Limiting Negations in Bounded Treewidth and Upward Planar Circuits

Jing He, Hongyu Liang, and Jayalal M.N. Sarma

Institute for Theoretical Computer Science,
Tsinghua University, Beijing, China
{hejing2929,hongyuliang86,jayalal.sarma}@gmail.com

Abstract. The decrease of a Boolean function $f : \{0,1\}^n \to \{0,1\}$, denoted by $d(f)$ is the maximum number of inverse indices in any increasing chain of inputs $x_1, \ldots, x_\ell \in \{0,1\}^n$, where i is an inverse index if $f(x_i) > f(x_{i+1})$. It follows from a theorem of Markov (JACM 1958) that the minimum number of negation gates in a circuit necessary and sufficient to compute any Boolean function f is $\lceil \log(d(f)+1) \rceil$. A recent result due to Morizumi (ICALP 2009) proves that $d(f)$ negations are necessary and sufficient when the computation is done by formulas. We explore the situation in between formulas (directed trees) and general circuits (DAGs), and related models. We obtain the following results:

1. We argue that for any Boolean function f, there is a circuit computing f, that uses $\lceil \log(d(f)+1) \rceil$ negations and has treewidth at most $\lceil \log(d(f)+1) \rceil + 1$. For $1 \leq k \leq \lceil \log(d(f)+1) \rceil$, we prove that $d(f) \cdot 8k/2^k$ negations are sufficient to compute any Boolean function f by circuits of treewidth at most k. Moreover, if there is a circuit family of size $s = s(n)$ and treewidth $k = k(n)$ computing $\{f_n\}$, then there exists a circuit family of size $s \cdot n^{O(1)} \cdot 2^{O(\min\{k, \log n\})}$ and treewidth at most $2k$ which computes $\{f_n\}$ and contains $O(\max\{nk/2^{2k}, \log n\})$ negation gates.
2. We obtain tight bounds on the number of negation gates required to compute specific functions such as $\textsc{Parity}_n$, $\overline{\textsc{Parity}_n}$ and $\textsc{Inverter}_n$ by one-input-face upward planar circuits. We extend these lower bounds to a larger class of functions (which also includes natural functions like Add and Subtract) and we show a direct sum theorem for this class with respect to the number of negations.
3. We demonstrate the limitations of the one-input-face constraint in the upward planar circuits by showing the explicit function which can be computed by a monotone upward planar circuit, but cannot be computed by any montone one-input-face upward planar circuit.
4. We prove that for every Boolean function f, there exists a multilective upward planar circuit which uses at most $\lceil \frac{d(f)+1}{2} \rceil$ negation gates for computing f.

1 Introduction

Proving super-polynomial size lower bounds for circuits computing explicit functions in NP is a central problem in circuit complexity theory. Theory of monotonicity in Boolean circuits became fruitful in this context and it culminated in

P. Hliněný and A. Kučera (Eds.): MFCS 2010, LNCS 6281, pp. 417–428, 2010.

the exponential size lower bound due to Razborov [13] for any monotone circuit computing the clique function. Tardos [17] demonstrated that there are explicit functions in P which requires exponential size for monotone circuits computing them. This area received a lot of attention and several important resource lower bounds were proved against monotone Boolean circuits [6,12]. Relaxing the monotonicity constraint, Amano and Maruoka [1] proved exponential size lower bounds for an explicit function when the circuit is allowed to use only $\frac{1}{6}\log\log n$ negations.

How far can one improve the above lower bound result in terms of the number of negations allowed in the circuit? This highlights the importance of exploring the power of negation gates in Boolean circuits. A very fundamental question in this direction is about the number of negations that is required to compute any Boolean function, called the inversion complexity of the function. Historically much earlier (in 1958), Markov [8] came up with a surprisingly tight bound for the inversion complexity of any Boolean function. Let $f : \{0,1\}^n \to \{0,1\}$, and let $(x_1, x_2, \ldots, x_n)$ and $(y_1, y_2, \ldots, y_n)$ be two Boolean vectors in $\{0,1\}^n$. Define $x \leq y$ if $x_i \leq y_i$ for all i. The decrease of function f with respect to an increasing chain of Boolean vectors $v_1, \ldots, v_m \in \{0,1\}^n$, is defined as the number of i such that $f(v_i) > f(v_{i+1})$. The decrease of the function, denoted by $d(f)$ is the maximum decrease over all increasing chains of Boolean vectors. Thus, the decrease of the function f can at most be n. Markov [8] showed a tight characterization of the inversion complexity of a function f as $\lceil \log(d(f)+1) \rceil$. However, the circuits that Markov constructed are of exponential size although they use only $O(\log n)$ negations. Complementing this, Fischer [4] showed that for every poly-sized circuit, there is an equivalent poly-sized circuit which uses at most $\lceil \log(n+1) \rceil$ negations.

Many years after Markov's and Fischer's results, Santha and Wilson [14] (see also [16]) showed a contrasting picture in the constant depth world: there are functions requiring super-logarithmic number of negation gates in any poly-sized constant-depth circuit computing them. Recently, Morizumi [11] studied the case of formulas and proved tight lower and upper bounds for the inversion complexity of Boolean functions. More precisely, he proved that the inversion complexity in formulas computing the function f is exactly $d(f)$. He also proved an analogue of Fischer's result in this context: if there is a polynomial size formula computing a function f, then there is a polynomial size formula for f which uses at most $\lceil \frac{n}{2} \rceil$ number of negations.

In this paper, we study circuit classes between formulas (directed trees) and general circuits (directed acyclic graphs). Many parameters interpolate between the two. Two important ones we consider here are upward planarity and the treewidth of the underlying undirected graph of the circuit. We defer the formal definition of these parameters to Section 2.

Roughly speaking, treewidth measures how formula-like the circuit is (the treewidth of a formula is 1). We consider circuits of treewidth k, and parameterize the number of negations in the circuit in terms of k. The power of such circuits has been considered earlier in the context of classical [5] and quantum

computation [9]. As noted in [5], leveled circuits (or graphs) of width k has treewidth at most $2k-1$, and they showed that width bounded and treewidth bounded circuit classes roughly interleave in terms of computational power. As noted earlier, the treewidth bounded classes also generalizes formulas. It was shown in [5] that the circuits of treewidth k and size s can also be simulated by formulas of size roughly s^{k^2}. However, the number of negations used in the construction is large even if the original circuit is monotone.

We explore the power of such circuits in the context of inversion complexity. To begin with, we argue that treewidth beyond $O(\log n)$ does not help in general(Theorem 5). We prove the following parameterized upper bound:

Theorem 1. *Let f be a Boolean function, $1 \leq k \leq \lceil \log(d(f)+1) \rceil$. There exists a circuit of treewidth at most k computing f using at most $d(f) \cdot 8k/2^k$ negations.*

However, as in the case of Markov's theorem, the size of the circuit in the above theorem could be exponentially large. We also obtain an analogue of Fisher's result in our context; i.e. the inverse complexity under size constraints.

Theorem 2. *Let $\{f_n\}$ be a family of Boolean functions. If there is a circuit family of size $s = s(n)$ and treewidth $k = k(n)$ computing $\{f_n\}$, then there exists a circuit family of size $s \cdot n^{O(1)} \cdot 2^{O(\min\{k, \log n\})}$ and treewidth at most $2k$ which computes $\{f_n\}$ and contains $O(\max\{nk/2^{2k}, \log n\})$ negation gates.*

Upward planar circuits are circuits whose underlying graph is upward planar (see Section 2). These circuit classes have been considered in the literature in many contexts [10,3,2,7]. In this model, we require that each input label appears at most once in the circuit and that all input vertices are in one face and all edges in the circuit go upwards in the plane. These are better known in the literature as *one-input-face upward planar circuits* (and are also considered by McColl [10] and Beynon and Buckle [3] in the context of monotone circuits.) For this model, we explore the inversion complexity of specific functions and prove tight upper and lower bounds for the PARITY function and the INVERTER function. More specifically we prove the following:

Theorem 3. *Let $n \geq 2$, $f \in \{\text{PARITY}_n, \overline{\text{PARITY}_n}\}$. The inversion complexity of the function f with respect to one-input-face upward planar circuits is $n-1$. The inversion complexity of INVERTER_n with respect to such circuits is n.*

We generalize this argument further to a non-trivial collection of Boolean functions (Theorem 7). For functions in this class, we show a direct sum theorem (Theorem 8) by proving a tight lower bound on the number of negations required to compute t functions simultaneously using a one-input-face upward planar circuit. We also exhibit the limitation of the one-input-face constraint by showing an explicit function which can be computed by a monotone upward planar circuit, but cannot be computed by any monotone one-input-face upward planar circuit (Theorem 9).

Although formulas are planar, the above model does not include them since formulas allow input labels to be duplicated. A planar circuit model where the

inputs can be duplicated is called *multilective planar circuit* (which was introduced in [15]). Formulas are clearly multilective planar circuits. For this more powerful class of circuits, we are able to improve the upper bound on the inversion complexity (denoted as $I_{M-UP}(f)$) slightly.

Theorem 4. *For every Boolean function f, $I_{M-UP}(f) \leq \lceil \frac{d(f)+1}{2} \rceil$.*

2 Preliminaries

We introduce some basic definitions in this section. Let $\mathbb{B}_{n,m}$ denote the set of Boolean functions $f : \{0,1\}^n \to \{0,1\}^m$. $\mathbb{B}_n$ stands for $\mathbb{B}_{n,1}$. For a function $f = f(x_1, \ldots, x_n)$, we say f *essentially depends on* x_i if $f|_{x_i=0} \neq f|_{x_i=1}$.

A circuit is an acyclic directed graph, in which all vertices of fan-in 0 (*input gates*) are associated with some variable $x \in \{x_1, \ldots, x_n\}$ or a constant $c \in \{0,1\}$, and all other nodes are either $\wedge, \vee$ or $\neg$. The *size* of a circuit is the number of gates contained in it, and the *depth* of a circuit is the length of the longest directed path from any input vertex to any output vertex. We refer to a standard text book [19] for more definitions.

A circuit is called *semilective* if for all $x \in \{x_1, \ldots, x_n\}$, at most 1 input vertices in the circuit is associated with x. It is called *multilective* otherwise. A *formula* is a multilective circuit all vertices of which have fan-out at most 1.

For a Boolean function f, the *inversion complexity* of f, denoted by $I(f)$, is the minimum number of negation gates contained in any circuit computing f. If restricting the circuits to be formulas (then f should be a single-output function), we get the definition of *inversion complexity of f in formulas*, denoted by $I_F(f)$. The inversion complexity of a family of Boolean functions can be similarly defined, as a function of n. In this notation, Markov [8] proved that $I(f) = \lceil \log(d(f) + 1) \rceil$ for every Boolean function f and Morizumi [11] proved that $I_F(f) = d(f)$.

A *tree decomposition* of a graph $G = (V, E)$ is given by a tuple $(T, (X_d)_{d \in V[T]})$, where T is a tree, each X_d is a subset of V called a *bag*, satisfying 1) $\bigcup_{d \in V[T]} X_d = V$, 2) For each edge $(u, v) \in E$, there exists a tree node d with $\{u, v\} \subseteq X_d$, and 3) For each vertex $u \in V$, the set of tree nodes $\{d : u \in X_d\}$ forms a connected subtree of T. Equivalently, for any three vertices $t_1, t_2, t_3 \in V[T]$ such that t_2 lies in the path from t_1 to t_3, it holds that $X_{t_1} \cap X_{t_3} \subseteq X_{t_2}$. The *width* of the tree decomposition is defined as $\max_d |X_d| - 1$. The *treewidth* $tw(G)$ of a graph G is the minimum width of a tree decomposition of G.

A *planar circuit* is a circuit in which each input label appears exactly once and the underlying undirected graph can be embedded on the plane without edge crossings. It is further called an *upward planar circuit* if it has some planar embedding in which all edges go upwards (monotonically increasing in the vertical direction), and is called *one-input-face upward planar* if besides upward planarity all input nodes are placed at the lowest level.

For every function f, let $I_{OUP}(f)$ denote the minimum number of negation gates required for computing f by any one-input-face upward planar circuit.

3 Bounded Treewidth Circuits

In this section we consider circuits with bounded treewidth, which naturally generalizes formulas. We allow circuits to be multilective, i.e., they may contain duplicated input variables. By a Boolean function we will mean a single-output Boolean function.

3.1 Inversion Complexity in Bounded Treewidth Circuits

Recall that Markov's theorem states that for every Boolean function f, the minimum number of negation gates contained in a circuit computing f is precisely $\lceil \log(d(f)+1) \rceil$; that is, $I(f) = \lceil \log(d(f)+1) \rceil$. We will first show that, if we only care the number of negation gates, then treewidth beyond $I(f)$ is useless.

Theorem 5. *For every Boolean function f, there is a circuit computing f that contains $\lceil \log(d(f)+1) \rceil$ negations and has treewidth at most $\lceil \log(d(f)+1) \rceil + 1$.*

We need the following definition of *connectors*. For any two Boolean functions f_0 and f_1 with the same set of input variables, a *connector of f_0 and f_1* is a function $\mu(y, y', x)$ satisfying that $\mu(i, 1-i, x) = f_i(x)$ for both $i = 0, 1$, where x is the input vector of f_0 and f_1. Markov showed that there always exists a connector of f_0 and f_1 containing at most $\max\{I(f_0), I(f_1)\}$ negation gates, which is then used in the construction of negation-limited circuits. We first demonstrate the existence of a special connector for which we can show a bound on the treewidth.

Lemma 1. *Every pair of Boolean functions $f_0(x)$ and $f_1(x)$ has a connector $\mu(y, y', x)$ which can be computed by a circuit containing $\max\{I(f_0), I(f_1)\}$ negation gates and having treewidth at most $1 + \max\{I(f_0), I(f_1)\}$.*

The outline of the proof of the above lemma is similar to that of Markov's connector, and proceeds by induction on m. The main observation is that at the induction step we can optimize on the treewidth while the negation gates are being combined. The proof of Theorem 5 also uses some observations about the treewidth while implementing Markov's construction. We skip the details.

Let $I_k(f)$ denote the minimum number of negation gates contained in any circuit computing f with treewidth at most k. In this notation, $I_1(f) = I_F(f) = d(f)$ and $I_{\lceil \log(d(f)+1) \rceil + 1}(f) = I(f) = \lceil \log(d(f)+1) \rceil$. Now we prove Theorem 1, which gives an upper bound on $I_k(f)$ for any $1 \le k \le \lceil \log(d(f)+1) \rceil$.

Proof of Theorem 1: We will prove, by induction on $d(f)$, that $I_k(f) \le d(f) \cdot 8k/2^k - 1$ for all $1 \le k \le \lceil \log(d(f)+1) \rceil$. The statement is obvious for $d(f) = 1$. Suppose $d(f) \ge 2$ and the theorem holds for all f' such that $d(f') < d(f)$. Let $S \subseteq \{0,1\}^n$ be the set of all input vectors x such that every chain Y starting with x satisfies that $d_Y(f) \le d(f)/2$. Then, for every chain Y ending at a vector $x \notin S$, $d_Y(f) \le d(f)/2$ (otherwise we can find a chain with decrease $\ge d(f)+1$ by concatenating Y and the chain witnessing $x \notin S$). We also have ($x \in S$ and $x \le y$) $\Rightarrow y \in S$.

Define two functions $f_0(x)$ and $f_1(x)$ as follows:

$$f_0(x) = \begin{cases} 1 & \text{if } x \in S, \\ f(x) & \text{if } x \notin S, \end{cases} \quad \text{and} \quad f_1(x) = \begin{cases} f(x) & \text{if } x \in S, \\ 0 & \text{if } x \notin S. \end{cases}$$

It is easy to see that $\max\{d(f_0), d(f_1)\} \leq d(f)/2$ and $f(x) = (h_S(x) \wedge f_1(x)) \vee (\overline{h_S(x)} \wedge f_0(x))$, where $h_S(x)$ is the characteristic function of the set S, which is monotone and hence can be computed by a monotone formula. If there exists C_i computing f_i for both $i = 0, 1$ such that C_i has treewidth at most k and contains at most t negation gates, we can obtain a circuit with treewidth at most k which computes f and contains at most $2t + 1$ negation gates. Having $t = d(f) \cdot 4k/2^k - 1$ will suffice.

If $k \leq \lceil \log(d(f_i) + 1) \rceil$ for both $i = 0, 1$, we are done by induction hypothesis. If $k > \lceil \log(d(f_i) + 1) \rceil$ for some $i \in \{0, 1\}$, we can get directly from Theorem 5 that there exists a circuit computing f_i with treewidth at most k which contains exactly $I(f_i) = \lceil \log(d(f_i) + 1) \rceil < k$ negation gates. Since $k \leq \lceil \log(d(f) + 1) \rceil$, we have $2^k \leq 2(d(f) + 1) \leq 4d(f)$, from which it follows that $I(f_i) \leq k - 1 \leq d(f) \cdot 4k/2^k - 1$. Therefore, for both $i = 0, 1$, there exists a circuit C_i computing f_i which has treewidth $\leq k$ and contains $\leq d(f) \cdot 4k/2^k - 1$ negation gates. This finishes the induction proof. □

3.2 Inversion Complexity under Polynomial Size Constraints

In this subsection we show Theorem 2 stated in the introduction. Let $f \in \mathbb{B}_n$. Borrowing the notation from [11], we say f' is a *pseudo* i^{th} *slice* of f iff $f'(x) = f(x)$ for all $x = (x_1, \ldots, x_n)$ such that $\sum_{j=1}^{n} x_j = i$. Let C be a circuit computing f of treewidth k and size s. For $i = 0, 1, \ldots, n$, we construct a circuit $C^{(i)}$, which computes a monotone pseudo i^{th} slice of f, by pushing all negations in C down to the input nodes by the De Morgan's law and then replacing $\overline{x_i}$ with $Th_i^{n-1}(x_{-i})$. Here we use x_{-i} to denote $(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$, and Th_m^n is the threshold function which equals 1 iff at least m out of n input variables are 1. It can be seen that pushing the negations down (with possible duplications of gates) can cause at most a doubling of the treewidth of the circuit (we skip the details of this construction). Using $n^{O(1)}$-sized formulas [18] for threshold functions in the above construction gives the following:

Lemma 2. *For a Boolean function $f \in \mathbb{B}_n$ which can be computed by a circuit of size s and treewidth k, and an integer $i \in \{0, 1, \ldots, n\}$, there exists a monotone circuit of size $s \cdot n^{O(1)}$ and treewidth $2k$ which computes a pseudo i^{th} slice of f.*

The basic idea is to make use of these pseudo slices to reconstruct f and hence save on the number of negation gates in the process. For this, we divide them into groups containing some consecutive slices of f. We will use groups of 2^{2k} concecutive slices each (which is a generalization of Morizumi's argument [11] which puts two neighboring slices into one group). Among each group, we find a circuit of limited treewidth and limited number of negation gates which plays the role of a "selector", in the sense that it selects which slice to use according to the

number of 1's in the inputs. Finally, a formula serving as a "universal selector" is applied to choose the correct group, which will not increase the treewidth.

Before presenting the proof, we need to introduce some notations. Let $X = \{x_1, x_2, \ldots, x_n\}$ be a set of variables, and $\mathcal{H} = \{h_1, h_2 \ldots, h_m\}$ be a set of Boolean functions each taking $x_1, \ldots, x_n$ as inputs. A circuit *augmented with* $\mathcal{H}$ is similarly defined as a normal circuit, except that every fan-in 0 vertex is now assigned with some function $h_i \in \mathcal{H}$. Functions in $\mathcal{H}$ are called *help functions*, and circuits augmented with $\mathcal{H}$ are also called *$\mathcal{H}$-circuits*. The normal definition of circuits can be seen as a special case in which $\mathcal{H} = \{x_1, x_2, \ldots, x_n\}$; that is, the help functions are exactly the variables themselves.

In the following we fix $\mathcal{H} = \{Th_i^n : i = 0, 1, \ldots, n\} \cup \{f^{(i)} : i = 0, 1, \ldots, n\}$, where $f^{(i)}$ is a pseudo i^{th} slice function of f. For every pair of integers a, b such that $0 \le a \le b \le n$, a Boolean function g is called a *(a, b)-selector of f* iff g is a pseudo i^{th} slice function of f for every i such that $a \le i \le b$. It follows that $f^{(i)}$ is a (i, i)-selector of f, and f is a $(0, n)$-selector of itself.

For a $\mathcal{H}$-circuit C, we say C is a *(a, b)-selector circuit of f* if it computes a (a, b)-selector of f. We call it a *good (a, b)-selector circuit of f* if in addition it satisfies the following "replacement rule": For every integer r such that $-a \le r \le n - b$, if we replace every input vertex Th_i^n of C by Th_{i+r}^n, and replace every input vertex $f^{(i)}$ of C by $f^{(i+r)}$, for every $0 \le i \le n$, then the resulting circuit is a $(a + r, b + r)$-selector circuit of f. Now we show the following existence lemma.

Lemma 3. *For every $a, b \in \{0, 1, \ldots, n\}$ such that $b - a + 1 = 2^k$ for some integer $k \ge 1$, there is a good (a, b)-selector circuit of f which has size at most 5^k, treewidth at most k and contains at most k negation gates.*

Proof. By induction on k. When $k = 1$, the circuit be $\left(Th_b^n \wedge f^{(b)}\right) \vee \left(\overline{Th_b^n} \wedge f^{(a)}\right)$ is a good (a, b)-selector circuit of f with size 4, treewidth 1 and contains 1 negation gate. Now suppose $k \ge 2$ and the theorem holds for all smaller k. Let C_{k-1} be a good $(a, a + 2^{k-1} - 1)$-selector circuit of f which has size at most 5^{k-1}, treewidth at most $k - 1$ and contains at most $k - 1$ negation gates. Let v be an arbitrary input vertex of C_{k-1}. If v is Th_i^n for some i, we replace it with $\left(Th_{a+2^{k-1}}^n \wedge Th_{i+2^{k-1}}^n\right) \vee \left(\overline{Th_{a+2^{k-1}}^n} \wedge Th_i^n\right)$. If v is $f^{(i)}$ for some i, we replace it with $\left(Th_{a+2^{k-1}}^n \wedge f^{(i+2^{k-1})}\right) \vee \left(\overline{Th_{a+2^{k-1}}^n} \wedge f^{(i)}\right)$. After we replacing every input node v, we combine all the negations connected to nodes assigned with $Th_{a+2^{k-1}}^n$ together to form a new node. Call the new circuit C_k. Since only one copy of $\overline{Th_{a+2^{k-1}}^n}$ is used, C_k contains exactly 1 more negation gates than C_{k-1}. The treewidth of C_k is at most k, as we first replace each input vertex with a tree (which will preserve the treewidth of C_{k-1}) and then combine some vertices together to form a new one (which will increase the treewidth by at most 1). To bound the size of C_k, we note that each input node of C_{k-1} is replaced with a circuit of size 4, and the number of input nodes does not exceed the number of gates in C_{k-1}. Therefore, the size of C_k is at most $5^{k-1} + 4 \cdot 5^{k-1} = 5^k$.

It remains to show that C_k is a good (a, b)-selector circuit of f. Due to our construction, when $\sum_{j=1}^n x_j < a + 2^{k-1}$ (that is, $Th_{a+2^{k-1}}^n(x) = 0$), C_k is

equivalent to C_{k-1}, and hence C_k is a $(a, a+2^{k-1}-1)$-selector circuit of f. When $\sum_{j=1}^{n} x_j \geq a+2^{k-1}$, C_k becomes the circuit obtained from C_{k-1} by replacing Th_i^n with $Th_{i+2^{k-1}}^n$ and replacing $f^{(i)}$ with $f^{(i+2^{k-1})}$, which is of course a $(a+2^{k-1}, a+2^k-1)$-selector circuit of f because C_{k-1} is good by the induction hypothesis. Hence, C_k is a (a,b)-selector circuit of f (remember that $b = a+2^k-1$). To see why C_k is good, we shift all the parameters (except n) by r for every $-a \leq r \leq b$, and can similarly prove that the resulting circuit is a $(a+r, b+r)$-selector circuit of f. This completes the induction step. □

Proof of Theorem 2: Let $f \in \mathbb{B}_n$ and C be a circuit computing it with size s and treewidth k. Let t be the minimum integer such that $n \leq 2^t - 1$, and let $n' = 2^t - 1$. Let $k' = \min\{k, t/2\}$. It is clear that $n' \leq 2n-1$ and $\log n \leq t \leq \lceil \log n \rceil + 1$. We add $n'-n$ dummy input variables to f and regard it as a function in $\mathbb{B}_{n'}$. For $l = 0, 1, \ldots, 2^{t-2k'}-1$, let C'_l be a $(2^{2k'}l, 2^{2k'}(l+1)-1)$-selector circuit of f constructed by Lemma 3, which has size at most $5^{2k'}$, treewidth at most $2k'$ and contains at most $2k'$ negations. Let C' be a formula-like circuit of the form $\bigvee_{l=0}^{2^{t-2k'}} \left(Th_{2^{2k'}l}^{n'} \wedge \overline{Th_{2^{2k'}(l+1)}^{n'}} \wedge C'_l \right)$ (different terms will use different copies of input vertices). It is easy to see that C computes exactly f and contains at most $(2k'+1)2^{t-2k'} = (2k'+1)(n'+1)/2^{2k'} \leq 2n(2k'+1)/2^{2k'}$ negation gates. Since $k' = \min\{k, t/2\}$ and $\log n \leq t \leq \lceil \log n \rceil + 1$, this is at most $\max\{2n(2k'+1)/2^{2k}, 2n(t+1)/2^t\} = \max\{O(nk/2^{2k}), O(\log n)\} = O(\max\{nk/2^{2k}, \log n\})$. Furthermore, C' has size at most $2^{t-2k'}(4+5^{2k'}) = n \cdot 2^{O(k')} = n \cdot 2^{O(\min\{k, t/2\})} \leq n \cdot 2^{O(\min\{k, \log n\})}$, and has treewidth at most $2k'$. Note that C' is not a "true" circuit, but one augmented with help functions $\{Th_i^n : i = 0, 1, \ldots, n\} \cup \{f^{(i)} : i = 0, 1, \ldots, n\}$. We replace every input node of C' with a circuit computing the corresponding help function. Since Th_i^n is computable by a poly-sized formula, and by Lemma 2 $f^{(i)}$ is computable by a monotone circuit of treewidth $2k$ and size $s \cdot n^{O(1)}$, the resulting circuit has size $s \cdot n^{O(1)} \cdot 2^{O(\min\{k, \log n\})}$ and treewidth at most $\max\{2k, 2k'\} = 2k$. This finishes the proof of Theorem 2. □

4 Inversion Complexity in Planar Circuits

4.1 Lower Bounds for One-Input-Face Upward Planar Circuits

In this section we will focus on the inversion complexity in one-input-face upward planar circuits. We will prove $\Omega(n)$ lower bounds of $I_{OUP}(f)$ for many functions f, including some tight results. Since $I(f) = O(\log n)$ for all $f \in \mathbb{B}_{n,m}$, an exponential gap between the number of negation gates used in general circuits and one-input-face upward planar circuits is obtained for a number of natural functions, including Parity, Inverter, Add and Subtract.

Theorem 6. *For* $f \in \{\text{Parity}_n, \overline{\text{Parity}_n}\}$ *where* $n \geq 2$, $I_{OUP}(f) = n-1$.

Proof. To prove that $n-1$ is an upper bound, just notice that $\overline{\text{Parity}_n} = \text{Equiv}(\text{Parity}_{n-1}(x_1, \ldots, x_{n-1}), x_n)$ (Equiv(x,y) computes $x \equiv y$), and that

both the XOR gate and the EQUIV gate can be simulated by planar circuits each containing one negation gate (by standard constructions).

Next we prove that $I_{OUP}(f) \geq n-1$ for $f \in \{\text{PARITY}_n, \overline{\text{PARITY}_n}\}$ where $n \geq 2$. The statement will be proved by induction on n. It is obvious when $n = 2$, since both functions are non-monotone. Now suppose $n \geq 3$ and the statement holds for all $n' < n$. Let $f \in \{\text{PARITY}_n, \overline{\text{PARITY}_n}\}$. For the sake of contradiction, let C be a one-input-face upward planar circuit computing f which contains at most $n-2$ negation gates. Without loss of generality, let the order of the input variables (from left to right in the input face) be $x_1, \ldots, x_n$.

Suppose G is the predecessor of the first negation gate in C (under some topological order of the underlying graph of C). Let g denote the function computed at the gate G. Let $S = \{x_i \mid g \text{ essentially depends on } x_i, 1 \leq i \leq n\}$. Let $l = \min\{i \mid x_i \in S\}$ and $r = \max\{i \mid x_i \in S\}$. Denote by PI the set of all prime implicants of g. Since g is monotone, every prime implicant of it only contains positive literals. First we argue that $S = \{x_i \mid l \leq i \leq r\}$. For this, assume the contrary that $x_i \notin S$ for some $l < i < r$. We set all variables in S to 1. Since g is monotone, this will fix G to be 1. Hence we can find one path from x_l to G on which each gate is fixed to be 1, and another path from x_r to G with the same property. The output gate of C (denoted by O) must lie out of the area defined by these two paths and the input face due to its non-monotonicity. Since C is upward planar, we know that every path connecting x_i and O must intersect the two 1-paths previously found. Thus the variable x_i is "disconnected" from O, and gives a contradiction since the function being computed is PARITY. Similarly we can show that any prime implicant is an "interval".

Lemma 4. *Let $p = x_{i_1} x_{i_2} \ldots x_{i_k} \in PI$ be any prime implicant of g, where $l \leq i_1 < \ldots < i_k \leq r$. Then $i_m = i_{m-1} + 1$ for all $2 \leq m \leq k$.*

Using the induction hypothesis (we skip the details) we can show that each prime implicant of g has size at least 2. In addition, we can prove the following: Let p_l and p_r be the prime implicants of g containing x_l and x_r, respectively. Then p_l and p_r do not intersect with each other; that is, they contain no common variables. Thus, PI contains at least 2 different prime implicants.

Now we are ready to prove Theorem 6. Let C_G be the induced sub-circuit of C with output gate G. More precisely, C_G contains all vertices of C (variables and gates) from which G is reachable, and all edges spanning them. So C_G computes the function g. Let $p_l = x_1 x_2 \ldots x_j$ and $p_r = x_k x_{k+1} \ldots x_r$ be the prime implicants of g containing x_l and x_r, respectively. By the arguments in the previous paragraph, we have $l < j < k < r$. Imagine the scenario where all variables except x_k are set to 0. Since f still essentially depends on x_k after this restriction, there exists a "switching path" $\mathcal{P}$ from x_k to the output gate of C such that flipping the value of x_k will cause all gates on $\mathcal{P}$ to change their values (given that other variables are set to 0). Also note that the output gate is not contained within the area of C_G. Due to the upward planarity of C, $\mathcal{P}$ must intersect the boundary of C_G. Let $\mathcal{P}_{in}$ denote the inside part of $\mathcal{P}$ respect to C_G. As C_G is monotone, any gate on $\mathcal{P}_{in}$ switches from 0 to 1 if x_k switches from 0 to 1, given that other variables are 0. Therefore, setting x_k to 1 will fix

all gates on $\mathcal{P}_{in}$ to evaluate to 1, regardless of the assignment to other variables. $\mathcal{P}_{in}$. The following two cases finishes the induction proof.
Case 1: $\mathcal{P}_{in}$ intersects the right boundary of C_G. We set $x_l = x_{l+1} = \ldots = x_{k-1} = 0$ and $x_k = x_{k+1} = \ldots = x_{r-1} = 1$. Under this restriction G will compute exactly x_r. But x_r cannot affect G now, since any path from x_r to G must intersect $\mathcal{P}_{in}$ This leads to a contradiction.
Case 2: $\mathcal{P}_{in}$ intersects the left boundary of C_G. We set $x_1 = x_2 = \ldots = x_{j-1} = 1$, $x_{j+1} = \ldots = x_{k-1} = 0$, $x_k = 1$ and $x_{k+1} = \ldots = x_r = 0$. Under this restriction G will compute exactly x_j. But any path from x_j to G must intersect $\mathcal{P}_{in}$, which again gives a contradiction. □

We next introduce the classes of functions for which we can apply a similar argument and prove linear lower bounds on $I_{OUP}(f)$. Although the definition seems restrictive, it can be shown that several natural functions fall into these classes. Define the class $\mathbb{W}^n_{k,n_0}$ for $k, n_0, n, m \in \mathbb{N}, k \geq 1$ of functions $f \in \mathbb{B}_{n,m}$ as the ones with the two properties: (1) If $n > n_0$, then for any variable $x \in X$ and any "uniform" restriction σ which maps all variables in $X \setminus \{x\}$ to the same constant $b \in \{0,1\}$, there exists a non-monotone output $y \in Y$ such that $y|_\sigma$ essentially depends on x. (2) If $n > n_0$, then for any variable $x_0 \in X$ and any constant $c_0 \in \{0,1\}$, there exists a set of k' variables $\{x_1, \ldots, x_{k'}\} \subseteq X \setminus \{x_0\}$ and a sequence of Boolean constants $c_1, \ldots, c_{k'}$, where $0 \leq k' \leq \min\{k-1, n-1\}$, such that $f|_\gamma \in \mathbb{W}^{n-k'-1}_{k,n_0}$, where γ denotes the restriction which maps x_i to c_i for all $0 \leq i \leq k'$. Now we state the following theorem about the functions in this class, the proof of which is essentially a generalisation of the proof of Theorem 6. We skip the details due to space constraints.

Theorem 7. $I_{OUP}(f) \geq \lceil \frac{n-n_0}{k} \rceil$ *if* $f \in \mathbb{W}^n_{k,n_0}$.

It can be verified that the class $\mathbb{W}$ contains some natural functions such as {Inverter, Add, Subtract, OrOfParity} (OrOfParity is the OR of t Parity's defined on pairwise-disjoint variables). Combined with the trivial upper-bound for Inverter, this gives:

Corollary 1.
- $I_{OUP}(\text{Inverter}_n) = n$ *for all* $n \geq 1$.
- $I_{OUP}(\text{OrOfParity}_{n_1,\ldots,n_t}) \geq \frac{\sum_{i=1}^t n_i}{2}$ *where all* n_i*'s are even.*
- *For* $f \in \{\text{Add}_{2n}, \text{Subtract}_{2n}\}$ *where* $n \geq 1$, $I_{OUP}(f) \geq n$.

A Direct Sum Theorem: We consider the direct sum of Boolean functions, i.e., a collection of different Boolean functions on pairwise disjoint variable sets. If f is the direct sum of $f_1, \ldots, f_t$, then each of the functions is called an *element* of f. What is the relationship between the inversion complexity of f and that of its elements? Trivially $I(f) \leq \sum_{i=1}^t I(f_i)$, but the result is far from tight; for example, when each f_i is $\text{Parity}_{\sqrt{n}}$ and $t = \sqrt{n}$, the LHS is $O(\log n)$ but the RHS is $\sum_{i=1}^t I(f_i) = n^{\Theta(1)}$. This indicates that computing the direct sum of functions (with large decreases) can benefit from interconnections between its seemingly independent elements. We will show that this is not always the case if we adopt planar circuits as our computation model.

Theorem 8. *Let* $f_i \in \mathbb{W}^{n_i}_{k_i,m_i}, \forall i \in [t]$. *If* f *is the direct sum of* f_i*'s, then* $I_{OUP}(f) \geq \sum_{i=1}^{t} \lceil \frac{n_i - m_i}{k_i} \rceil$.

The proof of this theorem is again based on induction, and Theorem 7 forms the base case of it. The induction step further generalizes the proof of Theorem 6. The only difference is that, when dealing with a special input variable x_i, we need to identify which function it belongs to. We skip the details of the proof in this short version. It is straightforward from Theorems 6,8 and the bounds for the functions that $I(f) = \sum_{i=1}^{t} I(f_i)$ if each f_i is PARITY, $\overline{\text{PARITY}}$ or INVERTER, and f is the direct sum of all f_i's. This differs from general circuits.

Limitation of the One-Input-Face Constraint: We show that restricting all input vertices to be on the same face (or equivalently on the exterior face, or at the lowest level in the plane) may increase the number of negation gates used for computing some functions. We prove a stronger result, by showing a monotone (multi-output) function which has a monotone upward planar circuit computing it, but cannot be computed by any monotone one-input-face cylindrical circuits. Here a circuit is called *cylindrical* if it can be embedded on a cylinder surface without edge crossings and every edge goes upwards. It is easy to see that cylindricality generalizes upward planarity. Let $\text{MINMAX}_n(x_1, x_2, \ldots, x_n) = (\bigwedge_{i=1}^{n} x_i, \bigvee_{i=1}^{n} x_i)$. Thus $\text{MINMAX}_n \in \mathbb{B}_{n,2}$ and computes the minimum and maximum values among all input variables. It is easy to construct a monotone upward planar circuit for it. Using an argument similar to that of the proof of Theorem 6, in the context of cylindricality, we can prove the following theorem:

Theorem 9. *Let* $n \geq 3$. *Then* MINMAX_n *can be computed by a monotone upward planar circuit, but cannot be computed by any monotone one-input-face cylindrical circuit.*

4.2 Multilective Upward Planar Circuits

In this subsection, we will outline the proof of Theorem 4. That is, for every single-output Boolean function f, $I_{M-UP}(f) \leq \lceil \frac{d(f)+1}{2} \rceil$. The proof follows a similar line to that of Markov's. Given a circuit C, we say a vertex (an input variable or a gate) in G is *out* if it lies in the outer-face of C. We say C is *out-negationed* if either C is monotone, or there is a negation gate in C which is out, and the sub-circuit below it is monotone.

The following statement essentially captures the argument. For $i = 0, 1$, let $f_i(x)$ be a Boolean function which can be computed by a out-negationed multilective upward planar circuit containing t_i negation gates. In addition suppose $t_1 = 1$. Then f_0 and f_1 have a connector $\mu(y, y', x)$ which can be computed by a multilective upward planar circuit containing at most $\max\{t_0, t_1\}$ negation gates and exactly one input node assigned with the variable y'. Moreover, this y'-node is out. The proof of Theorem 4 now follows by a careful induction combined with some crucial observations in Markov's original proof.

Acknowledgements. This work was supported in part by the National Natural Science Foundation of China Grant 60553001, and the National Basic Research Program of China Grant 2007CB807900,2007CB807901.

References

1. Amano, K., Maruoka, A.: A superpolynomial lower bound for a circuit computing the clique function with at most (1/6)log log n negation gates. SIAM Journal of Computing 35(1), 201–216 (2005)
2. Barrington, D.A.M., Lu, C.-J., Miltersen, P.B., Skyum, S.: On Monotone Planar Circuits. In: Proceedings of the 14th Annual IEEE Conference on Computational Complexity (CCC), pp. 24–31 (1999)
3. Beynon, M., Buckle, J.: On the planar monotone computation of boolean functions. Theor. Comput. Sci. 53(2-3), 267–279 (1987)
4. Fischer, M.: The complexity of negation-limited networks (a brief survey). LNCS, vol. 33, pp. 71–82. Springer, Heidelberg (1974)
5. Jansen, M., Sarma, M.N., Jayalal: Balancing Bounded Treewidth Circuits. In: Proeedings of CSR 2010 (to appear 2010)
6. Karchmer, M., Wigderson, A.: Monotone circuits for connectivity require superlogarithmic depth. In: Proc. of STOC 1988, pp. 539–550 (1988)
7. Limaye, N., Mahajan, M., Sarma, M.N.J.: Upper bounds for monotone planar circuit value and variants. Computational Complexity 18(3), 377–412 (2009)
8. Markov, A.A.: On the inversion complexity of a system of functions. J. ACM 5(4), 331–334 (1958)
9. Markov, I.L., Shi, Y.: Simulating quantum computation by contracting tensor networks. SIAM J. Comput. 38(3), 963–981 (2008)
10. McColl, W.F.: On the planar monotone computation of threshold functions. In: Mehlhorn, K. (ed.) STACS 1985. LNCS, vol. 182, pp. 219–230. Springer, Heidelberg (1984)
11. Morizumi, H.: Limiting negations in formulas. In: Albers, S., et al. (eds.) ICALP 2009, Part I. LNCS, vol. 5555, pp. 701–712. Springer, Heidelberg (2009)
12. Raz, R., Wigderson, A.: Monotone circuits for matching require linear depth. In: STOC 1990: Proceedings of the twenty-second annual ACM symposium on Theory of computing, pp. 287–292 (1990)
13. Razborov, A.A.: Lower bounds on the monotone complexity of some boolean functions. Soviet Math. Dokl. 281, 798–801 (1985)
14. Santha, M., Wilson, C.: Limiting negations in constant depth circuits. SIAM J. Comput. 22(2), 294–302 (1993)
15. Savage, J.E.: The performance of multilective vlsi algorithms. Journal of Computer and System Science 29(2), 243–273 (1984)
16. Sung, S.C., Tanaka, K.: Limiting negations in bounded-depth circuits: An extension of markovs theorem. In: Ibaraki, T., Katoh, N., Ono, H. (eds.) ISAAC 2003. LNCS, vol. 2906, pp. 108–116. Springer, Heidelberg (2003)
17. Tardos, E.: The Gap Between Monotone and Non-monotone Circuit Complexity is Exponential. Combinatorica 7, 393–394 (1987)
18. Valiant, L.G.: Short monotone formulae for the majority function. J. Algorithms 5(3), 363–366 (1984)
19. Vollmer, H.: Introduction to Circuit Complexity: A Uniform Approach. Springer, New York (1999)