Bounds on monotone switching networks for directed connectivity

Aaron Potechin
Mathematics department
MIT
Cambridge, MA
aaron@potechin.org

***Abstract*—We prove that any monotone switching network solving directed connectivity on N vertices must have size $N^{\Omega(\log N)}$**

***Keywords*-L; NL; computational complexity; switching networks;**

I. Introduction

L versus NL, the problem of whether non-determinism helps in logarithmic space bounded computation, is a longstanding open question in computational complexity. At present, only a few results are known. It is known that the problem is equivalent to the question of whether there is a log-space algorithm for the *directed connectivity* problem, namely given an N vertex directed graph G and pair of vertices s, t, find out if there is a directed path from s to t in G. In 1970, Savitch [9] gave an $O(\log^2 N)$-space deterministic algorithm for directed connectivity, thus proving that $NSPACE(g(n)) \subseteq DSPACE((g(n)^2))$ for every space constructable function g. In 1987 and 1988, Immerman [3] and Szelepcsenyi [10] independently gave an $O(\log N)$-space non-deterministic algorithm for directed *non-connectivity*, thus proving that $NL = co\text{-}NL$. For the problem of *undirected connectivity* (i.e. where the input graph G is undirected), a probabalistic algorithm was shown in 1979 using random walks by Aleliunas, Karp, Lipton, Lovász, and Rackoff [1], and in 2005, Reingold [8] gave a deterministic $O(\log N)$-space algorithm for the same problem, showing that undirected connectivity is in L. Trifonov [11] independently gave an $O(\lg N \lg\lg N)$ algorithm for undirected connectiivty.

In terms of monotone computation, in 1988 Karchmer and Wigderson [4] showed that any monotone circuit solving directed connectivity must have superlogarithmic depth, showing that monotone-$NC^1 \subsetneq$ monotone-L. In 1997 Raz and McKenzie [6] proved that monotone-$NC \neq$ monotone-P and for any i, monotone-$NC^i \neq$ monotone-NC^{i+1}.

So far, most of the work trying to show that $L \neq NL$ has been done using branching programs or the JAG model, introduced in [5] and [2] respectively. Instead, we explore trying to prove $L \neq NL$ using the switching network model, described in [7]. This model can be applied to any problem, and a general definition is given in Section II. In this paper, we focus on switching networks solving directed connectivity.

The best way to describe what such a switching network is through an example, see Figure I and the accompanying explanation. A formal specialized definition is given below:

Definition 1.1: A switching network solving directed connectivity on a set $V(G)$ of vertices with distinguished vertices s, t is a tuple $< G', s', t', \mu' >$ where G' is an undirected multi-graph with distinguished vertices s',t' and μ' is a labeling function such that:
1. Each edge $e' \in E(G')$ has a label of the form $v_1 \to v_2$ or $\neg(v_1 \to v_2)$ for some vertices $v_1, v_2 \in V(G)$.
2. Given a directed graph G with vertex set $V(G)$, there is a path in G' from s' to t' such that all of the labels are consistent with G, i.e. of the form e for some edge $e \in E(G)$ or $\neg e$ for some $e \notin E(G)$, if and only if there is a path from s to t in G.
We say that such a switching network solves directed connectivity on N vertices, where $N = |V(G)|$, and we take its size to be $|V(G')|$. A switching network solving directed connectivity is monotone if it has no labels of the form $\neg(v_1 \to v_2)$.

Notation: In this paper, we use lower case letters (i.e. a, e, f) to denote vertices, edges, and functions, and we use upper case letters (i.e. G, V, E) to denote graphs and sets of vertices and edges. We use unprimed symbols to denote vertices, edges, etc. in

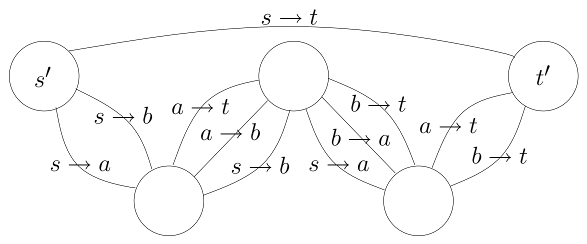

Figure 1. A switching network solving directed connectivity is an undirected multi-graph G' that takes a directed graph G and tells us if there is a path from s to t in G as follows: If an edge in G' has a label $a \to b$ for some vertices a and b in G, then we can take it if and only if the edge $a \to b$ is in G. Similarly, if an edge in G' has a label $\neg(a \to b)$, we can take it if and only if the edge $a \to b$ is not in G. Under these conditions, there is a path from s' to t' in G' if and only if there is a path from s to t in G.
In this figure, we have a switching network that solves directed connectivity when G has four vertices, s, t, a, and b. As needed, there is a path from s' to t' in G' if and only if there is a path from s to t in G. For example, if we have the edges $s \to a$, $a \to b$, and $b \to t$ in G, so there is a path from s to t in G, then in G', starting from s', we can take the edge labeled $s \to a$, then the edge labeled $a \to b$, then the edge labeled $s \to a$, and finally the edge labeled $b \to t$, and we will reach t'. If in G we have the edges $s \to a$, $a \to b$, $b \to a$, and $s \to b$ and no other edges, so there is no path from s to t, then in G' there is no edge that we can take to t', so there is no path from s' to t'.

the directed graph G, and we use primed symbols to denote vertices, edges, etc. in the switching network G'.

A. Our results

We have the following theorem:

Theorem 1.2: Any monotone swtiching network solving directed connectivity on N vertices has size at least $N^{\Omega(\log N)}$

In this paper, we prove only a superpolynomial lower bound. For a proof of an $N^{\Omega(\log N)}$ lower bound, see the full paper.

B. Proof overview

We now give a high level informal overview of the proof, ignoring details and subtleties.

The main idea involved in proving lower size bounds for monotone switching networks solving directed connectivity is as follows. Since G' solves directed connectivity, for every path P in G from s to t, there is a path P' in G' from s' to t' that uses only the edges of P. We show that this P' must include a vertex a'_P that gives significant information about P, i.e. there cannot be too many paths $P_1, P_2, \cdots$ in G such that each pair of paths P_i, P_j has very few vertices in common and all of these share the same vertex a' in G'. Then if we can find a large collection of paths such that each pair of paths has very few vertices in common, this will give a good lower bound on the number of vertices in G'.

In Section III, we apply this approach to prove that Theorem 1.2 is true for a restricted case of monotone switching networks solving directed connectivity, which we call certain-knowledge switching networks. Lemma 3.9 shows if we have a path P' from s' to t' in G' that uses only the edges of some path P from s to t in G, then we can pick a vertex a'_P in P' which tells us at least $\log k$ of the vertices in P, where k is the length of P. Thus, if two paths P_1 and P_2 of length k in G have less than $\log k$ vertices in common, then a'_{P_1} cannot be the same as a'_{P_2}, so each such path gives a distinct vertex in G'. It is not hard to find a large collection of paths from s to t in G of length k such that each pair of paths has less than $\log k$ vertices in common, and this completes the proof.

However, the way that a'_P gives information about P in certain-knowledge switching networks is somewhat artificial and cannot be extended to general monotone switching networks solving directed connectivity. In Section V, we introduce a fourier transformation technique and assign each vertex a' in G' a function $J_{a'} : \mathcal{C} \to \mathbb{R}$, where $\mathcal{C}$ is the set of all possible cuts of G.

In Section VI, we prove Theorem 6.1, showing that for each directed path P in G we can find a function $g_P : \mathcal{C} \to \mathbb{R}$ such that if P' is a path from s' to t' using only the edges of P, then $\sum_{a' \in V(P')} |J_{a'} \cdot g_P|$ is relatively large. Moreover, if P_1 and P_2 have very few vertices in common, then g_{P_1} and g_{P_2} are orthogonal. In this way, the vertices of P' give significant information about P. As shown in Theorem 5.14, this is sufficient to show a superpolynomial lower size bound.

II. Switching networks and L versus NL

We give the general definition of switching networks below.

Definition 2.1: A switching network is a tuple $< G', s', t', \mu' >$ where G' is an undirected graph with distinguished vertices s', t' and μ' is a labeling function that associates with each edge $e' \in E(G')$ a label $\mu(e')$ of the form $x_i = 1$ or $x_i = 0$ for some i between 1 and n. We say that this network computes the function $f : \{0,1\}^n \to 0,1$, where $f(x) = 1$ if and only if there is a path from s' to t' such that each edge of this path has a label that is consistent with x.

We take the size of a switching network to be

$|V(G')|$. A switching network is monotone if it has no labels of the form $x_i = 0$.

The following theorem is not new, see e.g. [7], but it is related to our results, so we include a proof sketch.

Theorem 2.2: If $f \in DSPACE(g(n))$ where $g(n)$ is at least logarithmic in n, then there is a switching network of size $2^{O(g(n))}$ computing f.

Proof Sketch: Start by modeling the Turing machine computing f with a branching program G in the natural way. Now note that for a given input x, since the Turing machine is deterministic, each vertex has at most one edge going out from it. This means that G has the structure of a forest where the root of each tree is either t, a vertex corresponding to a rejecting configuration, or a directed cycle. But then whether or not there is a path from s to t is unaffected by making all of the edges undirected. Thus, we can obtain a switching network that computes f simply by making all of the edges of this branching program undirected. The result follows immediately. ■

Corollary 2.3: If there is no switching network of polynomial size solving directed connectivity, then $L \neq NL$.

III. Certain-knowledge switching networks

In this section, we consider a subclass of monotone switching networks solving directed connectivity, which we call certain-knowledge switching networks, where we can assign each vertex $a' \in V(G')$ a simple knowledge set and there are simple reversible rules for moving from one knowledge set to another. We make the following definitions:

Definition 3.1: A knowledge set K is a directed graph with $V(K) = V(G)$, and we represent K by the set of its edges.

Given a knowledge set K, we can form a knowledge set $\bar{K}$ as follows:

If there is no path from s to t in K, then $\bar{K} = \{v_1 \to v_2$: there is a path from v_1 to v_2 in $K\}$.

If there is a path from s to t in K, then $\bar{K}$ is the complete directed graph on $V(G)$.

Call $\bar{K}$ the transitive closure of K.

Each transitive closure represents an equivalence class of knowledge sets. We say $K_1 = K_2$ if $\bar{K_1} = \bar{K_2}$ and we say $K_1 \subseteq K_2$ if $\bar{K_1} \subseteq \bar{K_2}$.

Remark 3.2: We will try to label each vertex a' in the switching network with a knowledge set $K_{a'}$ so that if $v_1 \to v_2 \in K_{a'}$ and there is a path from s' to a' in G' using only edges in G, then we know that either there is a path from v_1 to v_2 in G or there is a path from s to t in G, in which case we do not care which other paths are in G. In this way, $K_{a'}$ represents our knowledge about G when we are at the vertex a'.

If there is no path from s to t in K, then K and $\bar{K}$ represent exactly the same knowledge about G, so they are equivalent. If $K_{a'}$ contains a path from s to t, then if there is a path from s' to a' in G' using only edges in G, we know there is a path from s to t in G. Thus, we may as well merge a' and t'. To do this, we make all knowledge sets with a path from s to t equivalent by giving them the same transitive closure, the complete directed graph on $V(G)$.

Remark 3.3: The statement $K_1 \subseteq K_2$ does not imply that every edge in K_1 is in K_2. For example, if $K_1 = \{s \to a, s \to b\}$ and $K_2 = \{s \to a, a \to b\}$, then $\bar{K_1} = K_1$ and $\bar{K_2} = \{s \to a, a \to b, s \to b\}$, so $\bar{K_1} \subseteq \bar{K_2}$ and thus $K_1 \subseteq K_2$. It is best to think of the statement $K_1 \subseteq K_2$ as saying that the knowledge K_1 respresents is included in the knowledge K_2 respresents.

We would like to label each vertex in the switching network with a knowledge set. In order for these labels to be meaningful, we must know that for any $a', b' \in V(G')$ and $v_1, v_2 \in V(G)$, if we are at a' and use an edge with label $v_1 \to v_2$ to reach vertex b', then knowing there is a path from $v_1 \to v_2$ in G and having the knowledge represented by $K_{a'}$ are sufficient to give the knowledge represented by $K_{b'}$. This leads naturally to the following definition:

Definition 3.4: We say a monotone switching network solving directed connectivity is a certain-knowledge switching network if we can assign a $K_{a'}$ to each vertex $a' \in V(G')$ such that the following conditions hold:

1. $K_{s'} = \{\}$ and $K_{t'} = \{s \to t\}$.
2. If there is an edge with label $v_1 \to v_2$ between vertices a' and b', then
$K_{b'} \subseteq K_{a'} \cup \{v_1 \to v_2\}$ and $K_{a'} \subseteq K_{b'} \cup \{v_1 \to v_2\}$

Proposition 3.5: The condition that $K_{b'} \subseteq K_{a'} \cup \{v_1 \to v_2\}$ and $K_{a'} \subseteq K_{b'} \cup \{v_1 \to v_2\}$ is equivalent to the condition that we can obtain $K_{b'}$ from $K_{a'}$ using only the following reversible operations:

Operation 1: Add or remove $v_1 \to v_2$.

Operation 2: If $v_3 \to v_4, v_4 \to v_5$ are both in K, add or remove $v_3 \to v_5$ from K.

Operation 3: If $s \to t$ is in K, add or remove any edge except $s \to t$ from K

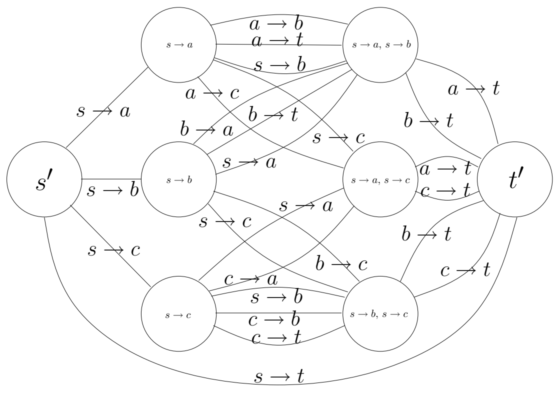

Figure 2. A certain-knowledge switching network that solves directed connectivity with five vertices, s, t, a, b, and c. The label inside each vertex represents the K for that vertex.

If this condition is satisfied, we say we can get from $K_{a'}$ to $K_{b'}$ with the edge $v_1 \to v_2$.

Remark 3.6: The operations in Proposition 3.5 are not a good starting point for definitions, but they are very effective for analyzing certain-knowledge switching networks solving directed connectivity. The reader would do very well to understand these operations thoroughly. In particular, note that each of these operations is reversible. This reflects the undirected nature of the switching network; we can undo any move that we make.

A. *certain-knowledge switching networks and Savitch's theorem*

While this model is restricted, it is not trivial. In particular, it is capable of capturing a variant of Savitch's algorithm. For details, see the full paper. This implies the following theorem:

Theorem 3.7: There is a certain-knowledge switching network of size $N^{O(\log N)}$ that solves directed connectivity on N vertices.

B. *Lower size bound on certain-knowledge switching networks solving directed connectivity*

We now prove Theorem 3.8, showing that the bound of theorem 3.7 is tight.

Theorem 3.8: Any certain-knowledge switching network that solves directed graph connectivity on N vertices has size at least $N^{\Omega(\log N)}$.

We will first show that the result follows from the following lemma. We will then give a proof sketch of the lemma. For a rigorous proof, see the full paper.

Lemma 3.9: If P is the path $s \to v_1, v_1 \to v_2, \cdots, v_{2^k} \to t$ in G, then any path P' in G' from s' to t' using only the edges of P must pass through at least one vertex a' such that $K_{a'} \neq K_{t'}$ and the union of the endpoints of the edges in $K_{a'}$ is a subset of $\{s, v_1, v_2, \cdots, v_{2^k}, t\}$ that contains at least $k+1$ of $v_1, v_2, \cdots, v_{2^k}$.

Proof of Theorem 3.8 using Lemma 3.9: For any prime p, if $k < p$, if we take all of the polynomials in $Z_p[x]$ of degree at most k, then any two distinct polynomials will have at most k values in common. Thus, if $p > 2^k$, given a polynomial $f(x)$ of degree at most k, if we take v_i to be vertex $p \cdot (i-1) + f(i)$ of G for $i = 1$ to 2^k, then the corresponding paths will share at most k vertices in common.

However, by Lemma 3.9, we can associate a vertex in G' to each such path, and no two such paths can share the same vertex. Hence, there are at least p^{k+1} vertices in G', and we can do this as long as $N \geq p^2 + 2$ and $k < \log p$. The result follows immediately. ■

Proof sketch of Lemma 3.9:

Definition 3.10: Call the vertices in $L = \{v_1, \cdots, v_{2^{k-1}}\}$ the left half of P and the vertices in $R = \{v_{2^{k-1}+1}, \cdots, v_{2^k}\}$ the right half of P.

Definition 3.11: K satisfies the lemma for the left half if $K \neq K_t$ and the union of the endpoints of the edges in K contains at least k of the vertices in L.

We define satisfying the lemma for the right half in a similar way.

We prove this lemma by induction. If there is a path $P' = s' \to v'_1 \to v'_2 \to \cdots \to v'_r \to t'$ in G' using only the edges in P, consider the sequence $K_{s'}, K_{v'_1}, \cdots K_{v'_r}, K_{t'}$. We get from each K to the next K using only the operations given by Proposition 3.5. Thus, we are trying to use these operations to obtain an edge from s to t (which represents a path from s to t in G).

To obtain an edge from s to t using only these operations, it is necessary (but not sufficient) to first obtain an edge from s to a vertex $r \in R \cup \{t\}$ and an edge from a vertex $l \in L \cup \{s\}$ to t. By the inductive hypothesis, to obtain an edge from s to a vertex $r \in R \cup \{t\}$, we must first reach a $K_{v'_i}$ that satisfies the lemma for the left half. If the union of the endpoints of the edges in $K_{v'_i}$ contains even one vertex in R, $K_{v'_i}$ will satisfy the lemma. If not, then either $K_{v'_i}$ already has an edge from a vertex $l \in L \cup \{s\}$ to t or there is neither an edge from $l \in L \cup \{s\}$ to a vertex $r \in R \cup \{t\}$ nor an edge from a vertex $r \in R$ to t, so there is no progress towards obtaining an edge from a vertex $l \in L \cup \{s\}$ to t. A similar argument holds if we try to obtain an edge from a vertex $l \in L \cup \{s\}$ to t.

If in trying to obtain an edge from s to a vertex $r \in R \cup \{t\}$ or an edge from a vertex $l \in L \cup \{s\}$ to t, when we reach such a $K_{v'_i}$, we always have no progress towards obtaining the other required edge, then we will never obtain an edge from $s \to t$. Thus, we can only obtain an edge from s to t if we reach such a $K_{v'_i}$ and the other required edge has already been obtained. But this means that we have reached a $K_{v'_j}$ such that either $K_{v'_j}$ contains an edge from s to a vertex $r \in R$ and the union of the endpoints of the edges in $K_{v'_j}$ does not contain any vertex in L or $K_{v'_j}$ contains an edge from a vertex $l \in L$ to t and the union of the endpoints of the edges in $K_{v'_j}$ does not contain any vertex in R.

If we could reach such a $K_{v'_j}$, we would indeed be close to obtaining the edge $s \to t$. However, reaching such a $K_{v'_j}$ without going through a $K_{v'_i}$ satisfying the lemma is impossible for the following reason:
When we first obtain an edge from s to a vertex $r \in R$, we must have the edges $s \to l$ and $l \to r$ for some $l \in L$. Removing these edges is just as difficult as obtaining them, which means that to remove them, we must pass through a $K_{v'_i}$ such that $K_{v'_i}$ satisfies the lemma for the left half. But if we also hold on to the edge $s \to r$, then $K_{v'_i}$ also contains a vertex in R, so $K_{v'_i}$ satisfies the lemma. A similar argument holds if we try to obtain an edge from a vertex $l \in L$ to t. ■

IV. Preliminary results on monotone switching networks

Not every monotone switching network solving directed connectivity is a certain-knowledge switching network. The monotone switching networks shown in Figures I and IV are not certain-knowledge switching networks. The reason why they are not certain-knowledge switching networks is because at the vertices in these switching networks, a depth-1 monotone formula (only ANDs) is insufficient to describe our knowledge about G. Instead, we need a depth-2 monotone formula (ORs of ANDs). Thus, we make the following definitions:

Definition 4.1: A state of knowledge J is a set $\{K_1, \cdots, K_m\}$ of knowledge sets.
Let $J_1 = \{K_{11}, K_{12}, \cdots, K_{1m}\}$ and let $J_2 = \{K_{21}, K_{22}, \cdots, K_{2n}\}$. We say $J_1 \subseteq J_2$ if for every j there exists a i such that $K_{1i} \subseteq K_{2j}$.
We say $J_1 = J_2$ if $J_1 \subseteq J_2$ and $J_2 \subseteq J_1$.
Let $J = \{K_1, \cdots, K_m\}$. Define $J \cup \{v_1 \to v_2\}$ to be $\{K_1 \cup \{v_1 \to v_2\}, \cdots, K_m \cup \{v_1 \to v_2\}\}$.
We say that we can get from J_1 to J_2 with the edge $v_1 \to v_2$ if
$J_2 \subseteq J_1 \cup \{v_1 \to v_2\}$ and $J_1 \subseteq J_2 \cup \{v_1 \to v_2\}$

Remark 4.2: The state of knowledge $J = \{K_1, \cdots, K_m\}$ represents knowing that the paths in K_1 are in G OR the paths in K_2 are in G OR $\cdots$ OR the paths in K_m are in G OR there is a path from s to t in G. Thus, J is characterized by its least informative K. The condition for $J_1 \subseteq J_2$ ensures that J_2 represents at least as much information about G as J_1.

Proposition 4.3: We can get from J_1 to J_2 with the edge $v_1 \to v_2$ if and only if for every i there exists a j such that $K_{2j} \subseteq K_{1i} \cup \{v_1 \to v_2\}$ and for every j there exists a i such that $K_{1i} \subseteq K_{2j} \cup \{v_1 \to v_2\}$.
To do this, the following 4 reversible operations are sufficient:
Operation 1: Add or remove $v_1 \to v_2$ from any K_i.
Operation 2: If $v_3 \to v_4, v_4 \to v_5$ are both in K_i, add or remove $v_3 \to v_5$ from K_i.
Operation 3: If $s \to t$ is in K_i, add or remove any path except $s \to t$ from K_i
Operation 4a: If $K_i \subseteq K_{i'}$ for some i, i', remove $K_{i'}$ from J.
Operation 4b: If for some knowledge set K and some i, $K_i \subseteq K$, add K to J.

Proposition 4.4: For any monotone switching network solving directed connectivity, we can assign a $J_{a'}$ to each $a' \in V(G')$ so that the following properties hold:
1. $J_{s'} = \{\{\}\}$ and $J_{t'} = \{\{s \to t\}\}$.
2. If there is an edge with label $v_1 \to v_2$ between a' and b', then it is possible to get from $J_{a'}$ to $J_{b'}$ with the edge $v_1 \to v_2$.

Proof: For each vertex $a' \in V(G')$, take $J_{a'}$ to be the set of all K such that using the edges of K, it is possible to reach a' from s' in G'. It is easy to check that both of the above properties are satisfied. ■
We will now describe a useful simplification for monotone switching networks that can be accomplished with an increase of at most a factor of N in the size of the network. For a proof, see the full paper.

Theorem 4.5: If there is a monotone switching network (G', s', t', μ') solving directed connectivity on N vertices, then there is a monotone switching network (G'', s'', t'', μ'') with $|V(G'')| \leq N|V(G')|$ such that for any vertex a'' of G'', for any K in $J_{a''}$, K consists only of edges of the form

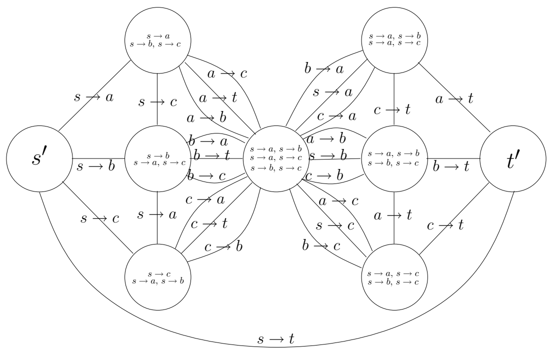

Figure 3. A monotone switching network that solves directed connectivity with five vertices, s, t, a, b, and c. The label inside each vertex gives the J for that vertex, with each line corresponding to one of its K.

$s \to v$ for some $v \in V(G)$.
Finally, we state a theorem that shows that in some sense, monotone switching networks can be reduced to certain-knowledge switching networks. For a proof, see the full paper.

Theorem 4.6: For any monotone switching network, if there is a path in G' from s' to t' using only edges that have a label in a subset E of $E(G)$, then there is a sequence of K_i, $0 \leq i \leq m$ with the following properties:
1. $K_0 = \{\}$. $K_m = \{s \to t\}$.
2. For each i, there exists an edge $e_i \in E$ such that it is possible to go from K_i to K_{i+1} using the edge e_i and the three given operations.
3. For each i, there exists a vertex a'_i on this path such that K_i is the union of some subset of $J_{a'_i}$.

V. Fourier analysis on monotone switching networks

Unfortunately, the above results are insufficient to prove a superpolynomial lower size bound on monotone switching networks solving directed connectivity. To prove a good lower size bound, more sophisticated techniques are needed. In this section, we introduce a fourier transformation technique for monotone switching networks solving directed connectivity and give a condition which is sufficient to prove a superpolynomial lower size bound.

An alternate way of solving directed connectivity is to look at cuts of G. There is a path from s to t if and only if there is no cut $C = (V_1, V_2)$ such that $s \in V_1$, $t \in V_2$, and there is no edge from a vertex in V_1 to a vertex in V_2. Thus, instead of describing each state of knowledge J in terms of paths in G, we can describe each J as a function of the cuts of G. We do this below.

A. Definitions and basic properties

Definition 5.1: We define an s-t cut (below we use cut for short) of G to be a subset C of $V(G)$ such that $s \in C$ and $t \notin C$. We denote the complement of C by C^c, and we say an edge $v_1 \to v_2$ crosses C if $v_1 \in C$ and $v_2 \in C^c$.
Let $\mathcal{C}$ denote the set of all cuts C. $|\mathcal{C}| = 2^{N-2}$.
Given a state of knowledge J, we want $J(C)$ to be 1 if given the information J represents we know that G contains an edge crossing C and -1 otherwise. This leads to the following definitions:

Definition 5.2: Given a cut C and a set of edges K, define $K(C)$ to be 1 if there is an edge in K that crosses C and -1 otherwise.

Definition 5.3: Given a cut C and a state of knowledge $J = \{K_1, \cdots, K_m\}$, define $J(C)$ to be 1 if for all i, $K_i(C) = 1$ and -1 otherwise.
It is easy to verify that for every knowledge set K and state of knowledge J, $K(C)$ and $J(C)$ are well-defined, i.e. if $K = K'$, $K(C) = K'(C)$ and if $J = J'$, $J(C) = J'(C)$.
Note that for all C, $J_{s'}(C) = -1$ and $J_{t'}(C) = 1$.
We define basis functions as follows:

Definition 5.4: Given a set of vertices V that does not include s or t, define $e_V(C) = (-1)^{|V \cap C|}$.
We define the dot product as follows:

Definition 5.5: Given two functions $f, g : \mathcal{C} \to \mathbb{R}$, $f \cdot g = 2^{2-N} \sum_{C \in \mathcal{C}} f(C)g(C)$
Note that $e_V(C)e_{V'}(C) = (-1)^{(V \Delta V') \cap C}$ for every cut C, where Δ denotes the symmetric difference of two sets, and hence the functions $\{e_V\}$ form an orthonormal basis for the vector space $\mathbb{R}^{\mathcal{C}}$ with the standard dot product $f \cdot g = 2^{2-N} \sum_{C \in \mathcal{C}} f(C)g(C)$.
We define Fourier coefficients as follows:

Definition 5.6: $\hat{f_V} = f \cdot e_V$

Proposition 5.7: For any function f, $f = \sum_V \hat{f_V} e_V$ and $f \cdot f = \sum_V \hat{f_V}^2$, where we are summing over all subsets V of $V(G)$ such that $s, t \notin V$.

Proposition 5.8: Given a monotone switching network G', if there an edge with label $v_1 \to v_2$ between vertices a' and b', then for any cut C, if $v_1 \notin C$ or $v_2 \notin C^c$, then $J_{a'}(C) = J_{b'}(C)$.

B. A general technique for obtaining lower size bounds

In this subsection, we show how to use this Fourier analysis to show lower bounds on monotone switching networks solving directed connectiivty.

Definition 5.9: Given a directed walk P' from v'_1 to v'_2 in G' and a label e, define $d(P', e) \in \mathbb{R}^{\mathcal{C}}$

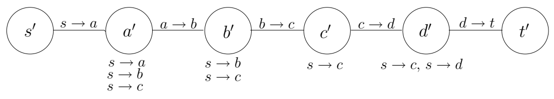

Figure 4.

This is a possible path in G' from s' to t'. Below, we express the six functions $J_{s'}$, $J_{a'}$, $J_{b'}$, $J_{c'}$, $J_{d'}$, and $J_{t'}$ in terms of the basis functions:
$J_{s'} = -e_{\{\}}$.
$J_{a'} = -\frac{3}{4}e_{\{\}} + \frac{1}{4}e_{\{a\}} + \frac{1}{4}e_{\{b\}} + \frac{1}{4}e_{\{a,b\}} + \frac{1}{4}e_{\{c\}} + \frac{1}{4}e_{\{a,c\}} + \frac{1}{4}e_{\{b,c\}} + \frac{1}{4}e_{\{a,b,c\}}$.
$J_{b'} = -\frac{1}{2}e_{\{\}} + \frac{1}{2}e_{\{a\}} + \frac{1}{2}e_{\{b\}} + \frac{1}{2}e_{\{a,b\}}$.
$J_{c'} = e_{\{c\}}$.
$J_{d'} = \frac{1}{2}e_{\{\}} + \frac{1}{2}e_{\{a\}} + \frac{1}{2}e_{\{b\}} - \frac{1}{2}e_{\{a,b\}}$.
$J_{t'} = e_{\{\}}$.

to be $\frac{1}{2}(\sum_{v' \in V_{sink}} J_{v'} - \sum_{v' \in V_{source}} J_{v'})$, where V_{sink} is the set of vertices in G' with an edge in P' with label e going into it and V_{source} is the set of vertices in G' with an edge in P' with label e going out from it, counted with multiplicity.

Remark 5.10: Together, all of the edges in P' allow us to go from v'_1 to v'_2, and the change in our knowledge is $J_{v'_2} - J_{v'_1}$. $d(P', e)$ measures the contribution to this change made by edges with label e.

Definition 5.11: Given a directed walk P' from v'_1 to v'_2 in G' using only the edges of some directed path P in G from s to t and a partition of the edges of P into two sets, E_1 and E_2, let $f_{P',P,E_1,E_2} = \sum_{e \in E_1} d(P', e) - \sum_{e \in E_2} d(P', e)$.

Definition 5.12: We say a cut C is (P, E_1, E_2)-invariant if all edges e in P that cross C are in E_1 or all edges e in P that cross C are in E_2. We say a function $g : \mathcal{C} \to \mathbb{R}$ is (P, E_1, E_2)-invariant if $f_{P',P,E_1,E_2} \cdot g$ is the same for all switching networks G' solving directed connectivity on $V(G)$ and paths P' in G' from s' to t'. If g is (P, E_1, E_2)-invariant, define $z(g, P, E_1, E_2)$ to be this constant.

Proposition 5.13: A function $g : \mathcal{C} \to \mathbb{R}$ is (P, E_1, E_2)-invariant if and only if $g(C) = 0$ for every C that is not (P, E_1, E_2)-invariant.

Proof: For any cut C that is not (P, E_1, E_2)-invariant, we can change the value of $f_{P',P,E_1,E_2}(C)$ without changing $f_{P',P,E_1,E_2}(C')$ for any other cut C'. To see this, given a G', create a new G' by creating a new s'. Let a' be the old s' and for each edge e such that e crosses C, create an edge with label e between s' and a'. This is still a valid monotone switching network solving directed connectivity and for all vertices v' except s', $v'(C) = 1$. Also, $a'(C') = 1$ if $C' = C$ and -1 otherwise. Thus, we can change $f_{P',P,E_1,E_2}(C)$ without changing $f_{P',P,E_1,E_2}(C')$ for any other C' by choosing whether to use an edge with label in E_1 or E_2 to go from s' to a'. Thus, if $g(C) \neq 0$, then g cannot be (P, E_1, E_2)-invariant.
If C cannot be crossed by any edge in E_1, then $\sum_{e \in E_1} d(P', e)(C) = 0$. $\sum_{e \in E_1} d(P', e)(C) + \sum_{e \in E_2} d(P', e)(C) = \frac{1}{2}(J_{t'}(C) - J_{s'}(C)) = 1$, so $\sum_{e \in E_2} d(P', e)(C) = 1$, and $f_{P',P,E_1,E_2}(C) = -1$. Similarly, if C cannot be crossed by any edge in E_2, then $f_{P',P,E_1,E_2}(C) = 1$. In either case, $f_{P',P,E_1,E_2}(C)$ is the same for all switching networks G' solving directed connectivity on $V(G)$ and all paths P' in G' from s' to t', so if $g(C) = 0$ for every C that is not (P, E_1, E_2)-invariant, then g is (P, E_1, E_2)-invariant, as needed. ■

Theorem 5.14: Let $V(G) = \{s, v_1, \cdots, v_{k_1}, t\}$ and let P be the path $s \to v_1 \to \cdots \to v_{k_1} \to t$. If we have a partition of the edges of P into two groups E_1 and E_2, and a function g_P such that g_P is (P, E_1, E_2)-invariant, $z(g_P, P, E_1, E_2)$ is nonzero, and $\hat{g_P}_V = 0$ for any set of vertices V such that $|V| < k_2$, then any monotone switching network solving directed connectivity on N vertices has size at least $\Omega(N^{\frac{k_2}{2}})$.

Proof: Without loss of generality, we may assume $g_P \cdot g_P = 1$. Given such a g_P, note that if we add vertices to $V(G)$ so that $|V(G)| = N$, we can keep the same g_P (expressed in fourier coefficients) and it will still be (P, E_1, E_2)-invariant and have the same $z(g_P, P, E_1, E_2)$. Let $M = z(g_P, P, E_1, E_2)$.
If we have another path P_2 from s to t of length $k_1 + 1$, by symmetry, we have a function g_{P_2} and a partion (E_3, E_4) of the edges of P_2 so that $g_{P_2} \cdot g_{P_2} = 1$, g_{P_2} is (P_2, E_3, E_4)-invariant, and $z(g_{P_2}, P_2, E_3, E_4) = M$. Moreover, if P and P_2 have less than k_2 vertices in common (excluding s and t), then $g_P \cdot g_{P_2} = 0$. If we have K paths $P_1, \cdots, P_K$ of length $k_1 + 1$ from s to t in G such that any pair of them have less than k_2 vertices in common (excluding s and t), then we have K orthonormal functions $\{g_{P_i}\}$.
Now assume G' solves directed connectivity on N vertices, and let N' be the number of vertices in G'. We wish to bound N' from below. Note that given any set of orthonormal functions $\{g_i\}$,

$$N' \geq \sum_i (\sum_{a' \in V(G')} |J_{a'} \cdot g_i|^2) \tag{1}$$

Using Cauchy-Schwarz (specifically $\sum_{j=1}^{N'} |c_j|^2 \geq \frac{1}{N'}(\sum_{j=1}^{N'} |c_j|)^2$ for any $c_1, \cdots, c_{N'}$),

$$N' \geq \sum_i (\sum_{a' \in V(G')} |J_{a'} \cdot g_i|^2) \tag{2}$$

$$N' \geq \frac{1}{N'} \sum_i \Big(\sum_{a' \in V(G')} |J_{a'} \cdot g_i| \Big)^2 \qquad (3)$$

$$N' \geq \sqrt{\sum_i \Big(\sum_{a' \in V(G')} |J_{a'} \cdot g_i| \Big)^2} \qquad (4)$$

If P' is a path in G' from s' to t' using only the edges of P, by the definition of f_{P',P,E_1,E_2},

$$M = |f_{P',P,E_1,E_2} \cdot g_P| \leq \sum_{a' \in V(P')} |J_{a'} \cdot g_P| \quad (5)$$

Also, we clearly have that

$$\sum_{a' \in V(P')} |J_{a'} \cdot g_P| \leq \sum_{a' \in V(G')} |J_{a'} \cdot g_P| \quad (6)$$

Combining 5 and 6, we get that

$$\sum_{a' \in V(G')} |J_{a'} \cdot g_P| \geq M \qquad (7)$$

$$\Big(\sum_{a' \in V(G')} |J_{a'} \cdot g_P| \Big)^2 \geq M^2 \qquad (8)$$

By symmetry, 8 holds for each g_{P_i}. Plugging 8 into 4,

$$N' \geq \sqrt{KM^2} = M\sqrt{K} \qquad (9)$$

Following similar logic as in the proof of Theorem 3.8, we can easily obtain make K at least $\Omega(N^{k_2})$, so N' is at least $\Omega(N^{\frac{k_2}{2}})$, as needed. ■

Corollary 5.15: If for all k, when $V(G) = \{s, v_1, \cdots, v_{2^k}, t\}$ and P is the path $s \to v_1 \cdots \to v_{2^k} \to t$, there exists a partition of the edges of P into two groups E_1 and E_2 and a function g_P such that g_P is (P, E_1, E_2)-invariant, $z(g_P, P, E_1, E_2)$ is nonzero, and $\hat{g_{P}}_V = 0$ for any set of vertices V such that $|V| \leq k$, then any monotone switching network solving directed connectivity must have superpolynomial size.

Proof: Applying Theorem 5.14 for a fixed k gives that any monotone switching network solving directed connectivity must have size at least $c(k)(N^{\frac{k+1}{2}})$, where $c(k)$ is a constant depending on k. Thus, the size of a monotone switching network solving directed connectivity grows faster than any polynomial, as needed. ■

Remark 5.16: Without a bound on $c(k)$, we cannot give an explicit lower bound on the size of a switching network solving directed connectivity, we can only say that it is superpolynomial. For these bounds, see Section 7 of the full paper.

VI. A SUPERPOLYNOMIAL BOUND

In this section, we prove the following theorem.

Theorem 6.1: For all k, if $V(G) = \{s, v_1, \cdots, v_{2^k}, t\}$ and P is the path $s \to v_1 \cdots \to v_{2^k} \to t$, there exists a partition of the edges of P into two groups E_1 and E_2 and a function g_P such that g_P is (P, E_1, E_2)-invariant, $z(g_P, P, E_1, E_2)$ is nonzero, and $\hat{g_{P}}_V = 0$ for any set of vertices V such that $|V| \leq k$.

By Corollary 5.15, this is sufficient to prove a superpolynomial lower size bound on monotone switching networks solving directed connectivity.

A. *Proof overview*

We now give an informal overview of the proof of Theorem 6.1.

It is instructive to first note how this function g_P relates to certain-knowledge switching networks and Lemma 3.9. If we let W be the set of all a' such that the union of the endpoints of the edges in $K_{a'}$ contains at least $k+1$ vertices, then Lemma 3.9 says that any path P' in G' from s to t using only the edges of P must pass through at least one vertex $w' \in W$. We can think of W as a barrier preventing us from easily going from s' to t'. The function g_P describes this barrier more precisely, as if we let W' be the set of all vertices a' such that $K_{a'} \cdot g_{P,E_1,E_2} \neq 0$, then P' must pass through at least one vertex $w' \in W'$. Also, $W' \subseteq W$.

Thus, the existence of such a g_P implies Lemma 3.9. Roughly speaking, we want to show the converse, that the existence of such a barrier for certain-knowledge switching networks implies the existence of such a g_P.

To show that a function $g : \mathcal{C} \to \mathbb{R}$ is (P, E_1, E_2)-invariant, we either need to show that $g(C) = 0$ for all cuts C that are not (P, E_1, E_2)-invariant, or we need to show that $f_{P',P,E_1,E_2} \cdot g$ is the same for all switching networks G' solving directed connectivity on $V(G)$ and all paths P' in G' from s' to t'. If we had an explicit formula for $g(C)$, it would be easiest to use the first approach. However, since we do not have such a general formula, we use the second approach.

To do this, we use Theorem 6.2, which says that if we add the condition that $f_{L',P,E_1,E_2} \cdot g_P = 0$ for all directed cycles L' of G' using only the edges of P, we only need to consider certain-knowledge switching networks G' such that all knowledge sets contain only edges of the form $s \to v$.

Since there are now at most $2^{2^k}+1$ knowledge sets and we must have $J_{t'} = -J_{s'}$, as noted in Lemma 6.4, we can arbitrarily choose the values $g \cdot J_{a'}$ for all $a' \in V(G')$ except t'. Lemma 6.5 shows that if we can split the vertices of G' into 4 groups with a mapping $b : V(G') \to \{0,1\} \times \{0,1\}$ that has ceratin properties, then using the freedom given by Lemma 6.4, we can create a g that satisfies the conditions given by Theorem 6.2 and thus satisifes Theorem 6.1. Lemma 6.7 shows that if we have a barrier W similar to the one provided by Lemma 3.9 with one additional property, then we can create a mapping $b : V(G') \to \{0,1\} \times \{0,1\}$ as required by Lemma 6.5. Finally, Lemma 6.8 modifies Lemma 3.9 so that it provides the barrier W with the needed additional property. Putting everything together, we can create a function g_P that satisfies Theorem 6.1.

B. Reduction to certain-knowledge switching networks

We have the following theorem. For the proof, see the full paper.

Theorem 6.2: If for a function $g : \mathcal{C} \to \mathbb{R}$, for any certain-knowledge G' such that all knowledge sets contain only edges of the form $s \to v$, $f_{P',P,E_1,E_2} \cdot g$ is independent of P' and $f_{L',P,E_1,E_2} \cdot g = 0$ for all directed cycles L' in G', then g is (P, E_1, E_2)-invariant.

C. Construction of g_P

In this subsection, we complete the proof of Theorem 6.1 by constructing a function $g_P : \mathcal{C} \to \mathbb{R}$ with the given properties.

Looking at certain-knowledge G' where each knowledge set only has paths of the form $s \to v$, there are only $2^{2^k} + 1$ possible knowledge sets: $s \to t$ and anything of the form $\cup_{v \in V}\{s \to v\}$ for some set of vertices V. Denote each such K by K_V.

Proposition 6.3: If $V' \not\subset V$, then $e_{V'} \cdot K_V = 0$ and $e_V \cdot K_V \neq 0$.

Lemma 6.4: For any set of values $\{a_V\}$, there is a function $g : \mathcal{C} \to \mathbb{R}$ such that for all V, $g \cdot K_V = a_V$. Furthermore, if there is a k such that if $|V| \leq k$, then $g \cdot K_V = 0$, then writing $g = \sum_{V'} c_{V'} e_{V'}$, if $|V'| \leq k$ then $c_{V'} = 0$.

Proof of Lemma 6.4: To see the first part of the lemma, pick an ordering of the V such that no V is a subset of an earlier V. Now pick each c_V in that order. Since if $V' \not\subset V$, then $e_{V'} \cdot K_V = 0$ and $e_V \cdot K_V \neq 0$, this means that when we pick each c_V, we can change the value of a_V without affecting any earlier a_V. Thus, we can freely choose each a_V.
To see the second part of the lemma, let V be a set such that $c_V \neq 0$ and for all proper subsets V' of V, $c_{V'} = 0$. Then by the above proposition, $a_V \neq 0$, as needed. This completes the proof. ■

Lemma 6.5: If we have a directed path P in G, a partition of the edges of P into two sets E_1 and E_2, and a mapping $b : V(G') \to \{0,1\} \times \{0,1\}$ such that if we write $b(v') = (b_1(v'), b_2(v'))$, then:
1. $b_1(s') = b_2(s') = 0$.
2. $b_1(t') = b_2(t') = 1$.
3. If $b_i(v_1') \neq b_i(v_2')$, $i \in \{1,2\}$, then there is no edge with label in E_i between v_1' and v_2'.
Then if we also have a $g : \mathcal{C} \to \mathbb{R}$ such that $g \cdot J_{v'} = b_2(v') - b_1(v')$ for all $v' \in V(G')$, then for any directed cycle L' in G' using only the edges of P, $f_{L',P,E_1,E_2} \cdot g = 0$ and for any path P' in G' from s' to t' using only the edges of P, $f_{P',P,E_1,E_2} \cdot g = 1$.

Proof of Lemma 6.5: This follows immediately from the following proposition:

Proposition 6.6: With the above conditions, if P'' is a path in G' from s' to a', then
$\frac{1}{2}(\sum_{e \in E_1} d(P'', e) - \sum_{e \in E_2} d(P'', e)) \cdot g = \frac{1}{2}(b_2(a') + b_1(a'))$

Proof of Proposition 6.6: We prove this by induction. It is clearly true for paths of length 0. Assume we have a path P'' from s' to some vertex $v_1' \in V(G')$ for which the proposition is true and an additional edge e' from v_1' to some vertex $v_2' \in V(G')$. Let P''' be P'' with the edge e' added.
If e' has a label in E_1, then $b_1(v_2') = b_1(v_1')$, so
$\frac{1}{2}(\sum_{e \in E_1} d(P''', e) - \sum_{e \in E_2} d(P''', e)) \cdot g = \frac{1}{2}(\sum_{e \in E_1} d(P'', e) - \sum_{e \in E_2} d(P'', e)) \cdot g + \frac{1}{2}(g \cdot J_{v_2'}) - \frac{1}{2}(g \cdot J_{v_1'})$
$= \frac{1}{2}(b_2(v_1') + b_1(v_1')) + \frac{1}{2}(b_2(v_2') - b_1(v_2')) - \frac{1}{2}(b_2(v_1') - b_1(v_1'))$
$= \frac{1}{2}(b_2(v_2') + b_1(v_2'))$, as needed.
Similarly, if e' has a label in E_2, then $b_2(v_2') = b_2(v_1')$, so
$\frac{1}{2}(\sum_{e \in E_1} d(P''', e) - \sum_{e \in E_2} d(P''', e)) \cdot g = \frac{1}{2}(\sum_{e \in E_1} d(P'', e) - \sum_{e \in E_2} d(P'', e)) \cdot g - \frac{1}{2}(g \cdot J_{v_2'}) + \frac{1}{2}(g \cdot J_{v_1'})$
$= \frac{1}{2}(b_2(v_1') + b_1(v_1')) + \frac{1}{2}(-b_2(v_2') + b_1(v_2')) - \frac{1}{2}(-b_2(v_1') + b_1(v_1'))$
$= \frac{1}{2}(b_2(v_2') + b_1(v_2'))$, as needed.

This completes the proof. ■

■

To find such a suitable mapping $b : V(G') \to \{0,1\} \times \{0,1\}$, we use the following lemma. For a proof, see the full paper.

Lemma 6.7: If there is a set of vertices W in G' such that any path P' from s' to t' using only edges with labels in P contains a vertex $w' \in W$ incident with both an edge in P' with label in E_1 and an

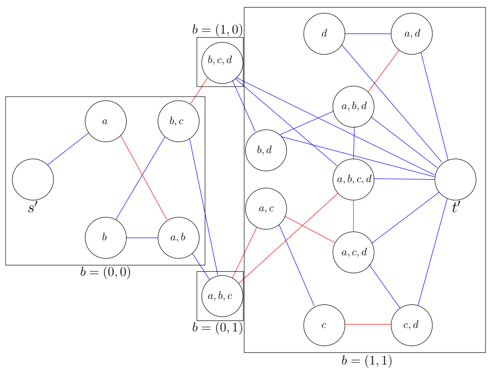

Figure 5. This is a partition of the vertices in G' into 4 groups with a mapping $b: V(G') \to \{0,1\} \times \{0,1\}$ as described in Lemma 6.5, where G' is a certain-knowledge switching network such that all knowledge sets contain only edges of the form $s \to v$, $P = s \to a \to b \to c \to d \to t$, $E_1 = \{s \to a, b \to c, d \to t\}$, and $E_2 = \{a \to b, c \to d\}$. In this diagram, each vertex has knowledge set K_V, where V is the set of vertices inside of the vertex. Edges with label in E_1 are blue and edges with label in E_2 are red. Taking $g = 4e_{\{a,b,c\}} - 4e_{\{b,c,d\}}$, $g \cdot J_{a'} = b_2(a') - b_1(a')$.

edge in P' with label in E_2, then there is a mapping $b: V(G') \to \{0,1\} \times \{0,1\}$ as described in Lemma 6.5 where all vertices v' such that $b_1(v') \neq b_2(v')$ are in W.

The final Lemma we need is a slight modification of Lemma 3.9:

Lemma 6.8: If P is the path $s \to v_1, v_1 \to v_2, \cdots, v_{2^k} \to t$, then setting $s = v_0$, $t = v_{2^k+1}$, taking E_1 to be all edges of the form $v_i \to v_{i+1}$ where i is even and taking E_2 to be the remaining edges, then if G' is a certain-knowledge switching network, any path P' in G' from s' to t' using only the edges in P must pass through at least one vertex a' such that $K_{a'} \neq K_{t'}$ and the union of the endpoints of the edges in $K_{a'}$ is a subset of $\{s, v_1, v_2, \cdots, v_{2^k}, t\}$ that contains at least $k+1$ of $v_1, v_2, \cdots, v_{2^k}$. Furthermore, a' is incident with both an edge in P' with label in E_1 and an edge in P' with label in E_2.

Proof of Lemma 6.8: The proof is identical to the proof of Lemma 3.9, except that in the inductive hypothesis we also require that a' is incident with both an edge in P' with label in E_1 and an edge in P' with label in E_2. ■

Proof of Theorem 6.1: We put everything together as follows. Using Lemma 6.8, we obtain a W which we can use in Lemma 6.7. This gives us a mapping $b: V(G') \to \{0,1\} \times \{0,1\}$ which we can use in Lemma 6.5. By Lemma 6.4, we can obtain a function $g_P : \mathcal{C} \to \mathbb{R}$ that satisfies all of the conditions of Lemma 6.5, so for any directed cycle L' in G' using only the edges of P, $f_{L',P,E_1,E_2} \cdot g_P = 0$ and for any path P' in G' from s' to t' using only the edges of P, $f_{P',P,E_1,E_2} \cdot g_P = 1$. Also, by Lemma 6.4, if $|V| \leq k$, $\hat{g_{P}}_V = 0$. Using Theorem 6.2, g_P is (P, E_1, E_2)-invariant and $z(g_P, P, E_1, E_2) = 1$, as needed. This completes the proof. ■

Acknowledgment

The author wishes to thank Boaz Barak and Eli-Ben-Sasson for their advice on this research and for their help in editing the article.

References

[1] R. Aleliunas, R. M. Karp, R. J.Lipton, L. Lovász, and C. Rackoff. Random walks, universal traversal sequences, and the complexity of maze problems. Proceedings of the 20th Annual Symposium on Foundations of Computer Science, p.218-223, 1979

[2] S. A. Cook and C. W. Rackoff. Space lower bounds for maze threadability on restricted machines. SIAM Journal on Computing, 9(3)636-652, Aug 1980

[3] N. Immerman. Nondeterministic Space is Closed Under Complementation, SIAM J. Comput. 17, pp. 935-938, 1988

[4] M. Karchmer and A. Wigderson. Monotone circuits for connectivity require superlogarithmic depth, Proceedings of ACM STOC'88, pp. 539-550, 1988.

[5] W. Masek. A fast algorithm for the string editing problem and decision graph complexity. Master's Thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1976.

[6] R. Raz, P. McKenzie. Separation of the monotone NC hierarchy, Proceedings of the 38th Annual Symposium on Foundations of Computer Science, pp 234-243, 1997

[7] A. Razborov. Lower Bounds for Deterministic and Nondeterministic Branching Programs, Proceedings of the 8th FCT, Lecture Notes in Computer Science, vol. 529, 1991, 47-60.

[8] O. Reingold. Undirected ST-connectivity in Log-Space, STOC 2005.

[9] W. J. Savitch. Relationship between nondeterministic and deterministic tape classes, J.CSS, 4, pp 177-192, 1970

[10] R. Szelepcsenyi. The method of forcing for nondeterministic automata, Bull. EATCS 33, pp. 96-100, 1987

[11] V. Trifonov. An O(log n log log n) space algorithm for undirected st-connectivity, Proceedings of the thirty-seventh annual ACM symposium on Theory of computing, May 2005.