A* matching *is a graph where no two edges in* E *share an end-point.*

We say that a graph $G = (V,E)$ *contains* a graph $G' = (V,E')$ if $E' \subseteq E$. We say that a graph $G = (V,E)$ *covers* the nodes $A \subseteq V$ if A is the set of end-points for edges in E. For example, a triangle covers 3 nodes while a matching covers $2|E|$ nodes.

Lemma 2.1 (a) $|\mathrm{Tr}_n| = \binom{n}{3}$.

(b) $|\mathrm{CB}_n| = 2^{n-1}$.

(c) *No complete bipartite graph contains a triangle.*

(d) *Given a star* S *of size* k *(*k *edges) over* n *nodes. The number of complete bipartite graphs which are disjoint (the two sets of edges are disjoint) from* S *is* 2^{n-k-1}*.*

(e) *Given a matching* M *of size* k *(*k *edges) over* n *nodes. The number of complete bipartite graphs which are disjoint (the two sets of edges are disjoint) from* M *is* 2^{n-k-1}*.*

(f) *Any graph with m edges contains either a star of size at least $\sqrt{m/2}$ or a matching of size at least $\sqrt{m/2}$.*

(g) *Any graph with n nodes and m edges is disjoint from at most $2^{n-\sqrt{m/2}-1}$ complete bipartite graphs.*

Proof

(a) There are $\binom{n}{3}$ ways to select 3 elements out of n.

(b) The number of subsets of V is 2^n. Since $Bip_A = Bip_{V \setminus A}$ the number of different complete bipartite graphs is 2^{n-1}.

(c) Obvious.

(d) Let S be a star of size k (over n nodes). If Bip_A is disjoint from S then the $k+1$ covered nodes (the centre and its neighbours) of S must all be in either A or $V \setminus A$. The remaining $n-k-1$ nodes can be distributed arbitrarily between A and $V \setminus A$ which implies the statement.

(e) Let M be a matching of size k (over n nodes). If Bip_A is disjoint from M then for each of the k edges both end-points must be in the same set (A or $V \setminus A$). k pairs of nodes together with $n-2k$ single nodes can be distributed between two sets in $2^{k+n-2k-1} = 2^{n-k-1}$ different ways as claimed.

(f) Let $G = (V, E)$ be an arbitrary graph with m edges. Assume that the largest star contained in G is of size s. We can construct a matching M as follows:

$M := \emptyset$
$R := E$;
while $R \neq \emptyset$ **do**
  pick an edge $\{i, j\}$ in R;
  add $\{i, j\}$ to M
  remove from R all edges with i or j as end-point
od;

Since the largest star in G had size s, the number of edges removed from R in each iteration is at most $2s-1$. Therefore the size of the constructed matching must be at least $\frac{m}{2s-1}$. Since $\max\{s, \frac{m}{2s-1}\} \geq \sqrt{s\,\frac{m}{2s-1}} > \sqrt{\frac{m}{2}}$ the statement follows.

(g) A consequence of d, e, and f. $\square$

Exercise 2.1 *Prove statements (c) and (g) above.*

3 Putting things together

We represent graphs by their incidence matrices. Since the graphs are undirected and have no self loops the incidence matrices are symmetric with zeros in the diagonal.

$$\left\{\begin{array}{llllll} 0 & x_{12} & x_{13} & \ldots & x_{1,n-1} & x_{1n} \\ x_{12} & 0 & x_{23} & \ldots & x_{2,n-1} & x_{2n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \\ x_{1,n-1} & x_{2,n-1} & x_{3,n-1} & \ldots & 0 & x_{n-1,n} \\ x_{1,n} & x_{2,n} & x_{3,n} & \ldots & x_{n-1,n} & 0 \end{array}\right\}$$

Figure 3: Symmetric incidence matrix for a graph.

Thus there is a one-to-one correspondence between graphs with n nodes and $\frac{n(n-1)}{2}$-bit vectors. Therefore functions on graphs with n nodes can be identified with $\mathrm{B}_{\frac{n(n-1)}{2}}$. We are going to use this correspondence heavily, so we define the following notions.

Definition 3.1

- *The undirected graph corresponding to $\vec{x} \in \{0,1\}^{\frac{n(n-1)}{2}}$ is denoted $G_{\vec{x}}$. This is the graph whose incidence matrix is depicted in Figure 3.*

- *If f is a function from graphs with n nodes to $\{0,1\}$ then we will also perceive f as a function in* $\mathrm{B}_{n(n-1)/2}$ *in the obvious way* $f(\vec{x}) = f(G_{\vec{x}})$ *and vice versa.*

Definition 3.2 *A k-circuit is a monotone circuit where we allow $\vee$-gates with up to k inputs as predecessors. Such gates are called* input*-gates. The size of a k-circuit is the number of ordinary gates.*

Let *triangle* $\in \mathrm{B}_{\frac{n(n-1)}{2}}$ be the function defined by:

$$triangle(n) = \begin{cases} 1 & \text{if } G_{\vec{x}} \text{ contains a triangle} \\ 0 & \text{otherwise} \end{cases}$$

Exercise 3.1 *Show that triangle separates triangles from complete bipartite graphs.*

Note that *triangle* can be computed by a monotone circuit of size $\mathrm{O}(\frac{n^3}{\log n})$. (See Problem 4.3.)

We are now ready to state and prove the main result of this note.

Theorem 3.1 *Any monotone circuit computing triangle must have size* $\Omega(\frac{n^3}{\log^4 n})$.

Proof The theorem is a consequence of two lemmas. The first lemma states that a k-circuit of size 0 necessarily separates Tr_n and CB_n very badly. The second lemma states that separation of Tr_n and CB_n can only be done a little better by a k-circuit of size s compared to circuits of size $s-1$. As a result a complete separation must require a "large" k-circuit.

Lemma 3.1 *If f is computed by a k-circuit with no ordinary gates then*

$$|\mathrm{Tr} \cap f^{-1}(1)| > 0 \Rightarrow |CB \cap f^{-1}(1)| > 2^{n-2}$$

.

Proof f is either constant or computed by one input gate. If f is constant, the statement is clearly true. Otherwise f is of the form $\bigvee_{r=1}^{t} x_{i_r j_r}$ for some $t \leq k$. Now if $f(\triangle_{ijm}) = 1$ then $\{\{i_1, j_1\}, \{i_2, j_2\}, \ldots, \{i_t, j_t\}\} \cap \{\{i, j\}, \{j, m\}, \{i, m\}\} \neq \emptyset$. Wlog assume that $\{i, j\}$ is in the intersection. Then $x_{ij} \leq f$ and consequently $f(Bip_A) = 1$ for all A's where one of i and j is in A and the other not. □

Lemma 3.2 *If f is computed by a k-circuit of size $s > 0$ and f (a, b)-separates* Tr_n *from* CB_n*, then there exists a function f' computed by a k-circuit of size at most $s - 1$ which either $(a, b + 2^{n-\sqrt{k/2}-1})$- or $(a + k^2, b)$-separates* Tr_n *from* CB_n*.*

Proof Let C be a k-circuit of size s computing f. There are two cases to consider.

Case 1 There is an $\vee$-gate v with two input gates as predecessors (Figure 4). Then

$$val(v) = \bigvee_{q=1}^{k_1} x_{i_q j_q} \vee \bigvee_{r=1}^{k_2} x_{i'_r j'_r} \; (= \bigvee_{t=1}^{k_3} x_{i''_t j''_t})$$

for some $k_1, k_2 \leq k$ and $k_3 \leq k_1 + k_2$.
If $k_3 \leq k$ then substitute for v an input gate v' computing $\bigvee_{t=1}^{k_3} x_{i''_t j''_t}$. The new circuit is a k-circuit of size $s - 1$ computing f so we are done.
If $k_3 > k$ then we substitute the constant 1 for v and propagate it towards the root by removing gates until a non constant gate is met. The details are omitted. Figure 4 indicates the transformation.

The resulting circuit C' is a k-circuit of size at most $s - 1$ computing a function f' dominating f ($f \leq f'$).

We will prove that f' accepts at most $2^{n-\sqrt{k/2}-1}$ more complete bipartite graphs than f from which the lemma follows.
Assume that $f'(Bip_A) = 1$ and $f(Bip_A) = 0$. Then $val(v)(Bip_A) = 0$ and hence $(A \times V \setminus A) \cap \{\{i''_t, j''_t\} \mid 1 \leq t \leq k_3\} = \emptyset$. The statement now follows by Lemma 2.1(g).

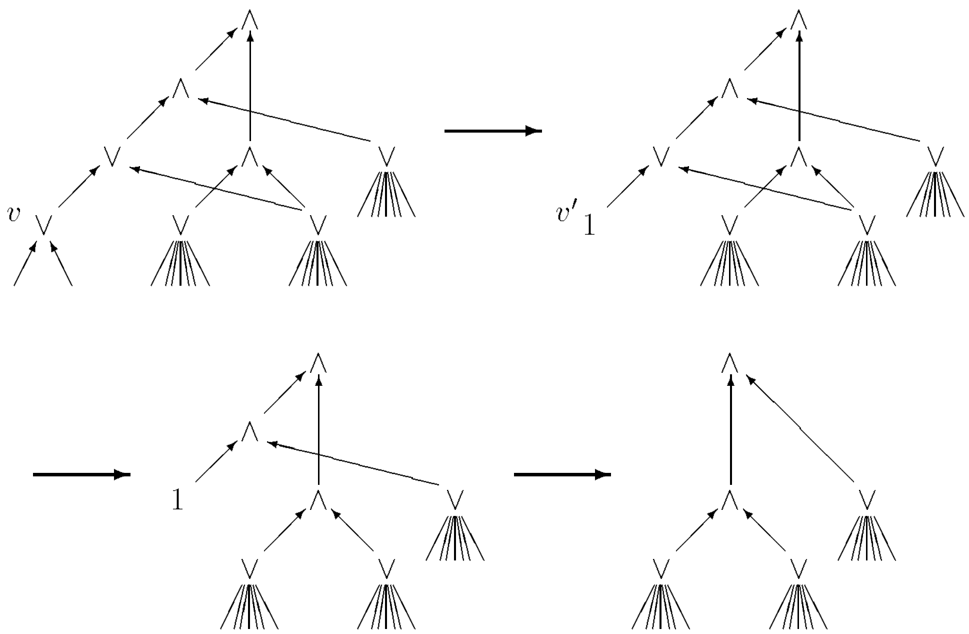

Figure 4: Transforming a k-circuit by removing an $\vee$-gate and propagating a constant.

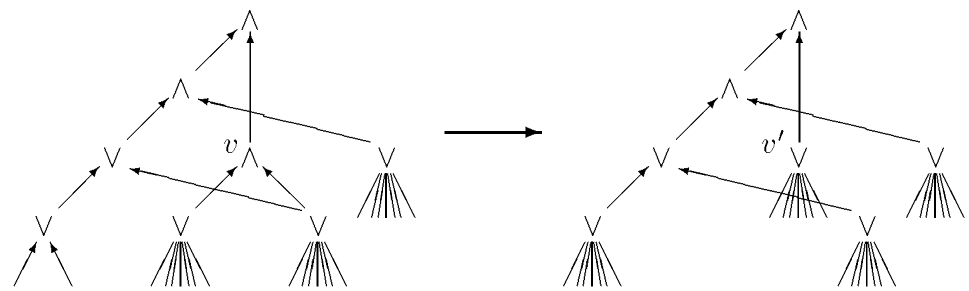

Figure 5: Transforming a k-circuit by removing an $\wedge$-gate.

Case 2 There is an $\wedge$-gate v in C with two input gates as predecessors. Then

$$val(v) = \bigvee_{q=1}^{k_1} x_{i_q j_q} \wedge \bigvee_{r=1}^{k_2} x_{i'_r j'_r} \; (= \bigvee_{q=1, r=1}^{k_1, k_2} x_{i_q j_q} x_{i'_r j'_r})$$

for some $k_1, k_2 \leq k$.

Since $xx = x$ is true for Booleans we may write $val(v)$ on the form

$$val(v) = \bigvee_{q=1}^{k_3} x_{i_q j_q} \vee \bigvee_{r=1}^{k_4} x_{i'_r j'_r} x_{i''_r j''_r}$$

where $k_3 \leq k$ and $k_4 \leq k^2$.
If we substitute gate v with an input gate v' computing $\bigvee_{q=1}^{k_3} x_{i_q j_q}$ we obtain a circuit C' computing a function f' dominated by f ($f' \leq f$). This follows because C and C' are monotone and $y_1 \leq y_1 \vee y_2$. If $k_3 = 0$ then $val(v') = 0$. In any case C' ends up being a k-circuit of size at most $s - 1$. See Figure 5.

The Lemma follows if we can prove that f' rejects at most k^2 more triangles than f. Assume that $f'(\triangle_{ijm}) = 0$ and $f(\triangle_{ijm}) = 1$. This implies that

$$x_{ij} x_{ik} x_{jk} \leq \bigvee_{r=1}^{k_4} x_{i'_r j'_r} x_{i''_r j''_r}$$

and consequently that for some $1 \leq r \leq k_4 \leq k^2$, $\{i'_r, j'_r, i''_r, j''_r\} = \{i, j, m\}$. This is possible for at most k^2 different sets $\{i, j, m\}$ concluding the argument.
$\square$

Proof (of Theorem 3.1) First recall that *triangle* separates Tr_n from CB_n (Lemma 2.1(c)).

Now assume that a k-circuit of size s computes *triangle*. By Lemma 3.2 it follows that there exists a k-circuit of size 0 that $(sk^2, s2^{n-\sqrt{\frac{k}{2}}-1})$-separates Tr_n from CB_n. Lemma 3.1 implies that

$$\binom{n}{3} - sk^2 > 0 \;\Rightarrow\; s2^{n-\sqrt{\frac{k}{2}}-1} \geq 2^{n-2}$$

or

$$s < \frac{\binom{n}{3}}{k^2} \;\Rightarrow\; s \geq 2^{\sqrt{\frac{k}{2}}-1}.$$

For $k = 18\log^2 n$ this amounts to

$$s < \frac{\binom{n}{3}}{18^2 \log^4 n} \Rightarrow s \geq \frac{n^3}{2} \, (= 2^{3\log n - 1}).$$

We conclude that an $18\log^2 n$-circuit computing *triangles* must be of size at least $\binom{n}{3} / \, 18^2 \log^4 n$ or $\Omega(n^3 / \log^4 n)$. $\square$

4 Problems

Problem 4.1 *Prove that there is at least* $2^{2^{n/2}}$ *different monotone functions in* B_n*. Hint: Count the number of* slice functions*.* f *is a slice function if there is a* k *such that*

$$f(\vec{a}) = \begin{cases} 1 & \text{if} \quad |\{i \mid a_i = 1\}| > k \\ * & \text{if} \quad |\{i \mid a_i = 1\}| = k \\ 0 & \text{if} \quad |\{i \mid a_i = 1\}| < k \end{cases}$$

Problem 4.2 *For a vector* $\vec{x} = (x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n)$*, let* $\vec{x}^{(i)}$ *denote the vector* $(x_1, \ldots, x_{i-1}, \overline{x}_i, x_{i+1}, \ldots, x_n)$*.*

f *is* locally monotone *if:*
For all $\vec{x} \in \{0,1\}^n$ *and* $1 \le i \le n$*:* $(x_i = 0) \Rightarrow f(\vec{x}) \le f(\vec{x}^{(i)})$*.*

Prove that f *is monotone if and only if* f *is locally monotone.*

We generalize circuits to compute a function from $\{0,1\}^n \to \{0,1\}^m$ by identifying an m-vector of gates as output gates.

Problem 4.3 The relevant oyster of the week. *Show that if the square of an* $n \times n$ *Boolean matrix can be computed by a monotone circuit of size* $S(n)$*, then triangle can be computed by a circuit of size* $S(n) + \mathrm{O}(n^2)$*. (This implies that triangle can be computed by a circuit of size* $\mathrm{O}(\frac{n^3}{\log n})$*.)*

Problem 4.4 The delicious oyster of the week. *Construct a circuit computing the function* $f : \{0,1\}^3 \to \{0,1\}^3$ *given by* $f(x,y,z) = (\overline{x}, \overline{y}, \overline{z})$ *using only two* $\neg$*-gates. There is no limit to the number of* $\wedge$*- and* $\vee$*-gates that can be used.*

Recent Publications in the BRICS Lecture Series

LS-96-4 **Sven Skyum.** ***A Non-Linear Lower Bound for Monotone Circuit Size*. December 1996. viii+14 pp.**

LS-96-3 **Kristoffer H. Rose.** ***Explicit Substitution – Tutorial & Survey*. September 1996. v+150 pp.**

LS-96-2 **Susanne Albers.** ***Competitive Online Algorithms*. September 1996. iix+57 pp.**

LS-96-1 **Lars Arge.** ***External-Memory Algorithms with Applications in Geographic Information Systems*. September 1996. iix+52 pp.**

LS-95-5 **Devdatt P. Dubhashi.** ***Complexity of Logical Theories*. September 1995. x+46 pp.**

LS-95-4 **Dany Breslauer and Devdatt P. Dubhashi.** ***Combinatorics for Computer Scientists*. August 1995. viii+184 pp.**

LS-95-3 **Michael I. Schwartzbach.** ***Polymorphic Type Inference*. June 1995. viii+24 pp.**

LS-95-2 **Sven Skyum.** ***Introduction to Parallel Algorithms*. June 1995. viii+17 pp. Second Edition.**

LS-95-1 **Jaap van Oosten.** ***Basic Category Theory*. January 1995. vi+75 pp.**