A superlinear lower bound for undirected monotone contact networks computing T^n_{n-1}

Magnús M. Halldórsson[*] Jaikumar Radhakrishnan[†] K. V. Subrahmanyam[‡]

Abstract

We show an $\Omega(n \log\log\log n)$ lower bound on the size of undirected monotone contact networks computing the threshold function T^n_{n-1}. This improves the lower bound of $2(n-1)$ due to Markov. We show that there exists a Boolean function of n variables that can be computed in linear size using directed monotone contact networks but that needs $\Omega(n \log n)$ size on undirected monotone contact networks.

1 Introduction

Proving non-trivial lower bounds on the size of circuits computing Boolean functions is of fundamental importance. Although no superlinear lower bound on circuit size is known for any 'explicitly defined' Boolean function, substantial progress has been made [BS] in restricted models of computation. Threshold functions have played a central role in the study of lower bounds in circuit complexity. In this paper, we consider threshold function computation using the monotone contact networks model. We show an $\Omega(n \log\log\log n)$ lower bound on the size of undirected monotone contact networks computing the $(n-1)$-st threshold function. The best lower bound for this function known previously, due to Markov [M], was $2(n-1)$.

Definition 1 *Let n and k be positive integers such that $1 \leq k \leq n$. The k-th* threshold function *T^n_k is a Boolean function on n variables that takes the value 1 precisely when there are at least k 1's in the input.*

Definition 2 *A monotone contact network is a graph where each edge has a variable as its label. (In non-monotone networks, negated variables are also allowed to appear as labels.) We use $\mathrm{vars}(N)$ to denote the set of variables of the network N. For a pair (v, w) of vertices, the contact network computes the Boolean function $f_{v,w}$ as follows. On an assignment $y : \mathrm{vars}(N) \to \{0, 1\}$, each edge is set to 0 or 1 according to the value of its label. Then $f_{v,w}(y) = 1$ if there is a path from v to w using only the edges with value 1, and $f_{v,w}(y) = 0$ otherwise. If N is a contact network with two distinguished vertices s (start) and t (terminal), then we refer to the function $f_{s,t}$ as the function computed by N and denote it by f_N. The size of a network is the number of edges in it. The contact network complexity of f is the size of the smallest contact network computing f.*

[*] School of Information Science, Japan Advanced Institute of Science and Technology – Hokuriku, Tatsunokuchi, Ishikawa 923-12, JAPAN. magnus@jaist.ac.jp

[†] Theoretical Computer Science Group, Tata Institute of Fundamental Research, Bombay, INDIA 400 005. Work done while visiting JAIST. jaikumar@tcs.tifr.res.in

[‡] SPIC Science Foundation, 92 G N Chetty Road, Madras 600 017. kv@ssf.ernet.in

In this paper, we will be concerned primarily with the monotone model; hence we will often drop the word *monotone*, and refer to the networks as just contact networks. Contact networks may be classified based on the nature of the underlying graph. When the underlying graph is undirected, such networks are called *undirected* contact networks. On the other hand, if the underlying graph is directed (the function value is 1 precisely when there is a directed path from s to t, using only edges with label 1), such networks are called *directed* contact networks. Unlike in many other circuit models, the presence of constant 1's as labels adds to the power of directed contact networks. Such networks, where constant 1's are allowed as labels, are called *contact-rectifier* networks. In the contact-rectifier networks model, the edges labelled 1 are ignored while computing the size of the network. We use $\mathrm{size}_U(f)$, $\mathrm{size}_D(f)$ and $\mathrm{size}_R(f)$ to denote the respective complexities of f in the undirected, directed and contact-rectifier network models.

1.1 Previous work

Contact networks and contact-rectifier networks appear in the pioneering works of Shannon [H56]. They have been extensively studied by Russian scientists, who were the first to show lower bounds on the contact network complexity of Boolean functions [L, M, Kr]. Interest in these models was revived when complexity measures in these models were shown to be related to the space complexity of Boolean functions in the Turing machine model (see [W, page 414]).

The computation of threshold functions in the contact networks model has been well studied. The problem of computing threshold functions using monotone contact-rectifier networks was completely solved by Markov [M]; he showed that $\mathrm{size}_{\mathrm{R}}(T_k^n) = k(n-k+1)$ (see also Moore and Shannon [MS]).

For directed monotone contact networks, Radhakrishnan and Subrahmanyam [RS] showed that $\mathrm{size}_{\mathrm{D}}(T_k^n) = O(k(n-k+2)\log(n-k+2))$ for $2 \le k \le n-1$. Since directed contact networks are special contact-rectifier networks, Markov's $k(n-k+1)$ lower bound applies to this model also. A lower bound of $\lfloor k/2 \rfloor\, n \log(n/(k-1))$ on the size of any directed monotone network computing T_k^n, $2 \le k \le n/2$, was obtained by Radhakrishnan [R], improving Markov's bound for small thresholds.

The study of threshold function computations in the undirected monotone contact networks model was initiated by Lupanov [L], who showed that $\mathrm{size}_{\mathrm{U}}(T_2^n) = \Omega(n \log n/\log\log n)$. This lower bound was later improved by Hansel [H64] and Krichevskii [Kr] to $\Omega(n \log n)$. Since *formulas* are special undirected contact networks in which the underlying graph is a series-parallel graph, upper bounds for monotone formulas apply to undirected monotone contact networks as well. Using the amplification method, Boppana [B] showed the existence of monotone formulas of size $O(k^{4.33} n \log n)$ computing T_k^n. Applying the amplification method directly to monotone contact networks, Dubiner and Zwick [DZ] showed the existence of undirected monotone contact networks of size $O(k^{3.99} n \log n)$ computing T_k^n and T_{n-k}^n. Since undirected networks can be converted to directed networks by replacing each undirected edge by a pair of directed edges an $\Omega(kn \log(n/(k-1)))$ lower bound for computing T_k^n, $2 \le k \le n/2$ follows from the lower bound for directed contact networks cited above. Similarly, for $k > n/2$, we get an $\Omega(k(n-k+1))$ lower bound from Markov's result for contact-rectifier networks.

1.2 The results in this paper

In many circuit models related to monotone contact networks (e.g. monotone contact-rectifier networks, monotone formulas) the complexities of computing T_k^n and T_{n-k+1}^n are the same. However, the results for directed contact networks stated above show that this is not true for directed monotone contact networks. In particular, while there exists an $\Omega(n \log n)$ lower bound for T_2^n, the dual function T_{n-1}^n can be computed in linear size. Thus directed networks are more suited for computing large thresholds than small thresholds.

It is natural to ask if this anomaly exists even in the undirected monotone contact networks model, and, in particular, if linear size *undirected* monotone contact networks exist for T_{n-1}^n even though there is an $\Omega(n \log n)$ lower bound for T_2^n. Our main result shows that T_{n-1}^n does not have linear size undirected monotone contact networks.

Theorem 1 $\text{size}_U(T_{n-1}^n) = \Omega(n \log\log\log n)$.

Although this does not tell us if T_2^n and T_{n-1}^n have the same complexity in the undirected model, it does show that T_{n-1}^n is strictly harder to compute in the undirected model than in the directed model. Inevitably, we need to use properties of undirected graphs that set them apart from directed graphs. For example, our proof makes essential use of the following property that holds only for undirected graphs: *every (s,t)-path includes an odd number of edges from every minimal (s,t)-cutset.* To get our result we combine such graph theoretic properties with a notion of approximate threshold computation.

A greater separation between the powers of the undirected and directed models can be established for a somewhat less natural function. This result is an easy consequence of the $\Omega(n \log n)$ lower bound for T_2^n.

Theorem 2 Let the function F be defined by $F(x_1, x_2, \ldots, x_n, y) \equiv T_2^n(x_1, x_2, \ldots, x_n) \wedge y$. Then, $\text{size}_D(F) \leq 3n$ whereas $\text{size}_U(F) = \Omega(n \log n)$.

The rest of the paper is organized as follows. Section 2 describes the main contribution of this paper, the lower bound proof leading to Theorem 1 above. The proof of Theorem 2 is described in section 3. Finally, in section 4, we conclude with some open problems.

2 The $\Omega(n \log\log\log n)$ lower bound for T_{n-1}^n

2.1 Preliminaries

In this section, all networks considered will be undirected. We allow constant 1's to appear as labels in our networks. It is easy to see that edges labelled 1 can be contracted without altering the function computed by the network. However, for ease in presentation, we shall often let them remain, remembering not to consider them when estimating the size of the network.

We shall use the following graph theoretic notation. In a graph G, a path from vertex v to vertex w will be called a *(v,w)-path.* For a contact network N, a set $S \subseteq E(N)$ is called a (v,w) *edge cutset* if there is no (v,w)-path in the graph obtained from N by deleting the edges in S. Extending this notation, we shall refer to a set $C \subseteq \text{vars}(N)$ as a (v,w) *variable cutset* if the set consisting of all the edges whose labels appear in C forms a (v,w) edge cutset. When the nature of the cutset is clear from the context, we shall omit the prefixes *edge* and *variable*, and just refer to the set as a (v,w)-*cutset.* We say that a path P in N *avoids* a variable $x \in \text{vars}(N)$ if P contains no edge with label x.

2.2 Overview

For monotone contact networks, the function T_{n-1}^n is characterized by two conditions: first, it accepts all inputs with $(n-1)$ 1's; second, it rejects all inputs with $(n-2)$ 1's. It turns out that we may relax the second condition considerably without making the task too easy. For example, in the case of formulas, the following holds [Kr]:

> Every formula that accepts all inputs with $(n-1)$ 1's and rejects all inputs with $(n-k)$ or fewer 1's, for $k \geq 2$, has size at least $n \log(n/(k-1))$.

This motivates the following definition. Assume that $k \geq 2$.

- We call G an (n,k)-network if G has n variables, G accepts all inputs with at least $(n-1)$ 1's, and G rejects all inputs with at least k 0's.

For example, a network computing T^n_{n-1} is an $(n,2)$-network. The fact about formulas can now be stated as follows:

> Every (n,k)-formula (that is, every (n,k)-series-parallel network) such that $n \geq 2^\tau(k-1)$, has size at least $n\tau$.

We would like to prove a similar statement for contact networks in general. First, we need a definition.

- A network G is *τ-robust* if G is an (n,k)-network for some (n,k) such that $n \geq k^{2^\tau}$.

We shall show that every τ-robust network on n variables has size at least $cn \log \tau$ for some constant c (independent of n and τ). Since a network computing T^n_{n-1} is $\log\log n$-robust, this would imply that

$$\text{size}_U(T^n_{n-1}) = \Omega(n \log\log\log n). \tag{1}$$

For $x \in \text{vars}(G)$, let $\mu_G(x)$ be the number of occurrences of x in G. Let $\mu(G)$ be the maximum multiplicity of a variable in G. That is,

$$\mu(G) = \max_{x \in \text{vars}(G)} \mu_G(x).$$

Let

$$l(\tau) = \min_{G: G \text{ is } \tau\text{-robust}} \mu(G).$$

We shall show that

$$l(\tau) \geq \log_3(\tau). \tag{2}$$

Then (1) will follow from the following lemma, which is an application of an observation due to Krichevskii [Kr].

Lemma 3 *If G is a τ-robust network with n variables, then G has size at least $(l(\tau)-1)n$.*

Proof. Among the τ-robust networks on n variables of the smallest size, let G be the one with the fewest variables x satisfying the condition $\mu_G(x) \geq l(\tau)$. Then, we claim that $\mu_G(x) \geq l(\tau)-1$, for all $x \in \text{vars}(G)$. For, suppose the variable x_i appears less than $l(\tau)-1$ times, that is, $\mu_G(x_i) \leq l(\tau)-2$. By the definition of $l(\tau)$ there is a variable x_j that appears at least $l(\tau)$ times. Construct a network G' as follows. Replace all occurrences in G of x_j by constant 1. Subdivide every edge with label x_i and label one part with x_i and the other with x_j. Create a new start vertex s' and add two new edges connecting s' to $s(G)$, one with label x_i and the other with label x_j.

If f_G is an (n,k)-function, then it can be verified that $f_{G'}$ is also an (n,k)-function. It follows that G' is a τ-robust network on n variables and $\text{size}(G') \leq \text{size}(G)$. Also $\mu_{G'}(x_n) = \mu_G(x_1)+1 \leq l(\tau)-1$, and $\mu_{G'}(x_1) \leq l(\tau)-1$. But then G' has fewer variables than G appearing at least $l(\tau)$ times, contradicting the minimality of G. Hence all variables must appear at least $l(\tau)-1$ times. It follows that $\text{size}(G) \geq n(l(\tau)-1)$. ∎

Our proof of inequality (2) uses induction on τ. We show that every τ-robust network contains a $(\tau/3)$-robust subnetwork M. Moreover, this $(\tau/3)$-robust network is not optimal, that is, $\mu(M) \geq l(\tau/3) + 1$. It then follows that $l(\tau) \geq \log_3(\tau)$. To obtain the non-optimal network M from N, we need to perform certain operations on N. We now describe these operations informally without giving the details. We shall repeatedly make use of the following simple facts about (n,k)-networks.

Observation 4

1. *N is an (n,k)-network if and only if the following two conditions hold.*
    - *For every variable $x_i \in \mathrm{vars}(N)$, there is an (s,t)-path in N avoiding x_i.*
    - *Every subset of $\mathrm{vars}(N)$ with k variables is an (s,t)-cutset.*
2. *If N is an (n,k)-network and $S \subseteq \mathrm{vars}(N)$ then the network obtained from N by setting all the variables in S to 1 is an $(n-|S|,k)$-network with variable set $\mathrm{vars}(N) - S$.*

In a network computing T^n_{n-1} no (s,t)-path can avoid both x_1 and x_2. Since we are dealing with (n,k)-networks, this useful property is not available to us automatically. However, with some reduction in the number of variables this property can be acquired. We shall show how a networks N_1 (say) can be obtained from N, such that, for all $x_j \in \mathrm{vars}(N_1) - \{x_1\}$, no (s,t)-path in N_1 can avoid both x_1 and x_j. If x_1 satisfies this condition then it is said to be *critical.*

Next, we obtain a network N_2 in which the variable x_1 appears only once. The reason this network is useful is as follows. Suppose the edge on which x_1 appears is (A,B). Since every path avoiding any other variable contains x_1, such paths must use the edge (A,B). Assume that they all use the edge in the direction, A to B. We focus on the subnetwork between s and A and the subnetwork between B and t. Now, if there are short paths (containing few variables) from s to A and from B to t, then there would be a short path from s to t. But this would contradict the robustness property of N_2. Hence, one of the two subnetworks must be reasonably robust. Also, as discussed above, for every variable they contain a path avoiding that variable. So one of these two networks will give us the desired subnetwork M. Then, it only needs to be shown that this subnetwork is non-optimal. For this we observe that in the network N_2 there is a path avoiding x_1. Now since x_1 is critical this path must include every other variable. It turns out that these occurrences are not essential for the computation of our subnetwork M. This implies that the network M is not optimal. The details for this part are discussed in section 2.4, when the 'special paths' are used. This is the only place where we use the fact that our network is undirected.

The details for the various steps are presented below. We shall use these steps when we prove the main lemma (Lemma 11, section 2.4). However, we suggest that the reader go directly to section 2.4 and obtain an overview of the argument presented there before verifying the details for the intermediate steps.

2.3 The intermediate steps

Making x_1 critical:

Lemma 5 *Let G be an (n, k)-network. Then, there is an (m, k)-network H, with $m \geq n/(k-1)$, such that*

(a) *$\mu(G) \geq \mu(H)$; and*

(b) *$x_1 \in \mathrm{vars}(H)$, and for every variable $x_j \in \mathrm{vars}(H) - \{x_1\}$, no (s,t)-path avoids both x_1 and x_j.*

Proof. We use induction on k. For $(k = 2)$, we have that G computes T^n_{n-1}. If x_1 is not one of the variables of G then rename some variable as x_1. Note that this network now has property (b). Hence we take H to be the network G itself with some variable renamed as x_1 if necessary.

For the induction step, assume that the lemma holds for $k = r \geq 2$; we shall show that it holds for $k = r + 1$. As before, if x_1 is not one of the variables of G, rename some variable of G as x_1. Let

$$S_1 = \{x_j \in \mathrm{vars}(G) : \text{ every } (s,t)\text{-path avoiding } x_j \text{ contains } x_1\}.$$

Let $m_1 = |S_1 \cup \{x_1\}|$. Let H_1 be the network obtained from G by fixing all variables outside $S_1 \cup \{x_1\}$ at 1. Clearly, H_1 has properties (a) and (b), and H_1 is an $(m_1, r + 1)$-network.

Let G' be the network obtained from G by fixing x_1 at 0 (deleting all edges with label x_1) and all variables in S_1 at 1. Clearly, $\mu(G') \leq \mu(G)$. We claim that G', with $\mathrm{vars}(G') = \mathrm{vars}(G) - (S_1 \cup \{x_1\})$, is an $(n - m_1, r)$-network. To justify this, we use Observation 4. First, we show that for every variable $x_j \in \mathrm{vars}(G')$, there is an (s,t)-path in G' avoiding x_j. Since G is an $(n, r + 1)$-network, there is an (s,t)-path in G avoiding x_j. Since $x_j \in \mathrm{vars}(G')$, $x_j \notin S_1$. By the definition of S_1, G has an (s,t)-path avoiding both x_1 and x_j. In G', this path may have some labels set on it to 1, but none to 0 (because it avoids x_1). Thus, this is an (s,t)-path in G' avoiding x_j. Next, we verify that every subset of $\mathrm{vars}(G')$ with r variables is an (s,t)-cutset of G'. Let C be a subset of $\mathrm{vars}(G')$ with r variables. Since G is an $(n, r+1)$-network, $C \cup \{x_1\}$ is an (s,t)-cutset in G. Hence, C is an (s,t)-cutset in G'. This completes the justification of our claim that G' is an $(n - m_1, r)$-network.

Since the statement holds when $k = r$, we get, by applying the induction hypothesis to the network G', an (m_2, r)-network H_2, such that

(a) $\mu(G') \geq \mu(H_2)$; and

(b) $x_1 \in \mathrm{vars}(H_2)$, and for every variable $x \in \mathrm{vars}(H_2) - \{x_1\}$, no (s,t)-path avoids both x_1 and x.

Further $m_2 \geq (n - m_1)/(r - 1)$. It follows that $m_2(r-1) + m_1 \geq n$. That is, $r \max\{m_2, m_1\} \geq n$ or $\max\{m_2, m_1\} \geq n/r$. Hence, one of H_1 and H_2 meets the requirements for H in the statement of the lemma. ∎

To just one x_1: Next we obtain a network in which x_1 appears only once.

Lemma 6 *If G is an (n, k)-network where $\{x_1, x_i\}$ is an (s,t)-cutset for every $x_i \in \mathrm{vars}(G) - \{x_1\}$, then there is an (m, k)-network H with $m \geq \frac{n-1}{2\mu(G)} + 1$, such that*

(a) *there is only one edge (A, B) with label x_1; and*

(b) *for each $x_j \in \mathrm{vars}(H) - \{x_1\}$, every (s,t)-path avoiding x_j has the form $p \cdot (A, B) \cdot q$.*

Proof. First we obtain a network H' that has property (a). H' will be an (n',k)-network with $n' \geq \frac{n-1}{\mu(G)} + 1$.

Let the edges with label x_1 in G be $e_1, e_2, \ldots, e_r$, where $r = \mu_G(x_1) \leq \mu(G)$. For $x_i \in \mathrm{vars}(G) - \{x_1\}$, let $g(x_i)$ be the smallest index j such that there is an (s,t)-path avoiding x_i that contains none of $e_{j+1}, \ldots, e_r$. (Every such path must use e_j.) That is, for $x_i \in \mathrm{vars}(G) - \{x_1\}$,

$$g(x_i) = \min\{j : \text{there exists an } (s,t)\text{-path avoiding } x_i \text{ and } e_{j+1}, \ldots, e_r\}.$$

For every variable $x_i \in \mathrm{vars}(G) - \{x_1\}$, there exists an (s,t)-path in G that avoids x_i. Because $\{x_1, x_i\}$ is a cutset, each such path must contain at least one of $e_1, \ldots, e_r$. Hence $g(x_i)$ is well defined for $x_i \in \mathrm{vars}(G) - \{x_1\}$. For $j = 1, 2, \ldots, r$, let

$$V_j = \{x_i \in \mathrm{vars}(G) - \{x_1\} : g(x_i) = j\}.$$

Since $\bigcup_{j=1}^{r} |V_j| = n-1$, there is a j such that $|V_j| \geq (n-1)/r$. For this j, let H' be the network obtained from G by setting all variables NOT in $\{x_1\} \cup V_j$ to 1, the edges $e_1, \ldots, e_{j-1}$ to 1 and deleting the edges $e_{j+1}, \ldots, e_r$. Clearly, the edge e_j $(= (A,B)$, say) is the only edge with label x_1. We now verify, using Observation 4, that H' is an $(|V_j|+1, k)$-network. First, we show that for every $x_i \in \mathrm{vars}(H')$ there is an (s,t)-path in H' avoiding x_i. There is such a path in G that does not use $e_{j+1}, \ldots, e_r$. Hence, there is an (s,t)-path in H' avoiding x_i. (Note that the (s,t)-path in G avoiding x_1 does not use any of $e_1, e_2, \ldots, e_r$, and therefore continues to serve in H'.) By our definition of V_j, for every $x_i \in V_j$, the path in H' avoiding x_j must use the edge e_j. Hence, $\{x_1, x_i\}$ is an (s,t)-cutset for all $x_i \in V_j$. Every subset S of V_j with k variables is an (s,t)-cutset of G. Since no occurrence of any variable in S has been replaced by 1, S is an (s,t)-cutset for H' also. It follows that every subset of $\mathrm{vars}(H')$ with k variables is an (s,t)-cutset in H'. Thus, we have verified that H' is an $(|V_j|+1, k)$-network.

To obtain the network H satisfying condition (b) we observe that for an $x_j \in \mathrm{vars}(H') - \{x_1\}$, all (s,t)-paths in H' avoiding x_j have either the form $p \cdot (A,B) \cdot q$ or the form $p \cdot (B,A) \cdot q$, but not both (for otherwise we would get a path avoiding both x_1 and x_j). For at least half the variables in $\mathrm{vars}(H') - \{x_1\}$, the path must have the same form. By fixing the remaining variables in $\mathrm{vars}(H') - \{x_1\}$ at the value 1 (invoking Part 2 of Observation 4), and suitably renaming A and B, we obtain an (m,k)-network with properties (a) and (b) such that $m \geq \frac{n-1}{2\mu(G)} + 1$. ∎

Getting special paths: We need the following definitions. We call a network N *optimal* if

$$\mu(N) = \min_{H: H \text{ computes } f_N} \mu(H).$$

For a network N and a variable $x \in \mathrm{vars}(N)$, we say that N is *x-optimal* if through every edge e with label x there is an (s,t)-path on which the label x appears only at e. First, we make the following easy observations.

Observation 7

1. *If N is not x-optimal then there is an edge in N whose label x can be replaced by* 1 *without changing f_N.*

2. *If N is not x-optimal for some variable $x \in \mathrm{vars}(N)$ and we replace some occurrence of some other variable by* 1*, then the new network is also not x-optimal.*

Lemma 8 *If N is optimal then N is x-optimal for some $x \in \mathrm{vars}(N)$.*

Proof. Suppose N is not x-optimal for any $x \in \mathrm{vars}(N)$. By repeatedly using Observation 7, we can replace one occurrence of each label $x \in \mathrm{vars}(N)$ by 1, without changing f_N. But then, the final network obtained would contradict the optimality of N. ∎

We use the following fact that relates (s,t)-paths and (s,t)-cutsets in undirected graphs.

Observation 9 *In an undirected graph, every (s,t)-path must include an* odd *number of edges from every minimal (s,t)-cutset.*

Lemma 10 *Suppose G is an (n,k)-network such that*

(a) $x_1 \in \mathrm{vars}(G)$ *appears only once;*

(b) *for the variable $x_j \in \mathrm{vars}(G) - \{x_1\}$, G is x_j-optimal, and every path avoiding x_j contains the x_1.*

Then there is an (s,t)-path in G avoiding x_1 and containing only one edge labelled x_j.

Proof. Property (b) implies that, the edges with labels x_1 and x_j form an (s,t)-cutset. Let C_j be a minimal such cutset. Note that C_j must include an edge e with label x_j and the lone edge with label x_1. Since G is x_j-optimal, there is an (s,t)-path on which x_j appears only at e. This path cannot contain x_1, for then it would contain exactly *two* edges from the minimal cutset C_j, contradicting Observation 9. ∎

2.4 The main lemma and the theorem

Lemma 11 $l(\tau) \geq \lfloor \log_3(\tau) \rfloor$.

Proof. Since $l(\tau)$ is a non-decreasing function of τ, it is enough to verify the claim for all τ's that are powers of 3. For $\tau = 1$, the claim is trivial. Hence, assume that τ (≥ 3) is a power of 3 and the claim holds for all smaller powers of 3. Let G be τ-robust, that is, G is an (n,k)-network with $n \geq k^{2^\tau}$. We shall show that $\mu(G) \geq \log_3(\tau)$. It will follow that $l(\tau) \geq \lfloor \log_3(\tau) \rfloor$.

Making x_1 critical: Using Lemma 5 we get an (n',k)-network G', with

$$n' \geq n/(k-1), \tag{3}$$

such that

(1a) $\mu(G) \geq \mu(G')$; and

(1b) $x_1 \in \mathrm{vars}(G')$, and for every variable $x_j \in \mathrm{vars}(G') - \{x_1\}$, no (s,t)-path avoids both x_1 and x_j.

To just one x_1: From G', using Lemma 6, we get an (m,k)-network H, with

$$m \geq \frac{n'-1}{2\mu(G')} + 1 \geq \frac{n-k+1}{2\mu(G)(k-1)} + 1, \tag{4}$$

such that

(2a) there is only one edge (A,B) with label x_1; and

(2b) for every $x_j \in \mathrm{vars}(H) - \{x_1\}$, every (s,t)-path avoiding x_j has the form $p \cdot (A,B) \cdot q$.

We assume that H is a minimal such network, that is, we cannot replace any variable label by 1 and retain these properties. Then Part 1 of Observation 7 implies that H has the following property.

(2c) H is x_j-optimal for every $x_j \in \mathrm{vars}(H)$.

Since H is an (m,k)-network, properties (2a) and (2b) imply that the following holds for H.

(2d) For $x_j \in \mathrm{vars}(H) - \{x_1\}$, there is an (s,A)-path p_j (and a (B,t)-path q_j) avoiding both x_1 and x_j.

Using special paths: Let H_1 be the network obtained from H by deleting the lone edge (A,B) with label x_1, setting $s(H_1) = s(H)$ and setting $t(H_1) = A$. Let H_2 be the network obtained from H by deleting (A,B), setting $s(H_2) = B$ and setting $t(H_2) = t(H)$.

Claim: H_1 and H_2 are not optimal.

Proof of claim. We shall only prove that H_1 is not optimal. That H_2 also is not optimal can be obtained by symmetry. Suppose H_1 is optimal. Using Lemma 8, we conclude that for some variable $x_i \in \mathrm{vars}(H_1)$, H_1 is x_i-optimal. Since H has properties (2a), (2b) and (2c), we get from Lemma 10 that there is an edge e and an (s,t)-path in H avoiding x_1 on which x_i appears only at e. Let the path be $P \cdot (a,b) \cdot Q$, where $e = (a,b)$. Since H_1 is x_i-optimal, there is an (s,A)-path in H_1 on which x_i appears only at e. Suppose this path is of the form $p \cdot (a,b) \cdot q$. Using property (2d) we get that $p_i \cup q \cup Q$ contains an (s,t)-path in H avoiding both x_1 and x_i. This contradicts (2b). On the other hand if the path has the form $p \cdot (b,a) \cdot q$, then $p \cup Q$ contains such a path, again contradicting (2b). (End of Claim.)

The induction: Assume that

$$\mu(G) < \log_3(\tau). \tag{5}$$

We shall show that either H_1 or H_2 contains a $(\tau/3)$-robust network. By the claim above, these networks are not optimal; therefore, we have $\mu(G) \geq \log_3(\tau/3) + 1 = \log_3(\tau)$, contradicting (5).

Suppose no (s,A)-path in H_1 avoids $k^{2^{\tau/3}}$ variables from $\mathrm{vars}(H_1)$. Then every subset of $\mathrm{vars}(H_1)$ of size $k^{2^{\tau/3}}$ is an (s,A)-cutset in H_1. Also, (2d) implies that for every $x_j \in \mathrm{vars}(H_1)$ there is an (s,A)-path in H_1 avoiding x_j. It follows from Observation 4 that H_1 is an $(m-1, k^{2^{\tau/3}})$-network on the variables $\mathrm{vars}(H_1) = \mathrm{vars}(H) - \{x_1\}$, where (4) gives

$$m - 1 \geq \frac{n-k+1}{2\mu(G)(k-1)}.$$

Now,

$$\left(k^{2^{\tau/3}}\right)^{2^{\tau/3}} \leq \frac{n-k+1}{2\log_3(\tau)(k-1)},$$

since $k \geq 2$, $\tau \geq 3$ and $n \geq k^{2^\tau}$. Hence H_1 is $(\tau/3)$-robust. By the claim above, H_1 is not optimal. Hence, using the induction hypothesis, we have

$$\mu(G) \geq \mu(H) \geq \mu(H_1) \geq l(\tau/3) + 1 \geq \log_3(\tau),$$

contradicting (5).

On the other hand, suppose some (s,A)-path P in H_1 avoids $m' \geq k^{2^{\tau/3}}$ variables from $\mathrm{vars}(H_1)$. Let V be the set of these variables. In this case, we consider the network H_2. Let $\hat{H}$

be the optimal network computing f_{H_2}. By the claim above, H_2 is not optimal. Therefore, we have

$$\mu(H_2) \geq \mu(\hat{H}) + 1. \tag{6}$$

Consider the network H' with $\text{vars}(H') = V$ obtained from $\hat{H}$ by setting all variables not in V to 1. We now show that H' is an (m', k)-network. Let S be a subset of V of with k variables. Since H is an (m, k)-network, S is an (s, t)-cutset in H. Since S does not contain any variables in P and $x_1 \notin S$, S must be a (B, t)-cutset in H. It follows that S is also a (B, t)-cutset in H_2 and hence an $(s(\hat{H}), t(\hat{H}))$-cutset in $\hat{H}$. Since H' is obtained from $\hat{H}$ by setting some variables outside S to 1, S is a $(s(H'), t(H'))$-cutset. Also, (2d) implies that for every variable $x_j \in V$, there is a (B, t)-path in H_2 avoiding x_j, and therefore an $(s(\hat{H}), t(\hat{H}))$-path in $\hat{H}$ avoiding x_j. Since H' is obtained from $\hat{H}$ by setting some variables to 1, there must exist an $(s(H'), t(H'))$-path in H' avoiding x_j. Thus, using Observation 4, it follows that H' is an (m', k)-network. Since $m' \geq k^{2^{\tau/3}}$, H' is $(\tau/3)$-robust. Then, using (6), we have

$$\begin{aligned} \mu(G) \ &\geq \ \mu(H) \ \geq \ \mu(H_2) \ \geq \ \mu(\hat{H}) + 1 \\ &\geq \ \mu(H') + 1 \ \geq \ l(\tau/3) + 1 \ \geq \ \log_3(\tau), \end{aligned}$$

again contradicting (5).

Thus (5) cannot hold and the induction is complete. ∎

Theorem 1 *$size_U(T^n_{n-1}) = \Omega(n \log\log\log n)$.*

Proof. The theorem is a direct consequence of Lemmas 3 and 11, and the fact that any network computing T^n_{n-1} is $(\log\log n)$-robust. ∎

3 Undirected vs. directed monotone contact networks

Theorem 2 *Let the function F be defined by $F(x_1, x_2, \ldots, x_n, y) \equiv T^n_2(x_1, x_2, \ldots, x_n) \wedge y$. Then, $size_D(F) \leq 3n$ whereas $size_U(F) = \Omega(n \log n)$.*

Proof. The network shown in Figure 1 shows that $\text{size}_D(F) \leq 3(n-1)$.

Figure 1: A network computing $F(x_1, x_2, \ldots, x_n, y)$

To show that $\text{size}_U(F) = \Omega(n \log n)$, let G be an undirected network that computes F. Let G' be the network obtained from G by fixing the variable y at 1. Since G is an undirected network, this amounts to contracting all edges labelled y. Now G' computes $T^n_2(x_1, x_2, \ldots, x_n)$, and using Krichevskii's lower bound [Kr], we have $\text{size}(G) \geq \text{size}(G') = \Omega(n \log n)$.

∎

4 Concluding remarks

We have obtained a lower bound of $\Omega(n \log\log\log n)$ on $\text{size}_U(T^n_{n-1})$, whereas the best upper bound known is $O(n \log n)$. Can this gap be reduced? For undirected networks, our lower bound improves Markov's $k(n-k+1)$ lower bound for computing T^n_k, if k is very large. It would be interesting to generalize this result, and obtain a lower bound of $\Omega(kn \log\log\log n)$ for T^n_{n-k+1}, or at least a bound of the form $\Omega(f(k) n \log\log\log n)$, where $f(k)$ goes to infinity with k.

We have shown that there exists a function on n variables that can be computed in linear size using directed monotone contact networks, but needs size $\Omega(n \log n)$ on undirected monotone contact networks. It appears, however, that a far greater separation exists. We restate a question of Stockmeyer (see Grigini and Sipser [GS, page 73]) in the language of contact networks.

Question: Is there a polynomial size monotone contact networks for computing the directed (s,t)-connectivity function?

Acknowledgment

Shiva Chaudhuri, Vivek Gore, and Rakesh Sinha read the earlier versions of this paper carefully, pointed out several errors and suggested many improvements. We are grateful to them for their contributions.

References

[B] R. Boppana. *Amplification of Probabilistic Boolean Formulas.* Advances in Computing Research Vol. 5 (S. Micali ed., JAI Press, Greenwich, CT), 1989, pp. 27–45.

[BS] R. B. Boppana and M. Sipser. *The Complexity of Finite Functions.* Chapter 14, The Handbook of Theoretical Computer Science, (J. van Leeuwen, ed.), Elsevier Science Publishers B. V., 1990, pp. 759–804.

[DZ] M. Dubiner and U. Zwick. *Amplification and Percolation.* Proc. of the 33rd IEEE FOCS, 1992, pp. 258–267.

[GS] M. Grigini and M. Sipser. *Monotone Complexity.* Boolean Function Complexity. LMS Lecture Note Series 169, Cambridge University Press, 1992, pp. 55–75.

[H64] G. Hansel. *Nombre minimal de contacts de fermature nécessaires pour réaliser une fonction booléenne symétrique de n variables.* C. R. Acad. Sci. Paris 258 (1964), pp. 6037–6040.

[H56] M. A. Harrison. *Introduction to Switching and Automata Theory.* McGraw-Hill Series in Systems Science, 1956.

[Kr] R. E. Krichevskii. *Complexity of contact circuits realizing a function of logical algebra.* Sov. Phys. Dokl. 8 (1964) pp. 770–772.

[L] O. B. Lupanov. *On comparing the complexity of realization of monotone contact networks containing only closing contacts and by arbitrary contact networks.* Sov. Phys. Dokl. 7 (1962) pp. 486–489.

[M] A. A. Markov. *On minimal switching-and-rectifier networks for monotone symmetric functions (in Russian).* In Problems of Cybernetics, vol. 8, pp. 117–121, Nauka, 1962.

[MS] E. F. Moore and C. E. Shannon. *Reliable circuits using less reliable relays.* J. Franklin Inst. 262, 1956, pp. 191-208 and pp. 281–297.

[R] J. Radhakrishnan. *Better Bounds for Threshold Formulas.* Proc. of the 32nd IEEE FOCS, 1991, pp. 314–323.

[RS] J. Radhakrishnan and K. V. Subrahmanyam. *Directed Monotone Contact Networks for Threshold Functions.* Information Proc. Letters 50 (1994) 199–203.

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