Groups whose word problem is a Petri net language

Gabriela Aslı Rino Nesin and Richard M. Thomas

Department of Computer Science, University of Leicester, Leicester LE1 7RH, U.K.

Abstract. There has been considerable interest in exploring the connections between the word problem of a finitely generated group as a formal language and the algebraic structure of the group. However there are few complete characterizations which tell us precisely which groups have their word problem in a specified class of languages. We investigate which finitely generated groups have their word problem a language accepted by a Petri net and give a complete classification, showing that a group has such a word problem if and only if it is virtually abelian.

Key words: Finitely generated group, word problem, terminal Petri net language.

1 Introduction

There has been considerable interest in exploring the connections between the word problem of a finitely generated group as a formal language and the algebraic structure of the group. Whilst the seminal work of Boone and Novikov in the 1950's showed that a finitely presented group could have a word problem that is not recursive, it was not really until the 1970's that languages lower down the Chomsky hierarchy were investigated. Anisimov showed in 1971 that a group has a regular word problem if and only if it is finite [1]. Whilst this result is not difficult to prove, asking such a question was an innovative idea and naturally led to an investigation as to what happens with other classes of languages.

Muller and Schupp showed in [16] (modulo a subsequent result of Dunwoody [4]) that a group has a context-free word problem if and only if it is virtually free. Indeed, the word problem of such a group must be deterministic context-free, and even an NTS language [2]. Apart from Dunwoody's result, this characterization uses other deep group theoretical results (such as Stallings' classification of groups with more than one end).

Within the class of context-free languages there are essentially not many other possibilities if we assume certain natural conditions on the class of languages. Herbst [7] showed that, if $\mathcal{F}$ is a cone (i.e. a class of languages closed under homomorphism, inverse homomorphism and intersection with regular languages) which is a subset of the context-free languages, then the class of groups whose word problem lies in $\mathcal{F}$ is either the class of groups with a regular word problem, the class of groups with a one-counter word problem or the class of groups with a context-free word problem. (The one-counter languages are those languages accepted by a pushdown automaton where we have a single stack symbol apart from a bottom marker.) Herbst also classified the groups with a one-counter word problem as being the virtually cyclic groups. This work was extended in [9] where it was shown that a group has a word problem which is a finite intersection of one-counter languages if and only if it is virtually abelian.

Whilst other classes of languages have been investigated there are very few complete characterizations. We investigate groups whose word problem is a terminal Petri net language and establish the following:

Theorem 1. *A finitely generated group G has word problem a Petri net language if and only if G is virtually abelian.*

Whilst this gives a correspondence between an important family of languages and a natural class of groups, there are many variations on Petri net languages which could potentially give rise to different classes of groups. Many of these modifications are so powerful that the class of languages is found to be equal to the class of recursively enumerable languages but there are other interesting possibilities, such as the class obtained by allowing λ-transitions in the Petri net, and it would be interesting to investigate the corresponding classes of groups.

The structure of this paper is as follows. We recall some basic definitions and facts about Petri nets and group theory in Sections 2 and 4 respectively. In Section 3 we comment on the equivalence of various definitions for Petri net languages. Given this background material, showing that a finitely generated virtually abelian group has a word problem which is a Petri net language is fairly straightforward, and we do this in Section 5. The proof of the converse is rather more involved and we provide that in Section 6. We finish in Section 7 by commenting how this class of groups relates to certain other classes which have arisen in considering word problems.

2 Petri nets

In this section we set out our conventions and notation for Petri nets and recall some properties of the class of languages they accept.

A *labelled Petri net* is a tuple $P = (S, T, W, m_0, \Sigma, l)$ where:

(i) S is a finite set, called the set of *places*; we will assume that an order is imposed on S and so it will be displayed as a tuple.
(ii) T is a finite set disjoint from S, called the set of *transitions*.
(iii) $W : (S \times T) \cup (T \times S) \to \mathbb{N}$ is the *weight function*, assigning a multiplicity to pairs of places and transitions. If $W(x, y) = n$ then we will write $x \xrightarrow{n} y$. If $W(x, y) = 0$ then we say there is no arrow from x to y.
(iv) $m_0 \in \mathbb{N}^S$ assigns to each place a natural number and is called the *initial marking*.
(v) Σ is a finite set called the *alphabet* and the *labelling function* $l : T \to \Sigma$ assigns a label to each transition.

The function l can be extended to a function $T^* \to \Sigma^*$ in the natural way (where we define λl to be λ). Note that l does not have to be bijective (if it were we would have a "free Petri net") but we do assume that l is a (total) function.

As usual we represent a labelled Petri net by a labelled directed graph, where the places are represented by circles, transitions by rectangles (we will denote transitions by their labels for simplicity), the weight function by arrows and arrow multiplicities by numbers on the arrows (with no arrow drawn if the multiplicity is zero and no number if the multiplicity is one). Markings (i.e. elements of $\mathbb{N}^S$) are represented by tokens or natural numbers in each place.

Now we describe the execution semantics of Petri nets. Let $m \in \mathbb{N}^S$ be a marking and $t \in T$ be a transition. We say that t is *enabled* at m if, for all places $s \in S$, we have $W(s,t) \leqslant m(s)$; we denote this by $m[t\rangle$. If t is enabled at m, we can fire t to get a new marking $m' \in \mathbb{N}^S$, defined by

$$m'(s) = m(s) + W(t,s) - W(s,t)$$

for all $s \in S$, and we write $m[t\rangle m'$ noting that asserting this automatically implies that $m[t\rangle$ must hold. We generalize this to sequences w of transitions (i.e. to elements w of T^*) and define $m[w\rangle m'$ in the obvious way.

We will need the notion of a labelled Petri net accepting a language; there are various possibilities and we consider the "terminal language" of a Petri net. To do this we extend the definition of a labelled Petri net $P = (S, T, W, m_0, \Sigma, l)$ to include a finite set of *terminal markings* $M \subset \mathbb{N}^S$ and write

$$P = (S, T, W, m_0, M, \Sigma, l)$$

The (*terminal*) *language* $L(P)$ recognized by P is the set

$$\{l(w) : m_0[w\rangle m \text{ some } m \in M, w \in T^*\}$$

We say that a language $L \subseteq \Sigma^*$ is a *Petri net language* (PNL for short) if there is a labelled Petri net whose terminal language is L and let $\mathcal{PNL}$ denote the class of all Petri net languages.

The class $\mathcal{PNL}$ has several nice closure properties (see references such as [12]), some of which we note here for future reference:

Proposition 2. *(i) $\mathcal{PNL}$ contains all regular languages.*
(ii) $\mathcal{PNL}$ is closed under union.
(iii) $\mathcal{PNL}$ is closed under intersection.
(iv) $\mathcal{PNL}$ is closed under inverse GSM mappings (and, in particular, under inverse homomorphisms).

Note that some authors define Petri net languages in a slightly different way: for example, they only allow one final marking, or disallow a final marking equal to the initial one. Several clever constructions (see pages 8-21 in [6]) show that these definitions are equivalent, up to the inclusion of the empty word λ in the language. We will survey some of these approaches in the next section.

3 Equivalence of the various definitions

In this section we will give various different definitions of Petri net languages and note their equivalence. Although we are not sure that all of what we note here has been explicitly proved in previous papers it does appear that these equivalences are all already known.

Our definition of a PNL is the same as that given by Jantzen in [12]. We will keep our terminology of labelled Petri net and PNL as above. The following definition is used by Petersen to define CSS (computation sequence sets) in [18]:

Definition 3. *A P-Petri net N is a 5-tuple (P, T, Σ, S, F) where P is a finite set of places, T is a finite set of transitions disjoint from P, Σ is the input alphabet (or the set of labels), $S \in P$ is a designated start place, $F \subseteq P$ is a designated set of final places, and each transition $t \in T$ is a triple consisting of a label in Σ, a multiset (bag) I of input places, and a bag O of output places.*

This is almost the same as our definition except for the designated start and final places. The labelling implies that there can be more than one transition with the same label, and the multiplicity of a place in a bag is just the multiplicity of its arrow to or from the transition in our original definition. Enabled transitions and so on are defined in the same way. We then have:

Definition 4. *The* Computation Sequence Set *of a P-Petri net is the set of all sequences of labels of transitions leading from the start marking (one token in the start place, none anywhere else) to one of the final markings (one token in one of the final places, none in any other place).*

Let CSS denote the class of languages which are computation sequence sets of a P-Petri net.

Hack's definition of a labelled Petri net in [6] is the same as the one given here (he actually splits the weight function into two separate forwards and backwards incidence functions, but this is not an essential difference). He then has:

Definition 5. *The set of H-terminal label sequences of a labelled Petri net N for a final marking $m_f \neq m_0$ is the set labels of sequences of transitions leading from m_0 to m_f.*

Essentially, the difference between our definition and Hack's is twofold: he only allows one final marking, and this final marking cannot be equal to the start marking. His motivation is that one then avoids having any H-terminal languages containing λ, as if one keeps the unique final marking condition but allows these languages to contain λ, then the class of H-terminal languages of labelled Petri nets would no longer be closed under union (see page 8 in [6]). Hack calls this class $\mathfrak{L}_0$ and we shall adopt this terminology.

It is known (see pages 19-20 of [6]) that $\mathfrak{L}_0$ and CSS are the same up to inclusion of the empty word:

Theorem 6. *For any language L, we have that $L \in CSS \iff L - \{\lambda\} \in \mathfrak{L}_0$*

We also have that

Theorem 7. $PNL = CSS$

It is clear that $CSS \subseteq PNL$ and the reverse inclusion is the interesting one. To show it, one can use a "standardisation" of the Petri nets described in [6]. If we have a Petri net N then it can be transformed into a P-Petri net without changing the terminal language.

4 Group theory

In this section we review the background material we need from group theory and establish some general facts about groups whose word problem is a Petri net language. For general information about group theory we refer the reader to [13, 19].

Let A be a finite set and let A^{-1} be another set disjoint from, but in a one-to-one correspondence with, A; we write a^{-1} for the element in A^{-1} corresponding to the element a in A. Let $\Sigma = A \cup A^{-1}$. We say that A is a *generating set* for a group G if we have a monoid homomorphism φ from Σ^* onto G such that $(a\varphi)^{-1} = a^{-1}\varphi$ for all $a \in A$; we normally then identify an element $x \in \Sigma$ with the image $x\varphi \in G$, so that A becomes a subset of G. A group with such a finite generating set is said to be *finitely generated*. The groups considered in this paper will all be finitely generated.

With this convention we define the *word problem* $W_A(G)$ of G with respect to the generating set A to be $\{\alpha \in \Sigma^* : \alpha =_G 1_G\}$ where the notation $\alpha =_G \beta$ (where $\alpha, \beta \in \Sigma^*$) denotes the fact that α and β represent the same element of G (i.e. that $\alpha\varphi = \beta\varphi$) and $\alpha =_G g$ (where $\alpha \in \Sigma^*$ and $g \in G$) denotes that fact that α represents the element g of G (i.e. that $\alpha\varphi = g$).

With this definition the word problem $W_A(G)$ is a subset of Σ^* and hence is a language; so we can consider which groups have their word problem in a given class of languages. This would seem to depend on the choice of A but, under certain mild assumptions of $\mathcal{F}$, this does not matter (see [8] for example):

Proposition 8. *If a class of languages $\mathcal{F}$ is closed under inverse homomorphism and the word problem of a group G with respect to some finite generating set lies in $\mathcal{F}$ then the word problem of G with respect to any finite generating set lies in $\mathcal{F}$.*

Given Proposition 2 (iv), we may talk about the word problem of a finitely generated group G being a PNL without reference to the choice of generating set. If $\mathcal{F}$ is any class of languages closed under inverse homomorphism then we will (mildly) abuse notation and write $G \in \mathcal{F}$ if the word problem of G lies in $\mathcal{F}$.

As the class $\mathcal{PNL}$ is closed under inverse homomorphisms and intersection with regular languages (the latter fact following from parts (i) and (iii) of Proposition 2), we have the following immediate consequence of Lemma 2 of [10]:

Proposition 9. *The class of finitely generated groups with word problem a PNL is closed under taking finitely generated subgroups.*

We also have the following:

Proposition 10. *If G and H are finitely generated groups with word problems in $\mathcal{PNL}$ then the word problem of the direct product $G \times H$ is also in $\mathcal{PNL}$.*

Proof. If P_1 and P_2 are Petri nets recognising the word problems of G and H with respect to finite generating sets A and B respectively (where $A \cap B = \emptyset$) then the disjoint union of P_1 and P_2 recognizes the word problem of $G \times H$. □

Of fundamental importance in what follows will be the so-called *Heisenberg group*, which is the group of matrices

$$\left\{ \begin{pmatrix} 1 & a & c \\ 0 & 1 & b \\ 0 & 0 & 1 \end{pmatrix} : a, b, c \in \mathbb{Z} \right\}$$

under multiplication. This is an example of a "nilpotent group". One way of defining this concept is to let $Z(G)$ denote the *centre* of a group G (i.e. the set of elements in G that commute with all the elements of G) and then define a series of normal subgroups $Z_1(G) \leqslant Z_2(G) \leqslant \ldots$ of G by:

$$Z_1(G) := Z(G), \quad Z_{i+1}(G)/Z_i(G) := Z(G/Z_i(G)) \text{ for } i \geqslant 1.$$

We say that G is *nilpotent* if $Z_i(G) = G$ for some $i \in \mathbb{N}$.

A generalization of this is to say that a group G is *virtually nilpotent* if G has a nilpotent subgroup H of finite index in G (where the *index* of a subgroup H is the number of distinct right cosets of the form Hg for $g \in G$). In general, if $\wp$ is any property of groups, then we say that G is *virtually* $\wp$ if G has a subgroup of finite index with the property $\wp$. It is a standard result that, if H has finite index in G, then H is finitely generated if and only if G is finitely generated. The following fact (see [10] for example) will be important here:

Proposition 11. *A finitely generated torsion-free virtually nilpotent group that does not contain the Heisenberg group is virtually abelian.*

The term "torsion-free" means that the group does not contain any non-trivial elements of finite order.

The notion of finite index will particularly relevant in this paper. Given that $\mathcal{PNL}$ is closed under union with regular sets and inverse GSM mappings by Proposition 2, we have the following immediate consequence of Lemma 5 in [10]:

Proposition 12. *If H is a finitely generated group with word problem in $\mathcal{PNL}$ and G is a group containing H as a finite index subgroup, then the word problem of G is also in $\mathcal{PNL}$.*

Returning to generating sets, we say that a group G with finite generating set A has *polynomial growth* if there is a polynomial $p(x)$ such that the number of distinct elements of G represented by words in $(A \cup A^{-1})^*$ of length at most n is bounded above by $p(n)$.

5 Virtually abelian implies PNL word problem

In this section we prove one direction of Theorem 1, showing that a finitely generated virtually abelian group G has its word problem in $\mathcal{PNL}$. We start with the case where G is abelian:

Proposition 13. *The word problem of a finitely generated abelian group is always a PNL.*

Proof. Let G be a finitely generated abelian group. According to the structure theorem for finitely generated abelian groups, G is expressible as a direct product

$$\mathbb{Z}^r \times \mathbb{Z}/a_1\mathbb{Z} \ldots \times \mathbb{Z}/a_m\mathbb{Z}$$

where $r \geqslant 0$, $m \geqslant 0$ and $a_i = p_i^{n_i}$ for some prime p_i and some natural number $n_i \geqslant 1$. As noted in the introduction, the word problem of a finite group such as $\mathbb{Z}/a\mathbb{Z}$ is regular, and hence a PNL. The word problem of $\mathbb{Z}$ with respect to some generating set $\{a\}$ is a PNL as shown in Figure 1.

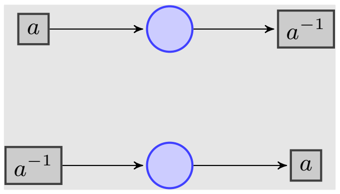

Fig. 1. A labelled Petri net recognizing the word problem of $\mathbb{Z}$. The empty marking is both initial and terminal, and there are no other terminal markings.

The result now follows from Proposition 10. □

Propositions 12 and 13 immediately give:

Corollary 14. *Any finitely generated virtually abelian group has word problem in $\mathcal{PNL}$.*

6 PNL word problem implies virtually abelian

Now we consider the converse to Corollary 14 which (together with Corollary 14) will establish Theorem 1. First we prove the following:

Proposition 15. *A finitely generated group with PNL word problem has polynomial growth.*

Proof. Let G be a group generated by by a finite set A, let $\Sigma = A \cup A^{-1}$, and assume that the word problem $W_A(G)$ of G is recognized by a Petri net $P = (S, T, W, m_0, M, \Sigma, l)$ with initial marking m_0 and set of terminal markings M.

We call markings which are reachable from m_0 in P and which allow the possibility of reaching a terminal marking *acceptable* markings. Note that, given an acceptable marking m, any two sequences of transitions reaching m from m_0 must represent the same element of G. This is because, if $m_0[t_1 \dots t_n\rangle m$ and $m_0[t'_1 \dots t'_k\rangle m$ and if m is acceptable, then there is a sequence of transitions w from m to some terminal marking m'. But then

$$m_0[t_1 \dots t_n\rangle m[w\rangle m' \text{ and } m_0[t'_1 \dots t'_k\rangle m[w\rangle m',$$

and hence both sequences of transitions label elements of $W_A(G)$, i.e.

$$(t_1 \dots t_n w)l =_G 1_G =_G (t'_1 \dots t'_k w)l,$$

from which we get that $(t_1 \dots t_n)l =_G (t'_1 \dots t'_k)l$.

So we have a natural mapping θ from the set of acceptable markings to G. As P recognizes the word problem of G, for each group element g there must be an acceptable marking m with $m\theta = g$, otherwise no word ww^{-1}, where w represents g, can be accepted by P. So the mapping θ is surjective.

In order to show polynomial growth, we want to show that there is a polynomial $p(n)$ such that the number of elements of G represented by a sequence of generators of length n is at most $p(n)$. Since the mapping θ is surjective it is therefore sufficient to bound the number of acceptable markings reachable by a sequence of transitions of length n by such a polynomial $p(n)$.

If a sequence $t_1 \dots t_n$ reaches an acceptable marking and if $t_{\sigma(1)} \dots t_{\sigma(n)}$ does as well for some permutation σ of $\{1, 2, \dots, n\}$, then the two sequences reach the same marking[1]; this follows directly from the effect on a marking of firing a transition. In counting the number of acceptable markings, one can therefore ignore the order in which the transitions fire: the only important thing is their multiplicities. If $T = \{u_1, u_2, \dots, u_k\}$ then there are at most as many acceptable markings induced by sequences of n transitions as there are possible values for $\mu(u_1), \dots, \mu(u_k) \in \mathbb{N}$ such that $\mu(u_1) + \dots + \mu(u_k) = n$, where $\mu(u_i)$ denotes the multiplicity of u_i in the transition sequence. It is now clear that the number of acceptable markings is bounded above by the polynomial $(n+1)^k$, as there are at most $n+1$ choices for each of the $\mu(t_i)$. □

Using Gromov's wonderful theorem [5] about groups with polynomial growth we immediately deduce the following:

Corollary 16. *A finitely generated group whose word problem is a PNL is virtually nilpotent.*

We now want to show that a finitely generated group whose word problem is a PNL is virtually abelian. As we will show later, it is enough to show that the Heisenberg group's word problem is not a PNL. To show this, we use Lambert's Pumping Lemma, a consequence of the decidability of the reachability problem for Petri nets. We state here a corollary to it (see Theorem 5.1 in [14]):

[1] Recall here that the t_i are actual transitions, not labels of transitions (i.e. generators); therefore this argument does not imply that G is abelian as, for example, being able to swap labels a and b in one such sequence does not mean that we would necessarily be able to do so in all such sequences.

Theorem 17. *Let $P = (S, T, W, m_0, \Sigma, l)$ be a labelled Petri net and m_f a final marking. Let $a \in \Sigma$. Defining*

$$\mathcal{L}(a) := \{|l(u)|_a : m_0[u\rangle m_f\}$$

we have that $\mathcal{L}(a)$ is infinite if and only if it contains an arithmetic sequence with a non-zero ratio.

Note that the use of only one final marking does not pose a problem, because of Theorems 6 and 7. Our result will follow from Theorem 18 below.

Theorem 18. *Let $\Sigma = \{a, b, A, B, C\}$. Then $L = \{a^i b^j A^i B^j C^{ij} : i, j \in \mathbb{N}\}$ is not a Petri net language.*

Proof. Assume that L is a Petri net language. If so, then we can intersect it with the language $K = \{a^n b^n A^n B^n C^k : n, k \in \mathbb{N}\}$ to get $\{a^n b^n A^n B^n C^{n^2} : n \in \mathbb{N}\}$. K is a Petri net language, recognisable by the Petri net

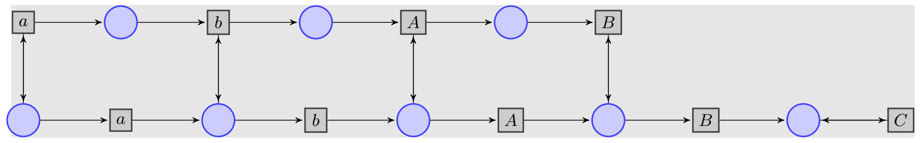

Fig. 2. A labelled Petri net recognizing K. The initial marking is a token in the bottom left place, and the terminal marking is a token in the bottom right place.

Since Petri nets are closed under intersection, we have that $L \cap K \in \mathcal{PNL}$ as well. By Theorem 17, $\mathcal{L}(C) = \{n^2 : n \in N\}$ would then contain an arithmetical sequence, a contradiction. □

We can now deduce the required result about the Heisenberg group:

Corollary 19. *The word problem of the Heisenberg group H is not a PNL.*

Proof. If a, b and c respectively denote the matrices

$$\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \text{ and } \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

then a, b and c generate H and every relation in H can be deduced from the relations

$$ac = ca, \; bc = cb \text{ and } a^{-1}b^{-1}ab = c;$$

see [13] for example. Let W denote the word problem of H with respect to $\{a, b, c\}$. To ease clutter, we let A represent a^{-1}, B represent b^{-1} and C represent c^{-1}. We claim that the language L from Theorem 18 is just $W \cap a^* b^* A^* B^* C^*$. To see this, we note that $ab =_G bac$. So we get

$$\begin{aligned} a^i b^j &=_G a^{i-1}abb^{j-1} =_G a^{i-1}bacb^{j-1} =_G a^{i-1}bab^{j-1}c =_G a^{i-1}babb^{j-2}c \\ &=_G a^{i-1}b^2ab^{j-2}c^2 =_G \dots =_G a^{i-1}b^j ac^j =_G \dots \\ &=_G b^j a^i c^{ij}. \end{aligned}$$

Now:

$$\begin{aligned} a^i b^j A^k B^l C^m =_G 1 &\iff b^j a^i c^{ij} A^k B^l C^m =_G 1 \iff b^j a^i A^k B^l c^{ij} C^m =_G 1 \\ &\iff i = k, j = l \text{ and } ij = m \end{aligned}$$

which is what we wanted to establish.

Since the class $\mathcal{PNL}$ is closed under intersection with regular languages, we have that $W \notin \mathcal{PNL}$. □

Our result now follows:

Proposition 20. *If a finitely generated group G has a PNL word problem, then G is virtually abelian.*

Proof. We know already that G is virtually nilpotent by Corollary 16. Assume that G is not virtually abelian; then G has a nilpotent but not virtually abelian subgroup K of finite index in G. In turn, it is known (see 5.4.15 (i) of [19] for example) that K must have a torsion-free subgroup L of finite index, and L must then be nilpotent but not virtually abelian. By Proposition 11 we have that $H \leqslant L \leqslant K \leqslant G$ where H is the Heisenberg group. So H is a finitely generated subgroup of G. Since G has a PNL word problem, so does H by Proposition 9, contradicting Corollary 19. □

Taken together with Corollary 14, this completes the proof of Theorem 1.

7 Relation to other classes of languages

In this section, we put our results into some context, comparing the class of groups with word problem in $\mathcal{PNL}$ with those in some other classes of languages. We let $\mathcal{OC}$ denote the class of one-counter languages, $\mathcal{CF}$ the class of context-free languages and $co\mathcal{CF}$ the class of co-context-free languages (i.e. languages that are complements of context-free languages).

We mentioned in the introduction that the groups whose word problem is a one-counter or context-free language have been classified. These families of languages are both incomparable with $\mathcal{PNL}$. Let $D_1^{'*}$ denote the one-sided Dyck language on one set of parentheses, which is both a one-counter language and a PNL. (In fact, $\mathcal{OC}$ is the smallest full AFL containing $D_1^{'*}$ and $\mathcal{PNL}$ is the smallest intersection-closed semi-AFL containing $D_1^{'*}$. See the remark after Proposition 1 in [3] for the first fact and Theorem 1 in [11] for the second). Define

$$L := (D_1^{'*} a)^* D_1^{'*}$$

where a is an arbitrary extra letter. Since $\mathcal{OC}$ is closed under concatenation and Kleene star, we have that $L \in \mathcal{OC}$. However, Hack shows in Theorem 9.8 of [6] that $L \notin \mathcal{PNL}$ (in fact, not even if one allows the Petri net to have empty or "invisible" transitions). So $\mathcal{OC}$ is not contained in $\mathcal{PNL}$. However, when we turn to word problems of groups, the situation changes:

Proposition 21. *If $G \in \mathcal{OC}$ then $G \in \mathcal{PNL}$.*

Proof. A group with one-counter word problem is virtually cyclic by [7] and hence virtually abelian; the result follows from Corollary 14. □

Of course, since $\mathcal{OC}$ is not contained in $\mathcal{PNL}$, finite intersections of one-counter languages are not necessarily in $\mathcal{PNL}$. However, this situation also changes when we restrict ourselves to word problems.

As we mentioned in the introduction, it was shown in [9] that a group has a word problem that is the intersection of finitely many one-counter languages if and only if it is virtually abelian. Therefore we immediately have:

Corollary 22. *$G \in \mathcal{PNL}$ if and only if $G \in \bigcap_{fin} \mathcal{OC}$.*

We mention in passing that, not only is $\bigcap_{fin} \mathcal{OC}$ not a subset of $\mathcal{PNL}$, but $\mathcal{PNL}$ is not a subset of $\bigcap_{fin} \mathcal{OC}$ either:

Proposition 23. *$L = \{a^n b^m : 1 \leqslant m \leqslant 2^n, 1 \leqslant n\}$ is in $\mathcal{PNL}$ but not $\bigcap_{fin} \mathcal{OC}$.*

Proof. It is known that $L \in \mathcal{PNL}$; see [11]. Assume that $L \in \bigcap_{fin} \mathcal{OC}$, say

$$L = L_1 \cap \ldots \cap L_n$$

where the L_i are one-counter languages. Let K be the regular language a^*b^*.

Since $L \subseteq K$, we have $L = (L_1 \cap K) \cap \ldots \cap (L_n \cap K)$ and so, without loss of generality, we can assume that $L_i \subseteq K$ for all i (as the intersection of a one-counter language and a regular language is one-counter). Since the Parikh mapping $\Phi : \Sigma^* \to \mathbb{N}^2$ defined by $w \mapsto (|w|_a, |w|_b)$ is bijective on K, we have

$$L\Phi = L_1\Phi \cap \ldots \cap L_n\Phi.$$

By Parikh's Theorem (see Theorem 2 in [17]) we know that any context-free language, and hence any one-counter language, has a semilinear Parikh image. Since the L_i are all one-counter, $L_i\Phi$ is semilinear for all i. Since any intersection of semilinear sets is semilinear, L would have a semilinear Parikh image. However L does not have a semilinear Parikh image [11], a contradiction. □

We finish with a comment relating groups with a word problem in $\mathcal{PNL}$ to those with a word problem in *co*$\mathcal{CF}$. The latter is a very interesting class of groups (see [10, 15] for example) but we do not yet have a classification as to which groups lie in this class. However, we can say the following:

Proposition 24. *Let G be a finitely generated group. If $G \in \mathcal{PNL}$ then $G \in$ co$\mathcal{CF}$. Furthermore, this inclusion is proper.*

Proof. By Proposition 6 in [10], all virtually abelian groups are in *co*$\mathcal{CF}$, and so the inclusion follows from Proposition 20.

To see the properness of the inclusion, consider the free group on two generators. This group is not virtually abelian, and so is clearly not in $\mathcal{PNL}$, but it is in *co*$\mathcal{CF}$ (see [10] for example). □

Acknowledgements

Some of the research for this paper was done whilst the authors were visiting the Nesin Mathematics Village in Turkey; the authors would like to thank the Village both for the financial support that enabled them to work there and for the wonderful research environment it provided that stimulated the results presented here. The second author also would like to thank Hilary Craig for all her help and encouragement.

References

1. Anisimov, A.V.: Group languages. Cybernetics and Systems Analysis **7** (1971) 594–601
2. Autebert, J.M., Boasson, L., Sénizergues, G.: Groups and NTS languages. Journal of Computer and System Sciences **35** (1987) 243–267
3. Boasson, L.: An iteration theorem for one-counter languages. In: Proceedings of the Third Annual ACM Symposium on Theory of Computing. STOC '71, New York, NY, USA, ACM (1971) 116–120
4. Dunwoody, M.J.: The accessibility of finitely presented groups. Inventiones Mathematicae **81** (1985) 449–457
5. Gromov, M.: Groups of polynomial growth and expanding maps. Publications Mathematiques de l'Institut des Hautes Etudes Scientifiques **53** (1981) 53–78
6. Hack, M.: Petri net languages. Computation Structures Group Memo 124, Project MAC, M.I.T. (1975)
7. Herbst, T.: On a subclass of context-free groups. RAIRO - Theoretical Informatics and Applications **25** (1991) 255–272
8. Herbst, T., Thomas, R.M.: Group presentations, formal languages and characterizations of one-counter groups. Theoretical Computer Science **112** (1993) 187–213
9. Holt, D.F., Owens, M.D., Thomas, R.M.: Groups and semigroups with a one-counter word problem. Journal of the Australian Mathematical Society **85** (2008) 197–209
10. Holt, D.F., Rees, S., Röver, C.E., Thomas, R.M.: Groups with context-free co-word problem. Journal of the London Mathematical Society **71** (2005) 643–657
11. Jantzen, M.: On the hierarchy of Petri net languages. RAIRO - Theoretical Informatics and Applications **13** (1979) 19–30
12. Jantzen, M.: Language theory of petri nets. In Brauer, W., Reisig, W., Rozenberg, G., eds.: Petri Nets: Central Models and Their Properties. Volume 254 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (1987) 397–412
13. Johnson, D.L.: Presentations of Groups. 2nd edn. Cambridge University Press (1997)
14. Lambert, J.: A structure to decide reachability in petri nets. Theoretical Computer Science **99**(1) (1992) 79 – 104
15. Lehnert, J., Schweitzer, P.: The co-word problem for the Higman-Thompson group is context-free. Bulletin of the London Mathematical Society **39** (2007) 235–241
16. Muller, D., Schupp, P.: Groups, the theory of ends, and context-free languages. Journal of Computer and System Sciences **26** (1983) 295–310
17. Parikh, R.J.: On context-free languages. Journal of the ACM **13** (1966) 570–581
18. Peterson, J.L.: Computation sequence sets. Journal of Computer and System Sciences **13**(1) (1976) 1 – 24
19. Robinson, D.: A Course in the Theory of Groups. 2nd edn. Springer (1995)

Appendix

We give here a proof of Theorem 7. As we commented in the paper, it is clear that $CSS \subseteq \mathcal{PNL}$ and the reverse inclusion is the interesting one. So we need to demonstrate that $\mathcal{PNL} \subseteq CSS$. To do this we describe the standardisation that will transform any labelled Petri net $N = (S, T, W, m_0, M.\Sigma, l)$ into a P-Petri net.

1. *The run place.* Add a place named *run* with a loop to all transitions in the original net N. This clearly does not change the language. The practicality of this place is that it enables one to activate and deactivate N at will; no transitions in N can fire unless there is a token in *run*.
2. *The first transitions.* (See page 12, section 2.2.2 in [6]) There were finitely many transitions $t_1, \ldots, t_n$ enabled at the initial marking m_0 of our labelled Petri net N. For each of these t_i we add a new transition t_i' with the same label as t_i. This t_i' will do two things: deposit the marking corresponding to $m_0[t_i\rangle$ (thus imitating t_i) and deposit a token in *run*, thus activating N.
3. *The start place* Now add a place named *start* with an arrow to each of the t_i'. It is easy to see the language is not changed. We now have a designated start place and can take our new initial marking to be one token in the start place and none anywhere else.
4. *The stop transitions.* These use the same principle as the first transitions - for any final marking of our original Petri net N, there are finitely many last transitions t_i leading to them - in other words, there are finitely many t_i such that there is a marking m_j where $m_j[t_i\rangle m$ for $m \in M$ a final marking. Again, add a new transition t_{ij}'' for each of these. Each t_{ij}'' will both take away the token in the *run* place and empty the penultimate marking it is associated to (so if m_j had k tokens in place p, there will be an arrow labeled k from p to t_{ij}''). Note that this only works if the last transition is not the first transition (i.e. m_j is not m_0). In that special case, t_{ij}'' has just a simple arrow from the *start* place, bypassing *run* altogether.
5. *The final places.* For each t_{ij}'', add a new place p_{ij}, with a simple arrow from t_{ij}'' to p_{ij}. If the empty word is not in the language, taking these as the final places and assuming each final marking to be a token in a single final place and none anywhere else is enough. If λ is in the language recognized by N, simply designate the start place to be a final place as well.

The modifications above can be straightforwardly seen not to change the terminal language of the net, and allow us to transform a labelled Petri net (according to our definition) into one of Petersen's P-Petri nets.