Electronic Colloquium on Computational Complexity - Reports Series 1996 - available via:
ECCC
TR96-039
FTP: ftp.eccc.uni-trier.de:/pub/eccc/
WWW: http://www.eccc.uni-trier.de/eccc/
Email: ftpmail@ftp.eccc.uni-trier.de with subject 'help eccc'

A Compendium of Problems Complete for Symmetric Logarithmic Space

Carme Àlvarez*
Departament Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
Pau Gargallo 5
08028 Barcelona, Spain
e-mail address: alvarez@lsi.upc.es

Raymond Greenlaw†
Department of Computer Science
University of New Hampshire
Durham, NH 03824
e-mail address: greenlaw@cs.unh.edu

June 26, 1996

Abstract

We provide a compendium of problems that are complete for *symmetric logarithmic space* (SL). Complete problems are one method of studying this class for which programming is nonintuitive. A number of the problems in the list were not previously known to be complete. A list containing a variety of open problems is also given.

*This work was partially supported by ESPRIT LTR Project no. 20244-ALCOM-IT.
†This research partially supported by National Science Foundation grant CCR-9209184, a Fulbright Scholarship Senior Research Award 1995, and a Spanish Fellowship for Scientific and Technical Investigations 1996.

1 Introduction

In this paper we describe problems that are logarithmic space many-one complete for symmetric logarithmic space (SL). Our hope in collecting these problems and extending this list is that more insight can be gained about the relationships between the complexity classes deterministic logarithmic space (DL), SL, and nondeterministic logarithmic space (NL). The symmetric Turing machine model introduced by Lewis and Papadimitriou in [17] is not an intuitive model to program due to the reversibility property of transitions and studying complete problems for SL is one approach to gain a better understanding of this class.

Lewis and Papadimitriou defined SL to capture the complexity of the undirected s-t connectivity problem (USTCON, see Problem 5.1). They proved that

$$\mathrm{DL} \subseteq \mathrm{SL} \subseteq \mathrm{NL}$$

and that USTCON is complete for SL. Many results that are relatively straightforward to prove about Turing machines become much more involved when carried over to symmetric Turing machines [17]. In a major advance in complexity theory and stirring further interest in SL, Nisan and Ta-Shma proved that SL is closed under complement.

Theorem 1.1 (Nisan and Ta-Shma [19])
The complexity class symmetric logarithmic space is closed under complement. That is, SL equals co-SL.

This result was achieved through a series of reductions and not by a technique related to inductive counting (Immerman [12] and Szelepcsényi [28]). A proof that NL equals co-NL using techniques similar to Nisan and Ta-Shma's has not been achieved although such a proof would be very interesting. Borodin et al. point out that the Immerman-Szelepcsényi proof technique does not seem to apply to yield a proof that SL equals co-SL due to the fact that symmetric Turing machines cannot "nondeterministically count" [2].

Figure 1 depicts the relationships currently known among the classes in the vicinity of SL. Many of the definitions involving these classes can be found in the excellent survey by Johnson [14] or the excellent paper by Borodin et al. [2]. Figure 7 of [14] and Figure 1 of [2] were combined and modified slightly to obtain our figure.

The remainder of this paper is organized as follows: §2 contains some background material about SL; in §3 a list of problems that are logarithmic space many-one complete for SL is given; sections 4, 5, 6, and 7 describe the SL-complete problems of type machine simulation, connectivity, graph theory, and miscellaneous, respectively; and §8 contains a number of open problems that are candidates for being SL-complete.

2 Preliminaries

The following notation will be useful for this paper. Let $G = (V, E)$ be an undirected graph. $\#\mathrm{cc}(G)$ denotes the number of connected components of G. $\overline{G}$ denotes the complementary graph of G, that is $\overline{G} = (V, (V \times V) - E)$. We use the notation $<_{\mathrm{lex}}$ to mean less than in lexicographic order or less than using the natural order specified by a problem instance. For definitions of basic complexity classes and techniques used in computational complexity theory, the reader is referred to Johnson [14] or Greenlaw, Hoover, and Ruzzo [9].

The complexity class symmetric logarithmic space is defined in terms of the *symmetric Turing machine* (STM) introduced by Lewis and Papadimitriou in [17]. The exact definition of STMs is very detailed and we provide only an intuitive description of the model here. A symmetric Turing machine can be thought of as a nondeterministic Turing machine which has the additional requirement that every move of the machine is "reversible." In order for this to be achievable the machine is allowed to scan two symbols at a time on each of its tapes. We can now formally define the notion of symmetric logarithmic space.

Definition 2.1 **Symmetric logarithmic space**, **SL**, *is the class of languages accepted by logarithmic space bounded symmetric Turing machines.*

We are interested in describing problems that are complete for SL under logarithmic space many-one reducibility.

Definition 2.2 *A language or problem L is* **SL-complete** *if $L \in SL$ and for all $L' \in SL$, L' is logarithmic space many-one reducible (denoted $\leq^m_{\log}$) to L. Problems L and L' are* **logarithmic space equivalent** *if and only if $L \leq^m_{\log} L'$ and $L' \leq^m_{\log} L$.*

In [17] they show that SL is closed under logarithmic space many-one reductions (Theorem 4, page 172). It is well-known that (deterministic)

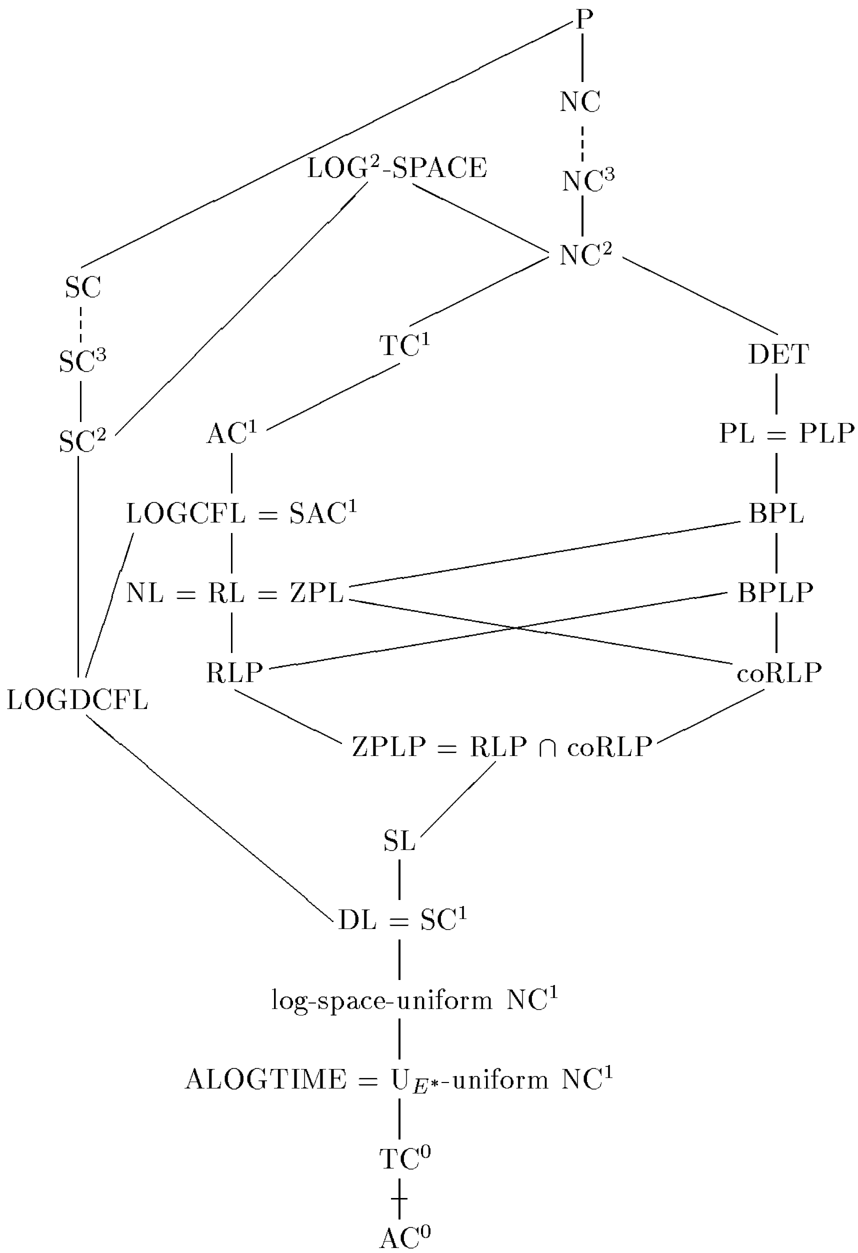

Figure 1: Complexity Classes Surrounding SL.

logarithmic space reductions are transitive, so in proving other problems are complete for SL one can exploit this fact. To demonstrate completeness for SL one must also show the problem under consideration is in SL. This is not always easy since symmetric Turing machines are difficult to reason about. The following lemma is helpful for showing problems are in SL.

Lemma 2.3 (Nisan and Ta-Shma [19, Corollary 3.1])
The class of languages accepted by deterministic logarithmic space bounded Turing machines having an oracle for SL is exactly equal to SL. That is, DL^{SL} *equals SL.*

This lemma gives another characterization of SL: those languages that are logarithmic space Turing reducible to a language in SL. This viewpoint is useful for showing additional languages are in SL. Since SL is closed under logarithmic space many-one reducibility [17], another way of viewing SL is as those languages that are logarithmic space many-one reducible to USTCON. Immerman provides a logical characterization of SL as (FO + posSTC), see [11] for details. SL is also equal to those languages that can be accepted by a uniform family of polynomial size contact schemes, see Razborov [23]. Nisan and Ta-Shma give one further characterization of SL by showing that SL equals $\mathrm{SL}^{\mathrm{SL}}$ when the oracle queries are asked in a deterministic way as defined by Ruzzo, Simon, and Tompa [25, page 224]. Here the requirement is that once the first symbol of a query is written on the oracle tape, the machine must behave deterministically until the query is asked and the oracle tape is erased.

3 SL-complete Problems

In this section we list the problems that are SL-complete. We welcome additions to this list or to the open problems list given in §8. For comparison purposes we note that Cook and McKenzie presented a list of problems that are complete for DL [4], and Jones, Lien, and Lasser gave a list of problems that are complete for NL [15].

Since SL equals co-SL (see Theorem 1.1), the complement of each of the problems listed is also SL-complete. In a couple of important cases we list both the problem and its complement separately. For each problem we provide a definition, reference, proof hint, and remarks. The proof hints vary widely in their level of utility. For problems that were previously known to be SL-complete, we typically provide just a brief hint; the reader should consult the original reference for more details. In addition, we omit details arguing that the reductions specified are in logarithmic space and occasionally the proof that a given problem is in SL. Note that providing a direct proof for showing a problem is in SL is often difficult.

The naming conventions we use are largely historical and thus there are some inconsistencies. It is difficult to put the problems into a natural order that addresses their historical importance, simplicity, proof order for reductions, and yet is convenient to search the list by. We split the problems up into four categories for organizational purposes: machine simulation, connectivity, graph theory, and miscellaneous. We provide an index of the problems below.

Machine Simulation

- 4.1 Generic Machine Simulation Problem (GMSP)
- 4.2 Symmetric Finite Automaton Nonemptiness (SFAN)
- 4.3 Symmetric Finite Automaton Acceptance (SFAA)

Connectivity

- 5.1 Undirected s-t Connectivity Problem (USTCON)
- 5.2 Undirected Non-s-t Connectivity Problem ($\overline{\text{USTCON}}$)
- 5.3 k Vertex Disjoint Paths (k-PATHS)
- 5.4 Membership in k-Connected Component (Mem$k$CC)
- 5.5 Connected Components Equal (CCE)
- 5.6 Connected Components Even (CCEven)
- 5.7 Spanning Forest Sizes Equal (SFSE)

Graph Theory

- 6.1 Nonbipartite Graph (NBG), 2-Colorability
- 6.2 Comparability Graph (ComG)
- 6.3 Minimum Weight Spanning Forest (MWSF)
- 6.4 Clique Cover-2 (CC-2)
- 6.5 Fixed Edge in Any Cycle (FEC)
- 6.6 Valid Node Ranking (VNR)
- 6.7 Valid Edge Ranking (VER)

Miscellaneous

- 7.1 Exclusive OR 2-Unsatisfiability ($\oplus$2UNSAT)
- 7.2 Exact Cover-2 (EC-2)
- 7.3 Hitting Set-2 (HS-2)
- 7.4 Generalized Word Problem Countably-generated 2 (GWPC(2))

The problem format is based on that employed by Garey and Johnson [7] and Greenlaw, Hoover, and Ruzzo [9].

4 Machine Simulation

4.1 Generic Machine Simulation Problem (GMSP)

Given: A string x, a description $\overline{M}$ of a symmetric Turing machine M, and a natural number s encoded in unary.
Problem: Does M accept x within space $\lceil \log s \rceil$?
Reference: Lewis and Papadimitriou [17], and THIS WORK.
Hint: The required symmetric universal Turing machine U, needed to show the problem is in SL, can be constructed from the deterministic one (see Hopcroft and Ullman [10], for example) and by applying Lemma 1 of [17, page 167]. U copies the current state and symbol of M to one of its work-tapes for decoding instructions and uses x on its input tape as input to M. The state requires space at most $\lceil \log |\overline{M}| \rceil$ since we may assume that the number of states of M is less than $|\overline{M}|$. Thus, to represent one state requires at most $\lceil \log |\overline{M}| \rceil$ space. A similar analysis can be made for the space required for the current symbol. The "input pointer" to x requires $\lceil \log |x| \rceil$ space. Therefore, the total space used by U is

$$\lceil \log |x| \rceil + 2 \cdot \lceil \log |\overline{M}| \rceil + \lceil \log s \rceil$$

which is $O(\log(|x| + |\overline{M}| + s))$. A direct reduction from any language L in SL to GMSP involves outputting the instance of L, a description of the corresponding SL machine N for L, and the space bound for N in unary.

4.2 Symmetric Finite Automaton Nonemptiness (SFAN)

Given: The description $\overline{M}$ of a *symmetric finite automaton*. A symmetric finite automaton $M = (Q, \Sigma, \Delta, s, F)$ is a nondeterministic finite automaton such that whenever $(q_1, \sigma, q_2) \in \Delta$ then so is (q_2, σ, q_1). Note, here there is no notion of "backing up" on the input tape.
Problem: Is $L(M)$ nonempty?
Reference: THIS WORK.
Hint: SFAN is in SL since we can reduce it to USTCON, Problem 5.1.

Given an instance $M = (Q, \Sigma, \Delta, s, F)$ of SFAN form the graph $G = (V, E)$, where $V = Q \cup \{t\}$ and

$$E = \{\{p, q\} \mid \text{ there is a } \sigma \in \Sigma \text{ with } (p, \sigma, q) \in \Delta\} \cup \{\{p, t\} \mid p \in F\}.$$

It is easy to see that $L(M)$ is nonempty if and only if s is connected to t in G.

To show SFAN is SL-hard reduce USTCON to it. Given an instance $G = (V, E)$, s, and t of USTCON define $N = (V, \{\sigma\}, \Delta, s, \{t\})$, where for all $u, v \in V$, $(u, \sigma, v) \in \Delta$ if and only if $\{u, v\} \in E$. Then it is easy to see N is symmetric and that s is connected to t in G if and only if there exists a k, $k \leq |V| - 1$, such that $\sigma^k \in L(N)$.

Remarks: The problem where M is deterministic, SDFAN, is also complete for SL. We can reduce an instance $G = (V, E)$, s, and t of USTCON to SDFAN as follows: form the symmetric deterministic finite automaton $N = (V, \Sigma, \Delta, s, \{t\})$, where $\Sigma = \{\langle u, v\rangle \mid u, v \in V \text{ and } u < v\}$ and for each edge $\{u, v\} \in E$ if $u <_{\text{lex}} v$ add the two transitions (u, σ, v) and (v, σ, u) to Δ where $\sigma = \langle u, v\rangle$ and if $v <_{\text{lex}} u$ add the two transitions as above but instead taking $\sigma = \langle v, u\rangle$. Then s is connected to t in G if and only if $L(N)$ is nonempty. *Reversible finite automata* have been studied in a number of settings each time with a slightly different definition (for example, see Angluin [1] for applications in learning theory and Pin [21, 22] for applications in formal language theory).

4.3 Symmetric Finite Automaton Acceptance (SFAA)

Given: A string x and a description $\overline{M}$ of a symmetric finite automaton (see Problem 4.2 for definitions).

Problem: Does M accept x?

Reference: THIS WORK.

Hint: We reduce SFAA to USTCON, Problem 5.1, to show SFAA is in SL. Given an instance $x = x_1 \cdots x_n$ and $M = (Q, \Sigma, \Delta, s, F)$ of SFAA we define the undirected graph $G = (V, E)$, where

$$V = \{p^i \mid 1 \leq i \leq n; p \in Q\} \cup \{s\} \cup \{t\}.$$

(For each state $p \in Q$ we create n copies $p^1, \ldots, p^n$ as vertices of G and we add a new dummy vertex $t \notin Q$ to V.) and

$$E = \{\{s, p^1\} \mid (s, x_1, p) \in \Delta\} \cup \{\{p^{i-1}, q^i\} \mid (p, x_i, q) \in \Delta, 2 \le i \le n\}$$
$$\cup \{\{f^n, t\} \mid f \in F\}.$$

It is easy to see that $x = x_1 \cdots x_n \in L(M)$ if and only if s is connected to t in G.

For SL-hardness reduce USTCON to SFAA. Given an instance $G = (V, E)$, s, and t of USTCON, define $N = (V, \{\sigma\}, \Delta, s, \{t\})$, where for all $u, v \in V$, $(u, \sigma, v) \in \Delta$ if and only if $\{u, v\} \in E$. In addition, add the transition (t, σ, t) on the final state to Δ. It is easy to see that N is symmetric and that s is connected to t in G if and only if there exists a k, $k \le |V| - 1$, such that $\sigma^k \in L(N)$. Since $(t, \sigma, t) \in \Delta$, we have that s is connected to t in G if and only if $\sigma^{|V|-1} \in L(N)$.

Remarks: We provide an alternative proof of the result below. The idea to prove SFAA is in SL is to stepwise guess an accepting path for x and verify that each step in the path represents a legal move for M. The only nondeterminism in this procedure is the guessing of the path, so we can apply Lemma 1 from [17, page 167] to obtain the required symmetric Turing machine program for SFAA.

For SL-hardness reduce an arbitrary language L in SL to SFAA. Let N be a logarithmic space bounded symmetric Turing machine accepting L. On input y of length n, N has $n^{O(1)}$ possible configurations. Let n^k be a bound on the number of configurations for some appropriate easy to compute constant k. Form the symmetric finite automaton M whose transitions correspond to N's configuration graph. Label each transition by a single symbol, σ. Additionally, incorporate two new states s_1 and s_2. s_1 will be the new initial state of M (and will be final if s was final). Add transitions on σ from s_1 to s_2 and s, from s_2 to s_1 and s, and from s to s_1 and s_2. Then N accepts input y if and only if M accepts input $\sigma^{n^k} \circ \sigma$.

5 Connectivity

5.1 Undirected s-t Connectivity Problem (USTCON)

Given: An undirected graph $G = (V, E)$ and two designated vertices s and t.
Problem: Are s and t connected?
Reference: Lewis and Papadimitriou [17].
Hint: The reduction is from an arbitrary language L in SL. Let M be a logarithmic space bounded symmetric Turing machine accepting L. Given an instance x of L form the configuration graph G of M on input x. Let s (t) be the initial (respectively, unique final) configuration of M. Then M accepts x if and only if there is a path from s to t in G.
Remarks: USTCON motivated Lewis and Papadimitriou to define the complexity class SL. This problem is also called UGAP by many authors (see Jones, Lien, and Laaser [15]). Frequently it is convenient to assume the vertices are numbered 1 through $|V|$ and then take s as 1 and t as $|V|$. Given an undirected graph $G = (V, E)$ and two designated vertices s and t, the problem of determining if the length of a shortest path from s to t is $|V| - 1$ is NL-complete (see [2, page 561]).

5.2 Undirected Non-s-t Connectivity Problem ($\overline{\text{USTCON}}$)

Given: An undirected graph $G = (V, E)$ and two designated vertices s and t.
Problem: Is it the case that s and t are not connected?
Reference: Nisan and Ta-Shma [19].
Hint: Reduce USTCON to $\overline{\text{USTCON}}$. This is the reduction used to show that SL is closed under complement.
Remarks: This problem is also called $\overline{\text{UGAP}}$ by many authors. See Problem 5.1 for additional comments.

5.3 k Vertex Disjoint Paths (k-PATHS)

Given: An undirected graph $G = (V, E)$ and two designated vertices s and t.
Problem: Are there k vertex disjoint paths from s to t?

Reference: Reif [24].
Hint: Observe 1-PATH is USTCON, Problem 5.1. To show the problem is in SL, Reif notes that for any graph $G = (V, E)$ and vertices $s, t \in V$, the k-PATHS instance G, s, and t has a "yes" answer if and only if for all $v_1, \ldots, v_{k-1} \in V - \{s, t\}$, the USTCON instances G', s, and t have "yes" answers, where G' is the graph obtained by deleting vertices $v_1, \ldots, v_{k-1}$ from G.
Remarks: The problem was originally proved complete for SL.

5.4 Membership in k-Connected Component (Mem$k$CC)

Given: An undirected graph $G = (V, E)$, a designated vertex v, and a set of k vertices $v_1, \ldots, v_k$.
Problem: Is v in the *k-connected component* determined by $v_1, \ldots, v_k$? A k-connected component is a maximal *k-connected* subgraph. A graph $H = (W, F)$ is k-connected if for all distinct vertices $w_1, w_2 \in W$, there exist k vertex disjoint paths from w_1 to w_2.
Reference: Reif [24].
Hint: Mem$k$CC is in SL since instance G, v, and $v_1, \ldots, v_k$ is "yes" if and only if

$$\bigwedge_{1 \le i \le k} k\text{-PATHS } G, v, \text{ and } v_i,$$

and Lemma 2.3 applies. For hardness reduce USTCON, Problem 5.1, to Mem$k$CC. Given an instance G, s, and t ask whether G, s, and t is an instance of Mem1CC.
Remarks: The problem was originally proved complete for SL.

5.5 Connected Components Equal (CCE)

Given: Two undirected graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$.
Problem: Is $\#\text{cc}(G_1)$ equal to $\#\text{cc}(G_2)$?
Reference: Nisan and Ta-Shma [19], and THIS WORK.
Hint: First we show that CCE is in SL. By Lemma 2.3 it suffices to show that CCE is in DL^{SL}. Let $\text{NCC}_i(G)$ equal 1 if $\#\text{cc}(G)$ equals i, and 0

otherwise. Consider the (monotone) formula

$$A = \bigvee_{i=1}^{\min\{V_1,V_2\}} (\mathrm{NCC}_i(G_1) \wedge \mathrm{NCC}_i(G_2)).$$

Noting that NCC_i can be computed in SL [19] (see Problem 8.1), it is easy to see that the value of A can be determined by a $\mathrm{DL}^{\mathrm{SL}}$ machine. Since the value of A provides the answer to CCE, we have that CCE is in SL.

Now we show that CCE is SL-hard. For this we logarithmic space many-one reduce $\overline{\mathrm{USTCON}}$, Problem 5.2, to CCE. Given an instance $G = (V, E)$, s, and t of $\overline{\mathrm{USTCON}}$ we form the corresponding instance of CCE as follows: $G_1 = G$ and G_2 given by $V_2 = V \cup \{u \mid \text{where } u \text{ is a new vertex not in } V\}$ and $E_2 = E \cup \{s, t\}$. Dummy vertex u is used for the purpose of adding one more connected component to G_2. It is easy to see that s is not connected to t in G if and only if $\#\mathrm{cc}(G_1)$ equals $\#\mathrm{cc}(G_2)$.

Remarks: Nisan and Ta-Shma show that a variant of CCE is reducible to USTCON.

5.6 Connected Components Even (CCEven)

Given: An undirected graph $G = (V, E)$.
Problem: Is $\#\mathrm{cc}(G)$ even?
Reference: Birgit Jenner, personal communication, 1996.
Hint: CCEven is in SL since CCE, Problem 5.5, is in SL and SL is closed under disjunctive logarithmic space reducibility. For SL-hardness reduce USTCON, Problem 5.1, to CCEven. Let $G = (V, E)$, s, and t be an instance of USTCON. Form the instance H of CCEven consisting of two copies of G and in one of the copies add an edge from s to t. If s is (is not) connected to t in G, then $\#\mathrm{cc}(H) = 2 \cdot \#\mathrm{cc}(G)$ (respectively, $\#\mathrm{cc}(H) = 2 \cdot \#\mathrm{cc}(G) - 1$). So, s is connected to t in G if and only if $\#\mathrm{cc}(H)$ is even.
Remarks: This problem is “between” Problems 5.5 and 8.1.

5.7 Spanning Forest Sizes Equal (SFSE)

Given: Two undirected graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$.
Problem: Is the number of edges in a spanning forest of G_1 equal to the number of edges in a spanning forest of G_2?
Reference: Nisan and Ta-Shma [19].

Hint: Recall an undirected graph G with n nodes has k connected components if and only if a spanning forest of G contains $n-k$ edges. Therefore, it is easy to see that this problem is logarithmic space equivalent to Problem 5.5.

6 Graph Theory

6.1 Nonbipartite Graph (NBG)

Given: An undirected graph $G = (V, E)$.
Problem: Is it the case that G is not bipartite?
Reference: Jones, Lien, and Laaser [15].
Hint: To show the problem is in SL, NBG is reduced to USTCON, Problem 5.1. Let $G = (V, E)$ be an instance of NBG. The idea is to construct a new graph by first forming two copies of each node, call them copy 0 and copy 1. For any edge $\{u, v\} \in E$ connect the 0 copy of u to the 1 copy of v and vice versa. This new graph, G', is not bipartite if and only if there is some node w such that the 0 copy of w is reachable from the 1 copy of w. To take care of the phrase "there is some node w" $|V|$ duplicates of G' are produced and new nodes s and t are introduced. s (t) is connected to the 0 (1) copy of the i-th node in copy i.

For hardness reduce USTCON to NBG. The idea is to make use of the fact that a graph is bipartite if and only if it has no cycle of odd length. Let $G = (V, E)$, s, and t be an instance of USTCON. Let d be a dummy node such that $d \notin V \cup E$. Form the instance $G' = (V', E')$ of NBG, where

$$V' = \{u, u' \mid u \in V\} \cup \{e, e' \mid e \in E\} \cup \{d\}$$

and

$$E' = \{\{u, e\}, \{e, v\}, \{u', e'\}, \{e', v'\} \mid e = \{u, v\} \in E\} \cup \{\{s, s'\}, \{t, d\}, \{t', d\}\}.$$

Then G' contains an odd length cycle if and only if s is connected to t in G.
Remarks: As noted in [15], this is equivalent to asking if G is not 2-colorable; the problem is a special case of *Chromatic Number* (see Garey and Johnson [7, page 191], for example). Since SL equals co-SL, the problems of asking whether G is bipartite or 2-colorable are also SL-complete. Before it was known that SL equals co-SL, Reif observed bipartite graph recognition was complete for his class Π_1CSYMLOG = co-SL [24, Theorem 5.11]. NBG

and Problems 6.4, 7.1, 7.2, and 7.3 were shown to be logarithmic space equivalent to USTCON in [15]. This was *prior* to SL being defined by Lewis and Papadimitriou [17]. Lewis and Papadimitriou observed that the results contained in [15] implied these problems were complete for SL when combined with the facts that USTCON is SL-complete and SL is closed under logarithmic space many-one reducibility.

6.2 Comparability Graph (ComG)

Given: An undirected graph $G = (V, E)$.
Problem: Is G a *comparability graph*? A graph G is a comparability graph if there exists a partial order P on V, say $<_p$, such that $\{u, v\} \in E$ if and only if either $u <_p v$ or $v <_p u$. That is, the edges in E correspond to pairs of elements in V that may be compared.
Reference: THIS WORK.
Hint: Reif originally showed the problem was in Π_1CSYMLOG = co-SL [24]. By Theorem 1.1 this implies ComG is in SL.

For SL-hardness we use the same reduction as given in Problem 6.1 but here we reduce $\overline{\text{USTCON}}$, Problem 5.2, to ComG. The claim is that s is not connected to t in G if and only if G' (see Problem 6.1) is a comparability graph. In order to prove the claim we rely on the following characterization of comparability graphs given by Gilmore and Hoffman [8]: An undirected graph $G = (V, E)$ is a comparability graph if and only if for every cycle C of G, if $\{u, v\} \notin E$ for every pair of vertices u and v at distance two in C, then C has an even number of edges.

First, notice in the graph G' excluding the "last" three edges in E' that each cycle (if any) has an even length. Also observe that both $\{u, u' \mid u \in V\}$ and $\{e, e' \mid e \in E\}$ are independent sets so there are not vertices of distance two apart in any cycle that are adjacent in G'. Finally, in adding the last three edges we create only odd cycles (if any) and none of these cycles have nodes at distance two apart in them adjacent in G'. As in Problem 6.1 s is connected to t in G if and only if there is an odd cycle in G' if and only if G' is not a comparability graph by arguing as above. From this the claim follows and ComG is SL-complete.

6.3 Minimum Weight Spanning Forest (MWSF)

Given: An undirected graph $G = (V, E)$, a designated edge e, and a weight function $w : E \mapsto \{1, 2, 3, \ldots\}$ assigning a distinct weight to each edge.
Problem: Is e in the minimum weight spanning forest of G? (Recall, the minimum weight spanning forest is unique for a weight function that assigns distinct positive integer weights.)
Reference: Stephen Cook; Reif [24].
Hint: Membership in SL follows by a reduction to USTCON, Problem 5.1. The idea is to use the greedy minimum weight spanning forest algorithm. An edge $e = \{u, v\}$ is in the minimum weight spanning forest if and only if u is not connected to v in the graph made up of all edges having lower weight than e (see Nisan and Ta-Shma [19]).

Reduce $\overline{\text{USTCON}}$, Problem 5.2, to MWSF for completeness. Let $G = (V, E)$, s, and t be an instance of $\overline{\text{USTCON}}$. If $\{s, t\} \in G$ output the instance $H = (\{s, t\}, \{\})$, $\{s, t\}$, and w undefined; otherwise form the instance $G' = (V, E \cup \{\{s, t\}\})$, $\{s, t\}$, and w of MWSF, where all edges $e \neq \{s, t\} \in E$ are given unique positive weights from $1, \ldots, |E| - 1$ and $w(\{s, t\}) = |E|$. Then s is not connected to t in G if and only if $\{s, t\}$ is in the minimum weight spanning forest of G'.
Remarks: The problem was shown complete for Π_1CSYMLOG in [24]. This class equals co-SL. At the time it was not known that SL is closed under complementation. The problem where we consider the lexicographically first minimum spanning forest without a weight function is also SL-complete since the ordering of the edges can be considered as unique distinct positive integer weights [19].

6.4 Clique Cover-2 (CC-2)

Given: An undirected graph $G = (V, E)$.
Problem: Is it the case that V cannot be covered by two cliques?
Reference: Jones, Lien, and Laaser [15].
Hint: Use the appropriate reductions given by Karp [16].

6.5 Fixed Edge in Any Cycle (FEC)

Given: An undirected graph $G = (V, E)$ and a designated edge $e = \{u, v\}$ in E.
Problem: Is there a cycle in G that contains e?
Reference: THIS WORK.

Hint: FEC is in SL since we can reduce it to USTCON, Problem 5.1. Given an instance G and $e = \{u, v\}$ of FEC form the graph G' from G by deleting edge e and ask if there is a path from u to v in G'. To show FEC is SL-hard reduce USTCON to FEC. Given an instance $G = (V, E)$, s, and t of USTCON if edge $\{s, t\} \in E$ then form the instance $G' = (V \cup \{u\}, E \cup \{\{u, s\}, \{u, t\}\})$ and $\{s, t\}$ of FEC. Otherwise, just add edge $\{s, t\}$ to G and ask if edge $\{s, t\}$ is in a cycle in this new graph. It is in a cycle if and only if s is connected to t in G.
Remarks: A related problem is to ask whether a designated vertex is in any cycle. This problem is easily seen to be in SL via a $\mathrm{DL}^{\mathrm{FEC}}$ machine. Furthermore, the problem is SL-complete since FEC can be reduced to it. The idea is to consider the dual graph. The *Cycle Free Problem* (CFP) is given an undirected graph to ask whether it is acyclic. Cook and McKenzie proved CFP is DL-complete [4]. They show the problem remains DL-complete if the input graph contains at most one cycle. Recall that a graph is bipartite if and only if it contains no cycle of odd length. It is interesting to compare the results here with those of Problem 6.1. The complement of FEC is also interesting and related to Problem 6.3: there is no cycle in G containing e if and only if every spanning forest of G includes e.

Another related problem called SameCycle, which asks if two nodes are in the same cycle, is formally defined as follows:

Given: an undirected graph $G = (V, E)$ and two designated vertices u and w.

Problem: Is there a *simple cycle* C that contains both u and w? A simple cycle $v_1, \ldots, v_k$ in a graph $G = (V, E)$ is such that $k \geq 3$, $\{v_i, v_{i+1}\} \in E$ for $1 \leq i \leq k-1$, $\{v_k, v_1\} \in E$, and $v_i \neq v_j$ for $1 \leq i \neq j \leq k$.

This problem is known to be NP-complete.

6.6 Valid Node Ranking (VNR)

Given: An undirected graph $G = (V, E)$ and a *node ranking* ρ of G. A node ranking of G is a mapping from V to the positive integers. A node ranking is *valid* if on every simple path between two distinct nodes u and w with $\rho(u) = \rho(w)$ there is a node v such that $\rho(v) > \rho(u)$.
Problem: Is ρ a valid node ranking of G?
Reference: Raymond Greenlaw and Birgit Jenner, personal communication, 1996.

Hint: To prove the problem is in SL, it suffices to show the complementary problem is. The idea is to guess two distinct vertices u and w, and a path P (one vertex at a time) "between" them that has no higher label on it. The highest label encountered thus far on P is recorded. If w is reached, one verifies that $\rho(u) = \rho(w)$ and that the highest value recorded on the guessed path between u and w is less than or equal to $\rho(u)$. The nondeterminism in the algorithm sketched above occurs in guessing the initial u and w, and in guessing a series of next nodes on the path. The remainder of the computation is deterministic and techniques from Lewis and Papadimitriou [17] may be employed to convert such an algorithm into a symmetric Turing machine program. If the ranking is not (is) valid, then there is some (respectively, is no) sequence of guesses that leads the program to accept.

To prove VNR is SL-hard, we reduce $\overline{\text{USTCON}}$, Problem 5.2, to it. Given an instance $G = (V, E)$, s, and t of $\overline{\text{USTCON}}$ construct an instance of VNR using the same graph and defining ρ as follows: $\rho(s) = \rho(t) = |V| - 1$, and $\rho(u) = i$ if $u \neq s, t$ and u is the i-th vertex in "lexicographic order" of $(V - \{s, t\})$ in the given encoding of G. That is, ρ assigns distinct labels less than $|V| - 1$ to all nodes except for s and t to which it assigns the same label $|V| - 1$. It is easy to see that s is not connected to t in G if and only if ρ is a valid node ranking of G.

Remarks: See, for example, de la Torre, Greenlaw, and Przytycka [5] for a discussion of the *Node Ranking Problem*, its sequential and parallel time complexities, and its applications. When restricted to trees, VNR is complete for DL (Raymond Greenlaw and Birgit Jenner, personal communication, 1996). The problem can be seen to be in DL using Euler tour techniques and may be shown DL-hard via a reduction from *Undirected Forest Accessibility*. Cook and McKenzie showed the latter problem was DL-complete [4, page 388].

6.7 Valid Edge Ranking (VER)

Given: An undirected graph $G = (V, E)$ and an *edge ranking* ρ of G. An edge ranking of G is a mapping from E to the positive integers. An edge ranking is *valid* if on every simple path between two distinct edges e_1 and e_2 with $\rho(e_1) = \rho(e_2)$ there is an edge e such that $\rho(e) > \rho(e_1)$.
Problem: Is ρ a valid edge ranking of G?
Reference: Raymond Greenlaw and Birgit Jenner, personal communication, 1996.
Hint: A similar proof to that given for Problem 6.6 yields the result.

Remarks: See, for example, de la Torre, Greenlaw, and Schäffer [6] for a discussion of the *Edge Ranking Problem*, its sequential and parallel time complexities, and its applications. When restricted to trees, VER is complete for DL (Raymond Greenlaw and Birgit Jenner, personal communication, 1996), see remarks for Problem 6.6. Note, in general the complexities of node and edge ranking problems do not seem to be the same (see [5, 6] for example).

7 Miscellaneous

7.1 Exclusive OR 2-Unsatisfiability ($\oplus$2UNSAT)

Given: A formula F that is the conjunction of a set of clauses $C_1, \ldots, C_m$, where each C_i consists of either one literal or is the EXCLUSIVE OR of two literals.
Problem: Is it the case that F is not satisfiable?
Reference: Jones, Lien, and Laaser [15].
Hint: Reduce NBG to $\oplus$2UNSAT and vice versa to show that $\oplus$2UNSAT is SL-hard and in SL, respectively (edges and clauses directly correspond).
Reference: 2UNSAT, where each clause contains two literals, is complete for NL (see Papadimitriou [20, Theorem 9.1 on page 184] and its corollary on page 185, for example).

7.2 Exact Cover-2 (EC-2)

Given: A "universe" set U and a family of n sets $S_i \subseteq U$ with the property that every element in U appears at most twice in the list $S_1, \ldots, S_n$.
Problem: Is it the case that there is no subfamily $S'_1, \ldots, S'_m$ with $m \leq n$, such that $S'_i \cap S'_j = \emptyset$ for $1 \leq i \neq j \leq m$ and $S'_1 \cup \cdots \cup S'_m = U$?
Reference: Jones, Lien, and Laaser [15].
Hint: Use the appropriate reductions given by Karp [16].

7.3 Hitting Set-2 (HS-2)

Given: A "universe" set U and a family of n sets $S_i \subseteq U$ with the property that $|S_i| \leq 2$ for $1 \leq i \leq n$.
Problem: Is it the case that there is no subset H of U such that $|H \cap S_i| = 1$

for all i?
Reference: Jones, Lien, and Laaser [15].
Hint: Use the appropriate reductions given by Karp [16].

7.4 Generalized Word Problem Countably-generated 2 (GWPC(2))

Given: A set $W = \{w_1, \ldots, w_t\}$ of distinct, *reduced* words over $\Sigma_n = \{s_0, s_1, \ldots, s_{n-1}\}$ of *length* 2 and a designated *word* w *over* Σ_n of length 2. Let Σ be any finite set of symbols and $\Sigma^{-1} = \{s^{-1} \mid s \in \Sigma\}$. A word over Σ is a string of symbols from $\Sigma \cup \Sigma^{-1}$; the length of the word is the length of the string. A word w over Σ is reduced if there are no consecutive occurrences in w of the symbols s and s^{-1}, or s^{-1} and s, for any $s \in \Sigma$. See [26] for further details.
Problem: Is $w \in \langle W \rangle$? For a finite set of reduced words $W = \{w_1, \ldots, w_t\}$ over Σ, $\langle W \rangle$ denotes the group generated by $w_1, \ldots, w_t$.
Reference: Stewart [26].
Hint: The logical characterization of SL by Immerman as (FO + pos-STC) [11] is used to show GWPC(2) is in SL. For the completeness proof see [26, Corollary on page 267].
Remarks: GWPC(2) is called the *Generalized Word Problem for finitely-generated subgroups of Countably-generated free groups with generators of length 2* in [26]. For general k certain variants of the problem are P-complete [27]. See Stewart [26, 27] for additional details and definitions.

8 Open Problems

In this section we list a number of open problems. For each problem we provide a definition, remarks, and a reference.[1] The goal in each case is to show that the problem is SL-hard under $\leq_{\log}^{m}$ reducibility. In some cases the problem is known to be in SL and in other cases it is not. The problems are not always stated as decision problems. We provide an index of the problems below.

[1] When to the best our knowledge we were the first to ask whether a given problem is SL-hard, we cite THIS WORK as a reference.

Open Problems

8.1 Number of Connected Components (NCC)
8.2 Chordal Graph (ChordalG)
8.3 Interval Graph (IntervalG)
8.4 Split Graph (SplitG)
8.5 Permutation Graph (PermG)
8.6 Unary $0-1$ Knapsack (UK)
8.7 Unary Knapsack with Signed Repetition (UKSR)
8.8 Bounded Degree Planarity (BDP)

8.1 Number of Connected Components (NCC)

Given: An undirected graph $G = (V, E)$ and a natural number k.
Problem: Is $\#\mathrm{cc}(G)$ equal to k?
Reference: FOLKLORE.
Remarks: NCC is in SL by [19]. Nisan and Ta-Shma show NCC is logarithmic space many-one reducible to USTCON. Note, NCC is logarithmic space many-one reducible to CCE by a straightforward reduction. If G and k comprise an instance of NCC, then take $G_1 = G$ and $G_2 = (\{1, \ldots, k\}, \emptyset)$. It is easy to see NCC is complete for SL under logarithmic space Turing reducibility by a reduction from CCE, Problem 5.5.

8.2 Chordal Graph (ChordalG)

Given: An undirected graph $G = (V, E)$.
Problem: Is G a *chordal graph*? A graph G is a chordal graph if every cycle C of length greater than three has a *chord*. A chord is an edge connecting two nonconsecutive vertices in C.
Reference: THIS WORK.
Remarks: Reif originally showed the problem was in Π_1CSYMLOG = co-SL [24]. By Theorem 1.1 this implies ChordalG is in SL.

8.3 Interval Graph (IntervalG)

Given: An undirected graph $G = (V, E)$.
Problem: Is G an *interval graph*? A graph G is an interval graph if its vertices can be put into a one-to-one correspondence with a set of intervals of the real line such that two vertices are adjacent if and only if their corre-

sponding intervals overlap.
Reference: THIS WORK.
Remarks: Reif originally showed the problem was in Π_1CSYMLOG = co-SL [24]. It is known that a graph G is an interval graph if and only if G is a chordal graph and $\overline{G}$ is a comparability graph. The result that IntervalG is in SL follows from the fact that ComG and ChordalG, Problems 6.2 and 8.2 respectively, are in SL.

8.4 Split Graph (SplitG)

Given: An undirected graph $G = (V, E)$.
Problem: Is G a *split graph*? A graph $G = (V, E)$ is a split graph if the vertices can be partitioned so that $V = V_1 \cup V_2$ and the graphs induced by the vertices in V_1 and V_2 using the edges from E are an independent set and a complete graph, respectively.
Reference: THIS WORK.
Remarks: Reif originally showed the problem was in Π_1CSYMLOG = co-SL [24]. It is known that a graph G is a split graph if and only if both G and $\overline{G}$ are chordal graphs. The result that SplitG is in SL follows from the fact that ChordalG, Problem 8.2, is in SL. We may ask two questions, one about G and the other about $\overline{G}$, written deterministically to the oracle tape and apply Lemma 2.3.

8.5 Permutation Graph (PermG)

Given: An undirected graph $G = (V, E)$.
Problem: Is G a *permutation graph*? A graph $G = (\{v_1, \ldots, v_n\}, E)$ is a permutation graph if there is a permutation π of $\{1, \ldots, n\}$ such that $\{v_i, v_j\} \in E$ if and only if $(i - j)(\pi^{-1}(i) - \pi^{-1}(j)) < 0$.
Reference: THIS WORK.
Remarks: Reif originally showed the problem was in Π_1CSYMLOG = co-SL [24]. It is known that a graph G is a permutation graph if and only if both G and $\overline{G}$ are comparability graphs. PermG can thus be logarithmic space many-one reduced to ComG. The reduction takes a graph G and outputs a new graph H consisting of a copy of G and a copy of $\overline{G}$ with the nodes relabeled. Note, H is a comparability graph if and only if both G and $\overline{G}$ are. The result that PermG is in SL then follows from the fact that ComG, Problem 6.2, is in SL.

8.6 Unary $0-1$ Knapsack (UK)

Given: A positive integer 0^y and a sequence $0^{y_1}, \ldots, 0^{y_n}$ of positive integers represented in unary.
Problem: Is there a sequence of $0-1$ valued variables $x_1, \ldots, x_n$ such that

$$y = \sum_{j=1}^{n} x_j \times y_j?$$

Reference: Monien and Sudborough [18] and Cook [3].
Remarks: This problem is known to be in NL [18]. Cook cites a personal communication with Martin Tompa that UK is unlikely to be NL-complete [3, page 9]. It is not known if this problem is in SL. See Problem 8.7 for related problems.

8.7 Unary Knapsack with Signed Repetition (UKSR)

Given: A positive integer 0^y and a sequence $0^{y_1}, \ldots, 0^{y_n}$ of positive integers represented in unary.
Problem: Is there a sequence of integers $x_1, \ldots, x_n$ such that

$$y = \sum_{j=1}^{n} x_j \times y_j?$$

Reference: Jenner [13].
Remarks: Jenner shows the problem is in SL by reducing it to USTCON. Also see [13] for several other interesting variants of the Knapsack Problem that are complete for NL and for several that may be complete for SL.

8.8 Bounded Degree Planarity (BDP)

Given: An undirected graph G whose vertices have bounded degree.
Problem: Is G planar?
Reference: THIS WORK.
Remarks: Reif originally showed the problem was in Π_3CSYMLOG [24]. Since the symmetric complementation hierarchy collapses using the results of Nisan and Ta-Shma [19], this places the problem in SL.

Acknowledgements

We thank Birgit Jenner and Luca Trevisan for helpful discussions about problems in SL, and Ricard Gavaldà and José Balcázar for a number of useful comments on a draft of this paper. This research was conducted while Ray was on sabbatical at the Universitat Politècnica de Catalunya in Barcelona; the department's hospitality is greatly appreciated.

References

[1] Dana Angluin. Inference of reversible languages. *Journal of the ACM*, 29(3):742–765, 1982.

[2] Allan Borodin, Stephen A. Cook, Patrick W. Dymond, Walter L. Ruzzo, and Martin Tompa. Two applications of inductive counting for complementation problems. *SIAM Journal on Computing*, 18(3):559–578, June 1989. See also 18(6): 1283, December 1989.

[3] Stephen A. Cook. A taxonomy of problems with fast parallel algorithms. *Information and Control*, 64(1–3):2–22, January/February/March 1985.

[4] Stephen A. Cook and Pierre McKenzie. Problems complete for deterministic logarithmic space. *Journal of Algorithms*, 8:385–394, 1987.

[5] Pilar de la Torre, Raymond Greenlaw, and Teresa Przytycka. Optimal tree ranking is in NC. *Parallel Processing Letters*, 2(1):31–41, March 1992.

[6] Pilar de la Torre, Raymond Greenlaw, and Alejandro A. Schäffer. Optimal edge ranking of trees in polynomial time. *Algorithmica*, 13(6):592–618, 1995.

[7] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[8] P. C. Gilmore and A. J. Hoffman. A characterization of comparability graphs and of interval graphs. *Canadian Journal of Mathematics*, 16:539–548, 1964.

[9] Raymond Greenlaw, H. James Hoover, and Walter L. Ruzzo. *Limits to Parallel Computation: P-Completeness Theory.* Oxford University Press, 1995.

[10] John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, 1979.

[11] Neil Immerman. Languages that capture complexity classes. *SIAM Journal on Computing*, 16(4):760–778, 1987.

[12] Neil Immerman. Nondeterministic space is closed under complementation. *SIAM Journal on Computing*, 17(5):935–938, October 1988.

[13] Birgit Jenner. Knapsack problems for NL. *Information Processing Letters*, 54(3):169–174, 1995.

[14] David S. Johnson. A catalog of complexity classes. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume A: Algorithms and Complexity, chapter 2, pages 67–161. M.I.T. Press/Elsevier, 1990.

[15] Neil D. Jones, Y. E. Lien, and William T. Laaser. New problems complete for nondeterministic log space. *Mathematical Systems Theory*, 10:1–17, 1976.

[16] Richard M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–104. Plenum Press, New York, 1972.

[17] Harry R. Lewis and Christos H. Papadimitriou. Symmetric space-bounded computation. *Theoretical Computer Science*, 19(2):161–187, August 1982.

[18] Burkhard Monien and Ivan Hal Sudborough. In Ron V. Book, editor, *Formal Language Theory.* Academic Press, 1980.

[19] Noam Nisan and Amnon Ta-Shma. Symmetric *logspace* is closed under complement. *Chicago Journal of Theoretical Computer Science*, 1995(1), June 1995.

[20] Christos H. Papadimitriou. *Computational Complexity.* Addison-Wesley, 1994.

[21] Jean-Eric Pin. On the languages accepted by finite reversible automata. In *Automata, Languages, and Programming: 14th International Colloquium*, number 267 in Lecture Notes in Computer Science, pages 237–249, Berlin, 1987. Springer-Verlag.

[22] Jean-Eric Pin. On reversible automata. In *Proceedings of the First LATIN Conference*, number 583 in Lecture Notes in Computer Science, pages 401–416, Saõ-Paulo, 1992. Springer-Verlag.

[23] Alexander A. Razborov. Lower bounds for deterministic and nondeterministic branching programs. In Lothar Budach, editor, *Fundamentals of Computation Theory: 8th International Conference, FCT '91*, volume 529 of *Lecture Notes in Computer Science*, pages 47–60, Gosen, Germany, September 1991. Springer-Verlag.

[24] John H. Reif. Symmetric complementation. *Journal of the ACM*, 31(2):401–421, April 1984.

[25] Walter L. Ruzzo, Janos Simon, and Martin Tompa. Space-bounded hierarchies and probabilistic computations. *Journal of Computer and System Sciences*, 28(2):216–230, April 1984.

[26] Iain A. Stewart. Complete problems for symmetric logspace involving free groups. *Information Processing Letters*, 40(5):263–267, December 1991.

[27] Iain A. Stewart. Refining known results on the generalized word problem for free groups. *International Journal of Algebra and Computation*, 2(2):221–236, June 1992.

[28] Róbert Szelepcsényi. The method of forced enumeration for nondeterministic automata. *Acta Informatica*, 26:279–284, 1988.