Solvable Black-Box Group Problems are Low for PP

V. Arvind and N. V. Vinodchandran
Institute of Mathematical Sciences, C.I.T Campus
Madras 600113, India
email: {arvind,vinod}@imsc.ernet.in

Abstract

Let $\mathcal{B} = \{B_m\}_{m>0}$ be a countable family of finite groups whose elements are uniquely encoded as strings of uniform length, and group operations are computable in time bounded by a polynomial in m. A *black-box* group over a group family $\mathcal{B}$ is a subgroup of some member of $\mathcal{B}$ and is presented by a generator set. In this paper we study the complexity of several algorithmic problems for abelian black-box groups and solvable black-box groups.

We design a suitable oracle algorithm that computes an independent set of generators for a given abelian black-box group. Using this we show that the problems of Membership Testing, Group Intersection, Order Verification, and Group Isomorphism over abelian black-box groups are in SPP. We also show that Group Factorization, Coset Intersection, and Double Coset Membership problems over abelian black-box groups are in LWPP. As a consequence, all these problems are low for PP and $\mathrm{C_=P}$.

We define the notion of *canonical* generator sets for classes of groups and show that solvable black-box groups have canonical generator sets. We design a suitable randomized oracle algorithm that computes a canonical generator set for a given solvable black-box group. Using this algorithm we show that Membership Testing, Order Verification, and Group Isomorphism for solvable groups are in $\mathrm{ZPP}^{\mathrm{SPP}}$ and hence low for PP. We also show that Group Intersection, Group Factorization, Coset Intersection, and Double Coset Membership for solvable groups are low for PP.

1 Introduction

This paper is motivated by the work of Babai and Szemerédi [6, 1] who introduced the theory of black-box groups as a general framework for studying the complexity of computational group theory problems. The computational complexity of several group theoretic problems is an intriguing question. They do not have any known polynomial-time algorithms, nor have they been shown to be NP hard. A closely related problem is Graph Isomorphism (deciding whether there is an isomorphism between two labeled graphs). While there is no known polynomial-time algorithm for it there is strong evidence that Graph Isomorphism is not NP-complete. In [8] it was shown that Graph Nonisomorphism is in AM implying that GI is in NP $\cap$ co-AM. It follows that GI cannot be NP-complete unless the polynomial hierarchy collapses to Σ_2^p [8, 18]. In [4] it is shown that for permutation groups, the Group Intersection and Coset Intersection problems are in NP $\cap$ co-AM, implying that these problems are low for Σ_2^p.

More recently, it was shown in [16] that GI is *low* for PP. In other words, GI is powerless as oracle for PP computations (where PP is the language class corresponding to #P). In [16] lowness for PP is also shown for several permutation group problems. In particular, Group Intersection, Group Factorization, Coset Intersection, and Double-Coset Membership problems for permutation groups are shown to be low for PP.

On the other hand, substantial work has also gone into designing efficient algorithms for more basic permutation group problems. For example, membership testing, normal closure of subgroups, reduction of generator sets, and various other problems for permutation groups have highly time/space efficient algorithms [13, 15, 5].

An important property of any subgroup G of S_n is that it can be decomposed into a series of n subgroups of G such that the ith subgroup G_i is the subgroup of G that fixes the first i elements of $\{1, 2, \ldots, n\}$. This special structure of permutation groups is exploited in the proofs of most of the above-mentioned algorithmic and lowness results.

However, for classes of finite groups other than permutation groups (such as matrix groups over finite fields), designing polynomial-time algorithms for many basic algorithmic problems appears challenging. Limited success is reported in [7, 17]. In [17] a deterministic polynomial-time algorithm for testing solvability of finite matrix groups is given. Assuming tractability of prime factorization and discrete log, it is possible to efficiently compute the order of solvable matrix groups [17].

In [6], Babai and Szemeredi introduce a general framework of *black-box groups*, in order to study the complexity of these problems for general classes of finite groups. Intuitively speaking, in this framework we have an infinite family of abstract groups. The elements of each group in the family are uniquely encoded as strings of uniform length. The groups operations (product, inverse etc) are assumed to be provided by a group oracle and hence are easily computable. Black-box groups are subgroups of groups from such a family and they are presented by generator sets. For example, matrix groups over finite fields and permutation groups presented by generator sets can be seen as examples of black-box groups.

A central problem considered in the papers of Babai and Szemerédi [6, 1] is Order Verification: given a black-box group G presented by a generator set and a positive integer n, verify that $|G| = n$. This problem is important because it turns out that several other problems reduce to Order Verification. Using randomization to compute approximate lower bounds and sophisticated group-theoretic tools, it is shown that Order Verification for general black-box groups is in AM. As a consequence, it turns out that several problems for black-box groups (e.g., Membership Testing, Group Intersection etc.) are in $\mathrm{NP} \cap \text{co-AM}$[1] and hence low for Σ_2^p. Another important result is from [3] which gives an efficient Monte Carlo algorithm for testing solvability of even general black-box groups.

The question that we are concerned with in this paper is whether computational problems over black-box groups are low for hard counting classes like PP and $\mathrm{C_{=}P}$. In the present paper we consider problems for black-box solvable groups. It is worth noting that solvable finite groups constitute a very large subclass of all finite groups. The celebrated Feit-Thompson Theorem states that all finite groups of odd order are solvable. We prove in this paper that various important computational problems for solvable black-box groups are low for the counting class PP.

We summarize our main results and the important ideas that go into their proofs. Since solvable groups are a generalization of abelian groups, it turns out that we first need to consider computational problems for abelian groups. Here the key fact that we use is that every finite abelian group has an independent set of generators. We first develop a suitable oracle algorithm that computes an independent generator set for an abelian black-box group. Using independent generator sets for abelian black-box groups we show that the problems of Membership Testing, Group Intersection, Order Verification, and Group Isomorphism for abelian black-box groups are in the low counting class SPP. We also show that Group Factorization, Coset Intersection, and Double Coset Membership problems for abelian black-box groups are in the low counting

[1] In fact, if the short presentation conjecture for finite simple groups is true then, relative to the group oracle, Membership Testing for black-box groups is in $\mathrm{NP} \cap \text{co-NP}$ [6]. On the other hand, since $\mathrm{NP} \cap \text{co-NP}$ is not known to be low for PP, this conjecture does not imply PP-lowness of Membership Testing.

class LWPP. As a consequence, all these problems are low for PP and $\mathrm{C_=P}$.

Intuitively speaking, solvable groups are groups that can be decomposed into a series of subgroups such that the successive factor groups are abelian. As a generalization of independent generator sets, we define *canonical* generator sets for classes of finite groups from any group family. We show that solvable finite groups have canonical generator sets. Using the randomized algorithm for computing the commutator series of a solvable black-box group from [3] we design a suitable randomized oracle algorithm that computes canonical generator sets for solvable black-box groups. Using this algorithm we are able to show that Membership Testing, Group Intersection, Order Verification, Group Isomorphism, Group Factorization, Coset Intersection, and Double Coset Membership problems for solvable black-box groups are low for PP.

The layout of the rest of the paper is as follows. In Section 2 we give definitions and formalize the notation. In Section 3 we show that solvable black-box groups have canonical generator sets. In Section 4 we describe our results on abelian groups and in Section 5 we describe our results on solvable groups. Finally, in Section 6, we summarize the results obtained in this paper.

2 Preliminaries

We fix the finite alphabet $\Sigma = \{0, 1\}$. We denote the base-2 logarithm function as log. Let $A, B \subseteq \Sigma^*$ be two languages. The language $0A \cup 1B$ is denoted by $A \oplus B$. Let FP denote the class of polynomial-time computable functions and NP denotes all languages accepted by polynomial-time nondeterministic machines.

Let $\mathbf{Z}$ denotes the set of integers. A function $f : \Sigma^* \to \mathbf{Z}$ is *gap-definable* if there is a polynomial-time nondeterministic (in short, NP) machine M such that, for each $x \in \Sigma^*$, $f(x)$ is the difference between the number of accepting paths and the number of rejecting paths of M on input x (denoted by $M(x)$). Let GapP [11] denote the class of gap-definable functions. For each NP machine M let gap_M denote the GapP function defined by it. The corresponding language class PP is defined as follows: A language L is in PP if there is a GapP function f such that: $x \in L$ iff $f(x) > 0$.

The counting classes of interest in this paper are UP, SPP and LWPP defined using GapP functions [11]. A language L is in UP if there is an NP machine M accepting L such that M has at most one accepting path on any input. A language L is in SPP if there is an $f \in$ GapP such that: $x \in L$ implies that $f(x) = 1$, and $x \notin L$ implies that $f(x) = 0$. A language L is in LWPP if there are functions $f \in$ GapP and $h \in$ FP such that: $x \in L$ implies that $f(x) = h(0^{|x|})$, and $x \notin L$ implies that $f(x) = 0$. It is known that UP $\subseteq$ SPP $\subseteq$ LWPP.

For an oracle $A \in \Sigma^*$ let PP^A denote the class of languages obtained by the natural relativization w.r.t. oracle A. A language $A \in \Sigma^*$ is said to be *low* for PP if $\mathrm{PP}^A = \mathrm{PP}$. In [11] it is shown that every language in LWPP is low for PP.

Let L be a language in NP accepted by an NP machine M. The prefix language of L with respect to M is defined as follows: $\mathit{Prefix}_M(L) = \{\langle x, y\rangle \mid \exists z \in \Sigma^*$ such that yz is an accepting path of M on input $x\}$. It is easy to see that $\mathit{Prefix}_M(L) \in$ NP.

Let M be an oracle NP machine and let $A \in$ NP accepted by an NP machine N and let $f \in$ FP. We say that M^A make $f(n)$*-guarded* queries if on length n inputs M^A only asks queries y for which $N(y)$ has either 0 or $f(n)$ accepting paths, for each n. In this terminology, we state a weaker version of a theorem from [16] which we frequently use in this paper.

Theorem 2.1 ([16]) *Let M be a nondeterministic polynomial-time oracle machine and let f be a polynomial-time computable function. If $A \in$ NP such that M^A makes $f(n)$-guarded queries then there is a polynomial q such that the function $h(x) = gap_{M^A}(x) * f(|x|)^{q(|x|)}$ is in* GapP.

2.1 Group-theoretic preliminaries

In this subsection, we give some basic group-theoretic definitions and formalize some notation. The reader can refer standard textbooks [9, 14] for details.

Definition 2.2 [9, 14] A *group* is a nonempty set G endowed with a binary operation $*$ such that G is closed under $*$. The operation $*$ is associative. There is an element $e \in G$, called the *identity* of G, such that $x * e = e * x = x$, for all $x \in G$. For every $x \in G$ there exists a unique $x^{-1} \in G$, called the *inverse* of x, such that $x * x^{-1} = x^{-1} * x = e$.

When there is no ambiguity we do not explicitly specify the group operation and denote the composition $x * y$ by xy.

A group G is a *finite group* if the cardinality of the set G (denoted by $|G|$) is finite. In the sequel G stands for a finite group. The *order* of G is defined as $|G|$. For an element $g \in G$, the *order* of g (denoted $o(g)$) is the smallest positive integer such that $g^{o(g)} = e$, where e is the identity of G. A subset H of G is called a *subgroup* of G (denoted $H < G$) if H is a group under the group operation of G.

A group G is *abelian* if $\forall g_1, g_2, \in G : g_1 g_2 = g_2 g_1$. Let S be a subset of a group G. The smallest subgroup of G containing S is called the group *generated by* S and is denoted $\langle S \rangle$. A subset S of G is a generator set for G if $G = \langle S \rangle$. Observe that, if S is a set of generators for G then G is abelian iff $\forall g_1, g_2, \in S : g_1 g_2 = g_2 g_1$.

Let H be a subgroup of a group G. For $g \in G$ the set $\{hg \mid h \in H\}$ denoted by Hg is a *right coset* of H in G. Similarly, the set $gH = \{gh \mid h \in H\}$ is a *left coset* of H in G.

A subgroup H of G is a *normal* subgroup of G if for all $g \in G$ it holds that $Hg = gH$. As a consequence it turns out that the set $G/H = \{Hg \mid g \in G\}$ is a group (the *quotient group* induced by H) under the binary operation $\cdot$ defined by $Hx \cdot Hy = Hxy$.

The element $xyx^{-1}y^{-1}$ is called the *commutator* of elements x and y in G. The subgroup G' of G generated by the set $\{xyx^{-1}y^{-1} \mid x, y \in G\}$ is called the *commutator subgroup* of G. The commutator subgroup G' is actually a normal subgroup of G and the factor group G/G' is abelian.

A group G is said to be *solvable* if the sequence $G = G_0 > G_1 > \ldots$, where each group G_i is the commutator subgroup of G_{i-1}, terminates in the trivial subgroup $\langle e \rangle$ in finitely many steps.

Let p be a prime. A p-group is a finite group whose order is a power of p. Let G be finite groups such that $|G| = p_1^{e_1} p_2^{e_2} \ldots p_r^{e_r}$. A fundamental theorem of group theory is that for each i there is a subgroup of G of order $p_i^{e_i}$. A subgroup of G of order $p_i^{e_i}$ is referred to as a *p_i-Sylow subgroup* of G.

Let $(X, *)$ and $(Y, .)$ be two groups. The *direct product* of the groups X and Y is defined as the group $(X \times Y, \circ)$, where $X \times Y$ is the cartesian product of sets X and Y, and for $(x_1, y_1), (x_2, y_2) \in X \times Y$ their $\circ$ composition is defined as $(x_1, y_1) \circ (x_2, y_2) = (x_1 * x_2, y_1 . y_2)$.

Definition 2.3 A *group family* is a countable sequence $\mathcal{B} = \{B_m\}_{m \geq 1}$ of finite groups B_m, such that there are polynomials p and q satisfying the following conditions. For each $m \geq 1$, elements of B_m are uniquely encoded as strings in $\Sigma^{p(m)}$. The group operations (inverse, product and testing for identity) of B_m can be performed in time bounded by $q(m)$, for every $m \geq 1$. The order of B_m is computable in time bounded by $q(m)$, for each m. We refer to the groups B_m of a group family and their subgroups (presented by generator sets) as *black-box* groups.[2]

[2]Note that black-box groups we define above are a restricted version of black-box groups introduced in [6].The black-box group defined in [6] is technically more general. There the black-box group is defined so as to incorporate factor groups.

For example let S_n denote the permutation group on n elements. Then, $\{S_n\}_{n\geq 1}$ is a group family of all permutation groups S_n. As another example let $GL_n(q)$ denote the group of all $n\times n$ invertible matrices over the finite field F_q of size q. The collection GL(q) $= \{GL_n(q)\}_{n\geq 1}$ is a group family.

Let $\mathcal{B} = \{B_m\}_{m>0}$ be a group family. The following decision problems which we consider in this paper are well-studied in computational group theory [10, 6, 16, 1].

Membership Testing $\stackrel{\triangle}{=} \{(0^m, S, g) \mid \langle S\rangle < B_m \text{ and } g \in \langle S\rangle\}$.

Order Verification $\stackrel{\triangle}{=} \{(0^m, S, n) \mid \langle S\rangle < B_m \text{ and } |\langle S\rangle| = n\}$.

Group Isomorphism $\stackrel{\triangle}{=} \{(0^m, S_1, S_2) \mid \langle S_1\rangle \text{ and } \langle S_2\rangle \text{ are isomorphic subgroups of } B_m\}$.

Group Intersection $\stackrel{\triangle}{=} \{(0^m, S_1, S_2) \mid \langle S_1\rangle, \langle S_2\rangle < B_m \text{ and } \langle S_1\rangle \cap \langle S_2\rangle \neq \langle e\rangle\}$.

Group Factorization $\stackrel{\triangle}{=} \{(0^m, S_1, S_2, g) \mid \langle S_1\rangle, \langle S_2\rangle < B_m \text{ and } g \in \langle S_1\rangle\langle S_2\rangle\}$.

Coset Intersection $\stackrel{\triangle}{=} \{(0^m, S_1, S_2, g) \mid \langle S_1\rangle, \langle S_2\rangle < B_m \text{ and } \langle S_1\rangle g \cap \langle S_2\rangle \neq \emptyset\}$.

Double-Coset Membership $\stackrel{\triangle}{=} \{(0^m, S_1, S_2, g, x) \mid \langle S_1\rangle, \langle S_2\rangle < B_m \text{ and } g \in \langle S_1\rangle x \langle S_2\rangle\}$.

3 Canonical generator sets

We define the notion of canonical generator sets and show that solvable black-box groups have canonical generator sets. We first state a fundamental theorem on finite abelian groups.

Theorem 3.1 [9] *Let G be a finite abelian group such that $|G| = p_1^{e_1}p_2^{e_2}\dots p_r^{e_r}$, where the p_i's are distinct primes. The group G can be expressed as the direct product of its Sylow subgroups $S(p_1), S(p_2), \dots, S(p_r)$ where $|S(p_i)| = p_i^{e_i}$ for $1 \leq i \leq r$. Furthermore, for $1 \leq i \leq r$, each Sylow subgroup $S(p_i)$ can be uniquely expressed as the direct product of cyclic groups of orders $p_i^{e_{i1}}, p_i^{e_{i2}}, \dots, p_i^{e_{is_i}}$ such that $e_{i1} \geq e_{i2} \geq \dots \geq e_{is_i}$ and $\Sigma_{j=1}^{s_i} e_{ij} = e_i$. This decomposition of G is unique.*

Let p be a prime and consider an abelian p-group S of order p^e. The above theorem implies that there is a *unique* sequence of natural numbers $e_1 \geq e_2 \geq \dots \geq e_m$ such that $\Sigma_{1\leq j\leq m} e_j = e$ and S can be expressed as a direct product of m cyclic groups of respective orders p^{e_j}, $1 \leq j \leq m$. The sequence $(e_1, e_2, \dots, e_m)$ is called the *type* of the p-group S.

Let G be a finite abelian group. An element $g \in G$, $g \neq e$, is said to be *independent* of a set X of elements of G if $\langle g\rangle \cap \langle X\rangle = \langle e\rangle$. A generator set S of G is an *independent set of generators* for G if every $g \in S$ is independent of $S - \{g\}$. As a consequence of Theorem 3.1, all finite abelian groups have independent generator sets.

The fact that abelian groups have independent generator sets plays an important role in the proofs of the PP-lowness results for abelian group problems in Section 4. Motivated by this we define the notion of canonical generator sets for arbitrary classes of finite groups.

Definition 3.2 Let $\mathcal{B} = \{B_m\}_{m>0}$ be any group family. Let $\mathcal{C}$ be a class of finite groups such that every $G \in \mathcal{C}$ is a subgroup of some $B_m \in \mathcal{B}$. The class of groups $\mathcal{C}$ has *canonical generator sets* if for every $G \in \mathcal{C}$, if $G < B_m$ then there is an *ordered* set $S = \{g_1, g_2, \dots, g_s\} \subseteq G$ such that each $g \in G$ can be uniquely expressed as $g = g_1^{l_1}g_2^{l_2}\dots g_s^{l_s}$, where $0 \leq l_i \leq o(g_i), 1 \leq i \leq s$. Furthermore, $s \leq q(m)$ for a polynomial q. S is called a *canonical generator set* for G.

If S is an independent generator set for an abelian group G then each $x \in G$ can be uniquely expressed as a product $\Pi_{g\in S} g^{i_g}$ for $1 \leq i_g \leq o(g)$, and it holds that $|S| \leq \log(|G|)$. Thus we have the following proposition.

Proposition 3.3 *Let $\mathcal{B}$ be any group family. The class of all abelian black-box groups from $\mathcal{B}$ has canonical generator sets.*

Note that the ordering of the elements of the canonical generator set is irrelevant for abelian groups since products commute. However, for classes of nonabelian groups the ordering of elements in a canonical generator set has to be prescribed.

We now show that the class of solvable black-box groups for any group family has canonical generator sets.

Lemma 3.4 *Let $\mathcal{B} = \{B_m\}_{m>0}$ be a group family such that $|B_m| \leq 2^{q(m)}$ for a polynomial q. Let $G < B_m$ be a finite solvable group and $G = G_0 > G_1 > \ldots > G_{k-1} > G_k = \langle e \rangle$ be the commutator series of G. Let $T_i = \{h_{i1}, h_{i2}, \ldots, h_{ik_i}\}$ be a set of distinct coset representatives corresponding to an independent set of generators for the abelian group $H_i = G_{i-1}/G_i$. Then the ordered set [3] $\bigcup_{i=1}^{k} T_i$ forms a canonical generator set for the group G and its cardinality is bounded by $q(m)$. Thus the class of solvable groups from $\mathcal{B}$ has canonical generator sets.*

Proof. Let $\mathcal{B} = \{B_m\}_{m>0}$ be a group family such that $|B_m| \leq 2^{q(m)}$ for a polynomial q. Let $G < B_m$ be a finite solvable group and $G = G_0 > G_1 > \ldots > G_{k-1} > G_k = \langle e \rangle$ be the commutator series of G. Recall that subgroups of solvable groups are solvable. We prove by induction on $k - i$, that for each $i : 1 \leq i \leq k$, $\bigcup_{j=i+1}^{k} T_j$ is a canonical generator set for G_i. For the base case, when $k - i = 1$ it clearly holds, since G_{k-1} is an abelian group and T_k is an independent set of generators for G_{k-1}. Suppose, as induction hypothesis, that $\bigcup_{j=i+1}^{k} T_j$ is a canonical generator set for G_i. Consider the group G_{i-1}. The factor group G_{i-1}/G_i is generated by the independent set of generators $X_i = \{h_{i1}G_i, h_{i2}G_i \ldots, h_{ik_i}G_i\}$. An element $g \in G_{i-1}$ belongs to exactly one right coset in G_{i-1}/G_i. Let that right coset be hG_i. Thus $g = hg'$ for some $g' \in G_i$. Now, since X_i is an independent generator set of the abelian group G_{i-1}/G_i, there are unique indices $\{l_{i1}, l_{i2}, \ldots, l_{ik_i}\}$ such that $hG_i = h_{i1}^{l_{i1}} \ldots h_{ik_i}^{l_{ik_i}} G_i$. Consequently, $\{l_{i1}, l_{i2}, \ldots, l_{ik_i}\}$ are unique indices such that $g = hg' = h_{i1}^{l_{i1}} \ldots h_{ik_i}^{l_{ik_i}} g''$, where $g'' \in G_i$. By induction hypothesis, $\bigcup_{j=i+1}^{k} T_j$ is a canonical generator set for G_i, implying that g'' can be uniquely expressed as a product of powers of the elements in $\bigcup_{j=i+1}^{k} T_j$. We have proved the induction step that every $g \in G_{i-1}$ can be uniquely expressed as a product of powers of the elements in $\bigcup_{j=i}^{k} T_j$. Thus, $\bigcup_{j=i}^{k} T_j$ is a canonical generator set for G_{i-1}.

Finally we show that $q(m)$ bounds $|\bigcup_{i=1}^{k} T_i|$. Observe that since for each i, $1 \leq i \leq k$ the set X_i is an independent generator set for the quotient group G_{i-1}/G_i, it holds that $|T_i| \leq \log(|G_{i-1}/G_i|)$. Therefore, it follows that $|\bigcup_{i=1}^{k} T_i| \leq \sum_{1 \leq i \leq k} \log(|G_{i-1}/G_i|) = \log(|G|) \leq q(m)$ since $|G| \leq 2^{q(m)}$.

■

4 Lowness results for abelian group problems

In this section we prove upper bounds for various abelian group-theoretic problems defined in Section 2. All these upper bounds implies that these problems are low for both classes PP and $\mathrm{C_{=}P}$. A key step in the proofs of these results is the design of a suitable algorithm which takes as input an arbitrary generator set for an abelian black-box group and outputs an independent generator set for that group. We first state some results from finite abelian group theory. Proofs of some of the easier-to-prove results are given here. For proofs of other results mentioned in this paper, reader can refer to standard text books in finite group theory [9, 14].

[3] The elements of the set $\bigcup_{i=1}^{k} T_i$ are ordered on increasing values of the index i, and lexicographically within each set T_i.

Proposition 4.1 *Let G be an abelian p-group generated by $\{g_1, g_2 \dots, g_s\}$. Let $(m_1, m_2, \dots, m_s)$ be the type of G. If g_l is an element of maximum order among the generators then g_l is of order p^{m_1}.*

Proof. Since $(m_1, m_2, \dots, m_s)$ is the type of G, the maximum possible order of an element in G is p^{m_1}. If all the generators of G are of order less than p^{m_1} then every element of G will be of order less than p^{m_1}. Thus g_l must be of order p^{m_1}. ∎

The following lemma is a classical result which actually indicates a method to compute an independent generator set for an abelian p-group. This lemma is crucial to the proof of the decomposition theorem (see Theorem 3.1) for abelian groups.

Lemma 4.2 ([9]) *Let G be a finite abelian p-group of type $(m_1, m_2, \dots, m_s)$. Let $g_1, g_2 \dots, g_i$ be i independent elements of G of orders $p^{m_1}, p^{m_2}, \dots, p^{m_i}$ respectively. The maximum order of any element in the factor group $G/\langle g_1, g_2 \dots, g_i\rangle$ is $p^{m_{i+1}}$. Let $g'_{i+1}\langle g_1, g_2 \dots, g_i\rangle$ be an element in the factor group $G/\langle g_1, g_2 \dots, g_i\rangle$ of order $p^{m_{i+1}}$. Further, let $(g'_{i+1})^{p^{m_{i+1}}} = g_1^{x_1} g_2^{x_2} \dots g_i^{x_i}$, and $y_j = x_j / p^{m_{i+1}}$, for $1 \le j \le i$. Then $g_{i+1} = g'_{i+1} g_1^{-y_1} g_2^{-y_2} \dots g_i^{-y_i}$ is an element of G of order $p^{m_{i+1}}$ which is independent of $\{g_1, g_2 \dots, g_i\}$.*

In the next lemma we show how to construct generator sets for the Sylow subgroups of an abelian group presented by a generator set.

Lemma 4.3 *Let $\mathcal{B} = \{B_m\}_{m>0}$ be a group family, and $G < B_m$ be an abelian black-box group generated by $\{g_1, g_2 \dots, g_s\}$. If $|B_m| = n = p_1^{e_1} p_2^{e_2} \dots p_r^{e_r}$ is the prime factorization of $|B_m|$, then for each j, the p_j-Sylow subgroup of G is generated by $X_j = \{g_i^{(n/p_j^{e_j})} \mid 1 \le i \le s\}$.*

Proof. It is not hard to see that for each j, the group $\langle X_j\rangle$ generated by X_j is a subgroup of the (unique) p_j-Sylow subgroup of G. Therefore, it is enough to show that each generator of G can be generated using products of approporiate elements from the different X_j. Consider a generator g_l. Let $o(g_l) = n_l$. Since $gcd(n_l, \Sigma n/p_j^{e_j}) = 1$, it follows that g_l is in the cyclic group generated by $\prod_{j=1}^{r} g_l^{n/p_j^{e_j}}$. ∎

We now prove a straightforward lemma which describes how to compute orders of elements in a given abelain group and also how to compute the order of cosets of some subgroup of a given abelain group.

Lemma 4.4 *Let $\mathcal{B} = \{B_m\}_{m>0}$ be a group family. Given the prime factorization $p_1^{e_1} p_2^{e_2} \dots p_r^{e_r}$ of $|B_m|$ as part of input, $o(g)$ for $g \in B_m$ can be computed in time polynomial in m. Let $\langle S\rangle$ be a normal subgroup of B_m. For any $g \in B_m$, the order of the coset $g\langle S\rangle$ can be computed in time polynomial in $|S|$ and m using a membership-test oracle.*

Proof. Let $n = |B_m|$. Observe that $o(g)$ can be expressed as $p_1^{d_1} p_2^{d_2} \dots p_r^{d_r}$, each $d_i \le e_i$. Clearly d_i is the smallest j such that $(g^{n/p_i^{e_i}})^{p_i^j} = e$. Since $1 \le d_i \le e_i$ and e_i is bounded by a polynomial in m, for every i, each d_i can be computed in time bounded by a polynomial in m.

Again, order of $g\langle S\rangle$ can be expressed as $p_1^{d_1} p_2^{d_2} \dots p_r^{d_r}$, each $d_i \le e_i$. Index d_i is the smallest j such that $(g^{n/p_i^{e_i}})^{p_i^j} \in S$. This j can be computed in time polynomial in $|S|$ and m using membership tests for $\langle S\rangle$. ∎

We require the following languages L_1 and L_2 throughout the paper as oracles. Since primality testing is in UP [12], it is easy to define an oracle L_1 in UP such that the prime factorization of a positive integer is computable in FP^{L_1}.

Let $\mathcal{B} = \{B_m\}_{m>0}$ be a group family. Now we will define a suitable language L_2 which we can use as oracle to compute the certificates of membership of an element in a group presented by a canonical generator set. We will actually define L_2 as a prefix language of another language L'_2 which is first defined below.

$$L'_2 \triangleq \{\langle 0^m, g, S\rangle \mid g \in B_m, S \subseteq B_m, \text{ and } \forall h \in S; \exists l_h \leq o(h) \text{ such that } g = \Pi_{h\in S} h^{l_h}\}$$

The following proposition shows that $L'_2 \in$ NP.

Proposition 4.5 *L'_2 is in* NP.

Proof Sketch. Consider an NP machine M'_2 which first computes $|B_m|$, then guesses the prime factorization of $|B_m|$ in a UP way [12], next M'_2 computes $o(h)$ for all $h \in S$ using Lemma 4.4. Then it guesses indices l_h for each $h \in S$ such that $l_h \leq o(h)$ and verifies in polynomial time that $g = \Pi_{h\in S} h^{l_h}$. ■

Let $\mathcal{C}$ be a class of groups having canonical generator sets over $\mathcal{B} = \{B_m\}_{m>0}$. Let $G \in \mathcal{C}$. Clearly we can test for membership in G of a given element $g \in B_m$ by making one query to oracle L'_2 if G is presented by a canonical generator set. Furthermore, on such queries, the NP machine M'_2 has a unique accepting path if $g \in G$.

Define L_2 as $\mathit{Prefix}_{M'_2}(L'_2)$. Given g and a canonical generator set S for G, using L_2 as oracle we can prefix search for indices $l_h, h \in S$, such that $g = \Pi_{h\in S} h^{l_h}$. For each such query made to L_2, the NP machine for L_2 (defined by suitably modifying M'_2) also has at most one accepting path. Thus, if S is a canonical generator set, the decomposition $g = \Pi_{h\in S} h^{l_h}$ can be computed by making only 1-guarded queries to L_2.

We are now ready to prove the key theorem of this section which yields the lowness results.

Theorem 4.6 *Let $\mathcal{B} = \{B_m\}_{m>0}$ be a group family. There is a deterministic oracle machine M such that*

1. *M takes as input $(0^m, S)$, $S \subseteq B_m$.*
2. *M checks if $G = \langle S\rangle$ is an abelian group and if so it outputs $|G|$ and a set of independent generators for G.*
3. *Furthermore, M runs in time polynomial in $|(0^m, S)|$, and makes 1-guarded queries to the NP oracle $L_1 \oplus L_2$.*

Proof. Let $\mathcal{B}$ be a group family. Let $G = \langle S\rangle < B_m$ be an abelian black-box group where $S = \{g_1, g_2, \ldots, g_s\}$. The idea of the proof is as follows. Firstly, using the UP oracle L_1, we compute the prime factorization of $|B_m|$. We next compute the generators for each of the Sylow subgroups from the given generator set S. This can be done in polynomial time using Lemma 4.3 and the prime factorization of $|B_m|$. Now, using oracle L_2, we convert the constructed generator set for each Sylow subgroup into an independent set of generators. The union of the independent generator sets for the different Sylow subgroups forms an independent generator set for G.

The following is the description of the machine M, which uses oracles L_1 and L_2 for constructing the set of independent generators.

Algorithm M:

Input $G = \langle 0^m, \{g_1, g_2 \ldots, g_s\}\rangle$;
if G is not abelian **then Output** 'G not abelian' and **stop**;

3 (* *Note that G is abelian iff $g_i g_j = g_j g_i$, for $1 \le i < j \le s$*
4 *which can be tested in $O(s^2)$ time.* *)
5 Using UP oracle L_1 compute prime factorization for $|B_m| = p_1^{e_1} p_2^{e_2} \dots p_r^{e_r}$;
6 **for** $i : 1 \le i \le s$ **do**
7 Compute the order of g_i; (* *Using Lemma 4.4* *)
8 **for** $i : 1 \le i \le r$ **do**
9 Compute the generators for the p_i-Sylow subgroup (* *Using Lemma 4.3* *);
10 Let X_i be the generator set for the p_i-Sylow subgroup, $1 \le i \le r$;
11 **foreach** p_i-Sylow subgroup **do**
12 $S_i := \phi$; $N_i := 1$;
13 **while** $X_i \not\subseteq \langle S_i \rangle$ **do** (* j^{th} iteration *)
14 (* *S_i stands for the set of independent generators so far constructed.* *)
15 Compute an element $h_j \in X_i$ such that the coset $h_j \langle S_i \rangle$
16 is of maximum possible order in $\langle X_i \rangle / \langle S_i \rangle$. Let $o(h_j \langle S_i \rangle) = p_i^{l_j}$; $N_i := p_i^{l_j} . N_i$;
17 (* *By part 2 of Lemma 4.4 this can be done in polynomial time*
18 *using L_2 as oracle for membership tests.* *)
19 Using L_2 as oracle, by prefix search compute indices
20 $\{x_g \mid g \in S_i, 1 \le x_g < o(g)\}$, such that $h_j^{o(h_j \langle S_i \rangle)} = \Pi_{g \in S_i} g^{x_g}$;
21 (* *L_2 is a prefix language designed precisely for this computation.* *)
22 $S_i := S_i \cup \{h_j \Pi_{g \in S_i} g^{-x_g / p_i^{l_j}}\}$; (* *Using Lemma 4.2* *)
23 **end-while**
24 **end-foreach**;
25 **Output** $\cup_{i=1}^{r} S_i$ as the independent generator set;
26 **Output** $\Pi_{1 \le i \le r} N_i$;
27 (**N_i is the order of the p_i-Sylow subgroup of G, for each i.* *)

Now we show that the above algorithm is correct. Firstly, as a consequence of Lemmas 4.4 and 4.3, observe that generators sets for each of the Sylow subgroups of G can be computed in polynomial time after computing the prime factorization of $|B_m|$ using oracle L_1.

In *lines* 11 to 24 of the algorithm, an independent set of generators is being computed for each Sylow subgroup of G. We focus attention on the p_i-Sylow subgroup G_i. Note that the set X_i generates G_i. Let the type of G_i be $(m_1, m_2, \dots, m_s)$. Let $\{g'_1, g'_2, \dots, g'_{j-1}\}$ be the set of independent generators constructed after $j-1$ iterations of the **while** loop in *line* 23. Now, observe that as a consequence of Proposition 4.1 and Lemma 4.2, in the j^{th} iteration of the **while** loop, in *lines* 15 and 16 the algorithm computes an element $h_j \in G_i$ such that the coset $h_j \langle S_i \rangle$ is of order $p_i^{l_j}$, where $l_j = m_j$. Furthermore, by Lemma 4.2 again, the element $h_j \Pi_{g \in S_i} g^{-x_g / p_i^{l_j}}$ is an element in G_i of order $p_i^{m_j}$, and it is independent of the set S_i. Thus after the j^{th} iteration of the **while** loop the set S_i is augumented to $S_i \cup \{h_j\}$. So finally, when the **while** loop condition is not satisfied, the set S_i will be an independent generator set of G_i, because $G_i = \langle X_i \rangle = \langle S_i \rangle$.

Observe that apart from the oracle calls to L_1 made in *line* 5, and the oracle calls to L_2 made in *lines* 15,16,19 and 20, the rest of the computation done by the algorithm is of time that can be bounded by a polynomial in the input size.

Oracle L_1 is in UP. As for queries to L_2, we observe that for every query $\langle y, S \rangle$ for which we wish to compute indices $x_g, g \in S$ such that $y = \Pi_{g \in S} g^{x_g}$, the set S consists of independent elements. Therefore, the queries made to L_2 are 1-guarded. Thus the machine M is a polynomial-time oracle machine that makes 1-guarded queries to both L_1 and L_2. We can combine L_1 and L_2 into a single oracle $L_1 \oplus L_2$. This completes the proof. ∎

As a consequence of the above theorem and Theorem 2.1 we can show that Membership Testing, Order Verification, and Group Isomorphism for abelian groups are all in SPP. It is interesting to note that Order Verification for general black-box groups is in AM $\cap$ co-AM [1].

Corollary 4.7 *Over any group family, Membership Testing, Order Verification, and Group Isomorphism for abelian black-box groups is in* SPP. *Hence these problems are low for* PP, $\mathrm{C_=P}$ *and the* $\mathrm{Mod}_k\mathrm{P}$ *classes for* $k \geq 2$.

Proof.

We first consider Membership Testing. Let $\mathcal{B}$ be a group family. Consider an oracle NP machine M' which on input $(0^m, S, g)$ checks first whether the set S indeed generates an abelian group. If S generates an abelian group, M' converts this set S to a canonical set of generators by simulating the algorithm M of Theorem 4.6. Then M' makes a 1-guarded query to the language $L_1 \oplus L_2$ to check $g \in \langle S \rangle$, if the answer is 'YES', it accepts. If the answer to the query in 'NO' then M' branches into an accepting and a rejecting path, thus producing a zero *gap*. Observe that M' makes only 1-guarded queries to $L_1 \oplus L_2$. Now, using Theorem 2.1, we can convert machine M' to an NP machine M'' which does not use any oracle and produces the same *gap* as that of M'. It follows that Membership Testing is in SPP.

By a similar construction as above we can show that Order Verification for abelian black-box groups is in SPP.

Finally we show that Group Isomorphism for abelian groups is in SPP. Let $(0^m, S_1, S_2)$ be an instance of Group Isomorphism. I.e., S_1 and S_2 generate abelian subgroups of $B_m \in \mathcal{B}$. Firstly, observe that any two abelian p-groups are isomorphic if and only if their types are identical. This follows from the fact that the type uniquely determines an abelian p-group up to isomorphism [9]. Furthermore, two abelian groups $\langle S_1 \rangle$ and $\langle S_2 \rangle$ are isomorphic iff $|\langle S_1 \rangle| = |\langle S_2 \rangle|$, and for every prime factor p of $|\langle S_1 \rangle|$, the p-Sylow subgroup of $\langle S_1 \rangle$ is isomorphic to the p-Sylow subgroup of $\langle S_2 \rangle$. Thus, in order to check that $\langle S_1 \rangle$ and $\langle S_2 \rangle$ are isomorphic, it is only required to verify that the p-Sylow subgroups of $\langle S_1 \rangle$ and $\langle S_2 \rangle$ respectively have identical types, for each prime factor p of $|\langle S_1 \rangle|$. Observe that given the generator set S_1 as input machine M of Theorem 4.6 in the course of its computation in *lines* 15 and 16 actually computes the type of each Sylow subgroup of $\langle S_1 \rangle$. Thus we can easily adapt machine M of Theorem 4.6 to define a machine M_{is} that computes the types of each Sylow subgroup of both $\langle S_1 \rangle$ and $\langle S_2 \rangle$. Now, M_{is} can make an easy inspection of the types of the Sylow subgroups of $\langle S_1 \rangle$ and $\langle S_2 \rangle$, to decide if they are isomorphic. Clearly, the machine M_{is} runs in time bounded by a polynomial in input length, and it makes 1-guarded queries to $L_1 \oplus L_2$. ■

The next theorem is a technical generalization of Theorem 4.6 to factor groups, with applications to other results in the paper.

Theorem 4.8 *Let* $\mathcal{B} = \{B_m\}_{m>0}$ *be a group family. There is a deterministic oracle machine* M_1 *that takes as input tuples* $(0^m, X, Y, Z)$, *where* $X, Y, Z \subseteq B_m$ *are finite sets. Suppose the input* $(0^m, X, Y, Z)$ *to* M_1 *satisfies the following properties*

1. *Let* G, H *and* K *be the groups generated by the sets* X, Y *and* Z. *The sets* Y *and* Z *are canonical generator sets for groups* H *and* K.
2. *The group* $H \cap K$ *is a normal subgroup of* G *and the factor group* $G/(H \cap K)$ *is abelian.*

Then the machine M_1 *outputs a list of coset representatives corresponding to some independent generator set for* $G/(H \cap K)$ *and also outputs* $|G/(H \cap K)|$. *Furthermore,* M_1 *runs in time polynomial in input length, and it makes 1-guarded queries to the NP oracle* $L_1 \oplus L_2 \oplus L_3$. *The behavior of the machine* M_1 *is not specified if the input does not satisfy the above two conditions.*

Proof. Let $\mathcal{B}$ be a group family. Let $G < B_m$ be a group presented by the generator set $S = \{g_1, g_2, \ldots, g_s\}$. Furthermore, suppose H and K presented by canonical sets of generators and $H \cap K$ is a normal subgroup of G such that $G/(H \cap K)$ is abelian. We can define a machine M_1 that is exactly like the machine M of Theorem 4.6. Observe that we have equivalents of Lemmas 4.3, 4.2 and Proposition 4.1 for abelian factor groups also.

The only new aspect of machine M_1 is that order computation of cosets of $G/(H \cap K)$ will be repeatedly required. Given the prime factorization of $|B_m|$, order computation of an element of $G/(H \cap K)$ can be done using part 2 of Lemma 4.4. Membership tests in $H \cap K$ can be done using two queries (one each for H and K) to oracle L_2. Since H and K are presented by canonical generator sets only 1-guarded queries will be made to L_2. However, for subsequent computation we require another oracle L_3 (similar to L_2) which we now define:

$L'_3 \triangleq \{(0^m, g, X_1, X_2, g_1, g_2, \ldots, g_k, o_1, o_2, \ldots, o_k) \mid g \in B_m, \forall i : g_i \in B_m \text{ and } \exists l_i \leq o_i : g(\prod_{1 \leq i \leq k} g_i^{-l_i}) \in \langle X_1 \rangle \cap \langle X_2 \rangle\}$. The following claim can be proved exactly as we had done for L'_2 and L_2.

Claim 4.8.1 *L'_3 is in* NP *witnessed by an* NP *machine M_3 which has at most one accepting path on inputs $(0^m, g, X_1, X_2, g_1, g_2, \ldots, g_k, o_1, o_2, \ldots, o_k)$ satisfying the following properties.*

1. *X_1 and X_2 are canonical generator sets for groups $\langle X_1 \rangle$ and $\langle X_2 \rangle$.*
2. *Let $\{g_1, g_2, \ldots, g_k\} \subseteq G$ where G is some subgroup of B_m such that $\langle X_1 \rangle \cap \langle X_2 \rangle$ is normal in G with the additional property that the quotient group $G/(\langle X_1 \rangle \cap \langle X_2 \rangle)$ is abelian.*
3. *Furthermore, the cosets $\{g_i(\langle X_1 \rangle \cap \langle X_2 \rangle) \mid 1 \leq i \leq k\}$ are independent elements of $G/(\langle X_1 \rangle \cap \langle X_2 \rangle)$, and $\forall i : o(g_i(\langle X_1 \rangle \cap \langle X_2 \rangle)) = o_i$.*

Let $L_3 = \textit{Prefix}_{M_3}(L'_3)$. Using the language L_3 as oracle the indices $l_i, 1 \leq i \leq k$ can be computed by prefix search.

We now give a brief description of M_1.

Machine M_1:

Input $G = \{g_1, g_2 \ldots, g_s\}, H = \{h_1, h_2 \ldots, h_q\}; K = \{k_1, k_2 \ldots, k_t\}$;
(* *H and K are presented by canonical sets of generators* *)
Using oracle L_1 compute prime factorization $p_1^{e_1} p_2^{e_2} \ldots p_r^{e_r}$ of $|B_m|$;
for $i := 1$ **to** s **do**
  Compute the order of $g_i(H \cap K)$ in $G/H \cap K$;
  (* *We can do it with 1-guarded queries to L_2 for membership tests in H and K.**)
endfor
for $i := 1$ **to** r **do**
  Compute the generator set X_i for the p_i-Sylow subgroup of $G/H \cap K$;
  (* *Using Lemma 4.3* *)
endfor
foreach p_i-Sylow subgroup **do**
  $S_i := \phi$; $N_i := 1$;
  while $X_i \not\subseteq \langle S_i \rangle$ **do**(* j^{th} iteration *)
  (* *S_i stands for the set of independent generators so far constructed.* *)
    Select an element $h_j(H \cap K) \in X_i$ such that the coset $h_j(H \cap K)\langle S_i \rangle$
    is of maximum possible order in $\langle X_i \rangle / \langle S_i \rangle$;
    Let $o(h_j(H \cap K)\langle S_i \rangle) = p_i^{l_j}$; $N_i := p_i^{l_j} N_i$;
    Using L_3 as oracle, by prefix search compute indices

```
20          {α_g | g(H ∩ K) ∈ S_i, 1 ≤ α_g < o(g(H ∩ K))}
21          such that h_j^{o(h_j(H∩K)⟨S_i⟩)} ∈ ∏_{g(H∩K)∈S_i} g^{α_g}(H ∩ K);
22          S_i := S_i ∪ {h_j ∏_{g(H∩K)∈S_i} g^{-α_g/p_i^{l_j}} (H ∩ K)};
23        end-while
24      end-foreach;
25      Output ∪_{i=1}^{r} S_i as the independent generator set;
26      Output Π_{1≤i≤r} N_i;
27      (*N_i is the order of the p_i-Sylow subgroup of G/(H ∩ K), for each i. *)
```

As in machine M of Theorem 4.6, notice in the above code that the structure of abelian groups given by Lemma 4.2 is being crucially used in *lines* 16-21, wherein an independent generator set is being constructed.

The test of the while-loop in *line* 14 can be done by making 1-guarded queries to oracle L_3 (by Claim 4.8.1) since at any stage it is ensured that S_i is an independent set of generators for $\langle S_i \rangle$. We now need to explain how the computation described in *lines* 16 to 21 can be carried out in polynomial time by making suitable 1-guarded queries to the NP oracle L_2 and L_3.

First, consider *lines* 16, 17 and 18 where an element $h_j(H \cap K) \in X_i$ such that the coset $h_j(H \cap K)\langle S_i \rangle$ is of maximum possible order in $\langle X_i \rangle / \langle S_i \rangle$ is selected. If the order of the coset $h_j(H \cap K)\langle S_i \rangle$ is $p_i^{l_j}$, then it implies that l_j is the smallest index such that $h_j^{p_i^{l_j}}(H \cap K) \in \langle S_i \rangle$. Since S_i is an independent set of elements of the p_i-Sylow subgroup of $G/(H \cap K)$, observe that $h_j^{p_i^{l_j}}(H \cap K) \in \langle S_i \rangle$ iff there exist a unique set of indices $\{\alpha_g \mid g(H \cap K) \in S_i\}$, such that $h_j^{p_i^{l_j}} \in (\Pi_{g(H\cap K)\in S_i} g^{\alpha_g})(H \cap K)$. The order $p_i^{l_j}$ of $h_j(H \cap K)\langle S_i \rangle$ can be computed by searching for the unique α_g for each g, such that $h_j^{p_i^{l_j}}(\Pi_{g(H\cap K)\in S_i} g^{-\alpha_g}) \in (H \cap K)$, where $1 \le \alpha_g < o(g(H \cap K))$, for each g. This can be computed with 1-guarded queries to L_3 by Claim 4.8.1.

Furthermore, in *lines* 18 to 21 the machine M_1 computes by prefix search (with appropriate 1-guarded queries L_3 as explained in Claim 4.8.1) this unique set of indices $\{\alpha_g \mid g(H \cap K) \in S_i\}$. ∎

Remark. The machine M_1 of Theorem 4.8 does meaningful computation only if the input satsifies the two specified conditions. In fact, M_1 does not test for the two conditions of the theorem. Actually, we will use M_1 as a subroutine in machines designed in subsequent results only in situations where we can explicitly test for the two conditions. The generality of machine M_1 allows us to adapt it as subroutine in several subsequent proofs.

The following is an immediate corollary useful in the next section.

Corollary 4.9 *Let $\mathcal{B} = \{B_m\}_{m>0}$ be a group family. There is a deterministic oracle machine that takes as input tuples $(0^m, X, Y)$, where $X, Y \subseteq B_m$ are finite sets. Let $G = \langle X \rangle$ and $H = \langle Y \rangle$ and suppose the input $(0^m, X, Y)$ satisfies the properties.*

1. *Y is a canonical generator set for H.*

2. *H is a normal subgroup of G and the factor group G/H is abelian.*

Then the machine outputs a list of coset representatives corresponding to some independent generator set for G/H and also outputs $|G/H|$. Furthermore, it runs in time polynomial in input length, and it makes 1-guarded queries to the NP oracle $L_1 \oplus L_2 \oplus L_3$, where L_3 is another

NP oracle, and L_1 and L_2 are as already defined. If any of the above conditions is not satisfied the machine rejects the input.

Proof. Take $H = K$ in Theorem 4.8. We claim the two conditions of Theorem 4.8 can be tested easily. As for the first condition, testing if H is a normal subgroup of G it is enough to check if $ghg^{-1} \in H$ for each generator g of G and each generator h of H. Since H is presented by a canonical generator set, membership testing in H can be done with 1-guarded queries to L_2. Observe that G/H is abelian iff $g_1g_2g_1^{-1}g_2^{-1} \in H$ for all pairs of generators g_1, g_2 of G. As before this can be tested with 1-guarded queries to L_2. Similarly, it can be tested with 1-guarded queries to L_2 if Y is a canonical generator set for H. Thus the machine tests for all these conditions first and invokes machine M_1 iff the conditions hold. If any of the conditions fail the input is rejected. ■

We now turn to the group intersection problem for abelian groups. The next result about computing $|G \cap H|$ for abelian black-box groups G and H is a corollary of Theorems 4.6 and 4.8.

Corollary 4.10 *Let $\mathcal{B} = \{B_m\}_{m>0}$ be a group family. There is a polynomial-time deterministic oracle machine M_{int} that takes $(0^m, X, Y)$ as input for $X, Y \subseteq B_m$ such that if $\langle X\rangle$ and $\langle Y\rangle$ are abelian groups then M_{int} outputs $|\langle X\rangle \cap \langle Y\rangle|$. Moreover, M_{int} makes only 1-guarded queries to $L_1 \oplus L_2 \oplus L_3$.*

Proof. The machine M_{int} first checks if the input groups are abelian and then simply uses the machine M described in Theorem 4.6 and the machine M_1 described in Theorem 4.8 as subroutines in the following manner. Using machine M, in time polynomial in m and with 1-guarded queries to $L_1 \oplus L_2$, independent generator sets for groups G and H can be computed, and also the order $|G| = \mu$ can be computed. Next, observe that $G \cap H$ is a normal subgroup of G (because G is abelian). Therefore, using machine M_1, in time polynomial in m, with 1-guarded queries to $\mathrm{L}_1 \oplus L_2 \oplus L_3$, the cardinality $|G/(G \cap H)| = \nu$ of $G/(G \cap H)$ can be computed. Finally, notice that $|G \cap H| = |G|/|G/(G \cap H)| = \mu/\nu$, which is output by the machine M_{int}. ■

Again, using Theorem 2.1 we can remove the oracle $L_1 \oplus L_2 \oplus L_3$ to which machine M_{int} of the above corollary makes 1-guarded queries. It yields the following immediate corollary.

Corollary 4.11 *Over any group family, Group Intersection for abelian black-box groups is in* SPP *and hence is low for* PP, $\mathrm{C_{=}P}$ *and the* $\mathrm{Mod}_k\mathrm{P}$ *classes for $k \geq 2$.*

Corollary 4.10 yields some further lowness results for abelian black-box group problems. We show that Group Factorization, Coset Intersection and Double-Coset Membership are low for PP and $\mathrm{C_{=}P}$ in the next corollary.[4]

Corollary 4.12 *Over any group family, Group Factorization, Coset Intersection, and Double Coset Membership for abelian black-box groups are low for* PP *and* $\mathrm{C_{=}P}$.

Proof Sketch. We prove this theorem only for Group Factorization. The proofs of the other problems are very similar. Let $(0^m, S_1, S_2, g)$ be an instance of Group Factorization for abelian black-box groups.

The following NP oracle machine M_{gf} takes $(0^m, S_1, S_2, g)$ as input, it makes 1-guarded queries to $L_1 \oplus L_2 \oplus L_3$, and it has $|B_m|$ accepting paths if there is a factorization for g in $\langle S_1\rangle\langle S_2\rangle$. Moreover, M_{gf} has running time bounded by a polynomial in input length.

[4] We omit the technical detail with which it is possible to show membership in LWPP.

Algorithm M_{gf}:

1 **Input** $(0^m, S_1, S_2, g)$;
2 Simulate machine M of Theorem 4.6 to compute independent generator sets
3 X and Y for $\langle S_1\rangle$ and $\langle S_2\rangle$ respectively;
4 Simulate machine M_{int} of Corollary 4.10 to compute $|\langle S_1\rangle \cap \langle S_2\rangle|$;
5 **Guess** $x \in \Sigma^{p(m)}$; Verify that $x \in B_m$;
6 **if** $x \in \langle X\rangle$ **and** $x^{-1}g \in \langle Y\rangle$ **then**
7 nondeterministically branch into $|B_m|/|\langle S_1\rangle \cap \langle S_2\rangle|$ accepting paths and stop
8 **else reject** and stop
9 **endif**
10 (* *Since X and Y are independent generator sets, membership*
11 *testing in $\langle X\rangle$ and $\langle Y\rangle$ can be done with 1-guarded queries to L_2.* *)

Using machine M of Theorem 4.6, M_{gf} first constructs independent generator sets X and Y for the groups $\langle S_1\rangle$ and $\langle S_2\rangle$ respectively.

Now, observe that $g \in \langle X\rangle\langle Y\rangle$ iff the set $\{x \in \langle X\rangle \mid x^{-1}g \in \langle Y\rangle\}$ is not empty. Suppose $g \in \langle X\rangle\langle Y\rangle$ and let x_0 be some element of $\{x \in \langle X\rangle \mid x^{-1}g \in \langle Y\rangle\}$. It is not hard to see that $\{x \in \langle X\rangle \mid x^{-1}g \in \langle Y\rangle\}$ is precisely the left coset $x_0\langle X\rangle \cap \langle Y\rangle$ of the group $\langle X\rangle \cap \langle Y\rangle$. Therefore it follows that $|\{x \in \langle X\rangle \mid x^{-1}g \in \langle Y\rangle\}| = |\langle X\rangle \cap \langle Y\rangle|$. Note that M_{gf} simulates machine M_{int} of Corollary 4.10 to compute $|\langle X\rangle \cap \langle Y\rangle|$.

Now, if the input is a 'yes' instance then $g \in \langle X\rangle\langle Y\rangle$ and in *line* 5 the number of $x \in \Sigma^{p(m)}$ such that $x \in \langle X\rangle$ and $x^{-1}g \in \langle Y\rangle$, is $|\langle X\rangle \cap \langle Y\rangle|$. Consequently *line* 7 of the machine description will produce a total of $|\langle X\rangle \cap \langle Y\rangle| \cdot |B_m|/|\langle X\rangle \cap \langle Y\rangle| = |B_m|$ accepting paths. On the other hand if the input is a 'no' instance then there are no elements $x \in B_m$ such that $x \in \langle X\rangle$ and $x^{-1}g \in \langle Y\rangle$, and hence M_{gf} has no accepting paths.

Clearly, the running time of M_{gf} can be bounded by a polynomial in m and the input length. As regards the queries made by M_{gf} to $L_1 \oplus L_2 \oplus L_3$, first note that the queries made when M and M_{int} are simulated are all 1-guarded. The other queries made are of the type $(0^m, x, X)$ and $(0^m, x, Y)$ in *line* 6 to L_2. But these queries are also 1-guarded since X and Y are independent generator sets. Now, using arguments as in the proof of Theorem 2.1 the lowness of Group Factorization for PP follows. ■

5 Lowness results for solvable group problems

In this section we prove several problems for solvable black-box groups to be low for PP. Since solvable groups are a generalization of abelian groups, the proofs for the lowness results of this section build on ideas developed in Section 4. We first consider a similar problem of designing a suitable algorithm, as in the case of abelian groups, which takes as input an arbitrary generator set for a black-box solvable group and outputs a canonical generator set for it. The idea behind the algorithm is as follows: given a black-box solvable group G, suppose we can compute the generator sets for each group in its commutator series $G = G_0 > G_1 > \ldots > G_{k-1} > G_k = \langle e\rangle$. Our aim is to convert the inductive argument of the proof of Lemma 3.4 into an algorithm. Thus, given a *canonical* generator set for G_i and a generator set for G_{i-1}, we wish to compute a list of coset representatives corresponding to some independent generator set of G_{i-1}/G_i. Theorem 4.8 of the previous section solves precisely this problem.

Now, coming to the problem of computing a canonical generator set for a black-box solvable group. Let G be a black-box solvable group. We require a suitable algorithm for computing the generator sets for each group in its commutator series $G = G_0 > G_1 > \ldots > G_{k-1} > G_k = \langle e\rangle$.

As a first step we apply a result due to Babai et al. [3]. It gives a randomized algorithm that computes the generator sets for each group in the commutator series of a given black-box solvable group. As a consequence it follows that testing solvability of black-box groups is in co-RP [3]. We abstract in the following technical lemma those aspects of [3, 10] that we require in this paper.

Lemma 5.1 ([3]) *Let $\mathcal{B} = \{B_m\}_{m>0}$ be a group family with $|B_m| \leq 2^{p(m)}$, for a suitable polynomial p. There is a randomized algorithm which takes $(0^m, S)$, for a finite set $S \subseteq B_m$ as input and outputs a sequence of finite sets $S_1, S_2, \ldots, S_{k-1}$, for $k < p(m)$ with the following properties.*

1. *For $i : 1 \leq i \leq k-1$, it holds that $|S_i| \leq c'(p(m))^3$, where $c' > 0$ is a constant.*
2. *For $i : 1 \leq i \leq k-1$, $S_i \subseteq G_i$.*
3. *If $G = \langle S \rangle$ is solvable with commutator series $G = G_0 > G_1 > \ldots > G_k = \langle e \rangle$ then with probability at least $1 - 2^{-m}$ the algorithm outputs $S_1, S_2, \ldots, S_{k-1}$ such that $\langle S_i \rangle = G_i$, for $1 \leq i \leq k-1$.*
4. *If $G = \langle S \rangle$ is not solvable then with probability at least $1 - 2^{-m}$ the algorithm outputs $S_1, S_2, \ldots, S_{k-1}$ such that $\exists a, b \in S_{k-1}$: $ab \neq ba$.*
5. *Furthermore, the algorithm runs in time bounded by a polynomial in the input size.*

Remark. The co-RP algorithm for testing solvability of black-box groups [3] simply runs the randomized algorithm of Lemma 5.1 and accepts the input group as solvable iff $ab = ba$ $\forall a, b \in S_{k-1}$.

We next state a useful technical lemma.

Lemma 5.2 *Let $\mathcal{B} = \{B_m\}_{m>0}$ be a group family. There exists a polynomial time deterministic oracle machine which takes as input a generator set for a group $G < B_m$ and a* canonical *generator set for a subgroup H of G, and test whether H is a normal subgroup of G that contains the commutator subgroup of G. Moreover, the machine makes 1-guarded queries to $L_1 \oplus L_2$. The machine behaviour is unspecified if H is not presented by a canonical generator set.*

Proof. Let S be a generator set for G and T be a canonical generator set for H. It is easy to see that H is a normal subgroup of G iff $\{ghg^{-1} \mid g \in S, h \in T\} \subseteq H$. Thus the machine will have to make $|S| \cdot |T|$ membership tests in the group H. Since T is a canonical generator set for H, this can be done with 1-guarded queries of the form $(0^m, ghg^{-1}, T)$ to oracle L_2. Oracle L_1 is used as usual to prime factorize $|B_m|$ so that the orders of elements in T can be easily computed.

To verify that H contains the commutator subgroup of G, it only remains to check that the factor group G/H is abelian. Since $\{gH \mid g \in S\}$ is a generator set for G/H, it suffices to check that $g_1g_2H = g_2g_1H$, $\forall g_1, g_2 \in S$. These are $|S|^2$ more membership tests of $g_2g_1g_2^{-1}g_1^{-1}$ in H which can be done with 1-guarded queries to L_2. ∎

Putting together the results stated above we now prove the key theorem of this section.

Theorem 5.3 *Let $\mathcal{B} = \{B_m\}_{m>0}$ be a group family. There is a coin-tossing randomized oracle machine R with the following properties*

1. *The machine R takes as input a pair $(0^m, X)$, $X \subseteq B_m$.*

2. *If the group $G = \langle X \rangle$ is a solvable group then with probability at least $1 - 2^{-m}$ R outputs a canonical generator set for G as well as outputs $|G|$. Furthermore, for the remaining fraction of the random coin tosses the machine outputs '?'.*

3. *If the group $G = \langle X \rangle$ is not solvable then for each random coin toss sequence the machine R either rejects the input or outputs '?'.*

4. *The machine R runs in time bounded by a polynomial in input length, and it makes 1-guarded queries to an NP oracle $L_1 \oplus L_2 \oplus L_3$.*

Proof. Let $G < B_m$ be the input group presented by a generator set. The idea of the proof is as follows. First, we use the randomized algorithm C of Lemma 5.1 which checks if G is solvable and if it is, outputs generator sets $S_1, S_2, \ldots, S_{k-1}$, where for each $i : 1 \leq i \leq k-1$, S_i generates G_i and $G = G_0 > G_1 > \ldots > G_k = \langle e \rangle$ is the commutator series for G. Now, observe that if we have a canonical generator set for G_i, then (from Corollary 4.9) using the machine of Theorem 4.8 for the factor group G_{i-1}/G_i we can compute with 1-guarded queries to $L_1 \oplus L_2 \oplus L_3$ a canonical generator set for G_{i-1}. Also note that to start with G_{k-1} is abelian. We can use the machine of Theorem 4.6 to construct an independent generator set for G_{k-1}. Proceeding 'bottom-up' we will finally have the canonical generator set for G.

We now give an outline of the randomized oracle machine for constructing a canonical generator set.

Machine R:

Input $(0^m, S_0)$;
(* *S_0 generates the group $G < B_m$.* *)
Using oracle L_1 compute prime factorisation of $|B_m|$;
Using machine C of Lemma 5.1 compute sequence of sets $S_1, S_2, \ldots, S_{k-1}$;
(* *Recall from Lemma 5.1 that if G is solvable For $i : 1 \leq i \leq k-1$,*
S_i generates G_i, where $G = G_0 > G_1 > \ldots > G_k = \langle e \rangle$ is the commutator series for G.
If G is not solvable then there are $a, b \in S_{k-1}$ that do not commute. *)
if $\exists a, b \in S_{k-1}$: $ab \neq ba$ **then**
Output 'G not solvable' and stop;
 Using machine M of Theorem 4.6 compute an independent
 generator set T_{k-1} for $\langle S_{k-1} \rangle$ and compute $|\langle S_{k-1} \rangle|$;
 (* *S_{k-1} generates G_{k-1}.* *)
 $T := T_{k-1}$; $K := |\langle S_{k-1} \rangle|$;
 for $i = k-1$ **downto** 1 **do**
  if $\langle T \rangle$ contains the commutator subgroup of $\langle S_{i-1} \rangle$ **then**
  (* *The* **if** *conditions above can be tested using the machine*
  described in Lemma 5.2. *)
  begin
  Using the machine of Theorem 4.8 compute
  a set of coset representatives T_{i-1} of an independent generator
  set for $\langle S_{i-1} \rangle / \langle T \rangle$;
  Using the machine of Theorem 4.8 compute $|\langle S_{i-1} \rangle / \langle T \rangle|$;
  Let $K_i := |\langle S_{i-1} \rangle / \langle T \rangle|$;
  $T := T_{i-1} \cup T$; $K := K * K_i$;
  end
  else Output '?';
 endfor;

28 **Output** the set T and number K;

It is easy to see the correctness of the above machine. Assume first that the input group G is solvable. From Lemma 5.1, it follows that in *lines* 4 of the above description, with probability at least $1 - 2^{-m}$ the randomized machine R correctly outputs generator sets for each group in the commutator series of G.

As the first case, consider a correct random construction $S_1, S_2, \ldots, S_{k-1}$, i.e., for $i : 1 \leq i \leq k-1$, S_i generates G_i. Observe that before the for-loop is entered it holds that T is a canonical generator set for G_{k-1}. Now, for an arbitrary $i < k-1$, suppose it holds at the beginning of the ith execution of the for-loop that T is a canonical generator set for G_i. From Lemma 5.2 it follows that the **if** test in *line* 15 succeeds. From Theorem 4.8 and Lemma 3.4 it follows that *lines* 19 to 22 correctly computes T_{i-1} and $T_{i-1} \cup T$ is a canonical generator set for G_{i-1}. Therefore, it follows that in this case the machine will output a set T that is a canonical generator set for G.It also follows from Theorem 4.8 that the number K which is output is $|G|$.

The next case is when the randomly constructed $S_1, S_2, \ldots, S_{k-1}$ is incorrect. There is an i such that S_i does not generate G_i. We know that the input S_0 does generate $G = G_0$. Therefore it must hold for some index i, that $\langle S_i \rangle$ is not the commutator subgroup of $\langle S_{i-1} \rangle$. Actually, in this case it follows from Lemma 5.1 that $\langle S_i \rangle$ is a proper subgroup of the commutator subgroup of $\langle S_{i-1} \rangle$. Let $i_0 \leq k-1$ be the largest such index. From the first case, it follows that at the end of the $k-1-i_0$th execution of the for-loop, the set T computed is a canonical generator set for $\langle S_{i_0} \rangle$. Now, since $\langle T \rangle$ is a proper subgroup of the commutator subgroup of $\langle S_{i_0-1} \rangle$, the **if** test of *line* 15 fails and therefore the machine outputs a '?'.

Now consider the case when G is not solvable. By Lemma 5.1 for the randomly constructed $S_1, S_2, \ldots, S_{k-1}$, with probability at least $1 - 2^{-m}$ there will be a noncommuting pair of generators in S_{k-1}. This is detected in *line* 8. Suppose $\langle S_{k-1} \rangle$ is abelian. Then for the sequence $S_0, S_1, S_2, \ldots, S_{k-1}$ there is a largest index $i : 0 < i \leq k-1$ such that $\langle S_i \rangle$ is not the commutator subgroup of $\langle S_{i-1} \rangle$. Machine R would have computed the canonical generator set T_i for $\langle S_i \rangle$, and in the next iteration in *line* 15 it is detected that $\langle T_i \rangle = \langle S_i \rangle$ does not contain the commutator subgroup of S_{i-1} (that must happen because by construction it holds always that $\langle S_i \rangle$ is contained in the commutator subgroup of $\langle S_{i-1} \rangle$). Thus machine R outputs '?'.

Finally, observe that the oracle $L_1 \oplus L_2 \oplus L_3$ is being accessed indirectly by machine R, via the machines described in Theorems 4.6, 4.8, and Lemma 5.2 that are used as subroutines. The access to the oracle is $L_1 \oplus L_2 \oplus L_3$ by all these machines is 1-guarded. ∎

As a consequence of Theorem 5.3 the next corollary states that several problems for solvable groups are low for PP.

Corollary 5.4 *Over any group family, Membership Testing, Order Verification and Group Isomorphism for black-box solvable groups are in* $\mathrm{ZPP}^{\mathrm{SPP}}$ *and hence are low for* PP.

Proof. Consider an instance $(0^m, S, g)$ of Membership Testing, where S generates a solvable subgroup of B_m. Using the machine R described in Theorem 5.3, we first construct a canonical set of generators T for $\langle S \rangle$. We use the randomized oracle machine R which has exponentially small error probability and makes 1-guarded queries to $L_1 \oplus L_2 \oplus L_3$. Observe that R makes 1-guarded queries to $L_1 \oplus L_2 \oplus L_3$ only *after* the randomized computation is over. Now, with the additional query $(0^m, T, g)$ to L_2 we can decide if $g \in \langle S \rangle$. Observe that this query to L_2 is also 1-guarded since T is a canonical generator set for $\langle S \rangle$.

This entire computation can be seen as a $\mathrm{ZP.P}^{L_1 \oplus L_2 \oplus L_3}$ computation, where only 1-guarded queries are made to $L_1 \oplus L_2 \oplus L_3$. Therefore from Theorem 2.1 it follows that it is a ZP.SPP computation. Since ZPP and SPP are low for PP lowness for PP also follows.

Let $\mathcal{B}$ be a group family. Consider an instance $(0^m, n, X)$ of Order Verification, where X is a generator set for a black-box solvable group $G < B_m$. We use as subroutine the machine R of Theorem 5.3, which computes $K = |\langle X\rangle|$. After this we only have to compare the numbers K and n and accept if the two integers are equal. By the same argument as in the proof of the above theorem it follows that Order Verification is in $\mathrm{ZPP}^{\mathrm{SPP}}$ which in turn implies lowness for PP.

For the second part, let $(0^m, X, Y)$ be an instance of Group Isomorphism, where X and Y generate solvable subgroups of $B_m \in \mathcal{B}$. Using the randomized machine C of Lemma 5.1, we first obtain generator sets for the groups in the commutator series of both $\langle X\rangle$ and $\langle Y\rangle$. Let $\langle X\rangle = G_0 > G_1 > \ldots > G_k = \langle e\rangle$ and $\langle Y\rangle = H_0 > H_1 > \ldots > H_k = \langle e\rangle$ be the two commutator series.

Observe that $\langle X\rangle$ and $\langle Y\rangle$ are isomorphic iff for each $i : 1 \leq i \leq k$, the abelian factor groups G_{i-1}/G_i and H_{i-1}/H_i are isomorphic. Now, we claim that testing the isomorphism of abelian factor groups can be done by querying an oracle that is low for PP (this is easily proved as in Corollary 4.7, if we use in that proof the machine described in Theorem 4.8 instead of the machine of Theorem 4.6). Thus, by querying an oracle that is low for PP we can check in polynomial time if $\langle X\rangle$ and $\langle Y\rangle$ are isomorphic. This completes the proof. ■

Remark. Observe that the randomized oracle machine R of the Theorem 5.3 can be suitably adapted to get an $\mathrm{ZPP}^{\mathrm{SPP}}$ algorithm for checking solvability for black-box groups.

Next we prove that over any group family, Group Intersection, Group Factorisation, Coset Intersection and Double Coset Membership for solvable groups are low for PP.

For the proof of these problems, we need the following easy-to-prove group-theoretic propositions.

Proposition 5.5 *Let G_1 and G_2 be two finite groups and let $H < G_1$. Then for all $g \in G_1$ such that $|gH \cap G_2| \neq 0$, $|gH \cap G_2| = |H \cap G_2|$.*

Proof. Let $g \in G_1$ such that $|gH \cap G_2| \neq 0$. Observe that for any $g' \in gH \cap G_2$ it holds that $gH \cap G_2 = g'H \cap G_2$. Pick such a g'. Consider the mapping $x \mapsto g'x$ from $H \cap G_2$ to $g'H \cap G_2$. It is easy to see that it is a well-defined mapping. Also, if $x_1 \neq x_2$ for $x_1, x_2 \in H \cap G_2$, then $g'x_1 \neq g'x_2$. Hence the mapping is one-to-one. The mapping is onto since for any $y \in g'H \cap G_2$, y can be uniquely expressed as $g'x$ for some $x \in H \cap G_2$. Since the mapping is a 1-1 correspondence it follows that $|gH \cap G_2| = |H \cap G_2|$. ■

Proposition 5.6 *Let G and H be two finite groups and let G' be a normal subgroup of G such that G/G' is abelian. Then $G' \cap H$ is a normal subgroup of $G \cap H$ and the factor group $G \cap H/G' \cap H$ is abelian. Moreover the set $M = \{gG' \mid gG' \cap H \neq \phi\}$ forms a subgroup of G/G', and M is isomorphic to the factor group $G \cap H/G' \cap H$.*

Proof. The proof is by a standard group theoretic argument. Let $x \in G' \cap H$ and $y \in G \cap H$. Since G' is normal in G, it holds that $yxy^{-1} \in G'$. Since $x, y \in H$, it holds that $yxy^{-1} \in H$. Thus, $G' \cap H$ is normal in $G \cap H$. Let $x, y \in G \cap H$. Then $xyx^{-1}y^{-1} \in G'$ since G/G' is abelian, and $xyx^{-1}y^{-1} \in H$ by closure of H. Thus, $xyx^{-1}y^{-1} \in G' \cap H$ implying that the factor group $G \cap H/G' \cap H$ is abelian.

Consider $M = \{gG' \mid gG' \cap H \neq \phi\}$. Since $M \subseteq G/G'$ and G/G' is a finite group, it suffices to check that M is closed under coset multiplication, to see that M is a subgroup. Observe firstly, that if $gG' \cap H \neq \emptyset$ then in fact it holds that $gG' \cap H = x(G' \cap H)$ for any element

$x \in gG' \cap H$. Now suppose $g_1G, g_2G' \in M$. Let $x \in g_1G' \cap H$ and $y \in g_2G' \cap H$. Then xy is in $g_1G'g_2G' = g_1g_2G'$, and $xy \in H$. Hence $g_1g_2G' \cap H \neq \emptyset$ implying that $g_1g_2G' \in M$. In fact, it also holds that $g_1g_2G' \cap H = xy(G' \cap H)$.

To see that M is isomorphic to $G \cap H/G' \cap H$, consider the mapping $gG' \mapsto x(G' \cap H)$ from M to $G \cap H/G' \cap H$, where $x \in gG' \cap H$ is some element. It is easy to see from the above discussion that this mapping is a surjective homomorphism. Since the kernel of this homomorphism is trivial, it is an isomorphism. ■

Lemma 5.7 *Let $\mathcal{B} = \{B_m\}_{m>0}$ be a group family. There is a deterministic polynomial-time oracle machine R_1 that takes $(0^m, X, Y, Z, K)$ as input. Let $G = \langle X \rangle$ and $H = \langle Y \rangle$. Suppose the input has the following properties.*

1. *G and H are solvable groups.*
2. *X and Y are canonical generator sets for G and H respectively.*
3. *Z is a canonical generator set for G' the commutator subgroup for G.*
4. *The number K is $|G' \cap H|$.*

Then the machine R_1 outputs $|G \cap H|$. Furthermore, R_1 makes $|B_m|$-guarded queries to A and 1-guarded queries to $L_1 \oplus L_2 \oplus L_3$, where A is an NP oracle. The behavior of machine R_1 is not specified for inputs that violate any of the above conditions.

Proof. Note that the groups G, G' and H of the theorem satisfy conditions of Proposition 5.6 and we can apply it here. The set $M = \{gG' \mid gG' \cap H \neq \phi\}$ is a subgroup of G/G', and M is isomorphic to the factor group $G \cap H/G' \cap H$. Thus $|M| = |G \cap H/G' \cap H| = |G \cap H|/|G' \cap H|$. Therefore, in order to compute $|G \cap H|$ it suffices for the machine R_1 to compute $|M|$. Furthermore, notice that G/G' is an abelian factor group. Therefore, we can compute the set of coset representatives of an independent set of generators T for G/G' using Theorem 4.8, since G' is presented by a canonical generator set. Recall that actually Theorem 4.8 yields $T = \bigcup_{1 \leq i \leq r} T_i$, where T_i is the independent generator set for the p_i-Sylow subgroup of G/G', where $|B_m| = p_1^{e_1} p_2^{e_2} \ldots p_r^{e_r}$. Let M_i be the p_i-Sylow subgroup of M for $1 \leq i \leq r$. Observe that $|M| = \prod_{1 \leq i \leq r} |M_i|$.

Thus, it suffices to give a machine that computes $|M_i|$ for each i. From such a machine it is easy to design the machine R_1 of the theorem.

We first define the oracle A to which our machine will make guarded acccess. Before defining A, we define the following language A'

$$\begin{aligned} A' := \{(0^m, g, Y, Z, g_1, \ldots, g_k, o_1, \ldots, o_k, K) \mid Y, Z \subseteq B_m, \forall i :\ \exists l_i \leq o_i : \\ g \textstyle\prod_{1 \leq i \leq k} g_i^{-l_i} \langle Y \rangle \cap \langle Z \rangle \neq \emptyset \text{ and } K \text{ divides } |B_m|\}. \end{aligned}$$

Consider the following NP machine N'_A that accepts A.

Machine N'_A

Input $(0^m, g, Y, Z, g_1, g_2 \ldots, g_k, o_1, o_2, \ldots, o_k, K)$;
Check that K divides $|B_m|$;
Guess $l_i \leq o_i$ $\forall i : 1 \leq i \leq k$
Guess $y \in \Sigma^{p(m)}$ and verify that $y \in B_m$;
Guess the L_2 witnesses to verify that $y \in \langle Y \rangle$
Guess the L_2 witnesses to verify $g(\prod_{1 \leq i \leq k} g_i^{-l_i})y \in \langle Z \rangle$;

Branch into $|B_m|/K$ paths and **accept**;

The guarded access property of language A' that we require is summarized in the following claim which is easy to see.

Claim 5.7.1 *Consider a 'yes' instance of A' for which Y and Z are the canonical generator sets for the groups G' and H of the lemma. Also suppose that for this instance $\{g_1, \ldots, g_k\} \subseteq G$, $g \in G$, and K is $|G' \cap H|$. Further, suppose that the set of indices l_i, are unique for this instance of A'. Then such a 'yes' instance of A' is accepted by the machine N_A with exactly $|B_m|$ accepting paths.*

Proof of Claim 5.7.1. The claim follows immediately from the fact that for any $x \in G$ such that $xG' \cap H \neq \emptyset$ it holds that there are exactly $|G' \cap H|$ elements $y \in G'$ such that $xy \in H$, since $|G' \cap H| = |xG' \cap H|$ for every $x \in G$ by Proposition 5.5. □

Let $A = \mathit{Prefix}_{N'_A}(A')$. Now A which can be used to compute the indices $l_1, l_2, \ldots, l_k$ by prefix search.

Now we will describe the machine that computes $|M_i|$ for any $i : 1 \leq i \leq r$. As already described we can assume that using the machine described in Theorem 4.8 we have computed for each $i : 1 \leq i \leq r$ an independent generator set T_i for the p_i-Sylow subgroup of G/G'. Notice that for each $i : 1 \leq i \leq r$ it holds that M_i is a subgroup of $\langle T_i \rangle$. In fact, since $\langle T_i \rangle$ is abelian, M_i is actually a normal subgroup of $\langle T_i \rangle$. We will compute an independent generator set for $\langle T_i \rangle / M_i$ and hence compute $|M_i|$. The reader will notice that it is the same strategy as used in Theorem 4.8.

Machine for computing $|M_i|$

1 **Input** X, Y, Z, T_i, K;
2 (* *$X, Y,$ and Z are the canonical generator sets for G,*
3 *G', and H respectively.* *)
4 (* *The number K is $|G' \cap H|$.* *)
5 $X_i := \{(gG')M_i \mid gG' \in T_i\}$;
6 (* *Notice that X_i generates $\langle T_i \rangle / M_i$* *)
7 $S_i := \phi$; $N_i := 1$;
8 (* *S_i will eventually be an independent generator set for $\langle T_i \rangle / M_i$* *)
9 **while** $X_i \not\subseteq \langle S_i \rangle$ **do**
10 (* *S_i stands for the set of independent generators so far constructed.* *)
11 Select an element $gG'M_i \in X_i$ such that the coset $gG'M_i\langle S_i \rangle$
12 is of maximum possible order in $\langle X_i \rangle / \langle S_i \rangle$; Let $o(gG'M_i) = p_i^l$; $N_i := p_i^l N_i$;
13 Using A as oracle by prefix search compute indices
14 $\{\alpha_h \mid hG'M_i \in S_i, 1 \leq \alpha_h \leq e_i\}$, such that $g^{p_i^l} G' \in \prod_{hG'M_i \in S_i} h^{\alpha_h} G'M_i$;
15 $S_i := S_i \cup \{g \prod_{hG'M_i \in S_i} h^{-\alpha_h/p_i^l} G'M_i\}$;
16 **end-while**
17 Compute $|\langle T_i \rangle|$ using algorithm M_1 of Theorem 4.8;
18 **Output** $|M_i|$ as $|\langle T_i \rangle| / N_i$;
19 (*N_i *is the order of $\langle T_i \rangle / M_i$.* *)

We now explain the computation in *line* 9. Let $gG'M_i \in X_i$. Checking if $gG'M_i$ is in $\langle S_i \rangle$ can be carried out by making the query $(0^m, g, Y, Z, g_1, g_2 \ldots, g_k, o_1, o_2, \ldots, o_k, K)$ to A, where $\{g_1, g_2 \ldots, g_k\}$ is the set of representatives in G corresponding to S_i. By Claim 5.7.1 the query is

$|B_m|$-guarded since the set of indices $l_i, 1 \leq i \leq k$ are unique (because S_i is an independent set of elements). Next consider *lines* 11 and 12. The order of $gG'M_i\langle S_i\rangle$ in $\langle X_i\rangle/\langle S_i\rangle$ is the smallest p_i power p_i^l such that $g^{p_i^l}G'M_i \in \langle S_i\rangle$. Now again, checking if $g^{p_i^l}G'M_i \in \langle S_i\rangle$ holds for some l can be carried out again by queries to A. Notice that since S_i are independent, these queries to A are again all $|B_m|$-guarded by Claim 5.7.1. In *lines* 13 and 14 notice that the unique indices $\{\alpha_h \mid hG'M_i \in S_i, 1 \leq \alpha_h \leq e_i\}$, such that $g^{p_i^l}G' \in \prod_{hG'M_i \in S_i} h^{\alpha_h}G'M_i$ are being computed by prefix search with queries to A. Since these indices are unique, the queries made in the prefix search are also guarenteed to be $|B_m|$-guarded. This completes the proof.

■

As a consequence, we have the following theorem.

Theorem 5.8 *Let $\mathcal{B} = \{B_m\}_{m>0}$ be a group family. Then there is a coin-tossing randomized polynomial time oracle machine that takes $(0^m, X, Y)$ as input which has the following behaviour. Let $\langle X\rangle = G < B_m$ and $\langle Y\rangle = H < B_m$.*

1. *If either G or H is not solvable, with probability at least $1 - 2^{-m}$ it outputs 'Invalid input'. For the remaining fraction of the random coin tosses the machine outputs '?'.*
2. *If both G or H are solvable, then with probability at least $1 - 2^{-m}$ it outputs $|G \cap H|$. For the remaining fraction of the random coin tosses the machine outputs '?'.*
3. *Moreover, the machine makes $|B_m|$-guarded queries to the NP oracle A and 1-guarded queries to $L_1 \oplus L_2 \oplus L_3$.*

Proof Sketch. Let $\mathcal{B} = \{B_m\}_{m>0}$ be a group family. As the first step use the randomized algorithm of Theorem 5.3 to check whether both G and H are solvable and if yes, to compute the canonical generator sets X' and Y' for G and H respectively. (The randomized algorithm R of Theorem 5.3 can be easily modified to output 'Invalid input' if either G or H is not solvable.)

In the next step, we can design a deterministic oracle machine that takes X' and Y' as input and outputs $|G \cap H|$.

Let $G = G_0 > G_1 > \ldots > G_{k-1} > G_k = \langle e\rangle$ be the commutator series for G. Theorem 5.3 actually yields canonical generator sets for each subgroup G_i in this series. To begin with, note that G_{k-1} is abelian and we have an independent generator set for it. Applying the algorithm of Lemma 5.7 to the groups G_{k-1} and H, the algorithm can compute $|G_{k-1} \cap H|$. Now, the algorithm of Lemma 5.7 can be used inductively to compute $|G_{i-1} \cap H|$ given independent generator sets for G_i and H as well as $|G_i \cap H|$, for each i.

Combining the two steps we have the desired randomized machine for computing $|G \cap H|$ that runs in time polynomial in input length, and it makes $|B_m|$-guarded queries to A and 1-guarded queries to $L_1 \oplus L_2 \oplus L_3$, where A is an NP oracle. ■

As a final consequence we have the following corollary.

Corollary 5.9 *Over any group family, Group Intersection, Group Factorization, Coset Intersection, and Double Coset Membership for solvable groups are low for* PP.

Proof. As a consequence of Theorem 5.8, Theorem 2.1, and the fact that BPP is low for PP it follows that Group Intersection for black-box solvable groups is low for PP.

Next, we consider Group Factorization. Consider the following NP machine, which takes $(0^m, S_1, S_2, g, K)$ as input, where S_1 and S_2 are the respective generator sets for two solvable groups $G, H < B_m$. This machine accepts the (extended) language $L = \{(0^m, S_1, S_2, g, K) \mid S_1, S_2$ generate solvable subgroups of $B_m \in \mathcal{B}$ and, $g \in \langle S_1\rangle\langle S_2\rangle$ and K divides $|B_m|\}$.

The NP machine for L

Input $(0^m, S_1, S_2, g, K)$;
Verify that K divides $|B_m|$;
Guess $w \in \Sigma^{p(m)}$;
Verify that $w \in \langle S_1 \rangle$ and $w^{-1}a \in \langle S_2 \rangle$ and
branch into $|B_m|/K$ paths and **accept** on each of them;
(* *Since there are precisely* $|G \cap H|$ *such guesses* w*, this*
machine has exactly $|B_m|$ *accepting paths when the input is*
accepted for $K = |G \cap H|$*.* *)

Note that the NP machine accepts input $(0^m, S_1, S_2, g, K)$ iff $(0^m, S_1, S_2, g)$ is a 'yes' instance of the group factorization problem. Moreover, if K in the input is $|G \cap H|$ then the NP machine accepts with exactly $|B_m|$ accepting paths. Now, consider a randomized oracle machine that takes $(0^m, S_1, S_2, g)$ as input. This machine first computes $|G \cap H|$ using the randomized oracle machine of Theorem 5.8, which makes only 1-guarded or $|B_m|$-guarded queries to an NP oracle. Next it computes canonical generator sets X and Y for G and H respectively using the randomized algorithm of Theorem 5.3, which makes only 1-guarded queries to $L_1 \oplus L_2 \oplus L_3$.

Finally, our machine queries the oracle L for $(0^m, X, Y, g, |G \cap H|)$ and accepts the input iff $(0^m, X, Y, g, |G \cap H|) \in L$. Observe that this query is a $|B_m|$-guarded query. Since we are using a ZPP machine which is making either 1-guarded or $|B_m|$-guarded queries to an NP oracle, using the same ideas as in the proof of Theorem 2.1 [16], it follows that group factorization is low for PP. ■

6 Conclusions

The study of computational problems over black-box groups was introduced in [6]. In [6, 1] the complexity of black-box group problems is investigated mainly in connection with Arthur-Merlin games and their lowness for Σ_2^p and AM is established. In this paper, we have made some new contributions to the study of computational problems over black-box groups. Using group-theoretic machinery specific to solvable groups and complexity-theoretic techniques developed in [16] we have shown that various problems over solvable and abelian black-box groups are in complexity classes that are low for PP.

In particular, we have generalized the notion of independent generator sets for abelian groups. We have defined the notion of canonical generator sets for classes of groups and show that solvable black-box groups have canonical generator sets. For proving the lowness results for solvable groups the crucial result we used was the randomized algorithm stated in Lemma 5.1 [3]. We have adapted this randomized algorithm to a suitable 'zero-error' randomized oracle algorithm that computes a canonical generator set for a given solvable black-box group. The considered problems for abelian groups turn out to be in SPP or LWPP. For solvable groups also these problems are all low for PP, in some cases they are in the class $\mathrm{ZPP}^{\mathrm{SPP}}$. We leave as an open question whether these problems over *nonsolvable* black-box groups are still low for PP.

In the table given below we summarize our main upper bound results for the considered problems over abelian and solvable groups. For the sake of completeness, we also include upper bounds for these problems for the special case of permutation groups. The upper bounds yield the lowness of these problems for PP.

Problem	Black-box Groups		Permutation
	Abelian	Solvable	
Membership Testing	SPP	$\mathrm{ZPP}^{\mathrm{SPP}}$	P [13]
Order Verification	SPP	$\mathrm{ZPP}^{\mathrm{SPP}}$	P [13]
Group Isomorphism	SPP	$\mathrm{ZPP}^{\mathrm{SPP}}$	
Group Intersection	SPP	Low for PP	SPP [16]
Group Factorization	LWPP	Low for PP	LWPP [16]
Coset Intersection	LWPP	Low for PP	LWPP [16]
Double-coset Membership	LWPP	Low for PP	LWPP [16]

Upper bounds on the complexity of various group-theoretic problems.

Acknowledgements. We thank the referees for their careful refereeing and valuable suggestions which have considerably helped in improving the presentation of the paper.

References

[1] L. Babai. Bounded round interactive proofs in finite groups. *SIAM Journal of Discrete Mathematics,* 5: 88-111, 1992.

[2] L. Babai. Trading group theory for randomness. *17th ACM Symp. Theory of Computing,* 421-429, 1985.

[3] L. Babai, G. Cooperman, L. Finkelstein, E. Luks and Á. Seress. Fast Monte Carlo algorithms for permutation groups. In *Journal of Computer and System Sciences,*50: 296-308, 1995.

[4] L. Babai and S. Moran. Arthur-Merlin games: A randomized proof system, and a hierarchy of complexity classes. *Journal of Computer and System Sciences,* 36: 254-276, 1988.

[5] L. Babai, E. Luks and Á. Seress. Fast management of permutation groups. *Proc. 28th IEEE Symposium on Foundations of Computer Science,* 272-282, 1988.

[6] L. Babai and M. Szemerédi. On the complexity of matrix group problems I. *Proc. 25th IEEE Symposium on Foundations of Computer Science,* 229-240, 1984.

[7] R. Beals and L. Babai. Las Vegas algorithms for matrix groups. *Proc. 25th ACM Symposium on Theory of Computing,* pp. 427-436, 1993.

[8] R. Boppana, J. Hastad and S. Zachos. Does coNP have short interactive proofs? Information Processing Letters, 25: 127-132, 1987.

[9] W. Burnside. *Theory of Groups of Finite Order*, Dover Publications, INC, 1955.

[10] G. Cooperman and L. Finkelstein. Combinatorial tools for computational group theory. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science,* Vol. 11, 1993.

[11] S. Fenner, L. Fortnow, S. Kurtz. Gap-definable counting classes. *Proc. 6th Structure in Complexity Theory Conference,* 30-42, 1991.

[12] M. Fellows and N. Koblitz. Self-witnessing polynomial time complexity and prime factorization. *Proc. 6th Structure in Complexity Theory Conference,* 107-110, 1992.

[13] M. Furst, J. E. Hopcroft and E. Luks. Polynomial time algorithms for permutation groups. *Proc. 21st IEEE Symposium of Foundations of Computer Science*, 36-45, 1980.

[14] M. Hall. *The Theory of Groups.* Macmillan, New York, 1959.

[15] C. Hoffmann. *Group-Theoretic Algorithms and Graph Isomorphism.* Lecture Notes in Computer Science #136, Springer Verlag, 1982.

[16] J. Köbler, U. Schöning, J. Torán. Graph isomorphism is low for PP. *Journal of Computational Complexity,* 2: 301-310, 1992.

[17] E. M. Luks. Computing in solvable matrix groups. *Proc. 33rd IEEE Symposium on Foundations of Computer Science,* 111-120, 1992.

[18] U. Schöning. Graph isomorphism is in the low hierarchy. *Journal of Computer and System Sciences,* 37: 312-323, 1988.