JOURNAL OF COMPUTER AND SYSTEM SCIENCES 37, 39–62 (1988)

Parallel Algorithms for Solvable Permutation Groups

EUGENE M. LUKS*

Department of Computer and Information Science, University of Oregon, Eugene, Oregon 97403

AND

PIERRE MCKENZIE†

Département d'I.R.O., Université de Montréal, Montreal, Canada H3C 3J7

Received September 15, 1986; revised October 26, 1987

A number of basic problems involving solvable and nilpotent permutation groups are shown to have fast parallel solutions. Testing solvability is in NC as well as, for solvable groups, finding order, testing membership, finding centralizers, finding centers, finding the derived series and finding a composition series. Additionally, for nilpotent groups, one can, in NC, find a central composition series, and find pointwise stabilizers of sets. The latter is applied to an instance of graph isomorphism. A useful tool is the observation that the problem of finding the smallest subspace containing a given set of vectors and closed under a given set of linear transformations (all over a small field) belongs to NC. © 1988 Academic Press, Inc.

1. INTRODUCTION

In recent years the asymptotic complexity of decidable group-theoretic problems has received much attention (e.g., [At75; Ba79; FuHoLu80a; Ho82; Lu82; BaKaLu83; McCo83; AvMa84a; AvMa84b; Ba85; Ka85a; Ka85b; Lu87; Re85; McCo87; Co85; KaTa]). The short-term impact of this work on computational group theory remains in question (it is an amazing fact that many practical group-theoretic algorithms in current use have non-polynomial worst-case time complexities [Ca84]), but the relevance of such work to the theory of computation is beyond doubts. For example, many subcases of the graph isomorphism problem (one of the few "classical" problems in the class NP believed not to be NP-com-

* Research supported by NSF Grants DCR-8403745 and DCR-8609491.
† Research supported by the Université de Montréal, by the Programme de Formation de Chercheurs et d'Action Concertée du Québec, and by the Natural Sciences and Engineering Research Council of Canada.

0022-0000/88 $3.00

plete), were solved in polynomial time using group-theoretic arguments [Ba79; FuHoLu80b; Lu82; Mi83]. Another example is the study of matrix group problems, which led to "Arthur versus Merlin" games and to the introduction of a finite complexity hierarchy collapsing "just above" NP [BaSz84; Ba85]. Yet another example is the existence of several free group problems complete for the class P under log space reducibility [AvMa84a; AvMa84b].

Naturally, the computational complexity of group-theoretic problems depends critically on the input description of the groups. This is illustrated best of all by the membership problem, which consists of determining whether a given test element g belongs to a specified group G. When G is specified by generators (indeed when G is even the trivial subgroup) within a finitely presented group then a result of Novikov-Boone [Ro73, p. 298] shows that membership is undecidable. When G is a group of matrices over a finite field $GF(p)$, membership is not known to be in NP [Ba85]. When G is a finite permutation group specified by generating permutations, membership can be decided in polynomial time using a variant of Sims' algorithm [Si70; FuHoLu80a]. When G is given explicitly by multiplication table, membership is a non-issue.

This paper is concerned solely with finite permutation groups. It is well known that all finite groups can be represented as permutation groups. Moreover, generating permutations provide economical descriptions of exponentially large sets (the groups generated). The sequential complexity of problems involving permutation groups specified by generating permutations (from now on we assume all groups so specified) has been studied intensively. Polynomial time algorithms now exist to find blocks of imprimitivity [At75], to test membership and to compute order [Si70; FuHoLu80], to compute generating permutations for groups occurring in derived series [FuHoLu80], to compute a composition series [Lu87], and to compute the Sylow subgroups of a group [Ka85a; Ka85b].

A natural question to ask about problems with good sequential solutions is how efficiently such problems can be solved on parallel computer models. We address this issue here and develop several new fast parallel algorithms for dealing with permutation groups. Specifically, we are concerned with the complexity class NC [Pi79; Co85], identified informally as the class of problems solvable in time $(\log n)^k$ for some k using a polynomial number of processors (hence problems in NC have polynomial time solutions [FiPi74; Sc76; Bo77]. We do not discuss the immediate practical merits of the class NC (see, for example, [Co85; Vi85] for pros and cons, respectively). Rather we adopt the viewpoint that the subclasses NC^k provide an interesting hierarchy which bears witness to the amount of "decomposability" (translating both into small parallel time requirements and into small sequential space requirements, see [Bo77; Co81]), and thus to the degree of difficulty, of a problem.

Prior to the work reported here, the parallel complexity of permutation group problems had been investigated in [McCo83; Mc84; Re85; McCo87]. McKenzie and Cook exhaust the case of Abelian permutation groups by showing that problems ranging from determining membership to obtaining a complete cyclic

decomposition belong to the class NC^4 (their weaker statement that such problems belong to "random" NC can indeed be strengthened owing to Mulmuley's completion of the problem of computing ranks of matrices over small fields in NC [Mu87]). The techniques used, combining the regularity property of transitive Abelian groups with the ability to express membership testing as solving a single system of linear congruences, do not seem to generalize. McKenzie [Mc84] compares non-Abelian permutation group problems with respect to their parallel complexity and develops an NC^4 nilpotency test. Reif [Re85] suggests a probabilistic NC membership test in a p-group assuming an oracle that delivers uniformly distributed random elements from the group.

Our main results are NC upper bounds for fundamental questions involving nilpotent and solvable groups which were not long known to have polynomial time solutions. Solvable groups constitute the largest class of groups that can be obtained using Abelian building blocks, that is, solvable groups have Abelian composition factors (the name "solvable" comes from Galois theory, as solvable groups are those which play a role in the expressibility of roots of a polynomial by radicals, see, for instance, [Ar42]). For solvable groups we show how to determine membership, how to compute order, normal closures, centralizers, centers, derived series, and composition series (see next section for definitions) in NC. This answers several open questions from [McCo87; Mc84]. The more restrictive nilpotent groups are characterized as being direct products of groups of prime power order. For nilpotent groups we further develop NC algorithms to compute pointwise set stabilizers and central series. We also suggest applications of the pointwise set stabilizer algorithm, including instances of the graph isomorphism problem.

The results of this paper were first announced in [LuMc85], the randomness in that paper having been obviated by [Mu87]. Recent work of Luks [Lu86] and Babai *et al.* [BaLuSe87] has since shown that general permutation group management (including order, membership, and pointwise set stabilizers) is in NC. In these extensions, the present paper has remained a fundamental tool. The techniques are not only necessary for the strictly solvable or nilpotent subclasses, but they are a component in the management of any groups that have some Abelian composition factors.

The organization of the paper is as follows. Section 2 introduces notation and background. Section 3 introduces notions to facilitate the manipulation of solvable groups. Section 4 discusses a linear algebra problem whose NC solution is fundamental to most subsequent algorithms. Section 5 presents nilpotent group algorithms. Section 6 solves the important pointwise set stabilizer problem for nilpotent groups and suggests applications to the setwise stabilizer problem and to graph isomorphism. Section 7 shows how to extend some of the techniques developed in Section 5 in order to provide solvable group algorithms. Finally, Section 8 concludes with open questions and suggestions for further work.

2. Background and Notation

We assume familiarity with the complexity classes NC^k, NC [Pi79], and P (see [HoU179]) generalized to include more than just decision problems (see [Co85]). Recall that NC^k is defined as the class of problems solvable by an $O((\log n)^k)$ depth and polynomial size uniform family of bounded indegree Boolean circuits, and that $NC = \bigcup_k NC^k$. Loosely speaking NC can thus be thought of as the class of problems solvable in polylog time on a parallel computer of feasible size and a reader ill at ease with Boolean circuits should adopt the latter point of view.

Descriptions of all the circuits discussed in this paper are computable by a Turing machine in log space, and these circuits probably meet the more stringent "alternating log time" uniformity condition introduced by Ruzzo [Ru81] and favored by Cook [Co85] (these different uniformity criteria may affect the subclass NC^1). Our notion of NC^1-reducibility is that in [Co85], except for our use of log space uniformity (log space uniform NC^1-reducibility is also used in [McCo87] to compare parallel complexities of Abelian permutation group problems). Occasionally we speak of NC^2-reducibility. We say that problem A NC^2-reduces to problem B if B is solved by an NC^2 family of circuits which is allowed the use of $O(\log n)$ oracle gates for A along any path from an input to an output. Note that NC^k for any k is not necessarily "closed" under NC^2-reducibility, but that NC is.

Our group-theoretic notation is mostly that of [Wi64]. We write $H \leqslant G$ when H is a subgroup of a group G. With S a set of elements in a given group, $\langle S \rangle$ represents the subgroup generated by S. Let g, h belong to a group G. The *commutator* $[g, h]$ is defined as the element $g^{-1}h^{-1}gh$ and we write g^h for the *conjugate* of g by h, that is, for $h^{-1}gh$. The *degree* of a permutation group G is the number of points actually moved by G, that is, not fixed by all elements in G. Group G *acts* on a set Ω if there is a homomorphism $\phi: G \to \mathrm{Sym}(\Omega)$. In such a case we write α^g for the image of $\alpha \in \Omega$ under $g \in G$, and Γ^g for the set of images of elements of $\Gamma \subseteq \Omega$ under g. G acts *faithfully* on Ω if G is isomorphic to its image within $\mathrm{Sym}(\Omega)$. Group G_Γ is the *setwise stabilizer* of Γ, that is, the group of all elements in G which map Γ to itself. The set $\Gamma \subseteq \Omega$ is a *G-orbit* if $\Gamma = \{\alpha^g \mid g \in G\}$ for some $\alpha \in \Omega$. If Γ is a union of G-orbits, then G^Γ denotes the *constituent* of G on Γ, that is, the image of the induced G-action on Γ. Group G acts *transitively* on $\Gamma \subseteq \Omega$ if Γ is a G-orbit; in that case G^Γ is a *transitive constituent*. When G acts transitively on Ω, a *G-block* is defined as any set $\Gamma \subseteq \Omega$ for which either $\Gamma^g \cap \Gamma = \varnothing$ or $\Gamma^g = \Gamma$ holds for each $g \in G$. If Γ is a G-block then G acts naturally on the *G-block system* $\{\Gamma^g \mid g \in G\}$. A G-block system partitions Ω into G-blocks of equal size. We say that G acts *primitively* on Ω if (it acts transitively on Ω and) Ω cannot be broken up into nontrivial (i.e., sizes $\neq 1$ or $|\Omega|$) G-blocks.

We refer the reader to [Ha59] for basic facts about *Abelian* (or commutative) groups, *normal closures*, *centralizers*, *centers*, *commutator subgroups*, *composition series*, *derived series*, *solvable* groups, *central series*, and *nilpotent* groups, though we recall the definitions of these concepts below. The *normal closure* of a subset S in a group G, denoted $\mathrm{NCL}_G(S)$, is the smallest normal subgroup of G containing S.

When G and H are subgroups of some larger group, we say G *normalizes* H if H is normal in $\langle G, H\rangle$. The *centralizer* $C_G(H)$ of H in G is $\{g \in G \mid gh = hg$ for each $h \in H\}$. The *center* of G is $C_G(G)$. The *commutator subgroup* $[G, G]$ of a finite group G is defined as $\langle [g, h] \mid g, h \in G\rangle$. Let $G = G_0 \geqslant G_1 \geqslant \cdots \geqslant G_k$ be a *subnormal series* for G, that is, one in which each term is normal in the preceding term. This series is a *composition series* for G if for each i G_{i+1} is a maximal normal subgroup of G_i. The series is the *derived series* for G if $G_{i+1} = [G_i, G_i]$ for each i; group G is *solvable* if the trivial group appears in its derived series. Finally, the series is a *central series* if for each i G_i/G_{i+1} is contained in the center of G/G_{i+1}; group G is *nilpotent* if it possesses a central series in which the trivial group appears (alternatively G is nilpotent if it is a *p-group*, that is, a group of prime power order, or if it is a direct product of p-groups for various p; note that Abelian groups are nilpotent). An Abelian group is said to be *elementary Abelian* if for a fixed prime p each non-trivial element of the group has order p.

Let $G = \langle S\rangle$ and let T be a complete set of right coset representatives of a subgroup H in G (that is, T contains exactly one element from each coset Hg of H in G). Then $\{ts[\phi(ts)]^{-1} \mid t \in T, s \in S\}$, where $\phi: G \to T$ is the corresponding coset representative function, is the set of *Schreier generators* of H relative to S and T (for a proof that the Schreier generators generate H see Lemma 7.2.2 in [Ha59]).

For our purposes, an *algebra* is a vector space equipped with an associative multiplication that distributes over linear combinations. For example, a set of matrices over $\mathbf{Z}_p$ closed under matrix addition and under matrix multiplication is an algebra over $\mathbf{Z}_p$.

Throughout this paper we assume a reasonable binary encoding of all the problems discussed (see, for instance, [Mc84]), and we follow [Mc84] in referring to an integer whose absolute value is in $O(n)$, when n is the length of the encoding of a problem instance, as a *tiny* integer. An example of a tiny integer is the degree of a permutation group specified by generating permutations. From now on, "computing a group" will always mean "computing a set of generators for the group."

3. Preliminaries

This section discusses the notions of structure forest and of power-commutator basis. Our algorithms use these concepts to transform the arbitrary set of input permutations generating a (solvable) group into a manageable description of the group.

Definition (structure forest). A structure forest for a permutation group G is a forest on which G acts as automorphisms (fixing the roots of the individual trees), whose leaves form the permutation domain, and such that the stabilizer within G of any node ($\equiv$ set of subtended leaves) acts primitively on the children of this node.

When G is nilpotent, the stabilizer of a node in a structure forest for G acts as a

cyclic group of prime order on the children of this node (this can be seen as a consequence of remarks on page 66 of [Ha59]). When G is solvable, the Pálfy–Wolf bound on the order of primitive solvable groups [Pa82; Wo82] states that the stabilizer of a node restricted to the children of this node has order at most $24^{-1/3}m^{3.24399\ldots}$ (m the number of children).

Typical sequential methods for constructing such a forest (requiring at a "primitive" node, the subgroup fixing the blocks) lead either to "blow-ups" in the sizes of generating sets, or to sequential "sifts" through linear-height towers of groups. As had occurred independently to Reif [Re85], we can avoid these pitfalls, so that

PROPOSITION 3.1. *NC^3 contains the problem of computing a structure forest for an* arbitrary *permutation group G.*

Proof. First we break up the point set into orbits [McCo83]. (Each orbit gives rise to a tree in the forest and we build each tree in parallel.) Now if G acts transitively on Ω, if Γ is a G-block, and if $\Delta \subseteq \Gamma \subseteq \Omega$ is a G_Γ-block, then Δ is in fact a G-block. This suggests picking a non-trivial G-block of smallest size, say Γ (in NC^2 [Si67; Mc84; Re85]). The previous statement guarantees that G_Γ then acts (transitively and) primitively on Γ. Hence Γ can be made a set of leaves with common parent. Images of Γ under G yield the other subtrees at the bottom level. The procedure is repeated with the parents so created (in effect with the G-action on the G-block system containing Γ). After $\log n$ iterations (hence NC^3) each tree is complete. ∎

We now define what we mean by a "manageable" group description:

DEFINITION (power-commutator basis of a group). A power-commutator basis (PCB) of a group G is an ordered sequence $(b_1, \rho_1), \ldots, (b_m, \rho_m)$, $b_i \in G$, $\rho_i > 1$ an integer, $1 \leqslant i \leqslant m$, such that

(1) each $g \in G$ is uniquely expressible in a "canonical form" $b_1^{\varepsilon_1} \cdots b_m^{\varepsilon_m}$, $0 \leqslant \varepsilon_i < \rho_i$, $0 \leqslant i \leqslant m$,

(2) for each pair of integers i, j, $1 \leqslant i < j \leqslant m$, the canonical expression for the commutator $[b_j, b_i]$ satisfies $\varepsilon_1 = \varepsilon_2 = \cdots = \varepsilon_i = 0$, and

(3) for each integer i, $1 \leqslant i \leqslant m$, the canonical expression for the element $b_i^{\rho_i}$ also satisfies $\varepsilon_1 = \varepsilon_2 = \cdots = \varepsilon_i = 0$.

Observe that a PCB for G directly reflects part of the group's structure. For example, the order of G is $\rho_1 \rho_2 \cdots \rho_m$. Also, it is easy to verify by induction that $(b_i, \rho_i), \ldots, (b_m, \rho_m)$ is a PCB for $G_i = \langle b_i, \ldots, b_m \rangle$, and so G_{i+1} is a normal subgroup of G_i with cyclic quotient. It follows that a PCB for a group G exists if and only if G is solvable.

Remark. In recent work [Lu86; Ba86; BaLuSe87], the PCB tool is replaced by a "strong generating set" (SGS). As used earlier in the sequential-computation

settings of [Ba79; FuHoLu80], an SGS is comprised of the union of coset representatives for the G_i mod G_{i+1} in any *polynomial tower*, i.e., any subgroup chain $G = G_1 \geqslant G_2 \geqslant \cdots \geqslant 1$, in which the indices $[G_i: G_{i+1}]$ are polynomially bounded. (This generalized a notion of Sims [Si70] who made use of the case where G_i is the subgroup of G that fixes the first i points in the permutation domain). Of course, an SGS is easily obtainable from a PCB by taking the $|\varepsilon_i|$ distinct powers of b_i to represent the cosets of G_i mod G_{i+1}. One can, in fact, modify the algorithms herein to conform to the construction of SGSs in general groups. One typical change would be in the description of elements to be "sifted" in Proposition 3.2 (compare with section 5 of [Lu86]). Despite the temptation to unify the notation and definitions, we have retained the PCBs in this paper. It appears to be the natural way to deal with groups that are known to be solvable.

For the purpose of computing PCBs it is necessary to extend the definition of PCBs as follows.

DEFINITION (PCB of a group relative to a normal subgroup). Given K a normal subgroup of a group G, a power-commutator basis for G relative to K (PCB of G rel K) is an ordered sequence $(b_1, \rho_1), \ldots, (b_m, \rho_m)$, $b_i \in G$, $\rho_i > 1$ an integer, $1 \leqslant i \leqslant m$, such that $\{(b_i K, \rho_i)\}_{1 \leqslant i \leqslant m}$ is a PCB for G/K. With respect to this PCB "sifting an element $g \in G$" means "computing the unique $h \in K$ such that the product gh^{-1} is expressible in the form $b_1^{\varepsilon_1} \cdots b_m^{\varepsilon_m}$, $0 \leqslant \varepsilon_i < \rho_i$, $1 \leqslant i \leqslant m$." If the PCB is understood, we denote the induced function $G \to K$ by SIFT; i.e., in the previous sentence, $h = \mathrm{SIFT}(g)$.

Observe that if $\psi: G \to H$ is a group epimorphism with $(b_i, \rho_i)_{1 \leqslant i \leqslant m}$ a PCB for H, then $(\psi^{-1}(b_i), \rho_i)_{1 \leqslant i \leqslant m}$ is a PCB for G rel $\operatorname{Ker} \psi$.

Our computation of PCBs hinges on the following proposition, which characterizes a normal subgroup K in terms of a PCB for G rel K.

PROPOSITION 3.2. *Let K be a normal subgroup of G, and let $\{(b_i, \rho_i)\}_{1 \leqslant i \leqslant m}$ be a PCB for G* rel *K. Denote by S the set of images under SIFT of the set comprised of generators for G, of commutators $[b_j, b_i]$, $1 \leqslant i < j \leqslant m$, and of powers $b_i^{\rho_i}$, $1 \leqslant i \leqslant m$. Then $K = \mathrm{NCL}_G(S)$.*

Proof. That $\mathrm{NCL}_G(S) \subseteq K$ follows by normality of K. So let $k \in K$. Since generators for G were sifted, k can be written as a product of PCB elements and of elements of S. Migrating occurrences of b_1 to the left (given that $b_1^{-1} s b_1 \in \mathrm{NCL}_G(S)$ whenever $s \in \mathrm{NCL}_G(S)$ and that $b_1^{-1} b_j b_1$ can be expressed without b_1 for $j > 1$, the latter because $[b_j, b_1] = b_2^* \cdots b_m^* s$ where $s = \mathrm{SIFT}([b_j, b_1])$ belongs to S by definition) and reducing the resulting exponent of b_1 modulo ρ_1 (reexpressing $b_1^{\rho_1}$ as required), then repeating for $b_2, b_3, \ldots, k$ is expressible as

$$b_1^{\varepsilon_1} b_2^{\varepsilon_2} \cdots b_m^{\varepsilon_m} \sigma, \qquad 0 \leqslant \varepsilon_i < \rho_i,\ 1 \leqslant i \leqslant m,$$

with $\sigma \in \mathrm{NCL}_G(S) \subseteq K$. It follows that $b_1^{\varepsilon_1} \cdots b_m^{\varepsilon_m} \in K$ and hence that

$\varepsilon_1 = \varepsilon_2 = \cdots = \varepsilon_m = 0$ (by the uniqueness criterion in the definition of PCB for G rel K). Therefore $k \in \mathrm{NCL}_G(S)$. Hence $K \subseteq \mathrm{NCL}_G(S)$. ∎

The next proposition will allow us to break up the computation of a PCB into several stages.

PROPOSITION 3.3. *Let $K_1 \leqslant K_2 \leqslant G$ with K_1 and K_2 each normal in G. Then a PCB for G rel K_1 is obtained by appending a PCB for K_2 rel K_1 to a PCB for G rel K_2.*

Proof. To show that $[b_j, b_i]$ is expressible appropriately when b_i belongs to the PCB for G rel K_2 and b_j to that for K_2 rel K_1, we appeal to the normality of K_2. Other properties are straightforward to verify. ∎

4. LINEAR ALGEBRA

This section is devoted to the problem LINEAR CLOSURE which consists of computing a basis for the smallest vector space containing a prescribed set of vectors and closed under the action of a prescribed set of linear transformations (all over a tiny field $\mathbf{Z}_p$). As we suggest below, LINEAR CLOSURE arises naturally during the computation of PCBs, though it seems to be of independent interest.

LEMMA 4.1. *Given a set T of $d \times d$ matrices over $\mathbf{Z}_p$ (p tiny) which includes the identity matrix, a basis for the linear span τ_i of all products of i matrices in T can be computed in NC, for $i = 1, 2, \ldots, d^2$.*

Proof. Observe that τ_{a+b} is spanned by the products of basis elements of τ_a and τ_b. Hence in j stages it is possible to compute a basis for the subspace τ_{2^j}, by forming in parallel at the jth stage the product of each pair of basis matrices for $\tau_{2^{j-1}}$, and then by computing in NC a basis for the new span using the techniques in [BoGaHo82] supplemented with [Mu87]. At most 2 log d stages therefore suffice to compute bases for each τ_i with i a power of 2. But then obtaining bases for the other τ_i can be done for each i in parallel, by combining the appropriate τ_{2^j}. ∎

THEOREM 4.2. *NC contains the following problem (LINEAR CLOSURE): Given a subset S of $\mathbf{Z}_p^d$ (p tiny), and a set T of linear transformations of $\mathbf{Z}_p^d$ (described by matrices), find (a basis for) the smallest subspace that contains S and that is closed under the action of T.*

Proof. First we obtain a basis B for the matrix algebra with identity, τ, generated by T, and second we compute a basis for $V = \mathrm{Span}(BS) \subseteq \mathbf{Z}_p^d$. Observe that a subspace of $\mathbf{Z}_p^d$ is closed under T if and only if it is closed under τ which is true if and only if it is closed under B. It follows that V is the desired subspace.

We analyse the parallel complexity of these two steps. We may assume that T contains the identity transformation for the addition of the latter does not affect the desired output. Thus, in the notation of the preceding lemma, $\tau_0 \subseteq \tau_1 \subseteq \tau_2 \cdots$. Since the space of linear transformations of $\mathbf{Z}_p^d$ has dimension d^2, there are at most d^2 strict inclusions in this sequence. But, as soon as $\tau_i = \tau_{i+1}$, the sequence is stable thereafter at τ. Thus $\tau = \tau_{d^2}$ and so computation of B is in NC by Lemma 4.1. Computation of a basis of V is then straightforward linear algebra, in NC by [Mu87]. ▮

The next technical lemma focuses on a scenario in which LINEAR CLOSURE arises in our algorithms. Let there be given:

— groups G and L with G normalizing L,

— a subset S of $G \cap L$, and

— a normal subgroup M of L that is also normalized by G and such that L/M is a direct product of cyclic groups of prime order (for various primes); we assume a representation domain in which the cyclic factors of L/M are known and in which we can work within L/M in NC.

LEMMA 4.3. *A PCB for* $\mathrm{NCL}_G(S)$ $\mathrm{rel}(G \cap M)$ *can be computed in NC.*

Proof. Let $h_1, h_2, \ldots, h_s \in L/M$ be elements of prime power order such that $L/M = \langle h_1 \rangle \times \langle h_2 \rangle \cdots \times \langle h_s \rangle$, and consider first the case in which L/M is an elementary Abelian p-group. For each generator g of G and for each h_i in parallel, we compute the image of h_i under the automorphism of L/M induced by conjugation by g, and we build matrix descriptions of the linear transformations induced by conjugation by generators of G. Since $\mathrm{NCL}_G(S)/M$ is the smallest $\mathbf{Z}_p$-subspace of L/M containing the $\mathbf{Z}_p$-vectors in SM and closed under the induced linear transformations, a basis for this smallest $\mathbf{Z}_p$-subspace is computable in NC by Theorem 4.2. An important remark is that all computations carried out within L/M are performed simultaneously, with inverse images, in L, so that the inverse images of the computed basis (which are automatically still in G) provide the desired PCB for $\mathrm{NCL}_G(S)$ $\mathrm{rel}(G \cap M)$.

Now suppose that various primes occur as orders of the elements h_i, and let p be such a prime. By raising an element in S to an appropriate power (see [McCo87]), we can get rid of all but the "p-part" of that element. Now since conjugation by G necessarily preserves the p-part of L/M, we can work on the "p-part" of $\mathrm{NCL}_G(S)/(G \cap M)$ independently, exactly as in the elementary Abelian case discussed above (using the "raised" elements of S instead of S directly). We do this in parallel for each prime occurring in L/M, and the union of the PCB elements computed independently for each prime from the desired PCB for $\mathrm{NCL}_G(S)$ $\mathrm{rel}(G \cap M)$. ▮

5. Nilpotent Groups

An essential tool for dealing with permutation groups is a membership test. The permutation group membership problem (GM) consists of determining whether a given permutation belongs to the generated group. Furst, Hopcroft and Luks showed that a variant of Sims' algorithm [Si70] for GM could be implemented in polynomial time [FuHoLu80a], but the only subcase known to be in NC was that of Abelian groups ([McCo83], see Introduction). [Mc84] further shows that GM for elementary Abelian groups is NC^1-equivalent to computing the rank of matrices over tiny fields $\mathbf{Z}_p$.

Asymptotically fast sequential algorithms for testing membership in an arbitrary permutation group G proceed by first constructing a linear length tower of subgroups of G fixing progressively more points [FuHoLu80a]. The set of "strong generators" computed by these algorithms is the union of complete sets of coset representatives for each successive quotient space in this tower. Not only does this procedure not seem to parallelize, but even if a set of strong generators were given as input, it would appear that one could only "sift" the test permutation through the underlying tower, one notch at a time, resulting in a linear time parallel solution at best.

Our new GM algorithm for nilpotent G proceeds instead by computing a PCB for G. This is done by constructing a (normal) series of length $\log n$, where n is the size of the point set on which G acts, through which it is possible to "sift" group elements in NC. For nilpotent groups, the linear length tower of the last paragraph, fixing progressively more points, will be computable in NC also, but only following the development of our pointwise set stabilizer algorithm in the next section.

THEOREM 5.1. *Computing a* PCB *for a nilpotent group G belongs to NC.*

Proof. Let F be the structure forest for G as computed per Proposition 3.1. Consider level i, $0 \leqslant i \leqslant \log n$, as the level of all the nodes at distance i from a root in F (n is the degree of G). Denote the action of G on nodes at level $\leqslant i$ by ϕ_i. Note that the kernel of this action, $\operatorname{Ker} \phi_i$, fixes all nodes at height $\leqslant i$ and that $G/\operatorname{Ker} \phi_i$ may be viewed as the induced action on the forest obtained by pruning all trees to height i. These kernels form a $\log n$ height group tower of normal subgroups of G and we proceed, inductively, finding PCBs for the quotients $G/\operatorname{Ker} \phi_i$ (employing Proposition 3.3). So suppose inductively that we have a PCB for G rel $\operatorname{Ker} \phi_k$, and that with respect to this PCB we can compute SIFT in NC. We write S for a known set (computed in NC by sifting, Proposition 3.2) for which $\operatorname{Ker} \phi_k = \mathrm{NCL}_G(S)$. It suffices to show how to extend our PCB to a PCB for G rel $\operatorname{Ker} \phi_{k+1}$.

At each level k node in parallel we compute in NC Schreier generators for the subgroup of G stabilizing that node. To see how to do this for the stabilizer G_α of a node α, observe that a complete set of coset representatives for G_α in G, with an easy-to-compute representative function, is obtained by keeping track of an element of G sending α to β for each node β found to belong to the G-orbit containing α (as

part of an NC computation). Note that the number of Schreier generators for each stabilizer of a level k node is at most the number of generators of G times the number of level k nodes. Each stabilizer acts primitively, thus as a cyclic group of prime order, on the children of the corresponding level k node. We form a group L, which G normalizes, by taking the direct product of these cyclic constituents of the stabilizers. (There are other ways to obtain a suitable L; we follow this one with a view toward the generalization to the general solvable case.) Now, writing M for $\mathrm{Ker}\,\phi_{k+1}$ and observing that L/M acts faithfully on the level $k+1$ nodes, we find ourselves in the scenario described prior to Lemma 4.3. Hence we apply that proposition in order to compute a PCB for $\mathrm{NCL}_G(S)$ rel$(G \cap M)$ in NC. Finally, since $\mathrm{NCL}_G(S) = \mathrm{Ker}\,\phi_k$ and $G \cap M = \mathrm{Ker}\,\phi_{k+1}$, we can append the PCB for $\mathrm{Ker}\,\phi_k$ rel $\mathrm{Ker}\,\phi_{k+1}$ to that for G rel $\mathrm{Ker}\,\phi_k$ (already available inductively), obtaining a PCB for G rel $\mathrm{Ker}\,\phi_{k+1}$ by Proposition 3.3. We point out that one can sift through the new PCB by sifting, in succession, through the two PCBs that form it; hence, the process remains in NC (sifting through the first PCB is in NC by the induction hypothesis, and sifting through the second PCB involves expressing an arbitrary element of L/M in terms of the generators of the cyclic factors of L/M, also in NC by [McCo87]). ▮

We can thus settle an open question from [McCo87]:

COROLLARY 5.2. *In a nilpotent group G, order computation and membership testing belong to NC. This holds, in particular, when G is a p-group.*

Proof. The order is readily computed from a PCB for G. To test membership of t in G we "sift" through the logarithmic length series of groups $\mathrm{Ker}\,\phi_k$ (fixing successive levels in a structure forest for G and explicitly known from the PCB for G) as follows. In the notation of the proof of Theorem 5.1, assume inductively that an element $g_k \in G$ is known satisfying $tg_k^{-1} \in \mathrm{Ker}\,\phi_k$ (that is, g_k and t have identical actions on the structure forest truncated to height k). The next "sifting" stage seeks $g_{k+1} \in G$ satisfying $tg_{k+1}^{-1} \in \mathrm{Ker}\,\phi_{k+1}$. For this we seek a solution $h \in \mathrm{Ker}\,\phi_k$ (if one exists) to the congruence $h \equiv tg_k^{-1} \bmod(\mathrm{Ker}\,\phi_{k+1})$; this amounts to finding a solution (if one exists) to a system of non-homogeneous linear equations in the direct product of vector spaces $\mathrm{Ker}\,\phi_k/\mathrm{Ker}\,\phi_{k+1}$, for which a convenient representation domain is available in the level $k+1$ nodes of the structure forest. This system can be solved in NC by [Mu87; BoGaHo82]. If h is found then $g_{k+1} = hg_k \in G$ satisfies $tg_{k+1}^{-1} \in \mathrm{Ker}\,\phi_{k+1}$; otherwise we conclude $t \notin G$. ▮

The following result forms an important tool in several subsequent algorithms.

THEOREM 5.3. *Computing the normal closure $\mathrm{NCL}_G(H)$ of a subgroup H of a nilpotent permutation group G belongs to NC.*

Proof. Writing $\mathbf{N}$ for $\mathrm{NCL}_G(H)$, we compute a PCB for $\mathbf{N}$ by duplicating the strategy described in the proof of Theorem 5.1. What changes is the specification of the elements whose images under SIFT, given a PCB for $\mathbf{N}$ rel$(\mathrm{Ker}\,\phi_k \cap \mathbf{N})$, form a

set S for which $\mathrm{Ker}\,\phi_k \cap \mathbf{N} = \mathrm{NCL}_G(S)$ ($\mathrm{Ker}\,\phi_k$ is the kernel which "shrinks" from G to 1 in $O(\log n)$ stages). The proof of Proposition 3.2 extends provided we now sift:

(i) the commutators and powers (as before) of the PCB elements of $\mathbf{N}$ rel($\mathrm{Ker}\,\phi_k \cap \mathbf{N}$),

(ii) the generators of H,

(iii) each PCB element conjugated by each generator of G.

The sifting of set (iii) guarantees normality (of $\mathbf{N}/(\mathbf{N} \cap \mathrm{Ker}\,\phi_{k+1})$ in $G/\mathrm{Ker}\,\phi_{k+1}$) after the spanned "vector space" is closed up under the action of G and its basis tacked onto the PCB for $\mathbf{N}/(\mathbf{N} \cap \mathrm{Ker}\,\phi_k)$. ▮

Let G be a permutation group specified by generators and let ϕ be an NC-computable G-action (that is, the permutation $\phi(g)$ is computable in NC for any $g \in G$). Problem KERNEL consists of computing $\mathrm{Ker}\,\phi$.

THEOREM 5.4. *When G is nilpotent KERNEL is in NC.*

Proof. In NC we can compute a PCB for $\phi(G)$ by Theorem 5.1 and hence also a PCB for G rel $\mathrm{Ker}\,\phi$ by having kept track of inverse images. Then sifting through the latter PCB yields S such that $\mathrm{Ker}\,\phi = \mathrm{NCL}_G(S)$ (Proposition 3.2), from which we compute $\mathrm{Ker}\,\phi$ by Theorem 5.3. ▮

THEOREM 5.5. *Let G be a permutation group in a class of groups X and let H be an* arbitrary *permutation group such that G normalizes H. Then computing the centralizer $C_G(H)$ NC^2-reduces to solving KERNEL for the class X.*

Proof. The technique is a parallelized version of an algorithm in [Lu87]. Write C for $C_G(H)$ and Ω for the relevant point set. We form, for each generator h of H, the set

$$\Gamma_h = \{(x, x^h) \mid x \in \Omega\} \subseteq \Omega \times \Omega.$$

Observing that $g \in G$ commutes with a generator h of H if and only if g (on $\Omega \times \Omega$) stabilizes Γ_h, imagine $\Omega \times \Omega$ colored (in NC) in such a way that two points share a color if and only if they belong to exactly the same set Γ_h. We claim that by refining the color partition (adding colors) until each intersection of a color class Γ with an image under G of some other class is either empty or equal to Γ (hence until each class is a G-block if we extend the definition of block to the case of non-transitive G), we maintain the property that $g \in G$ preserves each color class if and only if $g \in C$. We then obtain C as the kernel of the action of G on the set of color classes. It remains to analyze the complexity of the refinement process and to prove our claim.

As to the former, we begin by refining until each class is wholly contained in a G-orbit. Then we work on each G-orbit in parallel, seeking in parallel a non-trivial

intersection between certain images under G of the color classes. To determine these images, first expand the set of generators of G to a collection Φ that includes, for each x, y an element $g \in G$ such that $x^g = y$. In each round (of at most $2 \log n$ rounds), we use the images under Φ of the color classes to refine the color partition. Continue until the set of color classes is stable under the action of Φ (thus forming a G-block system on the orbit). We measure the progress in terms of the size m of the smallest color class. Suppose we do not yet have a block system. If all classes have size m, at least one class will be partitioned in the round and at least one part has size $\leqslant m/2$. If the classes are not uniform in size then the images of a size m class will cut all classes to size $\leqslant m$ in the round; if any of these are strictly smaller than m then the next round will partition the original size m class (if that has not already happened) and at least one segment has size $\leqslant m/2$.

To prove our claim, note that G normalizing H implies that C is normal in G. Hence if C preserves color class Γ then, for any $g \in G$, $\Gamma^{gC} = \Gamma^{g(g^{-1}Cg)} = \Gamma^g$ is preserved by C also. In other words we lose no element of C if we insist on preserving not only each original color class Γ but Γ^g for each $g \in G$ as well. That is, we lose no element of C if we refine until each class is a G-block. ∎

COROLLARY 5.6. *Let H be an* arbitrary *permutation group normalized by a nilpotent permutation group G. Computing the centralizer $C_G(H)$ belongs to NC.* ∎

A special case of Corollary 5.6 involves finding centralizers of normal subgroups (when $H < G$) and, if $H = G$, we get an important structural result:

COROLLARY 5.7. *The center of a nilpotent permutation group can be computed in NC.* ∎

Even more structural information on a nilpotent group is attainable:

THEOREM 5.8. *A central series for a nilpotent group can be computed in NC.*

Proof. Suppose $G = \langle S \rangle$. We may assume that G is a p-group, for general nilpotent G can be factored in NC as a direct product of p-groups [Mc84] and it is an easy matter to reassemble central series of the factors into one for G. In the notation of the proof of Theorem 5.1, we denote $\operatorname{Ker} \phi_k$ by $G^{(k)}$. Then we have constructed a normal series

$$G = G^{(0)} \geqslant G^{(1)} \geqslant \cdots \geqslant G^{(s)} = 1$$

in which each quotient $G^{(k)}/G^{(k+1)}$, denoted below by V, is an elementary Abelian p-group (vector space over $\mathbf{Z}_p$). Furthermore, we have a convenient representation domain in which to work with V in the k, $k+1$ slice of the structure forest. We need to show that, for each k, we can insert G-normal subgroups

$$G^{(k)} = H^{(0)} \geqslant H^{(1)} \geqslant \cdots \geqslant H^{(m)} = G^{(k+1)}$$

so that $[G, H^{(j)}] \leqslant H^{(j+1)}$. Equivalently, we need to insert G-invariant subspaces

$$V = V^{(0)} \geqslant V^{(1)} \geqslant \cdots \geqslant V^{(m)} = 0$$

so that, for $g \in G$, $v \in V^{(j)}$, $v - v^g$ is in $V^{(j+1)}$ (note that we switch to the additive notation in V in viewing $[g, v]$). Each g in G induces a linear transformation t_g (describable in NC as an $\mathbf{Z}_p$-matrix, in the spirit of the proof of Lemma 4.3) of V where $t_g(v) = v - v^g$. Let $T = \{t_s \mid s \in S\}$. Using Lemma 4.1 we compute τ_i, the linear span of the set of all products of i elements from T, for all $i = 1, 2, \ldots, \dim(V)$. We claim that we may take $V^{(j)} = \tau_j(V)$. It is immediate that $V^{(j+1)} = \text{Span } T(V^{(j)})$, so that $V^{(j)} \geqslant V^{(j+1)}$ follows inductively from $V \geqslant V^{(1)}$. To see that $V^{(j)}$ is G-invariant, it suffices to note that it is invariant under S, but, for $s \in S$, $v \in V^{(j)}$,

$$v^s = v - t_s(v) \in V^{(j)} + V^{(j+1)} \leqslant V^{(j)}.$$

This equation also gives the congruence

$$v^g \equiv v \bmod(V^{(j+1)})$$

for all g in a generating set, and so the congruence holds for all $g \in G$, whence $v - v^g \in V^{(j+1)}$. Finally, we need to show that, if $m = \dim(V)$, $V^{(m)} = 0$. For this, recall that the nilpotency of G implies that there is an M so that, for all h, h_1, $h_2, \ldots, h_M \in G$,

$$[h_M \cdots [h_2, [h_1, h]] \ldots] = 1.$$

But this implies that $\tau_M = 0$. Knowing, then, that the sequence $V^{(0)} \geqslant V^{(1)} \geqslant V^{(2)} \ldots$ will reach 0 eventually, we must simply conclude that this happens within m steps. For this, observe that once equality $V^{(j)} = V^{(j+1)}$ happens, then the sequence is stable (by induction) thereafter. But the sequence of dimensions $m = \dim V^{(0)}$, $\dim V^{(1)}$, $\dim V^{(2)}, \ldots$ can strictly decrease at most m times. ∎

We remark that the above proof may be extended to produce a central composition series by inserting, if necessary, arbitrary intermediate spaces so that dimensions go down by 1 in each step.

6. Pointwise Set Stabilizers for Nilpotent Groups

Consider the pointwise set stabilizer problem (POINTSET). Given a permutation group G and a subset of the points on which G acts, POINTSET consists of computing the largest subgroup of G fixing each point in the subset. (By contrast, the set stabilizer problem SET permits mapping points in the subset to other points in the subset.) Theorem 5.3 is instrumental in the proof of

Theorem 6.1. *POINTSET for nilpotent groups belongs to NC.*

Proof. Initially we mark nodes, in the structure forest F for G, which subtend leaves to be fixed. From generators for the group G_k fixing the marked nodes at level k (available inductively), we compute generators for G_{k+1}, again by looking at the induced action on the level k, $k+1$ trees extracted from F (but this time only those with marked roots, noting that G_k fixes these roots and that an unmarked root cannot have marked descendants). Iterating this process eventually yields generators for the group fixing exactly the marked leaves.

To describe how to compute generators for G_{k+1} from generators for G_k, consider the G_k-action on the aforementioned trees. Observe that the induced action of G_k on the children (if any) of a marked (thus fixed) level k node is trivial or cyclic of prime order. Hence, the subgroup fixing a marked child of this node in fact fixes all the children. Thus, G_{k+1} is the kernel of the induced action on the set of children of marked level k nodes and is obtainable by Theorem 5.4. ▮

Remark 6.2. The algorithm in the above proof exploits the fact that a pointwise-set-stabilizer H in the nilpotent permutation group G lies in a series that is subnormal (i.e., each group is normal in the next)

$$H = G_m < \cdots G_1 < G.$$

(Of course, it is also fortunate that, for pointwise-set-stabilizers, $m = O(\log n)$). In fact, the class of nilpotent groups is characterizable by the property that all subgroups lie in a subnormal series. Since any subgroup H of a group G turns up as the pointwise-set-stabilizer in some representation (e.g., of G acting on the set of right cosets of H), we see that the algorithm is necessarily invalid for non-nilpotent groups. Thus, while we extend the results of section 5, including kernels, to solvable groups, POINTSET for that class has to await more powerful methods (see [Lu86]).

Pointwise set stabilizers play an early and important role in the development of fast sequential algorithms [FuHoLu80a]. Though they arrive here at a much later stage, they are still of great value. For example, they can serve to solve special cases of the setwise stabilizer problem SET.

THEOREM 6.3. *Consider a class of nilpotent permutation groups for which the size of each transitive constituent of a group is polynomially bounded in the degree of the group. Then SET for this class belongs to NC.*

Proof. Let G be a group from the class and let Γ be the set to be stabilized. For each G-orbit Δ in parallel, we can compute (by enumeration of the small constituent in NC) the subgroup H of G^{Δ} which setwise stabilizes $\Gamma \cap \Delta$ and we can describe the (right) action of G^{Δ} on the cosets of H. Computing G_{Γ} then is an instance of POINTSET for nilpotent groups if we extend the action of G to the union of all such cosets (one group H per G-orbit Δ) and if we take the trivial cosets as the points to be fixed. We conclude by Theorem 6.1. ▮

The class of Abelian permutation groups satisfies the hypothesis of Theorem 6.3 because the order of a transitive Abelian group equals its degree [Wi64, p. 9]. Hence Theorem 6.3 implies a result of [McCo87] that SET for Abelian groups is in NC.

We turn to an application of our POINTSET algorithm to computing automorphism groups of graphs. Define **N** to be the class of vertex-colored graphs X for which $\mathrm{Aut}(X)^C$ is contained in a small (i.e., order polynomial in the size of the graph) nilpotent group that is computable in NC, for each color class C.

EXAMPLES. The automorphism group within each color class is nilpotent if, for example,

— it is a directed cycle (cyclic group)

— it is a connected trivalent graph with a distinguished edge (2-group)

— it is a p-ary tree with a cyclic orientation imposed on the children of each node (p-group).

Let m be the size of the color class. In the first example the group is transparent and of order m. In the second, it is computable in sequential time $O(m^3)$ and has size 2^r for some $r \leqslant m/2$ [GHLSW87], so we could allow m to be as large as $O(\log n)$. In the third, the conditions are satisfied with $m = O(\log_p n)$. Note that for a graph to be in **N** it is not essential that every color class be restricted; the containing group might be influenced by interconnections with other classes.

THEOREM 6.4. *If a graph is in* **N** *then generators for its automorphism group can be computed in NC.*

Proof. Let V be the set of vertices of a graph X in **N**, and write G for the direct product over each color class C of the small NC-computable nilpotent group containing $\mathrm{Aut}(X)^C$. Considering the action of nilpotent G on $V \times V$, observe that the restriction of G to a pair of color classes is small (being a small group or the direct product of two small groups). Hence G on $V \times V$ has small transitive constituents, and $\mathrm{Aut}(X)$ is obtainable in NC as the set stabilizer of the subset of $V \times V$ corresponding to edges in X, using Theorem 6.3. ∎

Instances of computability of automorphism groups typically facilitate isomorphism testing [Lu82]; for example, if one can compute the automorphism group of the disjoint union of two connected graphs then isomorphism is tested by seeing whether an automorphism switches the connected components (note, in the above examples, that the disjoint union of connected trivalent graphs with distinguished edges has a 2-group for automorphism group). Hence an NC isomorphism test follows directly from Theorem 6.4 if the union of the graphs belong to **N**. Unfortunately the disjoint union, for example, of two directed cycles does not have a nilpotent automorphism group (except if the cycles have size 2^c). One can handle the (solvable) group that arises using deeper techniques developed

in [Lu86]. However, it is worth noting that there is a direct extension of our automorphism group technique to isomorphism-testing that allows us to stay within the class of nilpotent groups.

First we adapt our techniques to solve generalizations of the POINTSET and SET problems. We generalize POINTSET to the POINTSET TRANSPORTER problem: given a group G acting on Ω and two *ordered* subsets $A = \{a_1, ..., a_m\}$ and $B = \{b_1, ..., b_m\}$ of Ω, compute $\{g \in G \mid A^g = B \text{ as } \textit{ordered} \text{ sets}\}$.

THEOREM 6.5. *POINTSET TRANSPORTER for nilpotent groups belongs to NC.*

Proof. Clearly the desired subset of G, if not empty, is the coset Hg for any $g \in G$ mapping A to B pointwise and for H the pointwise stabilizer of A. To obtain such a g (or to prove that none exist), we make use of the subnormal series from G to H formed by the groups G_k obtained, together with a structure forest for G, as a by-product of the NC-computation of H (see proof of Theorem 6.1). As in membership testing (see proof of Corollary 5.2) we "sift" through this series, except that here we begin with the map $t: A \to B$ which sends a_i to b_i for each i (observe that t is only a partial map from Ω to Ω). More precisely, we first verify in NC that the t-induced relation between the set of ancestors of elements of A and the set of ancestors of elements of B in the structure forest is a bijection which preserves G-orbits (in the negative case Φ is immediately returned as answer). Then we carry out the "sifting" as if t were a permutation, noting that only the immediate descendants of the level k ancestors of A (i.e., of the "marked" nodes at level k in Theorem 6.1) play a role at stage k in setting up the system of equations ruling the existence of $h \in G_k$ with an action identical to that of tg_k^{-1} on the level $k+1$ ancestors of A (where $g_k \in G$ with an action identical to that of t on the level k ancestors of A is available inductively). ∎

In a similar spirit we generalize SET to the SET TRANSPORTER problem, defined like the POINTSET TRANSPORTER problem but with A and B viewed as (usual, unordered) sets.

THEOREM 6.6. *In a class of nilpotent permutation groups for which the size of each transitive constituent of a group is polynomially bounded in the degree of the group, SET TRANSPORTER can be solved in NC.*

Proof. If not empty the answer is again a coset, this time $(G_A)g$, for any $g \in G$ mapping A to B (setwise). A straightforward extension of the proof of Theorem 6.3 produces in this case an instance of POINTSET TRANSPORTER (here we need to compute for each G-orbit Δ, in addition to the cosets of $G^{\Delta}_{\Delta \cap A}$ in G^{Δ}, an element $h \in G^{\Delta}$ mapping $\Delta \cap A$ to $\Delta \cap B$; these elements h prescribe the target cosets to be used in the POINTSET TRANSPORTER instance, and if no such h exists for some G-orbit then Φ is returned as answer). We conclude by Theorem 6.5. ∎

We remark that special cases of Theorems 6.5 and 6.6 concern Abelian groups, for which NC algorithms for POINTSET TRANSPORTER and SET TRANSPORTER do not seem to follow from [McCo87]. We also point out that the proof of Theorem 6.6 extends to solve (within the same class of groups) the MULTIPLE SET TRANSPORTER problem, in which it is required to transport, simultaneously, more than one set to a corresponding target. Finally, note that the proofs of Theorem 6.3 and 6.6 in effect describe NC^2 reductions from SET to POINTSET and from MULTIPLE SET TRANSPORTER to POINTSET TRANSPORTER in the class of nilpotent groups with polynomial-size transitive constituents. We observe that the same reductions carry over to any class of groups similarly restricted to having small transitive constituents.

We return now to our goal of testing isomorphism between two graphs in the class **N**. Suppose that X and Y are graphs in **N** for which we know a single isomorphism in each color class of X to the corresponding color class of Y. Then

THEOREM 6.7. *X and Y can be tested for isomorphism in NC.*

Proof. We imitate the algorithm for $\mathrm{Aut}(X)$ in Theorem 6.4 to compute $\mathrm{Iso}(X, Y)$, the set of all isomorphisms from X to Y, directly (an analogous approach is exploited in [GHLSW87]). Forming G as in the proof of Theorem 6.4, and gluing together the known isomorphisms between corresponding pairs of color classes, we form a set Gf that contains $\mathrm{Iso}(X, Y)$. As before, G acting on all pairs of vertices of X has small transitive constituents. Furthermore, $\mathrm{Iso}(X, Y)$ is the set of those gf in Gf which map edges of X to edges of Y, or equivalently $\mathrm{Iso}(X, Y) = Hf$ for H the subset of G comprised of those elements which map the set of edges of X to the image under f^{-1} of the set of edges of Y. In other words computing $\mathrm{Iso}(X, Y)$ reduces to an instance of SET TRANSPORTER involving G, and we conclude by Theorem 6.6. ∎

7. SOLVABLE GROUPS

First we aim at showing how to test solvability in NC, and we begin by proving a more general result.

Define a property as "hereditary" if whenever the property holds for a group G it holds for any subgroup and any quotient group of G, and whenever the property holds for both a normal subgroup N of G and for G/N it holds for G. Thus a hereditary property holds precisely for the class of groups with composition factors within some fixed collection of simple groups. Examples of hereditary properties include solvability, being a p-group, having bounded non-Abelian composition factors.

THEOREM 7.1. *Testing a permutation group for a hereditary property P NC^2-reduces to testing P for a primitive group.*

Proof. Generalizing a technique used in [Mc84] to test nilpotency in NC, we compute a structure forest for the input group G (Proposition 3.1). Write S for the set of nodes at level 1, that is, for the set of immediate descendants of the roots. Then P holds for G if and only if P holds for the G-action on S as well as for each stabilizer within G of a node in S restricted to the leaves subtended. To see this note that the direct product of the (restricted) stabilizers of each node in S contains the kernel of the G-action on S. Now testing the G-action on S for P can be done for each (primitive) transitive constituent in parallel (seeing that the direct product of these constituents includes the G-action). As for the stabilizers, we compute the Schreier generators for each of them in parallel (see proof of Theorem 5.1). Applying this argument recursively to each stabilizer in parallel (observing that we can always use G in the computation of Schreier generators), eventually we reach the bottom of the structure forest with a generating set bounded in size by the number of generators of G times the number of leaves in the forest. ▮

COROLLARY 7.2. *Testing a permutation group for solvability belongs to NC.*

Proof. Solvability is a hereditary property and the Pálfy–Wolf bound reduces testing solvability of a primitive solvable group to testing solvability of a group having order polynomial in its degree (hence in NC for one can generate the entire group). ▮

Since it is known that a bound on the non-Abelian composition factors imposes a polynomial-bound on the size of a primitive permutation group [BaCaPa82], testing for bounded non-Abelian composition factors is also in NC. We observe that this class of groups arises in testing isomorphism of graphs of bounded valence [Lu82].

Remark 7.3. The basic divide-and-conquer technique used in the proof of Theorem 7.1 has other applications. Suppose we consider the question of whether a prime p divides the order of G. This time we observe that property holds for G if and only if it holds for at least one transitive constituent, and it holds for a transitive group if and only if it holds either for the (primitive) group acting on a set of maximal blocks or for the subgroup fixing one block, in its action within that block. Consider, for example, groups with bounded non-Abelian composition factors. It is known [BaCaPa82] that primitive groups in this class have polynomially bounded order. Thus testing whether p divides the order is in NC. (It is now known that the exact order of groups is obtainable in NC. This was shown for groups with bounded non-Abelian composition factors in [Lu86] and extended to general groups in [BaLuSe87].) ▮

We now undertake the task of generalizing some of the results of Section 5 to the case of solvable groups. Not surprisingly, the first step in that direction is the computation of PCBs. Recall that the central ingredient to our PCB algorithm for nilpotent G (Theorem 5.1) was a normal series for G, of length $O(\log n)$, with Abelian factors. In the case of solvable G, the factors in the corresponding

logarithmic length series are not necessarily Abelian. However properties of solvable groups allow us to refine this series into a normal series with Abelian factors, this time of length $O((\log n)^2)$, on which to cast our solvable PCB algorithm. We give details in proving

THEOREM 7.4. *A PCB for a solvable permutation group can be computed in NC.*

Proof. Recall the proof of Theorem 5.1. The overall strategy here is identical to that described in the first paragraph of that proof. Using the same notation, let it be required to extend a PCB for G rel $\mathrm{Ker}\,\phi_k$, through which it is possible to sift in NC (inductively as before), to a PCB for G rel $\mathrm{Ker}\,\phi_{k+1}$. To complete the present proof it suffices to show how to perform this extension in NC.

Here the step from $\mathrm{Ker}\,\phi_k$ to $\mathrm{Ker}\,\phi_{k+1}$ is still too large ($\mathrm{Ker}\,\phi_k/\mathrm{Ker}\,\phi_{k+1}$ is not necessarily Abelian, let alone a vector space), so we need to refine this step of the normal series, inserting $O(\log n)$ groups whose successive factor groups can be viewed as vector spaces. We will proceed in effect by computing, using the known PCB for G rel $\mathrm{Ker}\,\phi_k$, a PCB for the action of G on the forest obtained by pruning all trees to level $k+1$; clearly this will yield a PCB for G rel $\mathrm{Ker}\,\phi_{k+1}$. Throughout the rest of this proof we therefore view G and $\mathrm{Ker}\,\phi_k$ as acting on the pruned trees, though in actual computations we also keep track of the faithful action of G on the original domain. We write K for $\mathrm{Ker}\,\phi_k$.

To describe the first stage, we compute, as in the nilpotent case, the stabilizers of the level k nodes and we denote by L their direct product (each factor acting only on the relevant level $k+1$ children), which is normalized by G as before. Then we form, as described next, a direct product L/M of elementary Abelian groups. At each level k node where L is non-trivial (*non-trivial* required for sufficient progress to be made at each stage) the group T induced by L on the children of that node contains a subgroup $U = \langle T^p, [T, T] \rangle$ (for p an appropriately chosen prime and for T^p the set of pth powers of elements of T), such that T/U is a direct product of cyclic groups of order p, i.e., a vector space over $\mathbf{Z}_p$. The group T is small by the Pálfy–Wolf bound and one can pick p in NC so that $U \neq T$. Note, for the purpose of computing M, that we wish to preserve the conjugation action of G on M. Hence it is important that the same "nontrivial" p be chosen when constructing the components of M corresponding to each level k node in a G-orbit (that is, to each level k node appearing in the same tree of the structure forest). This can be accomplished in NC using that all components of L corresponding to a given G-orbit are conjugate. Hence in practice only one component of M per G-orbit is obtained by brute force; the rest are obtained by "replication" via conjugation by appropriate (easily computed) elements of G. The upshot of this are groups L and M, each normalized by G, with L/M a direct product of known cyclic groups of prime order.

But then we are in the scenario described prior to Lemma 4.3, recalling that from the PCB for G rel K (available inductively) we can compute S such that $K = \mathrm{NCL}_G(S)$. Hence a PCB for K rel$(G \cap M)$ is computable in NC which, having

retained the full action on the leaves, we append to the PCB for G rel K, appealing to Proposition 3.2. This completes the first stage.

The next stage begins by replacing K by $(G \cap M)$ and L by M. Since M is still normalized by G, we can iterate the above computations to obtain the next "chunk" in the PCB for G rel $\mathrm{Ker}\,\phi_{k+1}$. We keep repeating these stages, pasting together the PCB chunks produced, until M becomes trivial (this occurs within $O(\log n)$ stages since at each stage the order of each component of L acting nontrivially on the level $k+1$ nodes is at least halved). When that happens we have the desired PCB for G rel $\mathrm{Ker}\,\phi_{k+1}$. ▮

As in the nilpotent case, it follows that

COROLLARY 7.5. *In a solvable group, order computation and membership testing can be done in NC.* ▮

Substituting the $O(\log^2 n)$ subgroup tower in the proof of Theorem 7.4, the proof of Theorem 5.3 extends directly to yield

THEOREM 7.6. *Computing the normal closure* $\mathrm{NCL}_G(H)$ *of a subgroup* H *of a solvable permutation group* G *belongs to NC.* ▮

Recall problem KERNEL which consists of computing the kernel of a G-action. As in the nilpotent case (Theorem 5.4), we obtain

COROLLARY 7.7. *For solvable groups KERNEL is in NC.* ▮

Applying Theorem 5.5, Corollaries 5.6 and 5.7 therefore also have analogs in the solvable case. In particular,

COROLLARY 7.8. *The center of a solvable permutation group can be computed in NC.* ▮

Normal closures in fact enable us to get at more of the structural underpinnings of the group.

COROLLARY 7.9. *Computing the derived series and a composition series of a solvable permutation group belongs to NC.*

Proof. The length of the derived series is $O(\log^2 n)$ (this is an easy consequence of the Pálfy–Wolfe bound on primitive solvable groups). Then successive commutator subgroups are obtained from Theorem 7.6 using the fact that $[G, G] = \mathrm{NCL}_G\{[g, h] \mid g, h \in S\}$ whenever S is a generating set for G. Now a composition series for G is formed by the subgroups $\langle \{b_j\}_{j \geqslant i} \rangle$ for $1 \leqslant i \leqslant m$ in the PCB computed in the proof of Theorem 7.4, since the ρ_is are prime integers. ▮

Note, though the derived series has poly-log length, a composition series need not.

8. Comments and Questions

As mentioned earlier, the results of this paper have now been extended to general permutation groups [Lu86; BaLuSe87]. In particular, order computation, membership-testing, and POINTSET, determination of composition factors, are shown to be in NC. However, there remain fundamental questions that are open even for nilpotent or solvable groups. We mention two favorites.

Of particular interest is the parallel complexity of set stabilizer (SET). At this time, we hesitate to recommend SET for general groups, since that problem is not even known to be in P, indeed, if it were then graph isomorphism would be in P [Lu82]. On the other hand SET is known to be in P for solvable groups [Lu82]. Nevertheless, only the case of Abelian groups is known to be in NC (see [Mc84; McCo87] or Section 6 of the present paper). It would be significant to extend this at least to nilpotent groups. By way of motivation, we mention that trivalent graph isomorphism NC reduces to SET for 2-groups [Lu86]. As with pointwise set-stabilizers (Section 6), set stabilizers in nilpotent groups can be located in a sequence of subgroups, each a kernel of a constructible action of its predecessor. However, unlike the construction for POINTSET, these subnormal series need not have poly-log length.

An interesting open question for solvable groups is the parallel complexity of the problem of finding Sylow p-subgroups. Kantor [Ka85a; Ka85b] has shown that Sylow p-subgroups of general permutation groups can be found in polynomial time (this was actually first done for the solvable case [KaTa]). In [Mc84] it was shown that, for nilpotent groups, Sylow p-subgroups are attainable in NC. Can this be done for solvable, or even more general, groups? Can one even find an element of order p in a general group, having observed that p divides $|G|$?

Note added in proof. The first author has shown that the problem of finding Sylow subgroups is in NC for solvable groups. However, it still appears open for general permutation groups.

References

[Ar42] E. Artin, "Galois Theory," Notre Dame Mathematical Lectures 2, 2nd ed., 1944.

[At75] M. D. Atkinson, An algorithm for finding the blocks of a permutation group, *Math. Comp.* **29** (1975), 911–913.

[AvMa84a] J. Avenhaus and K. Madlener, The Nielsen reduction and P-complete problems in free groups, *Theoret. Comput. Sci.* **32** (1984), 61–76.

[AvMa84b] J. Avenhaus and K. Madlener, On the complexity of intersection and conjugacy in free groups, *Theoret. Comput. Sci.* **32** (1984), 279–295.

[Ba79] L. Babai, "Monte Carlo Algorithms in Graph Isomorphism Testing," Tech. Rep. 79-10, Dép. Math. et Stat., Univ. de Montréal, 1979.

[Ba85] L. Babai, Trading group theory for randomness, *in* "Proceedings, 17th Annual ACM Symp. on Theory of Computing, 1985," pp. 421–429.

[Ba86] L. BABAI, A Las Vegas-NC algorithm for isomorphism of graphs with bounded multiplicity of eigenvalues, *in* "Proceedings, 27th IEEE Symp. on Foundations of Computer Science, 1986," pp. 303–312.

[BaCaPa82] L. BABAI, P. J. CAMERON, AND P. PÁLFY, On the order of primitive groups with restricted non-Abelian composition factors, *J. Algebra* **79** (1982), 161–168.

[BaKaLu83] L. BABAI, W. M. KANTOR, AND E. M. LUKS, Computational complexity and the classification of finite simple groups, *in* "Proceedings, 24th IEEE Annual Symp. on Foundations of Computer Science, 1983," pp. 162–171.

[BaLuSe87] L. BABAI, E. M. LUKS, AND A. SERESS, Permutation groups in NC, *in* "Proceedings, 19th Annual ACM Symp. on the Theory of Computing, 1987," pp. 409–420.

[BaSz84] L. BABAI AND E. SZEMERÉDI, On the complexity of matrix group problems I, *in* "Proceedings, 25th IEEE Annual Symp. on Foundations of Computer Science, 1984," pp. 229–240.

[Bo77] A. BORODIN, On relating time and space to size and depth, *SIAM J. Comput.* **6** (1977), 733–744.

[BoGaHo82] A. BORODIN, J. VON ZUR GATHEN, AND J. HOPCROFT, Fast parallel matrix and GCD computations, *Inform. and Control* **52** (1982), 241–256.

[Ca84] J. J. CANNON, An introduction to the group theory language Cayley, *in* "Computational Group Theory, Proceedings, London Mathematical Society Symposium on Computational Group Theory," (M. D. Atkinson, Ed.), pp. 145–183, Academic Press, New York/London, 1984.

[Co81] S. A. COOK, Towards a complexity theory of synchronous parallel computation, *in Enseign. Mathé. (2)* **27** (1981), 1–2.

[Co85] S. A. COOK, A taxonomy of problems with fast parallel algorithms, *Inform. and Control* **64** (1985), 2–22.

[FiPi74] M. FISCHER AND N. PIPPENGER, "M. J. Fisher Lecture Notes on Network Complexity," Universitat Frankfurt, 1974.

[FuHoLu80a] M. FURST, J. HOPCROFT, AND E. LUKS, Polynomial time algorithms for permutation groups, *in* "Proceedings, 21st IEEE Annual Symp. on Foundations of Computer Science, 1980," pp. 36–41.

[FuHoLu80b] M. FURST, J. HOPCROFT, AND E. LUKS," A Subexponential Algorithm for Trivalent Graph Isomorphism," Tech. Rep. 80-426, Dept. of Computer Science, Cornell University, 1980.

[GHLSW87] Z. GALIL, C. M. HOFFMANN, E. M. LUKS, C. P. SCHNORR, AND A. WEBER, An $O(n^3 \log n)$ deterministic and an $O(n^3)$ Las Vegas isomorphism test for trivalent graphs, *J. Assoc. Comput. Mach.* **34** (1987), 513–531.

[Ha59] M. HALL, "Theory of Groups," Macmillan Co., New York, 1959.

[Ho82] C. M. HOFFMANN, "Group-Theoretic Algorithms and Graph Isomorphism," Lecture Notes in Computer Science Vol. 136, Springer-Verlag, New York/Berlin, 1982.

[HoUl79] J. E. HOPCROFT AND J. D. ULLMAN, "Introduction to Automata Theory, Languages, and Computation," Addison-Wesley, Reading, MA, 1979.

[Ka85a] W. M. KANTOR, Polynomial-time algorithms for finding elements of prime order and Sylow subgroups, *J. Algorithms* **6** (1985), 478–514.

[Ka85b] W. M. KANTOR, Sylow's theorem in polynomial time. *J. Comput. System Sci.* **30** (1985), 359–394.

[KaTa] W. M. KANTOR AND D. E. TAYLOR, Polynomial-time versions of Sylow's theorem, *J. Algorithms*, in press.

[Lu82] E. M. LUKS, Isomorphism of graphs of bounded valence can be tested in polynomial time, *J. Comput. System Sci.* **25** (1982), 42–65.

[Lu86] E. M. LUKS, Parallel algorithms for permutation groups and graph isomorphism, *in* "Proceedings, 27th IEEE Symp. of Foundations of Computer Science, 1986," pp. 292–302.

[Lu87] E. M. LUKS, Computing composition factors of a permutation group in polynomial time, *Combinatorica* **7** (1987), 87–99.

[LuMc85] E. M. LUKS AND P. MCKENZIE, Fast parallel computation with permutation groups, *in* "Proceedings, 26th IEEE Annual Symp. on Foundations of Computer Science, 1985," pp. 505–514.

[Mc84] P. MCKENZIE, "Parallel Complexity and Permutation Groups," Doctoral thesis, Tech. Rep. 173/84, Dept. of Computer Science, University of Toronto, 1984.

[McCo83] P. MCKENZIE AND S. A. COOK, The parallel complexity of the Abelian permutation group membership problem, *in* "Proceedings, 24th IEEE Annual Symp. on Foundations of Computer Science, 1983," pp. 154–161.

[McCo87] P. MCKENZIE AND S. A. COOK, The parallel complexity of Abelian permutation group problems, *SIAM J. Comput.* **16**, No. 5 (1987).

[Mi83] G. L. MILLER, Isomorphism testing and canonical forms for k-contractable graphs, *in* "Proceedings, Fundamentals of Computation Theory, 1983," pp. 310–327.

[Mu87] K. MULMULEY, A fast parallel algorithm to compute the rank of a matrix over an arbitrary field, *Combinatorica* **7** (1987), 101–104.

[Pa82] P. PÁLFY, A polynomial bound on the orders of primitive solvable groups, *J. Algebra* **77** (1982), 127–137.

[Pi79] N. PIPPENGER, On simultaneous resource bounds, *in* "Proceedings, 20th IEEE Annual Symp. on Foundations of Computer Science, 1979," pp. 307–311.

[Re85] J. REIF, Probabilistic algorithms in group theory, *in* "Proceedings, 1985 Fundamentals of Computation Theory Conference," Lecture Notes in Computer Science Vol. 199, pp. 341–350, Springer-Verlag, New York/Berlin, 1985.

[Ro73] J. . ROTMAN, "The Theory of Groups," 2ed., Allyn & Bacon, Boston, 1973.

[Sc76] C. P. SCHNORR, The network complexity and the Turing machine complexity of finite functions, *Acta Inform.* **7** (1976), 95–107.

[Si67] C. C. SIMS, Graphs and finite permutation groups, *Math. Z.* **95** (1967), 76–87.

[Si70] C. C. SIMS, Computational methods in the study of permutation groups, *in* "Computational Problems in Abstract Algebra," (J. Leech, Ed.), pp. 169–183, Pergamon, Elmsford, NY, 1970.

[Wi64] H. WIELANDT, "Finite Permutation Groups," Academic Press, New York/London, 1964.

[Wo82] T. R. WOLF, Solvable and nilpotent subgroups of $GL(n, q^n)$, *Canad. J. Math.* **34** (1982), 1097–1111.