Symmetric Complementation

JOHN H. REIF

Harvard University, Cambridge, Massachusetts

Abstract. This paper introduces a new class of games called symmetric complementing games. These games are interesting since their related complexity classes include many well-known graph problems: Finding minimum spanning forests; k-connectivity and k-blocks; and recognition of chordal graphs, comparability graphs, interval graphs, split graphs, permutation graphs, and constant valence planar graphs. For these problems probabilistic sequential algorithms requiring simultaneously logarithmic space and polynomial time are given Furthermore, probabilistic parallelism algorithms requiring simultaneously logarithmic time and a polynomial number of processors are also given.

Categories and Subject Descriptors: F.1.2 **[Computation by Abstract Devices]**: Modes of Computation—*alternation and nondeterminism; parallelism; probabilistic computation*; F.1.3 **[Computation by Abstract Devices]**: Complexity Classes—*complexity hierarchies; reducibility and completeness; relations among complexity classes*, F.2.2 **[Analysis of Algorithms and Problem Complexity]**: Nonnumerical Algorithms and Problems—*computations on discrete structures*

General Terms: Algorithms, Theory

Additional Key Words and Phrases: Symmetric computation, parallelism, alternation, probabilistic computation, graph problems, connectivity, planarity

1. *Introduction*

1.1. PREVIOUS WORK. In the previous decade, considerable success has been made in the design of time-efficient sequential algorithms for many combinatorial problems on graphs (such as spanning trees, k-connectivity for $k = 1$, 2, and 3 [18], and planarity testing [19] using the technique of depth-first search. Also, breadth-first search has been used for time-efficient sequential algorithms for other graph problems. By applying well-known simulation results (e.g., [12], we can derive parallel space-efficient algorithms from these sequential time-efficient algorithms. Also, parallel time has been related to sequential space (by the simulation results, of, e.g., [8, 12]). It is intriguing therefore to ask

(i) Is there a general graph search technique that yields sequential algorithms with optimal space and (either by simulation results or directly) also yields parallel algorithms with optimal time?

We require that these algorithms be *reasonable*: that a sequential algorithm with space bound $S(n)$ uses no more than $2^{O(S(n))}$ sequential time (hence, if $S(n) =$

Presented at the 14th Annual Symposium on Theory of Computing, San Francisco, Calif., April 1982. This work was supported in part by the National Science Foundation Grant NSF-MCS79-21024, the Office of Naval Research Contract N00014-80-C-0647.

Author's address: Aiken Computation Laboratory, Harvard University, Cambridge, MA 02138.

$O(\log n)$, then $n^{O(1)}$ sequential time will be used) and that a parallel algorithm with time bound $T(n)$ use no more than $2^{O(T(n))}$ processors (again, note that $T(n) = O(\log n)$ implies $n^{O(1)}$ processors will be used). Hence, certain probabilistic Turing machines (TMs) have a time bound doubly exponential in their space bound and are therefore not reasonable.

The depth-first search techniques appear not to be applicable to (i) owing to their sequential nature. A possible technique for solving a graph problem is to reduce the problem efficiently to Boolean transitive closure, for which there is a known $O(\log^2 n)$ parallel time algorithm [7]. Using this technique, breadth-first search can be done in parallel time $O(\log n)^2$. Savage and Ja'Ja' [29], Hirschberg [16], and Hirschberg et al. [17] use parallel breadth-first search and parallel Boolean transitive closure to do planarity testing and to solve connectivity problems in $O(\log^2 n)$ parallel time. However, Boolean transitive closure has no known algorithm with less than $\Omega(\log^2 n)$ parallel time bound, and this bound seems very difficult to improve, whereas we show here the planarity testing and connectivity problems actually have $O(\log n)$ probabilistic parallel time algorithms.

A related problem is

(ii) Is there a logic, in which a significant class of combinatorial problems may be succinctly expressed, and such that validity of sentences in the logic may be decided efficiently or even optimally (with respect to sequential space or parallel time)?

A logic satisfying the conditions of (ii) could be used as the kernel of a language for parallel programming, where programs may be "compiled" into time-optimal code for parallel machines.

1.2. Our Results. This paper proposes solutions to (i) and (ii). A space-efficient sequential *probabilistic search* technique was first introduced by Aleliunas et al. [2] to test connectivity. We generalize the probabilistic search technique to yield optimal algorithms (in sequential space and also parallel time) for a complexity class Σ_*CSYMLOG, which contains many important combinatorial problems. Furthermore, we propose a restricted quantified Boolean logic Σ_*QBF$\oplus$ as a solution to (ii).

Our results are actually stated in a more general setting. This paper introduces a class of 1-player games of perfect information, which we call *complementing games*; the player is allowed moves that complement the value of successive plays. A complementing game is *symmetric* if all noncomplement moves are reversible (i.e., form a symmetric relation). These games are naturally related to a class of machines we call *symmetric complementing machines*. Symmetric nondeterministic machines were introduced in [24]; they can be viewed as a restricted class of our symmetric complementing machines with complement moves allowed only on termination. Of particular interest is the complexity class Σ_*CSYMLOG, which contains the outcome problem of symmetric complementing games with constant complement bound with game positions encoded in log space, and next move relations computable in log space. We show that the decision problem for a restricted quantified Boolean logic Σ_*QBF$\oplus$ is complete in Σ_*CSYMLOG. We also show that Σ_*CSYMLOG contains many well-known and common combinatorial problems:

(1) minimum spanning forests,
(2) k-connectivity and k-blocks;

and also the recognition problems for many classes of graphs:

(3) planar graphs of constant valence,
(4) chordal graphs,
(5) comparability graphs,
(6) interval graphs,
(7) split graphs,
(8) permutation graphs.

We present a *probabilistic algorithm* (this is an algorithm that makes probabilistic choices [27], but with *no* assumptions about the probability distribution of the inputs) for recognizing the languages of Σ_*CSYMLOG within space $O(\log(n))$ and simultaneous time $n^{O(1)}$, with error probability $<\epsilon$ for any given ϵ, $0 < \epsilon < 1$. As a consequence, problems (1)–(8) can be done probabilistically in space $O(\log(n))$ and simultaneously within polynomial time. Previously, the best known algorithms for problems (1)–(3) required deterministic space $\Omega(\log^2 n)$ [21], and for problems (4)–(8) space $\Omega(n)$.

Also, we give a *probabilistic parallel algorithm* (which employs the Hardware Modification Machines of [6], with independent distributed probabilistic choice) for recognizing the languages of Σ_*CSYMLOG within parallel time $O(\log n)$ and error probability $<\epsilon$, for any given ϵ, $0 < \epsilon < 1$. Thus, we also have parallel time $O(\log n)$ algorithms for problems (1)–(8). Our parallel algorithms require only a small polynomial number of processors. The best previously known parallel algorithms for problems (1) and (2) required parallel time $\Omega(\log^2 n)$ [16, 17, 29], problem (3) also required parallel time $\Omega(\log^2 n)$ [20], and we know of no previous parallel algorithms for problems (4)–(8). Furthermore, we show (by a counting argument) that for each input length $n \geq 0$, the probabilistic choice can be eliminated in both our sequential and parallel algorithms. This does not affect the efficiency of the algorithms but makes our algorithms nonuniform (i.e., we have a different algorithm for each input length).

1.3. Organization of This Paper. Section 2 defines complementing games and machines, and complexity notation. Section 3 provides a sequential decision algorithm for the problems of Σ_*CSYMLOG, which runs in probabilistic space $O(\log n)$ and can be modified to run in nonuniform deterministic space $O(\log n)$. Section 4 gives a probabilistic parallel algorithm that can simulate any sequential space $S(n)$-bounded probabilistic computation within parallel time $O(S(n))$. We also show in Section 4 that we can eliminate probabilistic choices in our parallel algorithm without degrading its running time, but introducing nonuniformity. Section 5 introduces our logic Σ_*QBF$\oplus$ and shows its decision problem is complete in Σ_*CSYMLOG. We also show in Section 5 that various combinatorial problems (including (1)–(8) mentioned above) are in Σ_*CSYMLOG.

2. *Preliminary Definitions*

2.1. Symmetric Relations. Let $R \subseteq D \times D$ be a relation on domain D. Let its inverse be $R^- = \{(b, a) \mid aRb\}$. R is *symmetric* if $R = R^-$. R is *deterministic* if, for all $a \in D$, there is at most one $b \in D$ such that aRb.

2.2. Complementing Games. A (1-player) *complementing game* consists of a quadruple $G = (P, W, \vdash, \vdash_c)$ where

(i) P is the set of *positions*; P is assumed to be a set of strings over a finite alphabet;

(ii) $W \subseteq P$ are the *winning positions*;
(iii) $\vdash \subseteq P \times P$ is the *next move relation*;
(iv) $\vdash_c \subseteq \vdash$ are the *complement moves*.

Fix an *initial position* $p_0 \in P$. A *move* is a pair $(p, p') \in \vdash$. The move (p, p') is an *initial move* (*complement move*) if $p = p_0((p, p') \in \vdash_c)$. A complement game G is *symmetric* if $\vdash_s = \vdash - \vdash_c$ is a symmetric relation (see 2.1). Let a *play* be a maximal length sequence of positions $p_0, p_1, \ldots$ where p_0 is the initial position, and $p_{i-1} \vdash p_i$ for $i = 1, 2, \ldots$ (we allow a trivial play p_0). For any finite $k \geq 0$, let p_0 have *complement bound* k if any play from p_0 has $<k$ complements, ignoring the initial move (ignoring the initial move allows us to maintain the duality between Σ_k and π_k). Suppose p_0 has finite complement bound k. Let OUTCOME(p_0) = **true** if there exists a finite play prefix $p_0, p_1, \ldots, p_j$, with no complement moves and where either $p_j \in W$ or there exists at least one complement move from p_j and OUTCOME(p') = **false** for each complement move $(p_j, p') \in \vdash_c$. Otherwise OUTCOME(p_0) = **false**.

2.3. MACHINE DEFINITIONS.[1] Let a *complementing* (*Turing*) *machine* be a 9-tuple $M = (\Sigma, \Gamma, b, t, Q, q_0, Q_A, \delta, \delta_c)$ where

Σ	is the finite *input alphabet*;
Γ	is the finite *tape alphabet*, with $\Sigma \subseteq \Gamma$;
$b \in \Gamma - \Sigma$	is the distinguished *blank symbol*;
t	is the *number of two-way infinite tapes*, where tape 1 is the input tape;
Q	is the finite *state set*;
$q_0 \in Q$	is the *initial* state;
$Q_A \subseteq Q$	are the *accepting states*;
$\delta \subseteq (Q \times (\Sigma^2 \times \{\textbf{left}, \textbf{right}\}) \times (\Sigma^2 \times \{\textbf{left}, \textbf{right}\})^{t-1})^2$	is the *transition relation*;
$\delta_c \subseteq \delta$	are the *complement transitions*.

The syntax of δ is slightly nonstandard so that we can easily define symmetric machines in a manner similar to [24]. (However, the semantics are not essentially different from the standard definitions.)

Suppose transition $((q, a_1b_1m_1, \ldots, a_tb_tm_t), (q', a_1'b_1'm_1', \ldots, a_t'b_t'm_t')) \in \delta$ is taken. Then the previous state was q and the new state is q'. Each tape $i \in \{1, \ldots, t\}$ moves its head one cell in direction m_i', and m_i is the reverse of direction m_i'. If $m_i' =$ **right**, then previously the head of tape i was scanning symbol a_i and "peeking" at symbol b_i located one cell to the right; in the new configuration these symbols a_ib_i are replaced by symbols $a_i'b_i'$ and the head is scanning symbol b_i'. The case m_i' = **left** is similar, except the head was previously scanning over symbol b_i while "peeking" at symbol a_i located one cell to the left; afterward, the head is scanning symbol a_i'. Let M be *symmetric* if noncomplement transitions $\delta_s = \delta - \delta_c$ are a symmetric relation. Let $\mathcal{I}$ be the *configurations* of M, defined in the usual way for TMs. We may extend δ in the usual way to the *next move relation* $\vdash \subseteq \mathcal{I} \times \mathcal{I}$. Let $\vdash_c \subseteq \vdash$ be the next moves, which are complements. Note that if M is symmetric, then the relation $\vdash_s = \vdash - \vdash_c$ is symmetric. Let $W \subseteq \mathcal{I}$ be the accepting configurations. $G_M = (\mathcal{I}, W, \vdash, \vdash_c)$ is the *computation game* of M.

[1]See conclusion of this paper for a discussion of previous and equivalent machine definitions for symmetric alternating machines.

Given an input string $\omega \in \Sigma^n$, we let the initial configuration $I_0(\omega)$ have the initial state q_0 and all tapes blank except that the input tape contains ω surrounded by infinite strings of blank symbols (since $b \notin \Sigma$, no endmarkers are required). The input tape head initially scans the first symbol of ω. $I_0(\omega)$ is the initial position of G_M. *Computation sequences* of M are plays of G_M from $I_0(\omega)$. Suppose M has a finite complement bound from $I_0(\omega)$. M *accepts* ω if OUTCOME($I_0(\omega)$) = **true**. If M has no complement moves, then M is a *nondeterministic machine*; furthermore, if $\vdash_s$ is also symmetric, then M is a *symmetric nondeterministic machine* as defined by Lewis and Papadimitriou [24].

2.4. Complexity Classes. Let complementing machine M have *space bound* $S(n)$ (*complement bound* $K(n)$) if, on any input of length $n \geq 0$, each computation sequence has no more than $S(n)$ nonblank cells on any work tape in each configuration (fewer than $K(n)$ complements on any computation sequence ignoring the initial moves).

Note that nondeterministic and co-nondeterministic machines have complement bound 1. For notational simplicity, we define a complementing machine with complement bound 0 to be a deterministic TM. Let $\mathscr{M}$ be a class of complementing machines. Let $\mathscr{M}$SPACE($S(n)$) be the languages accepted by those machines in $\mathscr{M}$ with space bound $S(n)$. Let $\Sigma_k\mathscr{M}$SPACE($S(n)$) ($\Pi_k\mathscr{M}$SPACE($S(n)$)) be the languages accepted by those machines in $\mathscr{M}$ with space bound $S(n)$, complement bound k, and no complement moves (only complement moves, respectively) for the initial moves. Let $\Sigma_*\mathscr{M}$SPACE($S(n)$) = $\bigcup_{k\geq 0}\Sigma_k\mathscr{M}$SPACE($S(n)$); that is, the machines operate in some constant number of complements, regardless of the input length.

In the context of complexity classes, we let D denote the class of deterministic TMs, let N denote the nondeterministic TMs, and let NSYM be the symmetric nondeterministic machines. Let C be the complementing machines, and let CSYM be the symmetric complementing machines. For example, the complexity class NSYMSPACE($S(n)$) = $\{L \mid L$ is accepted by a symmetric nondeterministic machine with space $S(n)\}$ previously investigated by Lewis and Papadimitriou [24]. The complexity class $\Sigma_{K(n)}$CSYMSPACE($S(n)$) = $\{L \mid L$ is accepted by a symmetric complementing machine with space bound $S(n)$, complement bound $K(n)$, and no complement initial moves$\}$ is of central importance to this paper. For notational simplicity, let NSYMLOG = NSYMSPACE(log(n)), and CSYMLOG = CSYMSPACE(log(n)).

Let $L_1 \leq_{\log} L_2$ denote that language L_1 can be many–one reduced in deterministic log space to language L_2. Let L_1 be $\leq_{\log}$ *equivalent* to L_2 if $L_1 \leq_{\log} L_2$ and $L_2 \leq_{\log} L_1$. Let L_2 be *complete* in a family of languages $\mathscr{L}$ if $L_2 \in \mathscr{L}$ and $L_1 \leq_{\log} L_2$ for each $L_1 \in \mathscr{L}$. Note that if $S(n) \geq \log n$, $L_1 \leq_{\log} L_2$, and $L_2 \in$ CSYMSPACE($S(n)$), then $L_1 \in$ CSYMSPACE($S(n)$), by Proposition 2.2.

2.5. Preliminary Results for Symmetric Complementing Machines. Let M be a symmetric complementing machine that accepts language $L \subseteq \Sigma^*$. If we augment M by an initial complementing move, then the resulting machine accepts $\Sigma^* - L$. On the other hand, if we remove an initial complementing move of M, then the resulting machine also accepts $\Sigma^* - L$. Hence, we have

Proposition 2.1. *For any* $L \subseteq \Sigma^*$, $L \in \Sigma_{K(n)}$ *CSYMSPACE*($S(n)$) *iff* $\Sigma^* - L \in \Pi_{K(n)}$ *CSYMSPACE*($S(n)$).

Lewis and Papadimitriou [24] show DSPACE($S(n)$) $\subseteq$ NSYMSPACE($S(n)$). Thus, NSYMSPACE($S(n)$), for $S(n) \geq \log n$, is closed under many–one deterministic log-space reductions. Their proof easily extends to CSYMSPACE($S(n)$).

PROPOSITION 2.2. *If L has a many-one deterministic log-space reduction to a language in $\Sigma_{K(n)}$ CSYMSPACE($S(n)$) and $S(n) \geq \log n$, then $L \in \Sigma_{K(n)}$ CSYMSPACE($S(n)$).*

Also, since any symmetric complementing machine is a complementing machine:

PROPOSITION 2.3. *DSPACE($S(n)$) $\subseteq \Sigma_{K(n)}$ CSYMSPACE($S(n)$) $\subseteq \Sigma_{K(n)}$ CSPACE($S(n)$)*

(Note that space-bounded complementing machines accept the same languages as space-bounded alternating machines. Both complementing machines and ordinary alternating machines without resource bounds accept any arithmetic set. However, this is not relevant to this paper.)

3. *A Space-Efficient Decision Algorithm for Symmetric Complementing Machines*

We give a $O(S(n)K(n))$ space sequential algorithm for recognizing the languages of $\Sigma_{K(n)}$ CSYMSPACE($S(n)$). The algorithm is probabilistic (see Sections 3.1 and 3.2), though we show it can be made deterministic by introducing nonuniformity (see Section 3.3).

3.1. PROBABILISTIC SEQUENTIAL MACHINES. We define a *probabilistic* TM to be a multitape deterministic Turing machine PM with a special read-only, one-way tape (distinct from the input and work tapes) containing an infinite binary sequence. The contents of this "random bitvector" tape are chosen randomly on each execution of PM. Let Σ be the input alphabet of PM and let $L \subseteq \Sigma^*$. For any $\epsilon(n)$, $0 \leq \epsilon(n) < 1$ say PM *recognizes L within error $\epsilon(n)$* if for all $\omega \in \Sigma^n$,

C1. $\omega \in L$ implies Pr(PM accepts ω) $\geq 1 \geq \epsilon(n)$,
C2. $\omega \notin L$ implies Pr(PM accepts ω) $< \epsilon(n)$.

To justify this definition, we note that Adleman's [1] definition of acceptance of probabilistic machines is similar to ours, except $\epsilon(n) = \frac{1}{2}$ in C1 and he strengthens condition C2 by requiring that $\omega \notin L$ imply PM does not accept ω on any probabilistic choice. Many probabilistic algorithms in number theory satisfy this more restrictive property, but it is too restrictive for many of the applications in this paper. On the other hand, Gill [13] defines acceptance of probabilistic machines with the max of the error of acceptance and rejection less than $\frac{1}{2}$ and called the polynomial time-bounded class BPP.

Note that a probabilistic machine may not be *reasonable* in the sense defined in the introduction (since Gill [13] gives a probabilistic machine with space bound $S(n)$ and expected time bound $2^{2^{O(S(n))}}$); however, the probabilistic machine implementing the PROB-SEARCH algorithm of Section 3.2 will be reasonable.

3.2. PROBABILISTIC SIMULATION OF SPACE-BOUNDED SYMMETRIC COMPLEMENTING MACHINES. We show

THEOREM 3.1. *For any $\epsilon(n)$, $0 < \epsilon(n) < 1$, there is a probabilistic TM that recognizes $L \in \Sigma_{K(n)}$ CSYMSPACE($S(n)$) within given error $\epsilon(n)$ and space $O(K(n)(S(n) + \log d(n)))$ and time $(d(n)2^{O(S(n))})^{K(n)}$, where $d(n) = K(n)(O(S(n)) + \log(O(K(n))) - \log \epsilon(n)$.*

Note that if $\epsilon(n)$ is constant, then we require space $O(K(n)S(n))$ and time $2^{O(K(n)S(n))}$. Thus

COROLLARY 3.1. *For each constant ϵ, $0 < \epsilon < 1$, and $L \in \Sigma_*$ CSYMLOG, there is a probabilistic TM that recognizes L within error ϵ and simultaneous space $O(\log n)$ and time $n^{O(1)}$.*

Although Theorem 3.1 suffices for our applications, we also show its space bounds can be improved.

THEOREM 3.2. *For each $L \in \Sigma_{K(n)}$ CSYMSPACE($S(n)$), there is a probabilistic TM that recognizes L within error $\epsilon(n)$ and space $O(K(n)(S(n) + \log(S(n) + \log d(n))))$.*

Our probabilistic search technique will utilize the following result:

LEMMA 3.1. [2]. *Let $G = (V, E)$ be any undirected, connected graph. Let a* random walk *r in G from any vertex $v \in V$ be constructed from trivial path v by repeatedly extending the front end of r by adding a random edge of E, which is connected to the current front end vertex of r. Let r be a random walk of length $2|E|(|V| - 1)$. Then $Pr(r$ visits all vertices in $V) \geq \frac{1}{2}$.*

Lewis and Papadimitriou [24] observe that this lemma immediately implies a space $O(S(n))$ probabilistic algorithm for NSYMSPACE($S(n)$). A generalized probabilistic search technique is used here to decide acceptance of symmetric complementing machines.

PROOF OF THEOREMS 3.1 AND 3.2. Let M be a symmetric complementing machine as defined in Section 2.2. We assume M has complement bound $K(n) \geq 1$ and constructible space bound $S(n) \geq \log n$ (otherwise, we use the standard technique of trying $S(n) = \log n$, $1 + \log n$, ... to the construction given below). Let $\mathscr{I}$ be the set of configuration of M and let $W \subseteq \mathscr{I}$ be the accepting configurations. Let $\vdash$ be the next move relation of M. Let $\vdash_c$, $\vdash_s \subseteq \vdash$ be the complement and noncomplement moves of $\vdash$, respectively. Fix some $n \geq 0$. Let $\mathscr{I}' \subseteq \mathscr{I}$ be the configurations that have $\leq S(n)$ nonblank cells on each work tape.

We define a recursive procedure that takes as input a configuration $I \in \mathscr{I}'$. Also, the procedure has a global variable t (which determines the procedure's probability of success).

```
procedure PROB-SEARCH_t(I)
  begin
    local integer i, set COMP
    i ← 0
    while i ≤ t do
      begin
        if I is accepting then return true
        COMP ← {I' ∈ 𝒥' | I ⊢_c I'}
        if COMP ≠ ∅ then
          if PROB-SEARCH_t(I') = false for all I' ∈ COMP then return true
        choose a random I' from {I' ∈ 𝒥' | I ⊢_s I'}
        I ← I'
        i ← i + 1
      end
    return false
  end
```

For each k, $1 \leq k \leq K(n)$ let $\mathscr{I}_k \subseteq \mathscr{I}'$ be the configurations that also have complement bound k.

Let $\epsilon_{k,t}$ be the max probability that PROB-SEARCH$_t(I)$ = **false** for any $I \in \mathscr{I}_k$ such that OUTCOME(I) = **true**. Let $\bar{\epsilon}_{k,t}$ be the max probability that PROB-SEARCH$_t(I)$ = **true** for any $I \in \mathscr{I}_k$ such that OUTCOME(I) = **false**. Thus, $\epsilon_{k,t}$ and $\bar{\epsilon}_{k,t}$ are the upper bounds of error probabilities for rejection and acceptance, respectively.

LEMMA 3.2. *There are constants b, $c \geq 1$, dependent only on M, such that for any $d \geq 1$, $k \geq 1$,*

$$max(\epsilon_{k,t}, \bar{\epsilon}_{k,t}) \leq k2^{-d}(tb)^{k-1}$$

where $t = 2dc^{S(n)}$.

PROOF. Let $\vdash'_s = (\mathscr{I}' \times \mathscr{I}') \cap \vdash_s$. Since M is symmetric $(\mathscr{I}', \vdash'_s)$ is an undirected graph. There exists a constant $b \geq 1$ that is an upper bound on the number of next configurations $\{I' \mid I \vdash I'\}$ possible from any given configuration I. Also there exists a constant $c_1 \geq 1$ such that $|\mathscr{I}'| \leq c_1^{S(n)}$. Thus, there exists a constant $c_2 \geq 0$ such that $|\vdash'_s| \leq c_2^{S(n)}$. Let $c = c_1 \cdot c_2$ so $t \geq 2d\,|\mathscr{I}'|\,|\vdash'_s|$.

For each pair of configurations $I, I' \in \mathscr{I}_k$ for which there is a noncomplementing computation sequence from I to I', by Lemma 3.1, we have Pr(r visits I') $\geq 1 - 2^{-d}$ for a random walk r in $(\mathscr{I}_k, \vdash'_s)$ starting at I and of length $2d\,|\mathscr{I}'|\,|\vdash'_s|$.

Now we prove Lemma 3.2 by induction on k. For $k = 1$, we show $\max(\epsilon_{1,t}, \bar{\epsilon}_{1,t}) \leq 2^{-d}$. The upper bound on the error probability for rejection and acceptance from any $I \in \mathscr{I}_1$ with no complement next move is $\leq 2^{-d}$ and 0, respectively. Thus, the total worst case error probability for rejection and acceptance from any $I \in \mathscr{I}_1$ is $\leq 2^{-d}$ and $\leq 2^{-d}$, respectively.

Since there are at most tb direct calls to PROB-SEARCH during a single execution of the body of the PROB-SEARCH procedure, for $k > 1$ we have

$$\epsilon_{k,t} \leq 2^{-d} + tb\bar{\epsilon}_{k-1} \quad \text{and} \quad \epsilon_{k,t} \leq tb\epsilon_{k-1}.$$

By the induction hypothesis,

$$\max(\epsilon_{k-1,t}, \bar{\epsilon}_{k-1,t}) \leq (k-1)2^{-d}(tb)^{k-2}.$$

Hence

$$\begin{aligned}\max(\epsilon_{k,t}, \bar{\epsilon}_{k,t}) &\leq 2^{-d} + tb\bar{\epsilon}_{k-1} \\ &\leq 2^{-d} + (k-1)2^{-d}(tb)^{k-1} \\ &\leq k2^{-d}(tb)^{k-1}.\end{aligned}$$

□

Let L be the language accepted by M. Suppose we are given some error function $\epsilon(n)$, $0 < \epsilon(n) < 1$. Let PM be the probabilistic TM that on input $\omega \in \Sigma^n$, computes PROB-SEARCH$_{t(n)}(I_0(\omega))$ and accepts iff the result is **true**, provided that $d(n) = K(n)\log(t(n)b) - \log \epsilon(n)$ and $t(n) = 2d(n)c^{S(n)}$. (Note that both $d(n)$ and $t(m)$ are decreasing functions of $\epsilon(n)$.) By Lemma 3.2, PM recognizes L within error $\epsilon_{K(n)} \leq \epsilon(n)$. Furthermore, PM has time bound $O(t(n))^{K(n)} = (d(n)2^{O(S(n))})^{K(n)}$. PM has space bound $O(K(n)(S(n) + \log d(n)))$, since we must store $b = O(1)$ configurations of size $O(S(n))$, and a "time counter" requiring space $\log t(n) = O(S(n) + \log d(n))$ to implement each of the $K(n)$ recursive cells. Thus we have proved Theorem 3.1. □

Although Theorem 3.1 is good enough for our applications to Σ_*CSYMLOG, it is nevertheless interesting to observe that we may decrease the space bound by using a trick due to Gill [13]. To avoid storing the "time counter" in the procedure PROB-SEARCH$_t$, we instead sample a random bit on each iteration of the **while** statement. If at any time there have been $\log t$ consecutive zero's chosen, then we immediately exit the **while** statement. This test replaces the test $(i \leq t)$ in the original text of PROB-SEARCH$_t$. To achieve error $\epsilon_{K(n)} \leq \epsilon(n)$, we must only

increase $d(n)$ by a factor of $1/(1 - \log\exp(1))$. Only $O(K(n)(S(n) + \log\log t(n))) = O(K(n)(S(n) + \log(S(n) + \log d(n))))$ space is required by this method (but note that we no longer have a deterministic time bound). Thus we have proved Theorem 3.2. □

3.3. ELIMINATING PROBABILISTIC CHOICES. Let a *nonuniform deterministic* TM be a deterministic TM augmented with a special read-only tape, called the *advice tape*, whose contents are fixed for all inputs of the same length n, but which may have different contents for distinct input lengths n and n'. (Neither the input tape nor the advice tape is considered in the space bound of this machine.) This nonuniform machine has *advice bound* $A(n)$ if, on inputs of length n, the advice tape has $A(n)$ cells (see [23]). We show

THEOREM 3.3. *Each* $L \in \Sigma_{K(n)}$ *CSYMSPACE*$(S(n))$ *is accepted by a nonuniform deterministic TM within space bound* $O(K(n)S(n))$, *time bound* $2^{O(K(n)S(n))}$, *and advice bound* $2^{O(S(n))}$.

COROLLARY 3.3. *Each* $L \in \Sigma_*$*CSYMLOG is accepted by a nonuniform deterministic TM within simultaneous space bound* $O(\log n)$, *time bound* $n^{O(1)}$, *and advice bound* $n^{O(1)}$.

PROOF OF THEOREM 3.3. We require a technical graph-theoretic result.

Let $G = (V, E)$ be a undirected regular graph, with valence b. We assume G has a fixed adjacency list representation, so for each vertex v we have a list $l(v)$ of vertices adjacent to v. Given a string $U \in \{1, \ldots, b\}^*$ and a vertex v, let $U(G, v)$ be the path $v = v_0, \ldots, v_{|\sigma|}$ such that v_i is the $U(i)$ element of list $l(v_{i-1})$ for $i = 2, \ldots, |\sigma|$. Let $\mathscr{G}_{n,b}$ be the class of all undirected, regular graphs with $\leq n$ vertices and valence b. Let $U \in \{1, \ldots, b\}^*$ be (n, b)-*universal* if for each graph $G \in \mathscr{G}_{n,b}$ and each vertex v of G, $U(G, v)$ visits all the vertices of G.

LEMMA 3.3 [2]. *For each* $b \geq 1$, *there is a* $c(b)$ *such that for each* $n \geq 0$ *there is a* (n, b)*-universal string* $U_{n,b}$ *of length* $\leq c(b)n^3\log n$.

Let M be a symmetric complementing machine of Section 3.2 with complement bound $K(n)$ and space bound $S(n)$. Let $(\mathscr{I}', \vdash'_s)$ be the undirected graph of valence b defined in the proof of Lemma 3.2. Clearly, we can add redundant transitions so that $(\mathscr{I}', \vdash'_s)$ is regular with valence b. Let NONUNIFORM-SEARCH$_t(I)$ be the deterministic procedure derived from PROB-SEARCH$_t(I)$ of Section 3.2 by using the $(|\mathscr{I}'|, b)$-universal string $U_{|\mathscr{I}'|,b}$ in place of probabilistic choice, for choosing the configurations to be explored in $(\mathscr{I}', \vdash'_s)$.

By the proof of Lemma 3.2, there exists some $c_1 \geq 1$ such that $|\mathscr{I}'| \leq c_1^{S(n)}$. Let $t(n) = c(b)(S(n)\log c_1)c_1^{3S(n)}$. Then Lemma 3.3 immediately implies

LEMMA 3.4. *For each input string* $\omega \in \Sigma^n$,

$$\textit{NONUNIFORM-SEARCH}_{t(n)}(I_0(\omega)) = \textbf{true} \quad \textit{iff} \quad M \textit{ accepts } \omega.$$

This procedure may be implemented by a nonuniform deterministic TM with space bound $O(K(n)S(n))$, time bound $t(n)^{K(n)} = 2^{O(K(n)S(n))}$, and advice bound $t(n) = 2^{O(S(n))}$. Thus, we have proved Theorem 3.3. □

Note that Reif [28] also describes how to eliminate probabilistic choice from probabilistic parallel computations with bounded error, and Adleman [1] and Bennett and Gill [3] describe how to construct circuits from sequential probabilistic computations with bounded error. However, none of these results imply Theorem 3.3.

4. *A Parallel Algorithm*

4.1. THE HARDWARE MODIFICATION MACHINE. Our parallel machine model is the hardware modification machine (HMM) of [8] (though we consider probabilistic and nonuniform variants of it below). The HMM was invented as the parallel analog of the storage modification machine of [31]. The HMM seems to be the simplest possible parallel machine with modifiable storage structure, and the HMM can be simulated within real time, with the same number of processors, by many other such parallel machines, including the P-RAM of [12] (this P-RAM model was assumed for the parallel graph algorithms of [20]), the PRAM of [30], and the SIMDAGs of [15]. A survey of parallel machine models and related complexity results are given in [6].

Intuitively, a HMM consists of a finite collection of deterministic finite state machines, which we call *processors*. The state transition functions of these processors are identical. Each processor also contains the same fixed, finite number of *input* and *output connections* for transmission of values, from a finite alphabet, between processors. On each step (the state transitions of the processor are synchronous), a processor will read the values of its input connections, which were set by its neighboring processors on the last step, write new values on each of its output connections (only one process is associated with each output connection), and enter a new state. In addition, a processor may reconnect any input connection to any machine that can be reached by a path of length ≤ 2 from the previous input connection. Also, a processor may reconnect an input connection to a new processor (with the same finite state control, initialized in some given state and with all input connections directed to its creator).

Given an input string $\omega \in \Sigma^n$, we assume the *initial configuration* of the HMM consists of a chain of $n + 1$ identical processors $PP_0, PP_1, \ldots, PP_n$, each in the same initial state and each with input connections connected back to itself, except that each PP_{i-1}, for $0 < i \leq n$, has a distinguished input connection to PP_i where the value output by PP_i is the ith symbol of the input string ω. (This initialization scheme is somewhat simpler than that defined by Dymond and Cook [8] but yields the same technical results of interest here.) The HMM *accepts* ω if PP_0 ever enters a distinguished accepting state q_A.

The *time bound* $T(n)$ (*processor bound* $P(n)$) of the HMM is the maximum number of steps (processors, respectively) taken on any accepting computation for any input of length n. Generally we assume the HMM is *uniform*: The processors have the same finite state transition function for all input strings. However, we consider in Section 4.4 *nonuniform HMMs*, which must only have the same finite state transitions for all input strings of the same length. The *advice bound* $A(n)$ of a nonuniform HMM is the number of tuples defining the processor state transition function for input of length n.

4.2. THE PROBABILISTIC HMM. In addition to the above for a uniform HMM, suppose we allow each processor PP_i *probabilistic choice* by providing a special read-only register r_i, which is set randomly to 0 or 1 each step, with each step independent of each other. Let PPM be the resulting *probabilistic* HMM. PPM recognizes *language* $L \subseteq \Sigma^*$ *within error* $\epsilon(n)$, $0 \leq \epsilon(n) < 1$, if for all $\omega \in \Sigma^n$,

C1. $\omega \in L$ implies $\Pr\{\text{PPM accepts } \omega\} \geq 1 - \epsilon(n)$,
C2. $\omega \notin L$ implies $\Pr\{\text{PPM accepts } \omega\} < \epsilon(n)$.

(Note that the conditions C1, C2 for probabilistic recognition are identical to those given in 3.1. Reif [28] gives complexity bounds for various other probabilistic

parallel machines and for both Adleman's [1] and also Gill's [13] bounded error definitions of probabilistic acceptance. If Adleman's definition of acceptance is used, then we can eliminate probabilistic choice in our parallel machines by introducing nonuniformity without any increase in parallel time. On the other hand, if Gill's bounded error definition of probabilistic acceptance is used, then we show in [28] that probabilistic parallel space $S(n)$ contains parallel space $S(n)$ with nondeterministic choice.)

4.3. PARALLEL SIMULATION OF PROBABILISTIC SEQUENTIAL COMPUTATIONS. Dymond and Cook [8] prove that

THEOREM 4.1. *If $L \in DSPACE(S(n))$ for $S(n) \geq \log n$, then L is recognized by a (deterministic) HMM in simultaneous parallel time bound $O(S(n))$ and processor bound $2^{O(S(n))}$.*

We generalize their results to probabilistic computations.

THEOREM 4.2. *Let PM be a probabilistic TM with space-bound $S(n) \geq \log n$ and time bound $T(n)$. Suppose for some $\epsilon(n)$, $0 < \epsilon(n) < 1$, $L \subseteq \Sigma^*$ is recognized by PM within error $\epsilon(n)$. Then there is a probabilistic HMM that recognizes L within error $\epsilon(n)$ and with $O(S(n) + \log T(n))$ parallel time and utilizes $T(n) \cdot 2^{O(S(n))}$ processors. Furthermore, this HMM is uniform.*

PROOF. Fix some input string $\omega \in \Sigma^n$ and let $I_0(\omega)$ be the initial configuration of PM. Let $\mathscr{I}'$ be the configurations of PM using $\leq S(n)$ tape cells. Clearly there exists a constant $c > 0$ such that $|\mathscr{I}'| \leq c^{S(n)}$. We assume $S(n)$ and $T(n)$, are constructible (otherwise we can in parallel use a diagonalization of $S(n) = 0, 1, \ldots$ and $T(n) = 0, 1, \ldots$).

Our simulating probabilistic HMM, which we call PPM, will utilize a processor $PP_{I,t}$ for each t, $0 \leq t \leq T(n)$ and $I \in \mathscr{I}'$. These processors can be created in binary tree fashion within $O(\log(T(n)|\mathscr{I}'|))$ time. Each processor $PP_{I,t}$ chooses a configuration I' randomly from those allowed from configuration I by PM. $PP_{I,t}$ then makes a distinguished *jump* connection to processor $PP_{I',t+1}$. These connections can be made in time $O(\log(|\mathscr{I}'|))$, again using binary trees for indexing. Thereafter, each process $PP_{I,t}$ repeatedly connects its jump connection to that which was its jump connection of distance 2 in the previous step. These steps are executed synchronously by all the processes, and the HMM is allowed to halt and accept only when process $PP_{I_0(\omega),0}$ has a jump connection to a process $P_{I,t}$, where I is an accepting configuration of PM.

Suppose $I_0, I_1, \ldots$ is an execution sequence of M, with particular probabilistic choices r. Suppose also that the RAMs of PPM make particular probabilistic choices r', such that $PP_{I_t,t}$ initially sets its jump connection to process $PP_{I_{t+1},t+1}$ for $t = 0, 1, \ldots, T(n) - 1$. Then it is easy to verify that PPM accepts ω (when making probabilistic choices r') iff PM accepts ω (when making probabilistic choices r. Since r and r' are chosen randomly, it follows that

$$\Pr\{\text{PM accepts } \omega\} = \Pr\{\text{PPM accepts } \omega\}.$$

Furthermore, if PPM accepts ω, then there is a path $PP_{I_0,0}, P_{I_1,1}, \ldots, P_{I_t,t}$ induced by the initial jump connections such that $I_0(\omega) = I_0, I_1, \ldots, I_t$ is an accepting computation of M, and $t \leq T(n)$. On each iteration, this path's length decreases by a factor of $\frac{1}{2}$. Thus, PPM accepts within parallel time

$$O(\log(T(n)|\mathscr{I}'|)) = O(\log(T(n)2^{O(S(n))})) = O(S(n) + \log T(n)). \qquad \square$$

Combining Theorems 3.1 and 3.2, we have

THEOREM 4.3. *For any $S(n) \geq \log n$, $K(n) \geq 1$, and for each $\epsilon(n)$, $0 < \epsilon(n) < 1$ and $L \in \Sigma_{K(n)}CSYMSPACE(S(n))$, there is a probabilistic HMM that recognizes L within error $\epsilon(n)$, with parallel time bound $O(K(n)(S(n) + \log d(n)))$ and processor bound $(d(n)2^{O(S(n))})^{K(n)}$ where $d(n)$ is defined as in Theorem* 3.1.

Note that if $\epsilon(n)$ is constant, then HMM has parallel time bound $O(K(n)S(n))$ and processor bound $2^{O(K(n)S(n))}$. Thus

COROLLARY 4.3. *For each constant ϵ, $0 < \epsilon < 1$, and $L \in \Sigma_* CSYMLOG$, there is a probabilistic HMM that recognizes L within error ϵ and with parallel time $O(\log n)$ and $n^{O(1)}$ processor bound.*

4.4. A NONUNIFORM PARALLEL ALGORITHM. By Theorems 3.3 and 4.1, we can eliminate probabilistic choice in our parallel algorithm, by introducing nonuniformity.

THEOREM 4.4. *For each $L \in \Sigma_{K(n)}CSYMSPACE(S(n))$ with $S(n) \geq \log n$, L is accepted by a nonuniform HMM with parallel time bound $O(K(n)S(n))$, processor bound $2^{O(K(n)S(n))}$, and advice bound $2^{O(S(n))}$.*

COROLLARY 4.4. *Each $L \in \Sigma_* CSYMLOG$ is accepted by a nonuniform HMM with simultaneous parallel time bound $O(\log n)$, processor bound $n^{O(1)}$ and advice bound $n^{O(1)}$.*

5. *Computational Problems in $\Sigma_* CSYMLOG$*

5.1. SYMMETRIC COMPLEMENTING GAMES. Let $G = (P, W, \vdash, \vdash_c)$ be a complementing game. Let the next move relation $\vdash$ be *log space* if there is a deterministic log space *next move transducer*, which, given any position $p \in P$, outputs $\{p' \mid p \vdash p'\}$. Let G have *position length bound* $S(n)$ (*complement bound* $K(n)$) if, for each position $p \in P$ of length n, all positions reachable from p have length not more than $S(n)$ (there are fewer than K complement moves on any play from p, respectively). The *outcome problem* for G is given $p \in P$, next move transducer $\vdash$, and recognizer for the winning positions W, compute OUTCOME(p).

Let CYSMGAMES($S(n)$, $K(n)$) be the class of languages that are outcome problems for symmetric complementing games with position length bound $S(n)$, complement bound $K(n)$, and next moves and winning positions recognizable in $O(\log n)$ space.

By our definition of complementing machines, we have

THEOREM 5.1. *For $S(n) \geq \log n$, $CSYMGAMES(S(n), K(n))$ is complete for the class of languages accepted by symmetric complementing machines with space bound $S(n)$ and complement bound $K(n)$.*

COROLLARY 5.1. *The outcome problems $\bigcup_{k \geq 0}$ $CSYMGAMES(\log n, k)$ are complete for $\Sigma_* CSYMLOG$.*

5.2. QBF$\oplus$. Given a set X of Boolean variables, let literals $(X) = X \cup \{\neg x \mid x \in X\} \cup \{$ **true**, **false**$\}$. Let CNF$\oplus$ be the set of Boolean formulas consisting of a conjunction of clauses, each clause consisting of the exclusive-OR $l \oplus l'$ of two literals l, l'. Note that $l \oplus l'$ is equivalent to $(\neg l) \oplus (\neg l')$.

Jones et al. [22] and Lewis and Papadimitriou [24] show the problem of testing CNF$\oplus$ unsatisfiability is deterministic log-space complete in NSYMLOG. Let Σ_kQBF$\oplus$ and Π_0QBF$\oplus$ be the truth values {**true**, **false**}. Inductively, let Σ_kQBF$\oplus$

be the set of quantified Boolean formulas F of the form $(\exists X)C_1 \wedge \cdots \wedge C_m$ where X is a set of Boolean variables and each clause C_i is either of form $l \oplus l'$ or of form $l \vee F'$ where $l, l' \in$ literals(X) and F' is a formula of Π_{k-1}QBF$\oplus$. Also, let Π_kQBF$\oplus$ be the set of quantified Boolean formulas F of the form $(\forall X)C_1 \vee \cdots \vee C_m$ where each clause C_i is of the form $l \oplus l'$ or of the form $l \wedge F'$ where $l, l' \in$ literals(X) and formula F' must be in Σ_{k-1}QBF$\oplus$. Let Σ_*QBF$\oplus = \bigcup_{k \geq 0} \Sigma_kQBF\oplus$. Note that all variables are bound in QBF$\oplus$ formulas.

THEOREM 5.2. *For all $k \geq 0$, the truth problem for Σ_kQBF$\oplus$ is complete in Π_kCSYMLOG, and the truth problem for Π_kQBF$\oplus$ is complete in Σ_kCSYMLOG.*

PROOF OF THEOREM 5.2. Our proof requires a technical Lemma. This Lemma is an easy generalization of a result of [22], which characterized truth of CNF$\oplus$ formulas.

LEMMA 5.1. *Let F be a formula of Σ_kQBF$\oplus$. F is false iff there exists a sequence of literals $l_0, \ldots, l_j$ such that $l_0 \oplus \neg l_1, \ldots, l_{j-1} \oplus \neg l_j$ are equivalent to clauses of F and $l_0 = \neg l_j$ or both* (1) *and* (2) *hold.*

(1) $l_0 =$ **true** *or $l_0 \vee F'$ is a clause of F where F' is a false formula in Π_{k-1}QBF$\oplus$.*
(2) $l_j =$ **false** *or $\neg l_j \vee F''$ is a clause of F where F'' is a false formula in Π_{k-1}QBF$\oplus$.*

It will also be useful to note that

PROPOSITION 5.1. *If F is a formula of Π_kQBF$\oplus$, then $\neg F$ is equivalent to a formula $\hat{F}$ of Σ_kQBF$\oplus$, where $\hat{F}$ is formed by switching the quantification symbols $\forall$, $\exists$ and also switching the logical connectives $\vee$, $\wedge$ in F. So F is* **false** *iff $\hat{F}$ is* **true**.

PROOF OF THEOREM 5.2 BY INDUCTION ON k. Π_0QBF$\oplus$ and Σ_0QBF$\oplus$ can easily be shown complete in Π_0CSYMLOG = Σ_0CSYMLOG = DSPACE($\log n$).

Suppose for some $k \geq 1$ the theorem holds for all $k' < k$. Let F be a Σ_kQBF$\oplus$ formula of length n. To decide F, we play a symmetric complementing game. Let the player begin by choosing a sequence of literals $l_0, \ldots, l_j$ such that $l_0 \oplus \neg l_1, \ldots, l_{j-1} \oplus \neg l_j$ are equivalent to clauses of F. Note that only the first literal and last literal need be stored, and this requires $O(\log n)$ space. This choice sequence is reversible since $(l_{i-1} \oplus \neg l_i) \equiv (l_i \oplus \neg l_{i-1})$. The player enters the accepting state (and thus wins) if either $l_0 = \neg l_j$ in both cases (1), (2) of Lemma 5.2 holds. This may require deciding formulas F', F'' of Π_{k-1}QBF$\oplus$. To do this, we allow the player two simultaneous complement moves from the current position. In these complement moves, we let the player test whether both F' and F'' are **false**. By the induction hypothesis, these tests are in Π_{k-1}CSYMLOG. Thus the symmetric complementing game can be implemented by a symmetric complementing machine with complement bound k and space bound $O(\log n)$. By Lemma 5.2, the player wins iff F is **false**. We have thus shown that testing falsehood of formulas in Σ_kQBF$\oplus$ is in Σ_kCSYMLOG. By Proposition 2.1, testing truth of formulas in Σ_kQBF$\oplus$ is in Π_kCSYMLOG.

Now let M be a symmetric complementing machine with complement bound k and space bound $\log n$. Let $\vdash_s$ be the noncomplement moves of M. For each k', $1 \leq k' \leq k$, let $\mathscr{I}_{k'}$ be the set of configurations of M with $\leq \log n$ nonblank cells per work tape and for which there is a complement bound of k'. Let $\mathscr{C}_{k'}$ be the configurations of $\mathscr{I}_{k'}$ that have a complement as a next move. By the induction hypothesis we can assume for each $I \in \mathscr{C}_{k-1}$ a formula $F'(I)$ of Π_{k-1}QBF$\oplus$ such that $F'(I)$ is **false** iff OUTCOME(I^*) = **false** for each complement move (I, I') from I.

For each $I \in \mathscr{I}_k$ we assume a distinct variable x_I. Let $X = \{x_I \mid I \in \mathscr{I}_k\}$. Let W' be configurations of M that are accepting and have $\leq \log n$ nonblank cells per work tape. For each $I \in W'$, let g_I be the formula $x_I \oplus \mathbf{true}$. Thus, g_I is **true** iff $x_I = \mathbf{false}$. For each $I \in \mathscr{G}_{k-1}$, let g_I be the formula $(\neg x_I) \vee F'(I)$. For each $I \in \mathscr{I}_k$ and $J \in W' \cup \mathscr{G}_{k-1}$, let $f_{I,J} = (\mathbf{true} \oplus \neg x_I) \wedge (\bigwedge_{I_1 \vdash_{s,k} I_2} x_{I_1} \oplus \neg x_{I_2}) \wedge g_J$, where $\vdash_{s,k} = \vdash_s \cap (\mathscr{I}_k \times \mathscr{I}_k)$. Thus, $(\exists X) f_{I,J}$ is **false** iff there exists a computation sequence $I_1, \ldots, I_j$ such that $I_1 = I$, $I_j = J$, and OUTCOME$(J) = \mathbf{true}$.

Now for each $I = \mathscr{I}_k$ and $J \in W' \cup \mathscr{G}_{k-1}$, let x_I^J be a new distinct variable and let $f_{I,J}'$ be derived from $f_{I,J}$ by substituting $x_{I'}^J$ for each instance of variable $x_{I'}$ for each $I' \in \mathscr{I}_k$. Let $x' = \{x_I^J \mid I \in \mathscr{I}_k$ and $J \in W' \cup \mathscr{G}_{k-1}\}$. For each $I \in \mathscr{I}_k$, let $F(I) = (\exists X') \bigwedge_{J \in W' \cup \mathscr{G}_{k-1}} f_{I,J}'$. Clearly $F(I)$ is in Σ_kQBF$\oplus$ and, furthermore, $F(I)$ is **false** iff OUTCOME$(I) = \mathbf{true}$. Hence $F(I_0(\omega))$ is **false** iff M accepts ω.

We have thus shown that the invalidity problem for Σ_kQBF$\oplus$ is complete in Σ_kCSYMLOG. By Propositions 2.1 and 5.1, the truth problem for Π_kQBF$\oplus$ is complete in Σ_kCSYMLOG. □

5.3. k-CONNECTIVITY. Given a graph $G = (V, E)$ and vertices $u, v \in v$, let k-PATHS(G, u, v) be the problem: Do there exist k paths from u to v that are mutually vertex disjoint? The problem 1-PATHS is commonly called the UGAP problem.

THEOREM 5.3. *UGAP is complete in NSYMLOG.*

PROOF [24]. Given an undirected graph G of n vertices with distinguished vertices u, v, we nondeterministically traverse a path in G from u and accept if the vertex v is reached. This can easily be done by a nondeterministic machine in space $O(\log n)$ to store the currently visited vertex. But this nondeterministic machine can be made symmetric since any edge can be traversed in both directions.

On the other hand, suppose M is a symmetric, nondeterministic machine with $\log n$ space bound and input string $\omega \in \Sigma^n$. Let $\mathscr{I}$ be the configurations of M, with space $\leq \log n$, let $\vdash_s \subseteq \mathscr{I} \times \mathscr{I}$ be the nondeterministic moves of M, and let $W \subseteq \mathscr{I}$ be the accepting configurations. We construct an undirected graph with vertices $V = \mathscr{I} \cup \{I_f\}$ where $I_f \notin \mathscr{I}$, and edges $E = \{\{I, I'\} \mid I \vdash_s I'\} \cup \{\{I, I_f\} \mid I \vdash_s I'$ for some $I' \in W\}$. Then M accepts ω iff there is a path in (V, E) from $I_0(\omega)$ to I_f. □

By Theorems 2.1 and 5.3,

COROLLARY 5.3. *The complement of the UGAP problem is complete in* Π_1*CSYMLOG.*

THEOREM 5.4. *For each $k \geq 1$, k-PATHS is complete in NSYMLOG.*

PROOF. By Menger's Theorem [4] for any graph $G = (V, E)$ and vertices $u, v \in V$, k-PATHS$(G, u, v) \leftrightarrow (\forall x_1, \ldots, x_{k-1} \in V - \{u, v\})$UGAP$(G', u, v)$ where G' is derived from G by deleting vertices $x_1, \ldots, x_{k-1}$ and all edges connected to these vertices. By Theorems 5.2 and 5.3, we can construct in deterministic log space a formula in Π_1CNF$\oplus$ that is **false** iff k-PATHS(u, v). Thus, by Theorem 5.2, k-PATHS is complete in NSYMLOG. □

Let a graph $G = (V, E)$ be *k-connected* if for all distinct vertices $u, v \in V$, there exists k paths from u to v that are vertex disjoint. Matula [25] defines a *k-block* of G to be a maximal k-connected subgraph of G.

(Note: To facilitate planarity testing, McLane [26] and Hopcroft and Tarjan [19] define "triconnected" components somewhat differently from 3-blocks. However, McLane's components are homeomorphic to the 3-blocks.)

By Menger's Theorem, any two k-blocks intersect at no more than $k-1$ vertices. Thus, some k vertices suffice to uniquely determine any k-block of G. Let k-BLOCK$(G, x, \{v_1, \ldots, v_k\})$ be the problem: Is vertex x in the k-block of G determined by $\{v_1, \ldots, v_k\}$?

COROLLARY 5.4. *k-BLOCK is complete in NSYMLOG.*

PROOF. By Menger's Theorem,

$$k\text{-BLOCK}(G, x, \{v_1, \ldots, v_k\}) \leftrightarrow \bigwedge_{1 \le i \le k} k\text{-PATHS}(G, x, v_i).$$

Again, we may apply Theorems 5.2 and 5.4 to show there is a formula in Π_1QBF$\oplus$ with **false** iff k-BLOCK$(G, x, \{v_1, \ldots, v_k\})$. □

5.4. MINIMUM SPANNING FORESTS. Here we show the Π_1CSYMLOG contains the problem of recognizing an edge of a (unique) minimum spanning forest.

Let $G = (V, E)$ be an undirected graph with a mapping $W: E \to \mathbb{N}^+$ labeling the edges with distinct positive integers. Consider the following well-known greedy algorithm for constructing a minimum weight spanning forest of G:

Input graph $G = (V, E)$ and edge weighting W.
(1) sort the edges $E = \{e_1, \ldots, e_m\}$ so that $W(e_i) < W(e_{i+1})$ for $i = 1, \ldots, m-1$.
(2) $SF \leftarrow \emptyset$
(3) **for** $i = 1$ **to** m **do**
  if $SF \cup \{e_i\}$ contains no cycles, **then** $SF \leftarrow SF \cup \{e_i\}$
Return (V, SF).

Note that the minimum spanning forest output by this algorithm is *unique* for a fixed W (even though, in general, there may exist many minimum spanning forests of a given graph). Let SPANNING-EDGE(G, W, e) be **true** if $e \in SF$ and **false** otherwise.

THEOREM 5.5. *SPANNING-EDGE is complete in Π_1CSYMLOG.*

PROOF. Let $e = \{u, v\}$ be an edge of $G = (V, E)$ and let $G_e = (V, \{e' \in E \mid W(e') < W(e)\}$. Then, SPANNING-EDGE$(G, W, e) \leftrightarrow \neg UGAP(G_e, u, v)$. The result then follows from Corollary 5.3. □

This construction was also independently discovered by C. Savage.

5.5. OTHER GRAPH RECOGNITION PROBLEMS CONTAINED IN Π_1CSYMLOG. Here we note that the recognition problems for many interesting and commonly found classes of graphs (including chordal graphs, comparability graphs, interval graphs, split graphs, and permutation graphs) are contained in Π_1CSYMLOG. Our proofs use known characterization lemmas.

Let $G = (V, E)$ be an undirected graph. Let its *complement* be $\bar{G} = (V, \{\{u, v\} \notin E \mid u, v \in V\})$. We define here some graphs commonly found in the literature. Each has a characterization lemma, which immediately implies, by Corollary 5.3, its recognition problem is in Π_1CSYMLOG (by a deterministic log-space reduction to the complement of the UGAP problem). G is a *chordal graph* if every cycle C of length >3 contains a *chord* (an edge connecting two nonconsecutive vertices of C).

LEMMA 5.2. *G is chordal iff for every vertex $v \in V$ and cycle C of length >3, if C contains v, then C has a chord $\{x, y\}$ such that both x and y are of distance ≤ 2 from v.*

PROOF. Repeatedly apply the chordal graph definition. □

G is a *comparability graph* if its edges may be transitively directed.

LEMMA 5.3 [14]. *G is a comparability graph iff for every cycle C of G, if $\{x, y\} \notin E$ for every pair of vertices x, y of distance 2 in C, then C has an even number of edges.*

G is an *interval graph* if its vertices can be put into 1–1 correspondence with a set of intervals on the real line, such that two vertices are connected by an edge of G iff their corresponding intervals have nonempty intersection.

LEMMA 5.4 [14]. *G is an interval graph iff G is a chordal graph and $\bar{G}$ is a comparability graph.*

G is a split graph if its vertex set V can be partitioned into sets V_1, V_2 such that $E(V_1) = \varnothing$ and $(V_2, E(V_2))$ is a complete graph.

LEMMA 5.5 [11]. *G is a split graph iff G and $\bar{G}$ are chordal graphs.*

$G = (V, E)$ is a *permutation graph* if $V = \{v_1, \ldots, v_n\}$ and there is a permutation σ of $\{1, \ldots, n\}$ such that $\{v_i, v_j\} \in E$ iff $(i - j)(\sigma^{-1}(i) - \sigma^{-1}(j)) < 0$.

LEMMA 5.6 [10]. *G is a permutation graph iff both G and $\bar{G}$ are comparability graphs.*

By the above Lemmas and Corollary 5.3,

THEOREMS 5.6–5.10. *The recognition problem for each of the graph classes (chordal graphs, comparability graphs, interval graphs, split graphs, and permutation graphs) are in Π_1CSYMLOG.*

G is *bipartite* if the vertex set V may be partitioned into disjoint sets V_1, V_2 such that $E \subseteq \{\{u, v\} \mid u \in V_1, v \in V_2\}$.

LEMMA 5.7. *G is bipartite iff G has no cycle of odd length.*

By using this characterization lemma, Jones et al. [22] show the recognition problem for nonbipartite graphs is $\leq_{\log}$ equivalent to the complement of UGAP. Thus, by Corollary 5.3,

THEOREM 5.11. *The bipartite graph recognition problem is complete in Π_1CSYMLOG.*

Also, Jones et al. [22] give restricted cases of the NP-complete problems CHROMATIC NUMBER, CLIQUE COVER, EXACT COVER, and HITTING SET and show that their restricted problems are $\leq_{\log}$ equivalent to UGAP and thus complete in NSYMLOG.

5.6. CONSTANT VALENCE PLANARITY TESTING IS IN Π_3CSYMLOG.

* 5.6.1. *Embedding Rotations* Let $G = (V, E)$ be an undirected graph with vertex set V and undirected edge set $E \subseteq \{\{u, v\} \mid \text{distinct } u, v \in V\}$. Let $D(E) = \{(u, v) \mid \{u, v\} \in E\} \cup \{(v, u) \mid \{u, v\} \in E\}$ be the set obtained by directing edges of E. Following Edmonds [9] (also see [32]), we define an *embedding* onto an oriented surface purely combinatorially; let an *embedding rotation* be a set $\theta = \{\theta_v \mid v \in V\}$ where θ_v is a cyclic ordering of the set of directed edges $D_v(E) = \{(x, y) \in D(E) \mid x = v\}$ of $D(E)$ departing from vertex v. Intuitively, θ_v gives the clockwise rotation of edges as they are embedded around vertex v, in a graph with a planar embedding.

Let $R(\theta) \subseteq D(E) \times D(E)$ be the relation such that $e_1^- R(\theta) e_2$ iff directed edge e_1 departs from the same vertex v that directed edge e_2 departs from, and e_2 appears immediately after e_1 in θ_v (where e_1^- is the reverse of edge e_1). (See Figure 1. Note

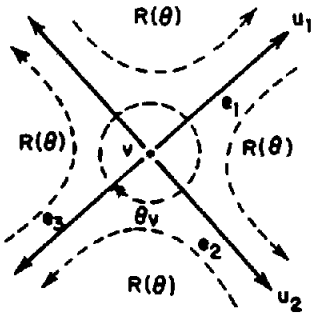

FIGURE 1

that e_3 immediately follows e_2, which immediately follows e_1 in Figure 1, so $e_1^- R(\theta) e_2$ and $e_2^- R(\theta) e_3$, but not $e_1^- R(\theta) e_3$.)

$R(\theta)$ is easily shown to be partitioned into a set of cyclic permutations. Let c be the number of connected components of G. Let $\lambda(\theta)$ be the number of orbits of $R(\theta)$. By Euler's formula, the embedding rotation θ is *planar* if $\lambda(\theta) - |E| + |V| = 2c$. Intuitively, if θ is planar and $c = 1$, then the orbits of $R(\theta)$ are in 1–1 correspondence with the connected regions of the embedding, with the borders of the regions oriented counterclockwise. If θ is planar and $c \geq 1$, then c orbits of R will be associated with the exterior regions of the embedding.

By Edmonds' [9] characterization, the graph G is *planar* iff it has a planar embedding rotation. To test if G is planar, we construct a formula H_G, which is true iff G is planar. (Previously, Ja'Ja' and Simon [21] gave a log-space construction of a 2CNF formula which is **true** if a triconnected graph G is planar, but may also be true if G is not planar.) Our construction makes no assumptions about the connectivity of G. However, we assume G has constant valence.

5.6.2. *The Basis Cycles.* Let us impose an arbitrary, fixed numbering of the edges E from $1, \ldots, |E|$. We consider this numbering to be an edge weighting of the graph $G = (V, E)$. The greedy algorithm of Section 5.4 constructs a *unique* minimum spanning forest (V, SF_G). For each direct edge (u, v), where $\{u, v\} \in E - SF_G$, there is a unique directed simple cycle $C_{(u,v)}$ containing (u, v) and directed edges derived from SF_G. We call $C_{(u,v)}$ a *basis cycle* Let $\mathscr{C}$ be the set of all basis cycles. By applying Theorem 5.5, we have

LEMMA 5.8. *Π_1CSYMLOG contains the test: Is edge sequence C in $\mathscr{C}$?*

5.6.3. *The Bridges of G.* Let C be a basis cycle in $\mathscr{C}$ of graph $G = (V, E)$. Let a *bridge* of C be a maximal edge set $B \subseteq E - \{\{u, v\} | (u, v) \in C\}$ such that $\forall e_1$, $e_2 \in B$ and there is a path p in B containing both e_1 and e_2, but visiting vertices of C only at the endpoints of p. Given C, we can construct in deterministic log space a graph (this graph is derived from G by substituting a distinct new vertex v_e and edge $\{u, v_e\}$ for each edge $e = \{u, v\}$ of E such that a vertex v appears in C) whose connected components are in 1–1 correspondence with the bridges of C. Thus, by Proposition 2.1 and Corollary 5.4, a symmetric complementing machine with logarithmic space and a single complement move on the first move can test whether B is a bridge of a given basis cycle; furthermore, by Lemma 5.8, this symmetric complementing machine can also enumerate all basis cycles in logarithmic space using a single further complement (yielding a total complement bound of 2). Thus we have

LEMMA 5.9. *The bridges of all the cycles of the basic $\mathscr{C}$ may be recognized in Σ_2CSYMLOG.*

5.6.4. *Embedding Formulas.* For each vertex $v \in V$ and each cyclic ordering θ_v of the set $D_v(E)$ of directed edges departing from v, we have a distinct Boolean variable $O_v(\theta_v)$. Note that since G has constant valence, there is only a constant number of such cyclic orderings. Let h_1 be the CNF⊕ formula consisting of the conjunction of subformulas $O_v(\theta_v) \oplus O_v(\theta_v')$ for all $v \in V$ and distinct θ_v, θ_v' of $D_v(E)$.

Also, for each undirected edge $e \in E$ and basis cycle $C \in \mathscr{C}$ not containing e, we have a Boolean variable OUT(C, e). (Intuitively, this variable will be **true** iff e is embedded into the exterior of the closed region of the sphere defined by C.) Let h_2 be the CNF⊕ formula consisting of the conjunctions of subformulas $O_v(\theta_v) \oplus \neg\text{OUT}(C, \{v, u_3\})$ for all $C \in \mathscr{C}$, and (u_1, v), $(v, u_2) \in C$, and $\{v, u_3\} \in E$, and θ_v orders (v, u_3) between (v, u_2) and (v, u_1).

Note that $h_1 \wedge h_2$ holds just when an embedding rotation θ is induced with the following property: For each $v \in V$ and basis cycle $C \in \mathscr{C}$ containing vertex v, if (u_1, v), (v, u_2) are directed edges of C and $\{v, u_3\}$ is any other edge containing v, then $\{v, u_3\}$ is embedded to the exterior of the region defined by C if θ_v cyclically orders (v, u_3) between (v, u_2) and (v, u_1) (see Figure 2).

By Proposition 2.2 and Lemma 5.8, both h_1 and h_2 can be constructed by a complementing symmetric machine with logarithmic space and complement bound 1. Finally, let h_3 be the formula consisting of the conjunction of subformulas $\text{OUT}(C, e_1) \oplus \neg\text{OUT}(C, e_2)$ for all $C \in \mathscr{C}$, and $e_1, e_2 \in E$, and e_1, e_2 are in the same bridge of C.

Note that h_3 holds just when for each bridge B of any cycle $C \in \mathscr{C}$, either all edges of B are embedded interior to the region defined by C, or all edges of B are embedded exterior to the region defined by C. By Proposition 2.2 and Lemmas 5.8 and 5.9, h_3 may be constructed by a symmetric complementing machine with logarithmic space and complement bound 2.

Let H_G be the formula $h_1 \wedge h_2 \wedge h_3$ with existential quantification on all the OUT and O variables.

LEMMA 5.10. *G is planar iff H_G is* **true**.

PROOF. Suppose G is planar. Then by Edmond's characterization, G has some planar embedding orientation θ. We use θ to define the truth values for the O_v and OUT variables. They clearly satisfy h_1, h_2, and h_3. Thus, we can satisfy H_G.

On the other hand, suppose H_G is true for some variable assignment. Let θ be the embedding rotation induced by the O_v variables. Let $\lambda(\theta)$ be the number of orbits of $R(\theta)$. Now we must show $\lambda(\theta) - |E| + |V| = 2$, implying by Euler's formula that θ is planar. Fix the spanning forest $SF = SF_G$. Suppose we delete an edge $e \in E - SF$ from G, so that the resulting graph is $G' = (V, E')$ with $E' = E - \{e\}$. We claim that if θ' is the resulting embedding rotation, then $\lambda(\theta) - |E| + |V| = \lambda(\theta') - |E'| + |V|$. Clearly $|E'| = |E| = 1$ so we must show $\lambda(\theta') = \lambda(\theta) - 1$.

Let $e = \{u, v\}$ be the undirected edge to be deleted from $E - SF$. Let π_1, π_2 be the orbits of $R(\theta)$ containing directed edges $d_1 = (u, v)$, $d_2 = (v, u)$, respectively, so $d_1^- = d_2$ and $d_2^- = d_1$. We claim now that $\pi_1 \neq \pi_2$.

For each $i \in \{1, 2\}$, let $C_{d_i} \in \mathscr{C}$ be the unique basis cycle that contains directed edge d_i. If π_i is a cyclic permutation of C_{d_i}, then C_{d_i} and π_i do not contain the reverse edge d_i^-, so $\pi_1 \neq \pi_2$.

Otherwise, for each $i = 1, 2$ let (x_i, y_i) be the first edge of π_i following d_i such that (x_i, y_i) is not in C_{d_i}. Let f_i be the directed edge of π_i immediately preceding

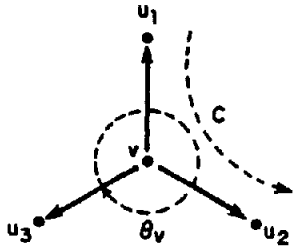

FIGURE 2

(x_i, y_i). By definition of orbits, (x_i, y_i) immediately follows f_i^- in θ_{x_i}. Note also that f_i must be a directed edge in C_{d_i}. Let g_i be the directed edge of C_{d_i} immediately following f_i. Then (x_i, y_i) is cyclically ordered between f_i^- and g_i in θ_{x_i}. Since h_2 is satisfied, $\mathrm{OUT}(C_{d_i}, \{x_i, y_i\})$ = **false** for $i = 1, 2$. But since C_{d_2} is the reverse of directed cycle C_{d_1}, g_2^- appears just before f_2^- in C_{d_1} and so $\mathrm{OUT}(C_{d_1}, \{x_2, y_2\})$ = **true**.

However, we claim $\mathrm{OUT}(C_{d_1}, \{x, y\})$ = **false** for all $\{x, y\} \in E$ such that (x, y) is contained in π_1, but not C_{d_1}. This has already been proved for $x = x_1$ and $y = y_1$. For any subsequence π' of π_1 contained on only a single bridge of C_{d_1}, since h_3 is satisfied, we have $\mathrm{OUT}(C_{d_1}, \{x, y\}) = \mathrm{OUT}(C_{d_1}, \{x', y'\})$ for all (x, y) and (x', y') in π'. Also, if $(x, y)\pi''(x', y')$ is a subsequence of π_1 where π'' is a subsequence of C_{d_1} but (x, y) and (x', y') do not appear in C_{d_1}, then, since h_2 is satisfied, we have $\mathrm{OUT}(C_{d_1}, \{x, y\}) = \mathrm{OUT}(C_{d_1}, \{x', y'\})$.

If (x_2, y_2) is in π_1, then we have just shown $\mathrm{OUT}(C_{d_1}, \{x_2, y_2\})$ = **false**, a contradiction. Hence, (x_2, y_2) is not in π_1, but by assumption is in π_2; so $\pi_1 \neq \pi_2$.

Without loss of generality, let $\pi_1 = d_1\pi_1'$ and $\pi_2 = d_2\pi_2'$, where neither π_1' nor π_2' contain d_1 or d_2. Hence, when e is deleted from $E - SF$, the orbits π_1 and π_2 of $R(\theta)$ are merged into a single new orbit $\pi_1'\pi_2'$ of $R(\theta')$, and no other orbits are modified.

Thus, $\lambda(\theta') = \lambda(\theta) - 1$, and we have shown $\lambda(\theta) - |E| + |V|$ remains invariant. We repeat this process (deleting edges not in SF) until we have only $E' = SF$. Let c be the number of maximal trees in the spanning forest SF; c is also a number of connected components of SF. Then $\lambda(\theta') = c$, $|SF| = n - c$, and $|V| = n$. Hence, our invariant is $\lambda(\theta) - |E| + |V| = \lambda(\theta') - |SF| + |V| = 2c$, which implies by Euler's formula that the embedding orientation θ is planar. Thus, by Edmonds characterization, G is planar. □

The formula H_G is constructed by a symmetric complementing machine with logarithmic space and complement bound 2. Furthermore, by Theorem 5.2, a single further complementation is required to test if H_G is **true**. Thus by Lemma 5.10.

THEOREM 5.12. *Planarity testing of constant valence graph is in* Π_3*CSYMLOG.*

6. *Conclusion*

It may be significant that the symmetric complementing machine introduced in this paper has applications to many combinatorial problems found in practice, such as spanning trees, k-connectivity, and planarity testing. In this case the theoretical study of a new machine type led us to the discovery of new techniques for practical combinatorial algorithm design. For example, applying our probabilistic decision algorithm for symmetric games to our proof that constant valence planarity testing is in Π_3CSYMLOG yields a new probabilistic algorithm for

planarity testing; furthermore, this algorithm has a quite different structure than any of the previously known deterministic planarity testing algorithms such as that of Hopcroft and Tarjan [19]. This indicates to us that the field of combinatorial algorithms, as well as the field of abstract computational complexity, would benefit by further study of unusual machine types and their decision algorithms.

Note. A. Borodin, J. Hopcroft, M. Paterson, L. Ruzzo, and M. Tompa also independently discovered the use of random walks to solve graph connectivity in logarithmic parallel time.

In a preliminary draft of this paper, we defined a restricted type of alternating machine whose nonalternation next moves are a symmetric relation. Dexter Kozen pointed out to us that such symmetric alternation machines are too restricted for our intended applications (in particular, they do not satisfy a complementation property such as Proposition 2.1).

The symmetric complementing machine described in this paper satisfies the required complementation property of Proposition 2.1 and also has efficient decision algorithms if the machine is both space and complementation bounded. Michael Sipser has also suggested an equivalent machine; this is a symmetric alternating machine M (with symmetric nonalternation moves), but with a modified definition of acceptance:

M accepts (*rejects*, respectively) from an existential (universal, respectively) configuration I if there exists a finite computation sequence $I = I_1, \ldots, I_{j-1}, I_j$, where $I_1, \ldots, I_{j-1}$ are existential (universal, respectively) and I contains an accepting (rejecting, respectively) state or I_j is universal (existential, respectively) and M accepts (rejects, respectively) from I_j. Also, *M rejects* (*accepts*, respectively) from existential (universal, respectively) configuration I iff M does not accept (does not reject) from I. Thus, Sipser's definitions for acceptance and rejection of these machines are duals. This is the same as the standard definition of acceptance of an alternating machine from an existential configuration, but differs from the standard definition of acceptance from a universal configuration so as to allow for complementations of languages.

ACKNOWLEDGMENTS. The author wishes to thank Larry Denenberg, Vassos Hadzilacos, Joe Halpern, Harry Lewis, A. Prasad, Michael Sipser, and Paul Spirakis and the referees for a careful reading and many useful comments on preliminary drafts of this paper.

REFERENCES

1. ADLEMAN, L Two theorems on random polynomial time. In *Proceedings of the 18th IEEE Symposium on Foundations of Computer Science* IEEE, New York, 1978, pp. 75–83.
2. ALELIUNAS, R., KARP, R.M., LIPTON, R.H., LOVÁSZ, L., AND RACKOFF, C. Random walks, universal traversal sequences, and complexity of maze problems. In *Proceedings of the 20th Annual Symposium on Foundations of Computer Science.* IEEE, New York, 1979, pp. 218–223.
3 BENNETT, C.H , AND GILL, J. Relation to a random oracle A, $P^A \neq NP^A \neq coNP^A$ with probability 1. *SIAM J Comput 10,* 1 (Feb. 1981), 98–113.
4 BONDY, J.A., AND MURTY, U.S R. *Graph Theory with Applications* Elsevier North-Holland. New York, 1977.
5. CHANDRA, A.K., KOZEN, D.C., AND STOCKMEYER, L.J. Alternation. *J. ACM 28,* 1 (Jan. 1981), 114–133.
6. COOK, S A Towards a complexity theory of synchronous parallel computation. *Extrait de L'Ensigeignement Mathematique T XXVII,* FASC 1–2 (1981), 100–124.
7 CSANKY, L. Fast Parallel Matrix Inversion Algorithms. *SIAM J Comput 5* (1976), 618–623.

8. DYMOND, P., AND COOK, S.A. Hardware complexity and parallel computation. In *Proceedings of the 21st Annual Symposium on Foundations of Computer Science.* IEEE, New York, 1980, pp. 360–372.
9. EDMONDS, J. A combinatorial representation for polyhedral surfaces. *Am. Math. Soc. Not. 7* (1960), 646
10. EVEN, S., PNUELI, A , AND LEMPEL, A Permutation graphs and transitive graphs. *J. ACM 19,* 3 (1972), 400–410.
11. FOLDES, S , AND HAMMER, P.L. Split graphs. In *Proceedings of the 8th Southeastern Conference on Combinatorics, Graph Theory and Computing,* (Baton Rouge, La., 1977). Utilitas Mathematica, Univ of Manitoba, Winnipeg, Man., Canada, pp. 311–315.
12. FORTUNE, S., AND WYLLIE, J. Parallelism in random access machines. In *Proceedings of the 10th ACM Symposium on Theory of Computing* (San Diego, Calif., May 1–3). ACM, New York, 1978, pp. 114–118.
13 GILL, J. Complexity of probabilistic Turing machines. *SIAM J Comput. 6,* 4 (1977), 675–695.
14. GILMORE, P.C., AND HOFFMAN, A.J. A characterization of comparability graphs and of interval graphs. *Canad J Math 16* (1964), 539–548.
15. GOLDSCHLAGER, L. A unified approach to models of synchronous parallel machines. In *Proceedings of the 10th Annual ACM Symposium on the Theory of Computing* (San Diego, Calif., May 1–3). ACM, New York, 1978, pp. 89–94.
16. HIRSCHBERG, D.S. Parallel algorithms for the transitive closure and the connected components problems. In *Proceedings of the 8th Annual ACM Symposium on the Theory of Computing.* (Hershey, Pa., May 3–5). ACM, New York, 1976, pp. 55–57.
17. HIRSCHBERG, D S , CHANDRA, A.K., AND SARWATA, D.V. Computing connected components on parallel computers. *Commun ACM 22,* 8 (Aug. 1979), 461–464.
18 HOPCROFT, J.E., AND TARJAN, R.E. Efficient algorithms for graph manipulation. *Commun. ACM 16,* 6 (1973), 372–378.
19. HOPCROFT, J.E., AND TARJAN, R.E. Efficient planary testing *J ACM 21,* 4 (Oct. 1974), 549–568.
20. JA'JA', J , AND SIMON, J Parallel algorithms in graph theory: Planarity testing. *SIAM J. Comput. 11,* 2 (May 1982), 314–328.
21. JA'JA', J , AND SIMON, J Some space-efficient algorithms. In *Proceedings of the 17th Allerton Conference* 1979, pp. 677–684. Tech Rep CS-80-14, Penn State Univ., 1980.
22. JONES, N.D., LIEN, Y.E , AND LAASER, W.T. New problems complete for nondeterministic log space. *Math Syst Theory 10* (1976), 1–17.
23 KARP, R.M., AND LIPTON, R.J. Some connections between nonuniform and uniform complexity classes. In *Proceedings of the 12th Annual ACM Symposium on Theory of Computing* (Los Angeles, Apr 28-30) ACM, New York, 1980, pp. 302–309. (Also presented at the Specker Symposium on Complexity (Zurich, February 1980)).
24 LEWIS, H.R., AND PAPADIMITRIOU, C.H. Symmetric space bounded computation. *Theor. Comput. Sci 19* (1982), 161–187.
25. MATULA, D. *k*-blocks and ultrablocks in graphs. *J Comb Theory, Ser. B 624* (1978), 1–13.
26. MCLANE, S. A combinatorial condition for planar graphs. *Fundam Math. 28* (1937), 22–32.
27. RABIN, M.O. Probabilistic algorithms. In *Algorithms and Complexity, New Directions and Recent Results,* J. Traub, Ed. Academic Press, New York, 1976, pp. 21–36.
28 REIF, J.H. On the power of probabilistic choice in synchronous parallel computations. In *Proceedings of the 9th International Colloquium on Automata, Languages and Programming* (Aarhus, Denmark, July 1982). Springer Verlag, Berlin, pp. 442–450.
29. SAVAGE, C., AND JA'JA', J. Fast efficient parallel algorithms for some graph problems. *SIAM J Comput 10,* 4 (Nov. 1981), 682–691.
30. SAVITCH, W. J., AND STIMSON, M.J. Time bounded random access machines with parallel processing. *J ACM 26,* 1 (Jan. 1979), 103–118.
31. SCHONHAGE, A. Storage modification machines. Technical Report, Mathematisches Institut, Universitat Tubingen, Germany, 1979.
32. WHITE, A. *Graphs, Groups and Surfaces* Elsevier North-Holland, New York, 1973.

RECEIVED AUGUST 1981; REVISED JUNE 1983; ACCEPTED JULY 1983