数理解析研究所講究録
第1554巻 2007年 131-138

Negation-Limited Complexity of Parity and Inverters

森住 大樹[1] 垂井 淳[2] 岩間 一雄[1]

[1] 京都大学大学院情報学研究科, {morizumi, iwama}@kuis.kyoto-u.ac.jp

[2] 電気通信大学情報通信工学科, tarui@ice.uec.ac.jp

Abstract

In *negation-limited* complexity, one considers circuits with a limited number of NOT gates, being motivated by the gap in our understanding of monotone versus general circuit complexity, and hoping to better understand the power of NOT gates. We give improved lower bounds for the size (the number of AND/OR/NOT) of negation-limited circuits computing Parity and for the size of negation-limited inverters. An *inverter* is a circuit with inputs $x_1, \ldots, x_n$ and outputs $\neg x_1, \ldots, \neg x_n$. We show that (a) For $n = 2^r - 1$, circuits computing Parity with $r-1$ NOT gates have size at least $6n - \log_2(n+1) - O(1)$, and (b) For $n = 2^r - 1$, inverters with r NOT gates have size at least $8n - \log_2(n+1) - O(1)$. We derive our bounds above by considering the minimum size of a circuit with at most r NOT gates that computes Parity for *sorted* inputs $x_1 \geq \cdots \geq x_n$. For an *arbitrary* r, we completely determine the minimum size. It is $2n - r - 2$ for odd n and $2n - r - 1$ for even n for $\lceil \log_2(n+1) \rceil - 1 \leq r \leq n/2$, and it is $\lfloor 3/2\, n \rfloor - 1$ for $r \geq n/2$. We also determine the minimum size of an *inverter for sorted inputs* with at most r NOT gates. It is $4n - 3r$ for $\lceil \log_2(n+1) \rceil \leq r \leq n$. In particular, the negation-limited inverter for sorted inputs due to Fischer, which is a core component in all the known constructions of negation-limited inverters, is shown to have the minimum possible size. Our fairly simple lower bound proofs use gate elimination arguments in a somewhat novel way.

1 Introduction and Summary

Although exponential lower bounds are known for the monotone circuit size [4], [6], at present we cannot prove a superlinear lower bound for the size of circuits computing an explicit Boolean function; the largest known lower bound is $5n - o(n)$ [7], [10], [8]. It is natural to ask: What happens if we allow a limited number of NOT gates? The hope is that by the study of *negation-limited* complexity of Boolean functions under various scenarios [3], [17], [15], [2], [1], [13], we understand the power of NOT gates better.

We consider circuits consisting of AND/OR/NOT gates, and the *size* of a circuit is the number of gates in it. An *r-circuit* is a circuit with at most r NOT gates. For a Boolean function f, let size(f), size$_r$(f), and size$_{\mathrm{mono}}$(f) respectively denote the minimum size of general circuits, r-circuits, and monotone circuits computing f.

An *inverter* for n Boolean inputs $x_1, \ldots, x_n$ is a circuit whose outputs are the negations of the inputs, i.e., $\neg x_1, \ldots, \neg x_n$. We denote this n-input n-output function by Inv$_n$. Beals, Nishino, and Tanaka [3] have shown that one can construct a size-$O(n \log n)$ depth-$O(\log n)$ inverter with $\lceil \log_2(n+1) \rceil$ NOT gates. The Boolean function Parity$_n(x_1, \ldots, x_n)$ is 1 if $\sum x_i$ is odd, and 0 otherwise. For general AND/OR/NOT circuits, Red'kin [12] has shown that size(Parity$_n$) = $4n - 4$.

Following previous works, which we will explain below, we consider the circuit complexity of Parity$_n$ and Inv$_n$ with a *tightly limited* number of NOT gates: We assume that $n = 2^r - 1$ and we consider computations of Parity$_n$ and Inv$_n$ with $r-1$ and r NOT gates respectively. For Parity$_n$ and Inv$_n$, $n = 2^r - 1$ is the

Table 1: the lower bounds of previous works and this paper for the negation-tightly-limited circuit size

	Parity	Inverter
[17]	$4n + 3\log_2(n+1) - O(1)$	
[3]		$5n + 3\log_2(n+1) - O(1)$
[14]/ [15]	$5.33n + \log_2(n+1)/3 - O(1)$	$7.33n + \log_2(n+1)/3 - O(1)$
this paper	$6n - \log_2(n+1) - O(1)$	$8n - \log_2(n+1) - O(1)$

maximum n such that computations are possible with $r-1$ and r NOT gates respectively. The Boolean function $\text{Majority}_n(x_1, \dots, x_n)$ is 1 if $\sum x_i \geq n/2$, and 0 otherwise. We give the following lower bounds.

Theorem 1 *For $n = 2^r - 1$,*

$$\begin{aligned} &\text{size}_{r-1}(\text{Parity}_n) \\ &\quad \geq 2n - \log_2(n+1) - 1 + \text{size}_{\text{mono}}(\text{Majority}_n) \\ &\quad \geq 6n - \log_2(n+1) - O(1). \end{aligned}$$

Theorem 2 *For $n = 2^r - 1$,*

$$\begin{aligned} &\text{size}_r(\text{Inv}_n) \\ &\quad \geq 4n - \log_2(n+1) + \text{size}_{\text{mono}}(\text{Majority}_n) \\ &\quad \geq 8n - \log_2(n+1) - O(1). \end{aligned}$$

Now we explain the previously known lower bounds shown in Table 1, and how we obtain our improvements focusing on Parity_n.

Let C be a circuit computing Parity_n with a tightly limited number of NOT gates as in Theorem 1. Then, the first NOT gate N, i.e., a unique NOT gate that is closest to the inputs, must compute $\neg\text{Majority}_n$, and the subcircuit C' at the immediate predecessor of N is a monotone circuit computing Majority_n. Long [9] has shown that such a monotone circuit has size at least $4n - O(1)$:

Proposition 1 *(Long [9])*

$$\text{size}_{\text{mono}}(\text{Majority}_n) \geq 4n - O(1).$$

We want to show that in addition to those gates in the subcircuit C', the circuit C must contain a certain number of gates; i.e., we want to show as good a lower bound as possible for the number of gates in $C - C'$. Tanaka, Nishino, and Beals [17] showed that there are at least $3\log_2(n+1)$ additional gates; Sung [14] and Sung and Tanaka [15] showed that there are at least about $1.33n$ additional gates; we show that there are at least about $2n$ additional gates. We show this in the following way.

We argue that a part of $C - C'$ must be computing what we call a sorted parity function, and we show that a circuit computing a sorted parity function has size at least about $2n$ if the number of NOT gates is tightly limited. A Boolean function $f : \{0,1\}^n \to \{0,1\}$ is a *sorted parity* function if for all sorted inputs $x_1 \geq x_2 \geq \cdots \geq x_n$, $f(x_1, \dots, x_n) = \text{Parity}(x_1, \dots, x_n)$. A function f is a sorted $\neg$parity function if for all sorted inputs $x_1 \geq x_2 \geq \cdots \geq x_n$, $f(x_1, \dots, x_n) = \neg\text{Parity}(x_1, \dots, x_n)$.

In fact, we completely determine the minimum size of circuits with at most r NOT gates computing Sorted Parity_n and Sorted $\neg\text{Parity}_n$, where a parameter r is an arbitrary nonnegative integer. From about $2n$, the minimum size decreases by 1 with each additional NOT gate. This decrease stops at about $1.5n$: one cannot make a circuit smaller using more NOT gates.

We also consider the minimum size of an *inverter for sorted inputs*, i.e., a circuit with Boolean inputs $x_1, \dots, x_n$ that outputs $\neg x_1, \dots, \neg x_n$ for all the sorted inputs $x_1 \geq \cdots \geq x_n$. The negation-limited inverter for sorted inputs due to Fischer [5] (shown in Figure 3) is a core component in all the known constructions of negation-limited inverters due to Fischer [5], Tanaka and Nishino [16], and Beals, Nishino and Tanaka [3]. (Explanations of all the three inverters can be found in [3].) We again completely determine the minimum size of inverters for sorted inputs with at most r NOT gates for any r. In particular, we show that Fischer's inverter for sorted inputs has the minimum possible size.

Table 2: the size and the number of AND/OR/NOT gates in a smallest circuit with $\leq r$ NOTs computing Sorted Parity; $t = \lceil \log_2(n+1) \rceil - 1$.

	size	AND	OR	NOT
$\lfloor n/2 \rfloor \leq r$	$\lfloor 3/2\, n \rfloor - 1$	$\lfloor n/2 \rfloor$	$\lceil n/2 \rceil - 1$	$\lfloor n/2 \rfloor$
$t \leq r \leq \lfloor n/2 \rfloor$, n odd	$2n - r - 2$	$n - r - 1$	$n - r - 1$	r
$t \leq r \leq \lfloor n/2 \rfloor$, n even	$2n - r - 1$	$n - r$	$n - r - 1$	r
$r < t$	not computable			

Table 3: the size and the number of AND/OR/NOT gates in a smallest circuit with $\leq r$ NOTs computing Sorted $\neg$Parity; $t' = \lceil \log_2(n+2) \rceil - 1$.

	size	AND	OR	NOT
$\lfloor n/2 \rfloor \leq r$	$\lceil 3/2\, n \rceil - 1$	$\lceil n/2 \rceil - 1$	$\lfloor n/2 \rfloor$	$\lceil n/2 \rceil$
$t' \leq r \leq \lfloor n/2 \rfloor$, n odd	$2n - r$	$n - r$	$n - r$	r
$t' \leq r \leq \lfloor n/2 \rfloor$, n even	$2n - r - 1$	$n - r - 1$	$n - r$	r
$r < t'$	not computable			

We think that our complete determination of $\text{size}_r(\text{Sorted Parity}_n)$ and $\text{size}_r(\text{Sorted Inv}_n)$ are interesting in their own. For the trade-off of size versus the number of NOT gates, an *asymptotically* tight result has been shown by Amano, Maruoka, and Tarui [2]. They showed that for $0 \leq r \leq \log_2 \log_2 n$, the minimum size of circuits computing Merge_n with r NOT gates is $\Theta(n \log n/2^r)$; thus they showed a smooth trade-off between the monotone case of $\Theta(n \log n)$ and the general case of $\Theta(n)$. But as far as we know, our results for Sorted Parity and inverters for sorted inputs are the first ones that establish *exact* trade-offs.

Our fairly simple lower bound proofs use gate elimination arguments in a somewhat novel way. The following are precise statements of our results.

Theorem 3 *The size and the number of AND/OR/NOT gates in smallest circuits with at most r NOT gates that compute Sorted* $\textit{Parity}_n$ *and Sorted* $\neg\textit{Parity}_n$ *are as shown in Table 2 and Table 3. In particular, for* $n = 2^s - 1$, *a smallest circuit with* $s - 1$ *NOT gates computing Sorted* $\textit{Parity}_n$ *has size* $2n - s - 1 = 2n - \log_2(n+1) - 1$.

Theorem 4 *For* $\lceil \log_2(n+1) \rceil \leq r \leq n$, *a smallest inverter for sorted inputs with at most r NOT gates has size* $4n - 3r$ *consisting of* $2n - 2r$ *AND gates,* $2n - 2r$ *OR gates, and r NOT gates. In particular, for* $n = 2^r - 1$, *a smallest inverter for sorted inputs with r NOT gates has size* $4n - 3r = 4n - 3\log_2(n+1)$.

2 Lower Bounds for Parity and Inverters

2.1 Preliminaries

Markov [11] precisely determined the minimum number of NOT gates necessary to compute a Boolean function. We state a special case of Markov's result relevant to our work. (Fischer [5] contains a good exposition of Markov's result.)

Proposition 2 *(Markov [11]) The maximum n such that* $\textit{Inv}_n$ *is computable by an r-circuit is* $n = 2^r - 1$. *The maximum n such that* $\textit{Parity}_n$ *is computable by an r-circuit is* $n = 2^{r+1} - 1$.

We will use the following result by Sung and Tanaka[15].

Lemma 1 *(Sung and Tanaka [15]) For* $n = 2^r - 1$,

$$\text{size}_r(\text{Inv}_n) \geq \text{size}_{r-1}(\text{Parity}_n) + 2n + 1.$$

2.2 Crossing wires

We introduce the notion of *crossing wire* and show simple lemmas. The lemmas are not strictly necessary for our proofs of the theorems, but their statements and proofs should be helpful for understanding our framework, and we think that the lemmas may be useful for further investigations of negation-limited circuits. A similar notion of *boundary gate* has been introduced by Sung [14]. We focus on wires as opposed to gates.

Fix a circuit $C$. A gate $g$ in $C$ is *black* if there is a path from some input to $g$ going through a NOT gate, including the case where $g$ itself is a NOT gate. Otherwise $g$ is *white*; also, inputs $x_1, \ldots, x_n$ are white.

Say that a wire going from gate $g$ to gate $h$ is a *crossing* wire if $g$ is white and $h$ is black. The white gates and inputs constitute the *monotone part* of $C$, and the black gates constitute the *nonmonotone part*.

Lemma 2 *Distinct crossing wires go into distinct gates.*

Proof. Let w_1 from gate g_1 to gate h_1 and w_2 from gate g_2 to gate h_2 be distinct crossing wires. By definition, g_1 and g_2 are white. If $h_1 = h_2$, this single gate is white; this contradicts the assumption that w_1 and w_2 are crossing wires. □

Lemma 3 *Let $C$ be a circuit computing a nonmonotone Boolean function $f$. Suppose that there are $a_0, \ldots, a_k \in \{0,1\}^n$ such that $a_0 < \cdots < a_k$ and $f(a_i) \neq f(a_{i+1})$ for $0 \leq i < k$. Then, $C$ contains at least $k$ crossing wires.*

Proof. The output gate $T$ of a nonmonotone circuit $C$ is black. Hence any path in $C$ from an input x_i to $T$ contains a crossing wire. If the values on all crossing wires remain the same, then the output remains the same. The value of a crossing wire changes only monotonically. The lemma follows. □

We note that the two lemmas above immediately yield an $n$ lower bound for the size of nonmonotone area of circuits computing Parity$_n$ and Inv$_n$.

2.3 Proofs of Theorems 1 and 2

We prove Theorems 1 and 2 using the lower bound for Sorted Parity$_n$ in Theorem 3, which will be proved in Section 3.

Proof of Theorem 1. Let $C$ be an $(r-1)$-circuit that computes Parity$_n$ for $n = 2^r - 1$. It is known that there is a NOT gate $N$ in $C$ such that the subcircuit C' at its immediate predecessor is a monotone circuit computing Majority$_n$. All the gates in C' are white, and by Proposition 1 the number of them is at least $\text{size}_{\text{mono}}(\text{Majority}_n) \geq 4n - O(1)$.

We can convert the nonmonotone, black part of $C$ into a circuit computing Sorted Parity for new inputs $y_1, \ldots, y_n$ as follows. Consider the chain $\langle a_0 = 0^n, a_1 = 10^{n-1}, \ldots, a_n = 1^n \rangle$, and the computation of $C$ on $a_0, \ldots, a_n$. When the input changes from $a_{i-1}$ to a_i $(1 \leq i \leq n)$, some crossing wires change the value from 0 to 1. Let W_i be the set of such crossing wires. Note that each W_i is nonempty and the sets W_i's are mutually disjoint.

Connect a new input y_i to all the gates $g$ in $C$ such that some crossing wire w in W_i goes into $g$. Let $D$ be the circuit thus obtained. Clearly, $D$ computes Sorted Parity for $y_1 \geq \cdots \geq y_n$, and the number of gates in $D$ is a lower bound for the number of black gates in $C$. By the lower bound for Sorted Parity$_n$ in Theorem 3, the size of $D$ is at least $2n - (r-1) - 2 = 2n - \log_2(n+1) - 1$.

Adding up the lower bounds for the number of white gates in $C$ and the number of black gates in $C$ yields the theorem. □

Theorem 2 immediately follows from Theorem 1 and Lemma 1. We note that instead of using Lemma 1, we can argue similarly as above using the lower bound in Theorem 4, and obtain a lower bound that is smaller by $2\log_2 n$ than the bound in Theorem 2.

3 Sorted Input Case: The Minimum Size Determined

The upper bounds of Theorem 3 and Theorem 4 can be shown by straightforward constructions as we will explain in section 3.2. We prove the lower bounds of Theorem 3 and Theorem 4 in section 3.1.

3.1 Lower bounds

We use well-known gate elimination arguments: We fix x_i, one at a time, to be 0/1 and eliminate some gates. A gate $g$ is eliminated if its value is fixed or else the value of one wire coming into $g$ is fixed. In the latter case, the other input wire of $g$ replaces all the out-going wires of $g$, and $g$ is eliminated. A lower bound for the total number of eliminations is a lower bound for the number of gates in a circuit. More information on gate elimination methods can be found, e.g., in Wegener's book [18].

Proof of the lower bound of Theorem 3. Assume that $n$ is odd and let $C$ be a circuit computing Sorted Parity$_n$ for $x_1 \geq \cdots \geq x_n$ at the top output gate $T$. Starting from $(0, 0, \ldots, 0)$, consider flipping and fixing $x_i = 1$ for $i = 1, \ldots, n-1$, in this order one at a time: Fix $x_i = 1$ after $x_1, \ldots, x_{i-1}$ have been fixed and remain to be 1. Each time we flip and fix $x_i = 1$, the value of $T$ changes flipping from 0 to 1 or 1 to 0. There must be a path $p$ from x_i to $T$ such that all the gates on $p$ flip the values when we fix $x_i = 1$. Call such a path a *propagating* path with respect to x_i.

Consider fixing $x_i = 1$. Let $p$ be a propagating path for x_i. Consider the gates on $p$ from x_i towards $T$. If all the gates on $p$ (including $T$) are ORs, fixing $x_i = 1$ will fix $T = 1$; this is a contradiction. Thus there is either an AND or a NOT in $p$. Let $g$ be the first non-OR gate in $p$. All the OR gates, if any, before $g$ are *fixed* to be 1 once we fix $x_i = 1$. Thus one input wire of $g$ is fixed to be 1.

(1) If $g$ is AND, $g$ is eliminated.

(2) If $g$ is NOT, $g$ is fixed to be 0 and is eliminated. In this case, there must be at least one AND/OR gate in $p$ beyond $g$: If all the gates beyond $g$ are NOTs, all their values are fixed; this is a contradiction. Hence at least one AND/OR gate (the first AND/OR beyond $g$) gets eliminated.

Now assume that the circuit $C$ contains $s$ NOT gates. From (1) and (2) we see that there are at least $n-1$ AND/OR gates; thus there are at least $n-1+s$ gates. This bound becomes meaningful when $s$ is large. In particular, combined with the bounds we derive below it will be easy to see that a smallest circuit for Sorted Parity$_n$ does not contain more than $\lfloor n/2 \rfloor$ NOT gates. By (1), at least $n-1$ AND/NOT gates are eliminated; thus the circuit contains at least $n-1-s$ ANDs.

Starting from $(1, 1, \ldots, 1)$, consider flipping $x_i = 0$ for $i = n, n-1, \ldots, 2$ in this order one at a time. Dual arguments yield the same lower bound for the number of ORs.

Consider the case where $n$ is even. In this case the circuit obtained after fixing $x_i = 1$ for $i = 1, \ldots, n-1$ must contain one NOT gate; thus at most $s-1$ NOT gates are eliminated, and hence the lower bound for the number of ANDs increases by 1. A similar increase occurs for odd $n$ and Sorted $\neg$Parity$_n$. □

For Theorem 4 we want to show a lower bound about twice as large by showing that the number of AND/OR gates eliminated is twice as large.

In the lower bound proof of Theorem 3 above, the eliminations of gates are always due to the fact that the value of a gate has been *determined* by having fixed some inputs. In the lower bound proof of Theorem 4, we also eliminate a gate when its value is not necessarily determined for an arbitrary input, but its value *must stay constant* for *sorted* inputs. With this additional argument we proceed similarly as in the lower bound proof of Theorem 3.

Proof of the lower bound of Theorem 4. Let $C$ be an inverter for $n$ sorted inputs $x_1 \geq \cdots \geq x_n$. Starting from $(0, 0, \ldots, 0)$, consider flipping and fixing $x_i = 1$ for $i = 1, \ldots, n$: Fix $x_i = 1$ after $x_1, \ldots, x_{i-1}$ have been fixed and remain to be 1. Each time we flip and fix $x_i = 1$, the output $\overline{x_i}$ changes flipping from 1 to 0. There must be a path $p$ from x_i to $\overline{x_i}$ such that all the gates on $p$ flip the values when we fix $x_i = 1$. Call such a path a *propagating* path for x_i.

Consider fixing $x_i = 1$. Let $p$ be a propagating path for x_i. Consider the gates on $p$ from x_i towards $\overline{x_i}$. If all the gates on $p$ are ORs, fixing $x_i = 1$ will fix $\overline{x_i} = 1$; this is a contradiction. Thus there is either an AND or a NOT in $p$.

Let $g$ be the first non-OR gate in $p$. The gate $g$ gets eliminated after fixing $x_i = 1$. Note that if $g$ is an AND, the value of $g$ is 1 after fixing $x_i = 1$ since all the gates, if any, before $g$ are ORs.

Let $h$ be the last non-OR gate in $p$. All the gates, if any, beyond $h$ are ORs. After fixing $x_i = 1$, the values of all the gates between $h$ and the output $\overline{x_i}$, including $h$ and $\overline{x_i}$, are 0.

We claim that we can fix $h$ to be 0 and thus eliminate $h$ from the circuit in the following sense. We have fixed $x_1, \dots, x_i$ to be 1; $x_{i+1}, \dots, x_n$ are 0 at present. We will further flip and fix $x_{i+1}, \dots, x_n$ to be 1 one at a time; but in this process the value of gate $h$ must remain to be 0 since if the gate $h$ has value 1, the output $\overline{x_i}$ gets flipped back from 0 to 1 contradicting to the fact that x_i has been fixed and remains to be 1. Since the gate $h$ will always be 0, we can fix $h$ to be 0 and eliminate $h$; the resulting circuit behaves in the same way. We note that if we set $x_{i+1}, \dots, x_n$ to be a *non*-sorted 0/1 sequence, it is possible that the gate $h$ evaluates to 1 even if $x_1, \dots, x_i$ are all 1.

It is possible that the gate $g$ and $h$ are the same NOT gate, i.e., $g = h$. But they can *not* be the same AND gate since after fixing $x_i = 1$, $h$ is 0 and $g$ is 1 if $g$ is an AND. Thus unless both $g$ and $h$ are NOTs, $g \neq h$. Therefore if the circuit $C$ contains $s$ NOT gates, we can eliminate a total of at least $2n - 2s$ AND gates, and hence $C$ contains at least $2n - 2s$ AND gates.

The dual argument about starting from $(1, 1, \dots, 1)$ and fixing $x_i = 0$ for $i = n, n-1, \dots, 1$ yields the same lower bound for the number of ORs. □

3.2 Upper bounds

Proof of the upper bound of Theorem 3. We can construct a smallest circuit computing Sorted Parity$_n$ with at most $r$ NOT gates for odd $n$ as follows. Constructions for even $n$ and for Sorted ¬Parity will be explained in the end.

CASE 1: $r = \lceil \log_2(n+1) \rceil - 1$ and $n = 2^{r+1} - 1$: See Figure 1.

CASE 2: $r = \lceil \log_2(n+1) \rceil - 1$ and $2^r \leq n < 2^{r+1} - 1$: See Figure 2.

In cases 1 and 2 it is easy to see that y_i's are sorted if x_i's are sorted, and that the circuit consists of $n - r - 1$ ANDs, $n - r - 1$ ORs, and $r$ NOTs.

CASE 3: $r > \lceil \log_2(n+1) \rceil - 1$: Construct a circuit of the following form:

$$(x_1 \wedge \overline{x_2}) \vee \cdots \vee (x_{2s-1} \wedge \overline{x_{2s}}) \\ \vee \text{Sorted Parity}_{n-2s}(x_{2s+1}, \dots, x_n),$$

where Sorted Parity$_{n-2s}$ is computed by a circuit in case 1 or case 2: Let $s$ be the maximum integer satisfying $2^{(r-s)+1} - 1 \geq n - 2s \geq 1$. Use $s$ NOT gates for $s$ pairs $(x_1, x_2), \dots, (x_{2s-1}, x_{2s})$, and use $\lceil \log_2(n - 2s + 1) \rceil - 1$ NOT gates for $x_{2s+1}, \dots, x_n$ as in cases 1 and 2. As for the size, the analysis for cases 1 and 2 applies for the subcircuit for $x_{2s+1}, \dots, x_n$, and we are using $s$ ANDs, $s$ ORs, and $s$ NOTs additionally.

For even $n$, construct a circuit as

$$\text{SortedParity}(x_1, \dots, x_{n-1}) \wedge \overline{x_n}.$$

For Sorted ¬Parity$_n$, construct a circuit as

$$\overline{x_1} \vee \text{Sorted Parity}(x_2, \dots, x_n).$$

□

Proof of the upper bound of Theorem 4. Construct a circuit as follows.

CASE 1: $r = \lceil \log_2(n+1) \rceil$, $n = 2^r - 1$: Figure 3 shows the circuit due to Fischer.

CASE 2: $r = \lceil \log_2(n+1) \rceil$, $2^{r-1} \leq n < 2^r - 1$: Use $\overline{x_p}$ instead of $\overline{x_{2^{r-1}}}$ similarly as in case 2 of Sorted Parity.

CASE 3: $r > \lceil \log_2(n+1) \rceil$: Similarly as in case 3 of Sorted Parity, apply $s$ NOTs directly to inputs $x_1, \dots, x_s$ to obtain outputs $\overline{x_1}, \dots, \overline{x_s}$, and use $\lceil \log_2(n - s + 1) \rceil$ NOT gates for $x_{s+1}, \dots, x_n$ to obtain $\overline{x_{s+1}}, \dots, \overline{x_n}$.

It is easy to see that the circuit thus constructed has size $4n - 3r$ consisting of $2n - 2r$ ANDs, $2n - 2r$ ORs, and $r$ NOTs. □

References

[1] K. Amano and A. Maruoka, A Superpolynomial Lower Bound for a Circuit Computing the Clique Function with at most $(1/6)\log\log n$ Negation Gates, *SIAM J. Comput.* **35(1)**, pp. 201–216, 2005.

[2] K. Amano, A. Maruoka and J. Tarui, On the Negation-Limited Circuit Complexity of Merging, *Discrete Applied Mathematics* **126(1)**, pp. 3–8, 2003.

[3] R. Beals, T. Nishino and K. Tanaka, On the Complexity of Negation-Limited Boolean Networks, *SIAM J. Comput.* **27(5)**, pp. 1334–1347, 1998.

[4] R. Boppana and M. Sipser, The Complexity of Finite Functions, *Handbook of Theoretical Computer Science, Volume A: Algorithms and Complexity*, J. v. Leeuwen editor, Elsevier/MIT Press, pp. 757–804, 1990.

[5] M. Fischer, Lectures on Network Complexity, *Technical Report* 1104, CS Department, Yale University, `http://cs-www.cs.yale.edu/homes/fischer`, 1974, revised 1996.

[6] D. Harnik and R. Raz, Higher Lower Bounds on Monotone Size, *Proc. of 32nd STOC*, pp. 378–387, 2000.

[7] K. Iwama, D. Lachish, H. Morizumi and R. Raz, An Explicit Lower Bound of $5n - o(n)$ for Boolean Circuits, *manuscript*, `http://www.wisdom.weizmann.ac.il/~ranraz`, 2005.

[8] K. Iwama and H. Morizumi, An Explicit Lower Bound of $5n - o(n)$ for Boolean Circuits, *Proc. of 27th MFCS*, LNCS vol. 2420, pp. 353–364, 2002.

[9] D. Long, The Monotone Circuit Complexity of Threshold Functions, Unpublished manuscript, University of Oxford, 1986.

[10] O. Lachish and R. Raz, Explicit Lower Bound of $4.5n - o(n)$ for Boolean Circuits, *Proc. of 33rd STOC*, pp. 399–408, 2001.

[11] A. A. Markov, On the Inversion Complexity of a System of Functions, *J. ACM* **5(4)**, pp. 331–334, 1958.

[12] N. Red'kin, Proof of Minimality of Circuits Consisting of Functional Elements, *Problemy Kibernetika* **23**, pp. 83–102, 1973 (in Russian); Translation: *Systems Theory Research* **23**, pp. 85–103, 1973.

[13] T. Sato, K. Amano, and A. Maruoka, On the Negation-Limited Circuit Complexity of Sorting and Inverting k-tonic Sequences, *Proc. of 12th COCOON*, LNCS vol. 4112, pp. 104–115, 2006.

[14] S. Sung, On Negation-Limited Circuit Complexity, *Ph.D. thesis*, Japan Advanced Institute of Science and Technology, 1998.

[15] S. Sung and K. Tanaka, Lower Bounds on Negation-Limited Inverters, *Proc. of 2nd DMTCS: Discrete Mathematics and Theoretical Computer Science Conference*, pp. 360–368, 1999.

[16] K. Tanaka and T. Nishino, On the Complexity of Negation-Limited Boolean Networks, *Proc. of 26th STOC*, pp. 38–47, 1994.

[17] K. Tanaka, T. Nishino and R. Beals, Negation-Limited Circuit Complexity of Symmetric Functions, *Inf. Process. Lett.* **59(5)**, pp. 273–279, 1996.

[18] I. Wegener, The Complexity of Boolean Functions, Wiley-Teubner Series in Computer Science, 1987.

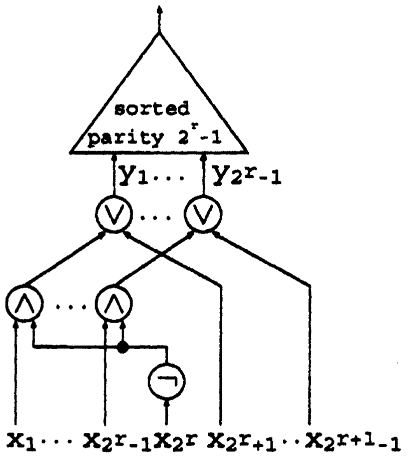
sorted
parity 2^r-1
$y_1 \cdots$
y_{2^r-1}
$x_1 \cdots x_{2^r-1} x_{2^r}\ x_{2^r+1} \cdots x_{2^{r+1}-1}$

Figure 1: Sorted Parity for $n = 2^{r+1} - 1$ with r NOTs

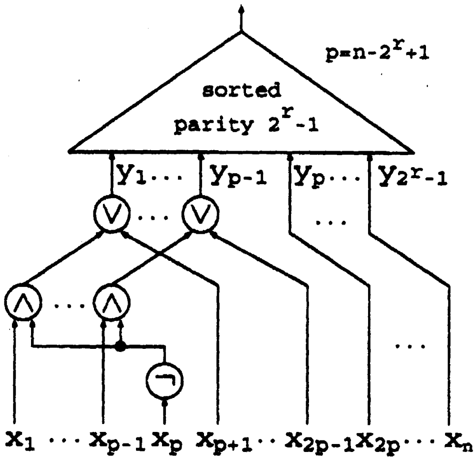

Figure 2: Sorted Parity for $2^r \leq n < 2^{r+1} - 1$ with r NOTs

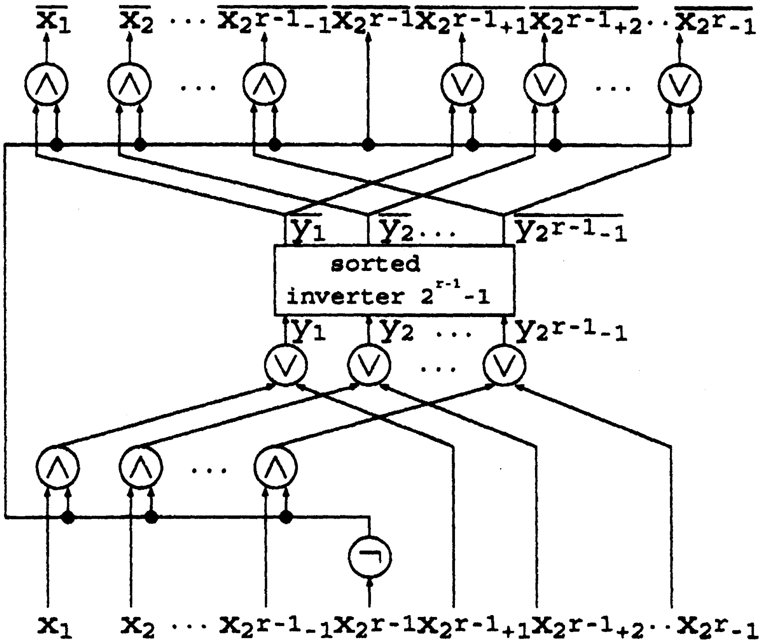
$\overline{x_1}$ $\overline{x_2}$ $\cdots$ $\overline{x_{2^{r-1}-1}}$ $\overline{x_{2^{r-1}}}$ $\overline{x_{2^{r-1}+1}}$ $\overline{x_{2^{r-1}+2}}$ $\cdots$ $\overline{x_{2^r-1}}$
$\overline{y_1}$
$\overline{y_2} \cdots$
$\overline{y_{2^{r-1}-1}}$
sorted
inverter 2^{r-1}-1
y_1
$y_2 \cdots$
$y_{2^{r-1}-1}$
x_1 x_2 $\cdots$ $x_{2^{r-1}-1} x_{2^{r-1}} x_{2^{r-1}+1} x_{2^{r-1}+2} \cdots x_{2^r-1}$

Figure 3: Fischer's inverter for sorted inputs $x_1 \geq \cdots \geq x_n$ with r NOTs, where $n = 2^r - 1$