International Electronic Journal of Algebra
Volume 2 (2007) 54-70

SUBGROUP LATTICE ALGORITHMS RELATED TO EXTENDING AND LIFTING ABELIAN GROUPS

Septimiu Crivei and Ştefan Şuteu Szöllősi

Received: 1 November 2006; Revised: 16 February 2007
Communicated by Sait Halıcıoğlu

Abstract. We develop algorithms for determining properties of finite abelian groups related to the notions of extending and lifting groups. Thus, we give efficient methods, on one hand to check the properties of being direct summand, essential, superfluous, coessential, complement (closed), supplement (coclosed) subgroup, and on the other hand to determine all subgroups with the mentioned properties of a given finite abelian group.

Mathematics Subject Classification (2000): 20K01, 20E15, 68R99
Keywords: Subgroup lattice, essential subgroup, superfluous subgroup, complement, supplement, extending abelian group, lifting abelian group.

1. Introduction

Extending and lifting modules have been intensively investigated in the last two decades due to their important applications to ring and module theory. The reader is referred to the monographs Dung et al. [4] and Clark et al. [2] for more information on them. A number of generalizations have been considered (for instance, see [5], [6], [9]) and their study is still developing. Establishing their structure and classifying them have been difficult tasks, even in special cases such as abelian groups. The structure of abelian groups that are extending or lifting is determined, but this is still to be done in the case of most of their generalizations.

From the computational point of view, a useful tool - still far from being used at its full capacity - is the computer algebra system GAP (Groups, Algorithms, and Programming) [10], able (among many other things) to compute the subgroup lattice of a finitely generated group. This is the framework where we develop algorithms for determining special subgroups of a finite abelian group, that are the bricks of working with extending groups, lifting groups and some of their generalizations. More precisely, we give efficient methods, on one hand to check the properties of being direct summand, essential, superfluous, coessential, complement (closed), supplement (coclosed) subgroup, and on the other hand to determine all subgroups with the mentioned properties of a given finite abelian group. Moreover, GAP has

This work was supported by the grant CNCSIS-AT 72/2006.

an optional package, called XGAP, able to visualize the generated subgroup lattice, and the results of our algorithms can be easily spotted out in XGAP. We believe that such algorithms, together with their implementation in GAP, will be useful tools both for easily obtaining examples as well as for testing some conjectures, before proving them rigorously.

Throughout G is a *finite* abelian group, unless specified otherwise. We denote by $A \leq G$ the fact that A is a subgroup of G. The definitions of the notions used in the paper are mainly taken from [2] and [4].

2. The subgroup lattice

Let us start by recalling a few elementary things on the subgroup lattice of a finite abelian group.

Proposition 2.1. *The set $S(G)$ of subgroups of G with respect to the inclusion is a modular self-dual lattice.*

Definition 2.2. Let $(L, \leq)$ be a lattice with 0. An element $0 \neq a \in L$ is called an *atom* if the interval $[0, a]$ consists only of 0 and a. The lattice L is called *atomic* if for every $0 \neq a \in L$, the interval $[0, a]$ has at least one atom.

One also has the notions of *dual atom* and *dually atomic lattice* when considering the notions of atom and atomic lattice in the dual lattice of $(L, \leq)$.

Proposition 2.3. *(i) The atoms of $S(G)$ are the simple subgroups of G and the dual atoms of $S(G)$ are the maximal subgroups of G.*

(ii) $S(G)$ is atomic and dually atomic.

We are going to see the subgroup lattice of an abelian group as a directed graph (digraph) with arcs always pointing "upwards". As usual, for two elements a and b of a lattice, $a < b$ means $a \leq b$ and $a \neq b$. If $a < b$ and there is no element c in the lattice such that $a < c < b$, we denote this by $a \prec b$. We construct the directed graph $\vec{\Gamma} = (V, E, \vec{\mathcal{G}})$, where V is a nonempty set of vertices, E is a set of arcs and $\vec{\mathcal{G}} : E \to V \times V$. In our construction $e = (x, y) \in E \iff a_x \prec b_y$, if a_x and b_y are the corresponding elements from $S(G)$.

Denote by $f : S(G) \to V$ the bijection between the subgroups of G (elements of $S(G)$) and the corresponding vertices in the digraph. Also, denote by $\delta(k)$ the number of divisors of $k \in \mathbb{N}$. Let us increasingly order the divisors d_i of $|G|$ and define, for each $i \in \{1, \ldots, \delta(|G|)\}$, $level(i) := \{H \in S(G) | \, |H| = d_i\}$. Note that if $e = (f(A), f(B)) \in E$, then $n_A < n_B$, where $A \in level(n_A)$, $B \in level(n_B)$. Also $(level(i))_{1 \leq i \leq \delta(|G|)}$ is a partition of $S(G)$.

Recall now the definition of neighborhood(s) of a vertex y in a digraph $\vec{\Gamma}$:

$$N^{in}_{\vec{\Gamma}}(y) = \{x \in V | \vec{\mathcal{G}}(x, y) \neq \emptyset\}, \quad N^{out}_{\vec{\Gamma}}(y) = \{x \in V | \vec{\mathcal{G}}(y, x) \neq \emptyset\}.$$

Remark. In general, we shall try to minimize references to GAP functions in order to allow the reader to follow just the description of our algorithms and not their implementation. But sometimes we shall mention the existence of some appropriate functions. For instance, we should note that one can build the subgroup lattice of G in GAP by using for example the built-in function `LatticeSubgroups(G)`. In GAP the subgroup lattice is a data structure allowing tests for minimality/maximality relations for each subgroup (i.e. we can check the condition $a \prec b$ with built-in functions). After the subgroup lattice of G is constructed, this information can be retrieved from GAP in numerous ways (one is to use the built-in functions `MaximalSubgroupsLattice(L)` and `MinimalSubgroupsLattice(L)`). In the description of algorithms throughout the paper we shall consider this information already known.

3. Direct summands

Many of the notions defined here and in the next sections are related to direct summands. So that, we begin by presenting some fast ways to test if a given subgroup of G is a direct summand, and to find out all the pairs of direct summands of G.

Let $A, B \in S(G)$ be such that $G = A \oplus B$. Then it is easy to see that $|G| = |A| \cdot |B| = |G : A| \cdot |G : B|$. Moreover, since the lattice $S(G)$ is self-dual, we immediately have the following result.

Proposition 3.1. *Let $A, B \in S(G)$. Then $G = A \oplus B$ if and only if $|G| = |G : A| \cdot |G : B|$ and $A \cap B = 0$.*

In other words, Proposition 3.1 tells us that if $A \in S(G)$, then we should look for a direct summand B of G such that $G = A \oplus B$ only among the subgroups of G of order $|G : A|$.

Remark. We mention that after the construction of $S(G)$ it is easy in GAP to retrieve all the subgroups of G with the same index.

Our functions `IsDirectSummand(G,A)` to check if a given subgroup A of G is a direct summand, and `DirectSummands(G)` to determine all pairs of direct summands of G are described in Algorithms 1 and 2. In Algorithm 2 we use the idea from Algorithm 1 put in terms of levels in the subgroup lattice.

Now let us consider separately the two properties involved in the writing $G = A \oplus B$, namely $A \cap B = 0$ and $A + B = G$. They will be important later on when we work with complements and supplements.

Denote $\mathcal{Z}_A = \{B \in S(G) \mid A \cap B = 0\}$. Clearly, every subgroup of G containing A or contained in A does not belong to $\mathcal{Z}_A$. We slightly modify the construction

Algorithm 1 `IsDirectSummand`(G,A)

Input : G finite abelian group, $A \leq G$
Output : $e = TRUE$ if A is a direct summand of G; $e = FALSE$ otherwise

```
n := δ(|G|), e := FALSE
for all B ∈ {H ∈ S(G) | |H| = |G : A|} do
  if A ∩ B = 0 then
    e := TRUE
    break
  end if
end for
return e
```

Algorithm 2 `DirectSummands(G)`

Input : G finite abelian group
Output : the set $\mathcal{D}$ of all pairs (A, B) of subgroups with $A \oplus B = G$

```
n := δ(|G|), D := ∅
for i:=1 to ⌈n/2⌉ do
  for all A ∈ level(i) do
    for all B ∈ level(n + 1 − i) do
      if A ∩ B = 0 then
        D := D ∪ {(A, B)}
      end if
    end for
  end for
end for
return D
```

of our digraph $\vec{\Gamma} = (V, E, \vec{\mathcal{G}})$. Thus we define a digraph $\vec{\Gamma}' = (V, E', \vec{\mathcal{G}}')$, where the only difference with respect to the initial one is that we "reverse" the arcs between the vertices corresponding to subgroups of A. Hence, if $D \not\leq A$, then $(f(C), f(D)) \in E' \iff (f(C), f(D)) \in E$, and if $D \leq A$, then $(f(C), f(D)) \in E' \iff (f(D), f(C)) \in E$.

Proposition 3.2. *Let $A, B, C \in S(G)$. If $A \cap B = C$, then there exists a path $p_{AB} : f(A) = a_0, \ldots, a_c = f(C), \ldots, a_m = f(B)$ in $\vec{\Gamma}'$ such that $n_i > n_{i+1}$ for every $0 \leq i < c$ and $n_j < n_{j+1}$ for every $c \leq j < m$, where $f^{-1}(a_k) \in level(n_k)$, $0 \leq k \leq m$.*

Proof. Let $A, B, C \in S(G)$ be such that $A \cap B = C$. Since $C \subseteq A$ and $C \subseteq B$, there are chains $C \prec H_1 \prec \ldots \prec H_r \prec A$ and $C \prec K_1 \prec \ldots \prec K_s \prec B$ of subgroups of G, hence there exist paths $p_{AC} : f(A) = a_0, \ldots, a_c = f(C)$ and

$p_{CB} : f(C) = a_c, \ldots, a_m = f(B)$ in $\vec{\Gamma}'$. Obviously, the concatenation of p_{AC} and p_{CB} leads to a path between $f(A)$ and $f(B)$ passing through $f(C)$. □

Corollary 3.3. *Let $A, B \in S(G)$. Suppose there is a path p_{AB} between $f(A)$ and $f(B)$ in $\vec{\Gamma}'$ such that $f(0) \notin p_{AB}$. Then $A \cap B \neq 0$.*

First let us solve the subproblem of computing the set $\mathcal{Z}_A$, based on the previous corollary. In practice we may proceed as follows:

- After the usual construction of the subgroup lattice and digraph $\vec{\Gamma}$, we isolate (i.e. remove all incoming and outgoing arcs) the vertex corresponding to the trivial subgroup ($f(0)$) and all the vertices $v \in V$ such that $A < f^{-1}(v)$.
- "Reverse" the arcs for the vertices corresponding to subgroups of A.
- Determine all $B \in S(G)$ such that there is a path p_{AB}. Since we have removed the trivial subgroup, for all these subgroups we have $A \cap B \neq 0$. For the remaining vertices $f(B')$ a path would have existed only through $f(0)$, hence $A \cap B' = 0$. Also, $A \cap 0 = 0$.

We wrap this method in Algorithm 3, function `NullIntersectors(G,A)`. Before running the main part of the algorithm, we could check if A is essential in G (see the corresponding section). If it is so, then clearly $\mathcal{Z}_A = \{0\}$.

Given a subgroup A of G, we have an algorithm dual to Algorithm 3 for finding the set $\mathcal{F}_A = \{B \in S(G) | A + B = G\}$. For later use, let us call this function `FullSummands(G,A)`.

4. Essential subgroups

Definition 4.1. A subgroup $A \leq G$ is called *essential* in G if for every $B \leq G$ the equality $A \cap B = 0$ implies $B = 0$. Notation: $A \trianglelefteq G$.

Definition 4.2. The *socle* of G is the sum of all simple subgroups of G, or equivalently, the intersection of all essential subgroups of G. Notation: $Soc(G)$. By convention, we take $Soc(G) = 0$ if G has no simple subgroup.

In the case of (not necessarily finite) abelian groups we have the following characterization.

Theorem 4.3. [1, Ex. S10.10] *Let $A \leq G$. Then $A \trianglelefteq G$ if and only if $Soc(G) \subseteq A$ and the quotient group G/A is torsion.*

The fact that a group is *torsion* can be described by the property that its subgroup lattice is atomic [1, Ex. M10.2], which holds if the group is finite.

The first function of this section checks whether a given subgroup H is essential in G or not and it is based on Theorem 4.3. GAP has a function `Socle(G)` that can be used.

Algorithm 3 `NullIntersectors(G,A)`

Input : G finite abelian group, $A \leq G$
Output : $\mathcal{Z}_A = \{B \leq G | A \cap B = 0\}$
  if `IsEssential(G,A)` **then**
    $\mathcal{Z}_A := \{0\}$
  else
    {reverse the arcs for all subgroups of A}
    $\mathcal{S}_A := \{A\}$, $\mathcal{S}' := \mathcal{S}_A$
    repeat
      $\mathcal{S}' := \{B \in S(G) | f(B) \in N^{in}_{\vec{\Gamma}}(f(H)), H \in \mathcal{S}'\}$, $\mathcal{S}_A := \mathcal{S}_A \cup \mathcal{S}'$
    until $0 \in \mathcal{S}_A$
    $Old := \emptyset$, $New := \emptyset$
    for all $e = (f(A), f(B)) \in E$ such that $A, B \in \mathcal{S}_A$ **do**
      $Old := Old \cup \{e\}$
      $New := New \cup \{(f(B), f(A))\}$
    end for
    $E := E \setminus Old$
    $E := E \cup New$
    {determine the set $\mathcal{E}_A$ of all proper extensions of A}
    $\mathcal{E}_A := \{A\}$, $\mathcal{E}' := \mathcal{E}_A$
    repeat
      $\mathcal{E}' := \{B \in S(G) | f(B) \in N^{out}_{\vec{\Gamma}}(f(H)), H \in \mathcal{E}'\}$, $\mathcal{E}_A := \mathcal{E}_A \cup \mathcal{E}'$
    until $G \in \mathcal{E}_A$
    $\mathcal{E}_A := \mathcal{E}_A \setminus \{A\}$
    {isolate the vertices in $\mathcal{E}_A$ and 0}
    $E := E \setminus (\{(a,b) \in E | f^{-1}(b) \in \mathcal{E}_A\} \cup \{(a,b) \in E | f^{-1}(a) \in \mathcal{E}_A\})$
    $E := E \setminus (\{(f(0), b) \in E\} \cup \{(a, f(0)) \in E\})$
    {compute the set $\bar{\mathcal{C}}_A$ of all subgroups B such that $A \cap B \neq 0$}
    $\bar{\mathcal{C}}_A := \{A\}$, $\bar{\mathcal{C}}' := \bar{\mathcal{C}}_A$
    repeat
      $\bar{\mathcal{C}}' := \{B \in S(G) | f(B) \in N^{out}_{\vec{\Gamma}}(f(H)), H \in \bar{\mathcal{C}}'\}$, $\bar{\mathcal{C}}_A := \bar{\mathcal{C}}_A \cup \bar{\mathcal{C}}'$
    until $\bar{\mathcal{C}}_A$ is not modified anymore
    $\mathcal{Z}_A := S(G) \setminus \left(\mathcal{E}_A \cup \bar{\mathcal{C}}_A\right)$
  end if
  return $\mathcal{Z}_A$

However, if one is interested in obtaining all the essential subgroups in a given subgroup lattice, the repeated use of the previous function turns out to be inefficient. With the method presented in our Algorithm 5 we can spot out all the

Algorithm 4 `IsEssential(G,A)`

Input : G finite abelian group, $A \leq G$
Output : $e = TRUE$ if $A \trianglelefteq G$; $e = FALSE$ otherwise
  if `Socle(G)` $\subseteq A$ **then**
    $e := TRUE$
  else
    $e := FALSE$
  end if
  return e

essential subgroups of G. We use the classical FIFO data structure of a queue. Recall that a queue Q is described by the following operations:

- *empty(Q)* returns an empty queue
- *push(Q,el)* returns a queue consisting of the elements of Q and *el* (inserted at the end)
- *pop(Q)* if Q is not empty, returns a queue consisting in all but the first element of Q
- *top(Q)* returns the first element of the queue Q (if it is not empty)

We use the classical idea of breadth-first traversal of the digraph. For keeping track of the already visited vertices, we use a boolean-valued list as a marker.

An important notion connected to essential subgroups is the following one.

Definition 4.4. $G \neq 0$ is called *uniform* if for every $0 \neq A, B \leq G$ we have $A \cap B \neq 0$, or equivalently, every non-zero subgroup of G is essential in G.

We should note that the structure of uniform abelian groups is well-known, the uniform finite abelian groups being exactly those isomorphic to cyclic groups $\mathbb{Z}_{p^n}$ for some prime p and natural number n. Nevertheless, one may write a simple function `IsUniform(G)` to determine if a given finite abelian group is uniform or not, based on the following observation. If A and B are two different atoms in the lattice $S(G)$, then $A \cap B = 0$ implies $A \not\trianglelefteq G$ and G is not uniform. However, if A is the only atom of $S(G)$, then $A = Soc(G) \subseteq H$ for every $H \leq G$, so that G is uniform.

5. Superfluous subgroups

Definition 5.1. A subgroup $A \leq G$ is called *superfluous* in G if for every $B \leq G$ the equality $A + B = G$ implies $B = G$. Notation: $A << G$.

Definition 5.2. The *radical* of G is the intersection of all maximal subgroups of G, or equivalently, the sum of all superfluous subgroups of G. Notation: $Rad(G)$. By convention, we take $Rad(G) = G$ if G has no maximal subgroup.

Algorithm 5 `EssentialSubgroups(G)`

Input : G finite abelian group
Output : the set $\mathcal{E}$ of all essential subgroups of G

```
E := ∅, u := f(Socle(G));
for all v ∈ V do
  marked[v] := FALSE
end for
Q := empty(Q)
Q := push(Q, u)
marked[u] := TRUE
while Q is not empty do
  x := top(Q), Q := pop(Q)
  for all y ∈ {z ∈ N_Γ^out(x) | marked[z] = FALSE} do
    Q := push(Q, y)
    marked[x] := TRUE
  end for
end while
for all v ∈ V do
  if marked[v] = TRUE then
    E := E ∪ {f^{-1}(v)}
  end if
end for
return E
```

The basic characterization of superfluous subgroups of an (not necessarily finite) abelian group is the following one.

Theorem 5.3. [1, Ex. D10.5] *Let $A \leq G$. Then $A << G$ if and only if $A \subseteq Rad(G)$ and A has no divisible quotient groups.*

The fact that a group is *divisible* can be described by the property that its lattice subgroup has no dual atoms [1, Ex. S10.9]. Since divisible groups are infinite, note that the condition on divisible quotient groups from the previous theorem is satisfied by any finite abelian group. In the same setting, now it is clear that superfluous subgroups in the lattice $S(G)$ are just the essential subgroups in the dual lattice of $S(G)$. Hence the algorithms for deciding if a given subgroup A of G is superfluous or not (called `IsSuperfluous(G,A)`), and for finding all the superfluous subgroups of G (called `SuperfluousSubgroups(G)`) are easily dualized from those in the section on essential subgroups. Because of duality it is reasonable to "mirror" the digraph: $e = (y, x) \in E \iff a_x \prec b_y$, if a_x and b_y are the corresponding elements from $S(G)$, i.e. the arcs point "downwards".

A dual notion to that of uniform group is the following one.

Definition 5.4. $G \neq 0$ is called *hollow* if for every $A, B < G$ we have $A + B < G$, or equivalently, every proper subgroup of G is superfluous in G.

By the self-dual property of $S(G)$, it is easy to see that a finite abelian group is hollow if and only if it is uniform. Hence, one can use the same function `IsUniform(G)` to test both these properties.

6. Coessential subgroups

A notion related to superfluous subgroups is that of coessential subgroup.

Definition 6.1. Let $B \leq A \leq G$. Then B is called a *coessential* subgroup of A in G if $A/B << G/B$.

GAP is able to work with factor groups, so that Algorithm 6 (for checking if a given subgroup B is a coessential subgroup of A in G or not) follows immediately.

Algorithm 6 `IsCoessential(G,A,B)`

Input : G finite abelian group, $B \leq A \leq G$
Output : $e = TRUE$ if B is a coessential subgroup of A in G;
$e = FALSE$ otherwise
$e :=$ `IsSuperfluous(G/B, A/B)`;
return e

However, if we are interested in finding all the coessential subgroups of a given subgroup in G, another method is required. We develop such a method based on the following theorem. In the case of a finite group we have:

Theorem 6.2. [2, 3.2] *Let $B \leq A \leq G$. Then $A = B + S$ for some $S << G$ if and only if B is a coessential subgroup of A in G.*

We proceed as follows.

- first determine the set $\mathfrak{S}'_A = \{S \leq A | S << G\}$
- compute $\mathcal{Q} = \{S \leq A | S << G$ and S not superfluous in A$\}$ (clearly, for any $C \in \{S \leq A | S << G$ and $S << A\}$ we cannot have $C + B = A$ for some $B < A$)
- $\mathfrak{C}_A := \bigcup_{S \in \mathcal{Q}} \{B \leq A | B + S = A\} \cup \{A\}$

A formal description is given in Algorithm 7.

Algorithm 7 `CoessentialSubgroups(G,A)`

Input : G finite abelian group, $A \leq G$

Output : the set $\mathfrak{C}_A$ of all coessential subgroups of A in G

$\mathfrak{C}_A := \{A\}$
$\mathfrak{S}'_A :=$ `SuperfluousSubgroups(G)`
$\mathcal{S}_A := \{A\}$, $\mathcal{S}' := \mathcal{S}_A$
repeat {compute all subgroups of A}
$\mathcal{S}' := \{B \in S(G) | f(B) \in N_{\bar{\Gamma}}^{out}(f(H)), H \in \mathcal{S}'\}$
$\mathcal{S}_A := \mathcal{S}_A \cup \mathcal{S}'$
until $0 \in \mathcal{S}_A$
$\mathfrak{S}'_A := \mathfrak{S}'_A \cap \mathcal{S}_A$
$\mathfrak{S}_A :=$ `SuperfluousSubgroups(A)`
$\mathcal{Q} := \mathfrak{S}'_A \setminus \mathfrak{S}_A$
$\mathfrak{C}_A := \emptyset$
for all $S \in \mathcal{Q}$ **do**
$\zeta_S :=$ `FullSummands(A,S)`
$\mathfrak{C}_A := \mathfrak{C}_A \cup \zeta_S$
end for
return $\mathfrak{C}_A$

7. Complement and closed subgroups

A notion generalizing direct summands is that of complement subgroup.

Definition 7.1. Let $A \leq G$. A subgroup B of G is called a:
(i) *complement of A in G* if it is maximal in the set of subgroups C of G with $A \cap C = 0$.
(ii) *complement* if there is $A \leq G$ such that B is a complement of A in G.

Theorem 7.2. *(i)* [4, p.6] *Every subgroup of G has a complement.*
(ii) Every direct summand of G is a complement subgroup.

Strongly related to complement subgroups are the closed subgroups.

Definition 7.3. A subgroup $A \leq G$ is called *closed* in G if A has no proper essential extension in G.

Theorem 7.4. [4, p.6] *Let $A \leq G$. Then A is a complement subgroup if and only if A is closed in G.*

We first determine if a given subgroup A of G is a complement (closed) subgroup. Afterwards, we compute all its complements in G, and finally we determine all the complement (closed) subgroups of G.

In order to check if a subgroup is closed, note that if $A, B \in S(G)$ with $A \subseteq B$, then $A \trianglelefteq B$ if and only if A and B contain the same atoms in $S(G)$. Hence a subgroup A of G is closed if and only if it has no extension containing the same atoms as A. So, if A is not closed, then it must have a minimal extension containing the same atoms as A.

Algorithm 8 `IsClosed(G,A)`

Input : G finite abelian group, $A \leq G$
Output : $e = TRUE$ if A is a complement (closed) subgroup of G;
$e = FALSE$ otherwise
if $A = G$ **then**
$e := TRUE$
return e
end if
if $|\{a \in N_{\vec{\Gamma}}^{out}(f(0)) \mid f^{-1}(a) \subseteq A\}| < min\{|\{a \in N_{\vec{\Gamma}}^{out}(f(0)) \mid f^{-1}(a) \subseteq f^{-1}(b)\}| \mid b \in N_{\vec{\Gamma}}^{out}(f(A))\}$ **then**
$e := TRUE$
else
$e := FALSE$
end if
return e

According to Definition 7.1, we have to build up the set of all subgroups $C \leq G$ with $A \cap C = 0$. The description of `Complements(G,A)` is given in Algorithm 9 and makes use of Algorithm 3. The breadth-first traversal of the digraph can be implemented using the method in Algorithm 5, with a queue data structure.

Algorithm 9 `Complements(G,A)`

Input : G finite abelian group, $A \leq G$
Output : the set $\mathcal{C}_A$ of all complements of A in G
{determine the set $\mathcal{Z}_A$}
$\mathcal{Z}_A :=$ `NullIntersectors(G, A)`
{the elements of $\mathcal{C}_A$ are the maximal elements of $\mathcal{Z}_A$}
$\mathcal{C}_A := \{B \in \mathcal{Z}_A | N_{\vec{\Gamma}}^{out}(f(B)) \subseteq \{f(N) | N \in S(G) \setminus \mathcal{Z}_A\}\}$
return $\mathcal{C}_A$

The function `ClosedSubgroups(G)`, which determines the closed (complement) subgroups of G, is presented in Algorithm 10. The construction of the digraph and the notations are the same as in Section 4. We use again the idea from Algorithm 8, using also the fact that if a subgroup of G contains a certain set of atoms of

$S(G)$, then any of its extensions contains at least those atoms. For a vertex $v \in V$, we denote by $atm[v]$ the set of all vertices $a \in V$ such that $f^{-1}(a)$ is an atom of $S(G)$ contained in $f^{-1}(v)$.

Algorithm 10 `ClosedSubgroups(G)`

Input : G finite abelian group
Output : the set $\mathcal{C}$ of all closed subgroups of G

```
if G = 0 then
  C := {0}
  return C
end if
C := ∅
for all v ∈ V do
  marked[v] := FALSE
  atm[v] := ∅
end for
for all a ∈ N^out_Γ(f(0)) do
  atm[a] := {a}
end for
Q := empty(Q)
Q := push(Q, f(0))
while Q is not empty do
  x := top(Q)
  Q := pop(Q)
  for all y ∈ N^out_Γ(x) do
    atm[y] := atm[y] ∪ atm[x]
    if marked[y] = FALSE then
      Q := push(Q, y)
      marked[y] := TRUE
    end if
  end for
end while
C := {H ∈ S(G) | |atm[f(H)]| < min{|atm[y]| | y ∈ N^out_Γ(f(H))}}
return C
```

Now let us recall the definition of a closure of a subgroup in a group.

Definition 7.5. Let $A \leq G$. A subgroup C of G such that $A \subseteq C$ is called a *closure* of A in G if C is a maximal essential extension of A in G, or equivalently, $A \trianglelefteq C$ and C is closed in G.

Theorem 7.6. [4, p.6] *Every subgroup of G has a closure in G.*

As an application, one can easily write a function `Closures(G,A)` to get all the closures in G of a subgroup A of G by using the functions `ClosedSubgroups(G)` and `IsEssential(G,A)`.

8. Supplement and coclosed subgroups

Another notion generalizing direct summands, dual to a complement subgroup, is that of supplement subgroup.

Definition 8.1. Let $A \leq G$. A subgroup B of G is called a:

(i) *supplement of A in G* if it is minimal in the set of subgroups C of G with $A + C = G$.

(ii) *supplement* if there is $A \leq G$ such that B is a supplement of A in G.

We have the following result, whose first part holds because G is finite.

Theorem 8.2. *(i) Every subgroup of G has a supplement.*

(ii) Every direct summand of G is a supplement subgroup.

Definition 8.3. A subgroup A of G is called *coclosed* in G if there is no proper coessential subgroup of A in G.

We have the following result for a not necessarily finite abelian group G.

Theorem 8.4. [2, 20.3] *If G is* weakly supplemented *(i.e. every subgroup of G has a supplement), then a subgroup A of G is a supplement if and only if it is coclosed in G.*

Note that every supplement subgroup of a (not necessarily finite) abelian group is a complement subgroup [7, Theorem 4.1.4]. Since every finite abelian group is clearly weakly supplemented, we can use [7, Theorem 4.3.1] to get the following stronger result.

Theorem 8.5. *The following are equivalent for a subgroup A of G:*

(i) A is a complement.

(ii) A is closed in G.

(iii) A is a supplement.

(iv) A is coclosed in G.

Of course, if a subgroup A of G is a complement for a subgroup B of G, this does not mean that A is a supplement for the same subgroup B of G. Hence one needs a function `Supplements(G,A)` to determine all supplements of A in G, dual to `Complements(G,A)`. But the function `IsClosed(G,A)` checks also if A is a

supplement (coclosed) subgroup of G, whereas the function `ClosedSubgroups(G)` gives also the set of all supplement (coclosed) subgroups of G.

Now let us recall the definition of a coclosure of a subgroup in a group.

Definition 8.6. Let $A \leq G$. A subgroup C of G such that $C \subseteq A$ is called a *coclosure* of A in G if C is a minimal coessential subgroup of A in G, or equivalently, C is a coessential subgroup of A in G and C is coclosed in G.

For a finite group we have the following result.

Theorem 8.7. *Every subgroup of G has a coclosure in G.*

One can easily write a function `Coclosures(G,A)` to determine all the coclosures in G of a subgroup A of G, dual in some sense to `Closures(G,A)`.

9. Extending groups and lifting groups

Now let us make a few considerations on extending and lifting abelian groups.

Definition 9.1. G is called *extending* if every subgroup of G is essential in a direct summand of G.

Proposition 9.2. *G is extending if and only if every complement subgroup of G is a direct summand of G.*

Definition 9.3. G is called *lifting* if every proper subgroup A of G contains a direct summand D of G such that $A/D << G/D$ (i.e. D is a coessential subgroup of A in G).

Proposition 9.4. [2, 22.3] *G is lifting if and only if it is* amply supplemented *(i.e. for every subgroups A, B of G with $A + B = G$ there is a supplement C of A with $C \subseteq B$) and every supplement subgroup is a direct summand.*

Note that every finite abelian group is clearly amply supplemented, so that we have the following consequence.

Corollary 9.5. *A finite group G is lifting if and only if every supplement subgroup of G is a direct summand.*

The structures of extending and lifting abelian groups are well-known. Having in mind also Theorem 8.5, it should not be surprising to get the following consequence, which shows, together with their structure, that for a finite abelian group the extending and lifting properties coincide.

Theorem 9.6. [8, p.19 and p.98] *A finite abelian group G is extending if and only if it is lifting if and only if each p-component of G is isomorphic to a direct sum $(\bigoplus_I \mathbb{Z}_{p^n}) \bigoplus (\bigoplus_J \mathbb{Z}_{p^{n+1}})$, where p is a prime, $n = n(p)$ is a natural number and the cardinals I, J may be zero.*

Even if the structure of extending and lifting abelian groups is known, one can also easily check with a function `IsExtending(G)` if G is extending (and also lifting) by comparing the results of the functions `ClosedSubgroups(G)` and `DirectSummands(G)`.

10. Implementation

All the algorithms presented or just mentioned here have been implemented using the programming language of GAP [3]. Here it is the list of implemented functions, where G is a finite abelian group and A, B are subgroups of G:

- `IsDirectSummand(G)`
- `DirectSummands(G)`
- `NullIntersectors(G,A)` / `FullSummands(G,A)`
- `IsEssential(G,A)` / `IsSuperfluous(G)`
- `EssentialSubgroups(G)` / `SuperfluousSubgroups(G)`
- `IsUniform(G)`
- `IsCoessential(G,A,B)`
- `CoessentialSubgroups(G,A)`
- `IsClosed(G,A)`
- `Complements(G,A)` / `Supplements(G,A)`
- `ClosedSubgroups(G)`
- `Closures(G,A)` / `Coclosures(G,A)`
- `IsExtending(G)`

11. An example

Let us consider the abelian group $\mathbb{Z}_4 \oplus \mathbb{Z}_2 \oplus \mathbb{Z}_2$, that is, the abelian group G with the presentation

$$G = \langle a, b, c \mid 4a = 0, 2b = 0, 2c = 0 \rangle.$$

It has 27 subgroups, which are the following (listed in the order given by GAP):

$H_1 = 0$	$H_{10} = \langle 2a, b \rangle$	$H_{19} = \langle c + b, 2a + b \rangle$
$H_2 = \langle b \rangle$	$H_{11} = \langle b, 2a + c \rangle$	$H_{20} = \langle 2a, b, c \rangle$
$H_3 = \langle c \rangle$	$H_{12} = \langle 2a, b + 3a \rangle$	$H_{21} = \langle a, b \rangle$
$H_4 = \langle 2a \rangle$	$H_{13} = \langle c, 2a + b \rangle$	$H_{22} = \langle 2a, b, c + 3a \rangle$
$H_5 = \langle b + c \rangle$	$H_{14} = \langle a \rangle$	$H_{23} = \langle a, c \rangle$
$H_6 = \langle 2a + b \rangle$	$H_{15} = \langle 2a, b + 3a \rangle$	$H_{24} = \langle 2a, b + 3a, c \rangle$
$H_7 = \langle 2a + c \rangle$	$H_{16} = \langle 2a, c + 3a \rangle$	$H_{25} = \langle a, c + b \rangle$
$H_8 = \langle 2a + b + c \rangle$	$H_{17} = \langle 2a, c + b \rangle$	$H_{26} = \langle 2a, b + 3a, c + 3a \rangle$
$H_9 = \langle b, c \rangle$	$H_{18} = \langle 2a, a + b + c \rangle$	$H_{27} = G$

Instead of listing here the respective GAP commands and their output, we consider only the associated abstract mathematical objects. A detailed description of the GAP functions is given in [3]. Among our implemented functions, we consider only those which produce a list of subgroups with one of the previously mentioned properties. The results given by the other functions, which decide whether a given subgroup has a certain property, are basically included in these ones.

Let us start with functions having the group G as their only input parameter. The function `DirectSummands(G)` produces a list of 57 pairs of direct summands of G. For instance, we obtain the following pairs of direct summands involving the subgroup $H_9 = \langle b, c\rangle$: (H_9, H_{14}), (H_9, H_{15}), (H_9, H_{16}), (H_9, H_{18}). Also, one can see that G has 22 direct summands, namely H_1, H_2, H_3, H_5, H_6, H_7, H_8, H_9, H_{11}, H_{13}, H_{14}, H_{15}, H_{16}, H_{18}, H_{19}, H_{21}, H_{22}, H_{23}, H_{24}, H_{25}, H_{26}, H_{27}. Using the functions `EssentialSubgroups(G)` and `SuperfluousSubgroups(G)`, one can find out the essential subgroups of G, namely H_{20} and H_{27}, and the superfluous ones, namely H_1 and H_4. The list of all closed (hence also coclosed) subgroups of G is given by the function `ClosedSubgroups(G)` and is the same as the list of direct summands of G.

We can immediately check using the corresponding functions that our group is not uniform (hence also not hollow), but it is extending (hence also lifting). If someone does not know the structure of extending abelian groups (i.e., our previous Theorem 9.6), but wants to state a conjecture concerning the structure of these abelian groups, the function `IsExtending(G)` can be used for some (or even all) abelian groups of reasonable large orders (using also the GAP library of small groups, see [10]), in order to help him to find the correct statement.

Now take the subgroup $H_{10} = \langle 2a, b\rangle$. The subgroups of G having zero intersection with H_{10}, respectively having sum G with H_{10} are given by the functions `NullIntersectors(G,H10)` and `FullSummands(G,H10)`. These subgroups are H_1, H_3, H_5, H_7 and H_8, respectively H_{23}, H_{24}, H_{25}, H_{26} and H_{27}. Using the corresponding functions, we obtain that the coessential subgroups of H_{10} in G are H_2, H_6 and H_{10}, the complements of H_{10} in G are H_3, H_5, H_7 and H_8, the supplements of H_{10} in G are H_{23}, H_{24}, H_{25} and H_{26}, the closures of H_{10} in G are H_{21} and H_{22}, and the coclosures of H_{10} in G are H_2 and H_6.

Finally, regarding computer speed, let us note that the results of our functions are obtained fast, practically instantaneous for a group as $G = \mathbb{Z}_4 \oplus \mathbb{Z}_2 \oplus \mathbb{Z}_2$. Moreover, the subgroups determined by them may be visualized in the lattice of subgroups by using the GAP package XGAP.

Acknowledgements. The authors would like to thank the referee for the comments and suggestions, which improved the presentation of the paper.

References

[1] G. Călugăreanu, S. Breaz, C. Modoi, C. Pelea and D. Vălcan, *Exercises in Abelian group theory*, Kluwer Texts in the Mathematical Sciences, 25, Dordrecht, Kluwer, 2003.

[2] J. Clark, C. Lomp, N. Vanaja and R. Wisbauer, *Lifting modules. Supplements and projectivity in module theory*. Frontiers in Mathematics, Birkhäuser, Basel, 2006.

[3] S. Crivei, G. Olteanu and Ş. Şuteu Szöllősi, `ELISA`. A collection of GAP algorithms related to extending and lifting abelian groups. (`http://www.gap-system.org/Packages/undep.html`) (`http://math.ubbcluj.ro/~crivei/GAP_project`).

[4] N.V. Dung, D.V. Huynh, P.F. Smith and R. Wisbauer, *Extending modules*, Pitman Research Notes in Mathematics Series, 313, Longman Scientific and Technical, 1994.

[5] A. Harmancı, D. Keskin and P.F. Smith, On $\oplus$-supplemented modules, Acta Math. Hungar., 83 (1999), 161–169.

[6] D. Keskin and W. Xue, Generalizations of lifting modules, Acta Math. Hungar., 91 (2001), 253–261.

[7] E. Mermut, *Homological approach to complements and supplements*, Ph.D. thesis, Dokuz Eylül University, Izmir, 2004.

[8] S.H. Mohamed and B.J. Müller, *Continuous and discrete modules*, London Math. Soc. Lecture Notes Series, 147, Cambridge Univ. Press, Cambridge, 1990.

[9] P.F. Smith and A. Tercan, Generalizations of CS modules, Comm. Algebra, 21 (1993), 1809–1847.

[10] The GAP Group, *GAP – Groups, Algorithms, and Programming, Version 4.4.7*; 2006, (`http://www.gap-system.org`).

Septimiu Crivei * **and Ştefan Şuteu Szöllősi** **
Faculty of Mathematics and Computer Science,
Babeş-Bolyai University,
Str. M. Kogălniceanu 1, 400084 Cluj-Napoca, Romania
E-mails: * crivei@math.ubbcluj.ro, ** szollosi@gmail.com