**DIMACS Technical Report 95-31**
**August 1995**

# Improved construction of negation-limited circuits

by

Robert Beals[1] ,[2] ,[3]

[1]DIMACS Postdoc
[2]School of Mathematics Institute For Advanced Study Olden Lane Princeton, New Jersey, 08540
[3]Research supported by: NSF Mathematical Sciences Posdoctoral Fellowship, Institute for Advanced Study, and the Sloan Foundation

---

DIMACS is a cooperative project of Rutgers University, Princeton University, AT&T Bell Laboratories and Bellcore.

DIMACS is an NSF Science and Technology Center, funded under contract STC–91–19999; and also receives support from the New Jersey Commission on Science and Technology.

## ABSTRACT

A theorem of Markov states that any system of boolean functions on $n$ variables may be computed by a boolean circuit containing at most $\lceil \log_2(n+1) \rceil$ negation gates. We call such a circuit *negation-limited*. A circuit with inputs $x_1, \ldots, x_n$ and outputs $\neg x_1, \ldots, \neg x_n$ is called an *inverter*. Fischer has constructed negation-limited inverters of size $O(n^2 \log n)$ and depth $O(\log n)$. Recently, Tanaka and Nishino have reduced the circuit size to $O(n \log^2 n)$ at the expense of increasing the depth to $\log^2 n$. We construct negation-limited inverters of size $O(n \log n)$, with depth only $O(\log n)$. We also improve a technique of Valiant for constructing monotone circuits for slice functions (introduced by Berkowitz).

# 1 Introduction

## 1.1 Background

The theory of monotone boolean circuit complexity has met with considerable success [11, 3, 14, 6, 10]. Good lower bounds for both size and depth of monotone circuits (i.e., circuits without negation gates) computing many explicit functions are now known. However, the theory of general boolean circuits, *with* negation gates, is much less well understood. Exponential gaps between monotone and general circuit complexity are known, both for circuit size [14] and circuit depth [10]. However, the effect of negation gates on circuit complexity remains to a large extent a mystery.

This motivates the study of *negation-limited* circuit complexity: what is the effect on circuit complexity of restricting the number of negations available? Markov [8] gives an explicit formula (see below) for the maximum number $b(n)$ of negation gates *required* to compute any system of boolean functions of $n$ variables, without regard to circuit complexity. Fischer [7] shows that restricting the number of negations in a circuit to $b(n)$ entails only a polynomial blowup in circuit size. This is in sharp contrast to the situation for monotone circuits: E. Tardos [14] has shown that there is an exponential gap between the general complexity and the monotone complexity of some monotone functions. In related work, Santha and Wilson [12] have studied the negation-limited complexity of constant depth circuits, obtaining both upper and lower bounds for the number of negation gates required by circuits of a given depth. Recently, Tanaka and Nishino [13] have found an alternative to Fischer's construction; they decrease the size of the circuits at the expense of increasing the depth. We further reduce the size, while simultaneously reducing the depth to that of Fischer. We also exhibit a relationship between negation-limited complexity and the monotone complexity of certain types of monotone functions.

## 1.2 Definitions and preliminaries

Let $F$ be a collection of boolean functions $f_1, \ldots, f_m$ defined on $\{0,1\}^n$. We denote by $C(F)$ or $C(f_1, \ldots, f_m)$ the *circuit complexity* of $F$, i.e., the size (number of gates) of the smallest circuit of AND, OR, and NEGATION gates with inputs

$$x_1, \ldots, x_n$$

and outputs

$$f_1(x_1, \ldots, x_n), \ldots, f_m(x_1, \ldots, x_n).$$

We denote by $C^k(F)$ or $C^k(f_1, \ldots, f_m)$ the size of the smallest such circuit which uses no more than $k$ negation gates. If the system of functions cannot be computed with only $k$ negation gates, then $C^k(F)$ is undefined. The *inverter* is the collection $I_n$ of functions $f_1, \ldots, f_n$, where for all $1 \leq i \leq n$, $f_i(x_1, \ldots, x_n) = \neg x_i$. For $0 \leq k \leq n$, the $(k,n)$-*inverter* is the collection $I_n^k$ of functions $f_1, \ldots, f_n$, where for all $1 \leq i \leq n$,

$$f_i(x_1, \ldots, x_n) = \begin{cases} 0 & \text{if } \sum_{j=1}^n x_j < k \\ \neg x_i & \text{if } \sum_{j=1}^n x_j = k \\ 1 & \text{if } \sum_{j=1}^n x_j > k. \end{cases}$$

For $n > 0$ let $b(n)$ denote $\lceil \log(n+1) \rceil$, the number of bits in the binary representation of $n$ (all logarithms in this paper are base 2). Markov [8] has shown that $b(n)$ negation gates suffice to compute any system $f_1, \ldots, f_m$ of functions, and that $b(n)$ negation gates are indeed necessary for the inverter. Thus, $C^k(F)$ is always defined for $k \geq b(n)$. We call $C^{b(n)}(F)$ the *negation-limited* complexity of the system of functions $F$.

In this paper, we give improved upper bounds on the negation-limited complexity of the inverter. These in turn yield upper bounds on the negation-limited complexity of an arbitrary system of functions via the standard technique of using DeMorgan's laws to push all negations in a circuit to the inputs:

**Proposition 1.1** *For any system $F$ of $n$-input boolean functions,*

$$C^{b(n)}(F) \leq 2C(F) + C^{b(n)}(I_n). \qquad \spadesuit$$

A circuit with no negation gates is *monotone*. A *monotone function* $f$ is a boolean function which satisfies $f(x_1, \ldots, x_n) \leq f(y_1, \ldots, y_n)$ whenever $x_i \leq y_i$ for all $i$. Monotone functions are exactly those computed by monotone circuits. Observe that the $(k, n)$-inverter is a system of monotone functions. Berkowitz [5] shows that the monotone complexity of the $(k, n)$ inverter is polynomial, with a construction of size $O(n^2 \log n)$ and depth $O(\log n)$. Valiant [15] constructs monotone $(k, n)$-inverters of size $O(n \log^2 n)$ and depth $O(\log^2 n)$. We construct monotone $(k, n)$-inverters of size $O(n \log n)$ and depth $O(\log n)$, and we show that there is a close relationship between $C^0(I_n^k)$ and $C^{b(n)}(I_n)$. (For more background on monotone circuits, we refer to Section 4 of Boppana and Sipser's article [6].)

Markov [8] constructs inverters using monotone functions and $b(n)$ negations, but he does not consider the complexity of the monotone functions that he uses. Akers [2] gives the first explicit construction of a negation-limited inverter. His circuit uses $b(n)$ negation gates and positive weight threshold gates, and has size $O(n)$ and depth $O(\log n)$. For the remainder of this paper, we restrict our attention to circuits of AND, OR, and NEGATION gates. Fischer [7] gives such a circuit for $I_n$ with size $O(n^2 \log^2 n)$ and depth $O(\log^2 n)$, using only $b(n)$ negation gates. Sorting networks play a key role in Fischer's construction (and in all subsequent constructions). Using the sorting network of Ajtai, Komlós, and Szemerédi [1] (cf. [9]) in Fischer's construction reduces the size to $O(n^2 \log n)$ and the depth to $O(\log n)$. Recently Tanaka and Nishino [13] have investigated the negation-limited complexity of the inverter, giving both an upper bound of $O(n \log^2 n)$ and a lower bound of $5n + 3\log(n+1) - 6$. For the upper bound, their construction has depth $\theta(\log^2 n)$, and they ask whether some sort of size-depth tradeoff is possible (noting that Fischer's construction can be effected in depth $O(\log n)$).

We construct negation-limited circuits for $I_n$ with size $O(n \log n)$ and depth $O(\log n)$. This improves on previous negation-limited circuits for $I_n$ by a factor of at least $(\log n)$ in size. Note that the only previous construction with logarithmic depth had super-quadratic size.

The remainder of this paper is organized as follows. In Section 2 we describe elements of previous constructions which are common to ours. In Section 3 we give the heart of our construction of negation-limited inverters. In Section 4 we show the relationship between negation-limited inverters and monotone $(k, n)$-inverters. Finally, in Section 5 we conclude with a brief discussion of open problems.

# 2 The Fischer and Tanaka–Nishino constructions

We review briefly the circuits of Fischer [7] and Tanaka–Nishino [13]. These circuits have several elements in common with each other, and with ours as well. The most important of these is a sorting network.

A *comparator* is a circuit element with two inputs and two outputs. The inputs are taken from some ordered set $U$. The first output of the comparator is the maximum of the two inputs, while the second output is the minimum of the two inputs. A sorting network is a circuit, with $n$ inputs and $n$ outputs, composed entirely of comparators. For any $n$-tuple $x_1, \ldots, x_n$ of inputs from $U$, the $n$ outputs $y_1, \ldots, y_n$ of the sorting network are a permutation of the $x_i$ and are in decreasing order. Ajtai, Komlós, and Szemerédi [1] (cf. [9]) have constructed sorting networks using $O(n \log n)$ comparators, organized in $O(\log n)$ levels of $\lfloor n/2 \rfloor$ comparators each. Note that if the set $U$ is $\{0, 1\}$, then a comparator can be constructed without negations. Indeed, for $x, y \in \{0, 1\}$, $\min(x, y) = x \wedge y$ and $\max(x, y) = x \vee y$. Thus, the Ajtai–Komlós–Szemerédi result yields a monotone circuit of size $O(n \log n)$ and depth $O(\log n)$ for sorting $n$ bits.

Both Fischer and Tanaka–Nishino first sort the $n$ input bits $x_1, \ldots, x_n$. The outputs $y_1, \ldots, y_n$ of the sorting network are fed into a subcircuit $M_n$ (due to Fischer [7]) with the following properties:

1. $M_n$ has $n$ binary inputs $y_1, \ldots, y_n$ and $n$ outputs $z_1, \ldots, z_n$.

2. $M_n$ has size $O(n)$, depth $O(\log n)$, and uses $b(n)$ negation gates.

3. If $y_1 \geq y_2 \geq \ldots \geq y_n$ then $z_i = \neg y_i$ for $1 \leq i \leq n$.

The negations of the $x_i$ may now be calculated monotonically from the $x_i$, the $y_i$, and the $z_i$. Fischer does this using $O(n)$ sorting networks in parallel, resulting in a circuit of size $O(n^2 \log n)$ and depth $O(\log n)$ if the Ajtai–Komlós–Szemerédi sorting network is used. Tanaka and Nishino, on the other hand, use the monotone $(n, 2n)$-inverter of Valiant [15] which inverts $2n$ inputs provided that exactly $n$ of them equal 1 (as is the case with $x_1, \ldots, x_n, z_1, \ldots, z_n$). Valiant's circuit is monotone, with size $O(n \log^2 n)$ and depth $O(\log^2 n)$.

Note that in both the Fischer and the Tanaka–Nishino constructions, the circuit is arranged in 3 phases: the sorting network, the subcircuit $M_n$, and the top phase which computes the final answer. (We view circuits as having their inputs at the bottom and outputs at the top, so the final phase of a circuit with several phases is the top phase). The bottom two phases require only size $O(n \log n)$ and depth $O(\log n)$ (in fact, the second phase has linear size). Also, both the Fischer and the Tanaka–Nishino constructions require no more than the $x_i$, the $y_i$, and the $z_i$ as inputs to the top phase. We show that by allowing the top phase access to intermediate results from the sorting network (first phase), the top phase may be computed by a size $O(n \log n)$, depth $O(\log n)$ circuit. Our top phase may be thought of as an "upside-down" sorting network. We describe this in the next section.

# 3 Description of the circuit

Our negation-limited circuit for $I_n$, like its predecessors, begins with a sorting network and Fischer's $M_n$ circuit. We present a new approach to the third phase, which achieves considerable savings in complexity over previous constructions by means of a novel application of the Ajtai–Komlós–Szemerédi result.

Before going into the details of the circuit, we describe briefly the intuition. The differences between the three constructions ( [7, 13] and ours) seem to stem from how one thinks of the outputs of the $M_n$ subcircuit. Of course, in all three constructions, the outputs of $M_n$ are computing exactly the same system of $n$ functions. Nevertheless, these outputs are treated completely differently by the three approaches. In Fischer's original negation-limited inverter [7], the inputs and outputs of $M_n$ are thought of as the threshold functions (of $x_1, \ldots, x_n$) and their negations, respectively. Fischer uses these bits to select the output of an appropriate subcircuit, according to the number $k$ of ones in the input. There are $n+1$ such subcircuits, one for each $0 \le k \le n$. Tanaka and Nishino [13] avoid having to consider $(n+1)$ different cases in parallel by noticing that the $x_i$, together with the outputs of $M_n$, are a string of $2n$ bits containing exactly $n$ ones (such a string can be inverted monotonically using Valiant's monotone $(n, 2n)$-inverter [15]). In their construction, the outputs of $M_n$ are thought of as *balancing* the inputs $x_1, \ldots, x_n$. We take a different view. We view the outputs of $M_n$ as being a *permutation* of the negations of the $x_i$. It is the task of the third phase to *rearrange* these bits into their proper position. This rearrangement essentially undoes the sorting performed by the first phase, one level at a time.

We now present the details. Let $\ell$ be the depth of the sorting network used in the first phase. For each level in the sorting network, the third phase has a corresponding level, composed of a monotone, linear size, constant depth circuit. The ordering of the levels, however, is reversed between the first phase and the third phase. That is, we number the levels in the sorting network from the bottom (numbered 1) to top (numbered $\ell$), and we number the levels in the third phase from 1 at the top to $\ell$ at the bottom. When numbered in this way, level $i$ of the sorting network corresponds to level $i$ of the third phase. We will use $S_i$ to denote level $i$ of the sorting network, and $T_i$ to denote level $i$ of the top phase.

For $1 \le i \le \ell$, let $x_1^i, \ldots, x_n^i$ denote the inputs to $S_i$, and let $x_1^{\ell+1}, \ldots, x_n^{\ell+1}$ denote the outputs of $S_\ell$. The subcircuit $T_i$ receives $2n$ input bits: the $n$ input bits $x_1^i, \ldots, x_n^i$ of $S_i$, and the negations $\neg x_1^{i+1}, \ldots, \neg x_n^{i+1}$ of the $n$ output bits of $S_i$. For $1 \leq j \leq n$, $\neg x_j^{i+1}$ is computed one level below by $T_{i+1}$, or by $M_n$ in the case $i = \ell$. Using only linear size, constant depth, monotone circuitry, $T_i$ outputs the negations $\neg x_1^i, \ldots, \neg x_n^i$ of the $n$ input bits to $S_i$. If $i > 1$, these are passed up to the next level, $T_{i-1}$. Note that at the top level, $T_1$ outputs the negations of the inputs to $S_1$, i.e., $T_1$ outputs $\neg x_1, \ldots, \neg x_n$. The overall structure of the circuit is shown in Figure 1 (page 10. Note that for readability, the wires connecting the outputs of $S_{i-1}$ to the inputs of $T_i$ are not shown.

To complete the description of the circuit for $I_n$, we need to describe a monotone, linear size, constant depth circuit for $T_i$. Since $S_i$ is simply $\lfloor n/2 \rfloor$ comparators operating in parallel, it suffices to give a (constant size and depth) monotone circuit for each comparator. This we do in the following Lemma:

**Lemma 3.1** *Let $x$ and $y$ be boolean variables, and let $u = \neg\max(x, y)$ and $v = \neg\min(x, y)$. Then $\neg x$ and $\neg y$ may be computed monotonically from $x, y, u,$ and $v$.*

**Proof:** A quick check shows that $\neg x = u \vee (y \wedge v)$ and $\neg y = u \vee (x \wedge v)$. ♠

Suppose that $S_i$ has a comparator with inputs $x_{j_1}^i, x_{j_2}^i$ and outputs $x_{k_1}^{i+1}, x_{k_2}^{i+1}$. Then $T_i$ will have a corresponding circuit element with inputs

$$\neg x_{k_1}^{i+1}, \neg x_{k_2}^{i+1}, x_{j_1}^i, x_{j_2}^i$$

and outputs

$$\neg x_{j_1}^i, \neg x_{j_2}^i.$$

By the above Lemma, this can be done monotonically. This is illustrated in Figure 2 (page 11). As $S_i$ is a linear size, depth 1 circuit of comparators, we can construct $T_i$ to be a monotone, linear size, depth 2 circuit.

We are now ready to state and prove our main result:

**Theorem 3.2** *The negation-limited complexity of the inverter $I_n$ is $O(n \log n)$. In fact, $I_n$ may be computed by negation-limited circuits of size $O(n \log n)$ and depth $O(\log n)$.*

**Proof:** We describe the circuit. As in Fischer [7] and Tanaka–Nishino [13], the input variables $x_1, \ldots, x_n$ are the inputs of a sorting network with $\ell$ levels of $O(n)$ gates each. By Ajtai–Komlós–Szemerédi [1], we may assume $\ell = O(\log n)$. Still following Fischer and Tanaka–Nishino, the outputs $y_1, \ldots, y_n$ of the sorting network are the inputs to Fischer's $M_n$ circuit, which has size $O(n)$ and depth $O(\log n)$ and uses $b(n)$ negation gates. Since the inputs to $M_n$ are sorted, the outputs $z_1, \ldots, z_n$ satisfy $z_i = \neg y_i$ for $i = 1 \ldots n$.

From the $z_i$, and the inputs to the various levels of the sorting network, the subcircuits $T_\ell, \ldots, T_1$ described above compute the negations of the inputs to successively lower levels in the sorting network. It is clear by induction on $(\ell - i)$ that $T_i$ outputs the negations $\neg x_1^i, \ldots, \neg x_n^i$ of the inputs to $S_i$. At the top of our circuit, $T_1$ computes the negations $\neg x_1, \ldots, \neg x_n$ of the input variables.

The circuit has depth $3\ell + O(\log n) = O(\log n)$. As each level of the circuit has linear size, the total number of gates used is $O(n \log n)$. The only negation gates used are the $b(n)$ negation gates used by $M_n$, as desired. ♠

# 4 Monotone $(k, n)$-inverters

Berkowitz [5] introduces the notion of a *slice function*. The boolean function $f : \{0,1\}^n \to \{0,1\}$ is a $k$th slice if

$$f(x_1, \ldots, x_n) = \begin{cases} 0 & \text{if } \sum_{i=1}^n x_i < k \\ 1 & \text{if } \sum_{i=1}^n x_i > k. \end{cases}$$

For input strings containing exactly $k$ ones, the behavior of $f$ is not constrained. Any slice function is monotone. Note that an alternative, nonconstructive proof of Lemma 3.1 may be obtained by observing that of the four bits $x, y, u, v$, exactly two are equal to one, so any function of $x, y, u, v$ may be extended to a (monotone) slice function. The importance of monotone $(k, n)$-inverters is underscored by the following straightforward analogue of Proposition 1.1:

**Proposition 4.1** *For any system $F$ of $n$-input $k$th slice functions,*

$$C^0(F) \leq 2C(F) + C^0(I_n^k). \qquad \spadesuit$$

The following allows us to focus on the behavior of our circuit on inputs with exactly $k$ ones:

**Proposition 4.2** *$I_n^k$ is the unique system of monotone functions which agrees with $I_n$ on all inputs with exactly $k$ ones.*

**Proof:** Suppose that $f_1, \ldots, f_n$ are a system of monotone functions which agree with $I_n$ on all inputs with exactly $k$ ones. Suppose that the number of ones in $x_1, \ldots, x_n$ is strictly greater than $k$. Let $i$ be any integer between 1 and $n$. We must show that $f_i(x_1, \ldots, x_n) = 1$. By the monotonicity of $f_i$, it suffices to exhibit $y_1, \ldots, y_n$ such that $f_i(y_1, \ldots, y_n) = 1$ and for all $1 \leq j \leq n$, $y_j \leq x_j$. For this, we pick the $y_j$ so that exactly $k$ of them are one, $y_i = 0$, and each $y_j \leq x_j$ (we obtain the $y_j$ by changing some of the $x_j$ from one to zero, including $x_i$ if necessary, until exactly $k$ bits are one). The proof for the case that fewer than $k$ of the $x_j$ are one is analogous. $\spadesuit$

Berkowitz [5] constructs monotone $(k, n)$-inverters of size $O(n^2 \log n)$ and depth $O(\log n)$. Valiant [15] gives monotone circuits of size $O(n \log^2 n)$ and depth $O(\log^2 n)$ for the $(k, n)$-inverter (see also Wegener [16, 17]). Our techniques can be used to improve on Valiant's results by a factor of $(\log n)$ in size and depth:

**Theorem 4.3** *The monotone complexity of the $(k, n)$-inverter is $O(n \log n)$. In fact, $I_n^k$ has monotone circuits of size $O(n \log n)$ and depth $O(\log n)$.*

**Proof:** We use our negation-limited circuit for $I_n$, replacing the $M_n$ subcircuit by a circuit which outputs $k$ zeroes followed by $(n - k)$ ones. Note that the only negations in our circuit for $I_n$ occur in the $M_n$ subcircuit. Also, if there are exactly $k$ ones among the inputs to our negation-limited inverter, then the $M_n$ subcircuit will output $k$ zeroes followed by $(n - k)$ ones. Therefore, the new circuit will agree with $I_n$ (i.e., it will compute the negations of its input bits) whenever there are exactly $k$ ones in the input. By Proposition 4.2, we are done. ♠

Since the inputs to $M_n$ are sorted, the functions computed by the negation gates depend only on the number of ones in the input. Therefore, in the above construction, instead of replacing the outputs of $M_n$ by constants, we could have replaced the outputs of the negation gates by constants. One might ask whether it is always possible to obtain a monotone $(k,n)$-inverter from a negation-limited inverter by replacing the negation gates with constants. We shall see (perhaps surprisingly) that something close to this is true. Before formalizing this, we need the following Lemma of Tanaka–Nishino (cf. [13, Lemma 4.2]):

**Lemma 4.4 (Tanaka–Nishino)** *Suppose $n+1$ is a power of 2, and let $f$ be the function computed by some negation gate of a negation-limited circuit for $I_n$. Then $f(x_1,\ldots,x_n)$ depends only on the number of $i$ such that $x_i = 1$.*♠

Now we can make precise the relationship between negation-limited inverters and monotone $(k,n)$-inverters:

**Theorem 4.5** *Let $n$ be a positive integer. Then $C^0(I_n^k) \le C^{b(n')}(I_{n'})$, for some $n' < 2n$. In fact, a monotone $(k,n)$-inverter can be constructed with the same size and depth as a negation-limited circuit for $I_{n'}$.*

**Proof:** Let $n' = 2^r - 1$, where $2^r$ is the smallest power of 2 greater than $n$. Note that $2^r$ is at most $2n$. Let $\Delta$ be any negation-limited circuit for $I_n$. By Lemma 4.4, if the number of ones in the input is constant, then the negation gates in $\Delta$ compute constant functions. Replacing these gates by appropriate (according to $k$) constants yields a monotone $(k,n')$-inverter. To obtain a monotone $(k,n)$-inverter, we simply fix the first $(n'-n)$ input bits to 0, and ignore the first $(n'-n)$ output bits. This yields a monotone circuit $\Delta'$ computing $I_n^k$ with the same size and depth as $\Delta$. ♠

In fact, *all* known constructions of monotone $(k,n)$-inverters may be seen as negation-limited inverters with the negation gates replaced by constants. This modification applied to Fischer's circuit for $I_n$ (using the AKS sorting network [1]) yields Berkowitz's $(k,n)$-inverter. There is no previously published negation-limited inverter corresponding to Valiant's $(k,n)$-inverter, so we describe one here. First we give a sketch of Valiant's construction.

Valiant's circuit is very similar to ours (with the $M_n$ removed), except that instead of using comparators, *merging networks* are used. A merging network takes as inputs two sorted lists $u$ and $v$, and outputs the list $w = \mathrm{merge}(u,v)$ obtained by sorting the concatenated list $uv$. Batcher [4] shows that two $m$ elements lists can be merged by a network of $O(m\log m)$ comparators. Thus, if $u$ and $v$ are binary strings of length $m$, they can be merged by a monotone circuit of size $O(m\log m)$ (this circuit has depth $O(\log m)$). If $u$ is a binary string, we denote by $\neg u$ the string obtained from $u$ by replacing each symbol by its negation, and we denote by $\mathrm{reverse}(u)$ the string obtained from $u$ by reversing the order of the symbols. Note that if $u$ is in sorted (i.e., decreasing) order, then so is $\mathrm{reverse}(\neg u)$.

The bottom half of Valiant's circuit is a sorting network, built out of $\log n$ layers of merging networks (the $i$th layer merges pairs of consecutive substrings of length $2^i$). The top half is analogous to our top phase, with the following taking the place of Lemma 3.1:

**Lemma 4.6 (Valiant)** *Suppose that the sorted binary strings $u$ of length $s$ and $v$ of length $t$, together with the string $w = \mathrm{reverse}(\neg\, \mathrm{merge}(u, v))$ are given. Then $\mathrm{reverse}(\neg u)$ and $\mathrm{reverse}(\neg v)$ may be calculated by merging networks.*

**Proof:** By symmetry, we only need to consider $\mathrm{reverse}(\neg u)$. This is easily seen to be the middle $s$ bits of $\mathrm{merge}(v, w)$. ♠

(Note that the case $s = t = 1$ of Valiant's Lemma provides yet another proof of Lemma 3.1.)

Since $k$ is fixed, the negations of the output bits of the top layer of the bottom half are constants. (Actually this top layer of the bottom half is superfluous for the monotone $(k, n)$ inversion problem, and Valiant does without it. We leave it in for simplicity, and because it is necessary if we are to obtain a negation-limited inverter.) Successively higher layers of the top half calculate the reverses of the negations of inputs to merging networks in successively lower layers of the bottom half. At the top layer of the top half, the negations of the input bits are calculated (note that reversing a string of length one has no effect).

To obtain a negation-limited inverter from Valiant's circuit, we simply feed the output of the bottom half to Fischer's $M_n$ circuit, which computes the negation of the output of the top layer of the bottom half. This is then given as input to the top half. (The circuit constructed in this way is about half the size of that of Tanaka and Nishino [13], since we modify Valiant's $(k, n)$ inverter instead of using Valiant's $(n, 2n)$ inverter.) Therefore Valiant's techniques for constructing monotone $(k, n)$-inverters can also be used to construct negation-limited inverters. It is curious that this is true of all known $(k, n)$-inverters, since no converse of Theorem 4.5 is known to hold.

# 5 Concluding remarks and open problems

Current techniques for constructing negation-limited inverters seem to have reached their limit: as long as sorting is used as a preprocessing phase, size at least $cn \log n$ (for some constant $c$) will be required (see Fischer [7] for a survey, including monotone lower bounds for several sorting-like problems). From the point of view of obtaining lower bounds, Theorem 4.5 is encouraging. A superlinear lower bound on the size of monotone $(k, n)$-inverters would immediately give a superlinear lower bound on the size of a negation-limited inverter. Nevertheless, it remains open whether negation-limited inverters require superlinear size, and what relationship exists between $k$ and $C^k(I_n)$ for $b(n) \leq k \leq n$ (although trivially $C^n(I_n) = n$).

A *synchronous* circuit is one in which all input-output paths have the same length. Our circuit is not synchronous, since intermediate results calculated early on in the sorting network are required late in the third phase (also Fischer's $M_n$ circuit is not synchronous, but it may be made so with only $O(n \log n)$ additional gates). Our circuit may be made synchronous by adding $O(n \log^2 n)$ gates. The problem of finding a smaller synchronous circuit, or showing that none exists, is open. Tanaka and Nishino give a lower bound of $4n \log(n + 1) + 2n$ for the size of a synchronous negation-limited inverter. It may be worth noting that in our circuit, the bits which need to be "delayed" (i.e., used many levels higher than they are calculated) are required in a last-in, first-out order: the bits which are calculated near the top of the sorting network are used near the bottom of the third phase, and vice-versa.

## References

[1] M. Ajtai, J. Komlós, and E. Szemerédi, *An $O(n \log n)$ sorting network*, in Proceedings of the 15th ACM Symposium on the Theory of Computing, 1983, pp. 1–9.

[2] S. B. Akers, Jr., *On maximum inversion with minimum inverters*, IEEE Trans. Comput., 17 (1968), pp. 134–135.

[3] N. Alon and R. Boppana, *The monotone circuit complexity of boolean functions*, Combinatorica, 7 (1987), pp. 1–22.

[4] K. Batcher, *Sorting networks and their applications*, in AFIPS Spring Joint Computing Conference, 1968, pp. 307–314.

[5] S. J. Berkowitz, *On some relationships between monotone and non-monotone circuit complexity*, tech. rep., Computer Science Department, University of Toronto, 1982.

[6] R. B. Boppana and M. Sipser, *The complexity of finite functions*, in Handbook of Theoretical Computer Science, Vol. A, J. van Leeuwen, ed., North-Holland, Amsterdam, 1990, pp. 757–804.

[7] M. J. Fischer, *The complexity of negation-limited networks—a brief survey*, in Automata theory and formal languages: 2nd GI conference, vol. 33 of Lecture Notes in Computer Science, Springer-Verlag, Berlin, 1975, pp. 71–82.

[8] A. A. Markov, *On the inversion complexity of a system of functions*, J. Assoc. Comput. Mach., 5 (1958), pp. 331–334.

[9] N. J. Pippenger, *Communication networks*, in Handbook of Theoretical Computer Science, Vol. A, J. van Leeuwen, ed., North-Holland, Amsterdam, 1990, pp. 805–833.

[10] R. Raz and A. Wigderson, *Monotone circuits for matching require linear depth*, J. Assoc. Comput. Mach., 39 (1992), pp. 736–744.

[11] A. A. Razborov, *Lower bounds on the complexity of some Boolean functions*, Soviet Mathematics Doklady, 31 (1985), pp. 354–357.

[12] M. Santha and C. Wilson, *Limiting negations in constant depth circuits*, SIAM J. Comput., 22 (1993), pp. 294–302.

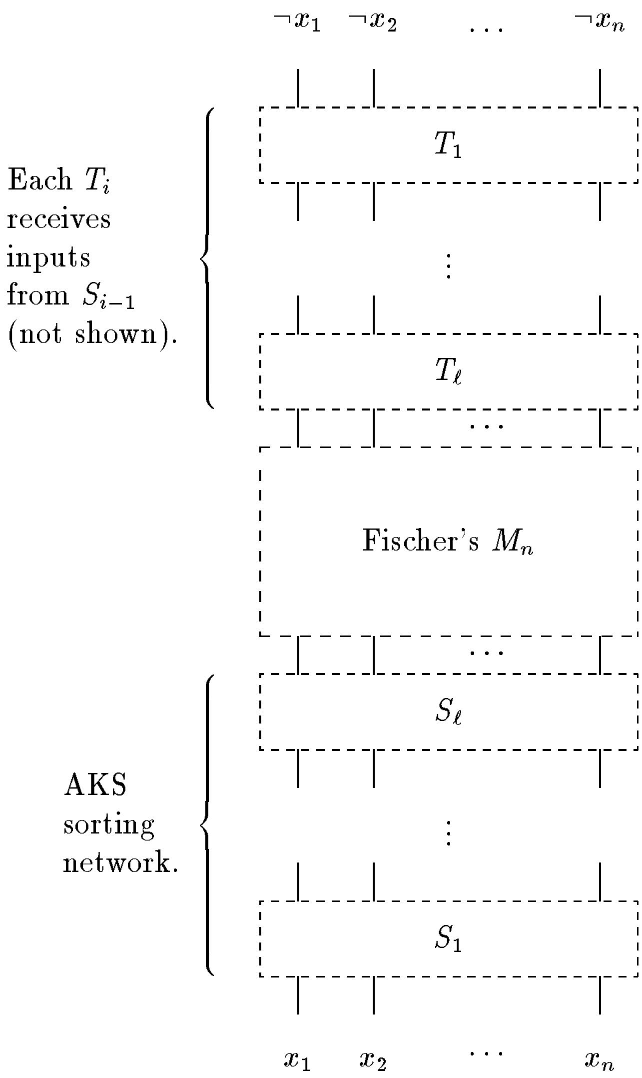


Figure 1: Overall structure of the circuit.

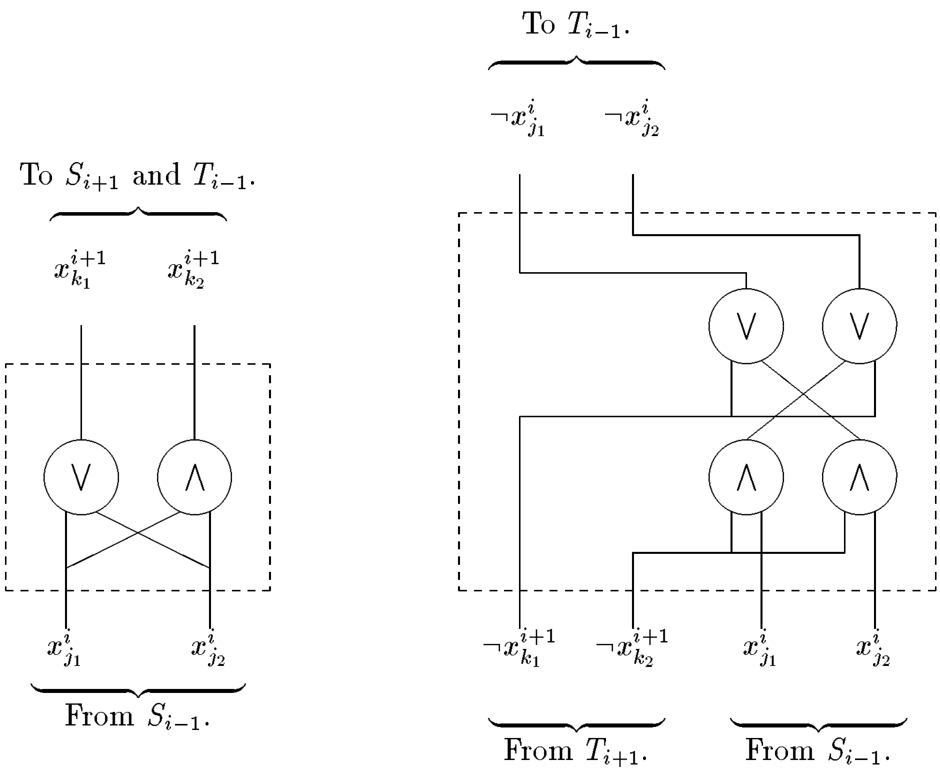


Figure 2: Corresponding circuit elements of $S_i$ (left) and $T_i$ (right).

[13] K. Tanaka and T. Nishino, *On the complexity of negation-limited boolean networks*, in Proceedings of the 26th ACM Symposium on the Theory of Computing, 1994, pp. 38–47.

[14] E. Tardos, *The gap between monotone and non-monotone circuit complexity is exponential*, Combinatorica, 8 (1988), pp. 141–142.

[15] L. G. Valiant, *Negation is powerless for boolean slice functions*, SIAM J. Comput., 15 (1986), pp. 531–535.

[16] I. Wegener, *On the complexity of slice functions*, Theoretical Comput. Sci., 38 (1985), pp. 55–68.

[17] ——, *The complexity of boolean functions*, John Wiley & Sons and B. G. Teubner, 1987.