Formal Languages, Word Problems of Groups and Decidability

Sam A.M. Jones and Richard M. Thomas

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

Abstract. This paper considers questions relating formal languages to word problems of groups with a particular emphasis on the decidability of some problems that arise. We investigate the decidability of certain natural conditions that characterize word problems for various classes of languages and we then turn our attention to the question of a language actually being a word problem. We show that this is decidable for the classes of regular and deterministic context-free languages but undecidable for the one-counter languages.

1 Introduction

The word problem of a finitely generated group G (i.e. the set of words representing the identity element of G) is a fundamental notion in group theory and there have been some intriguing connections between this and formal language theory. In particular, various classifications have been obtained as to which groups have their word problems lying in certain classes of languages (see Section 4). We focus on subfamilies of the context-free languages and, given a result of Herbst (Theorem 7) which says that, under certain closure assumptions, there are essentially only three cases, we concentrate on those particular families, namely the regular languages, the one-counter languages and the context-free languages.

There is also a simple necessary and sufficient criterion (see Theorem 9) for a language to be the word problem of a group. This involves the conjunction of two conditions (universal prefix closure and deletion closure - see Section 2 for definitions of these concepts) and we consider the question of the decidability of these two conditions in Sections 5 and 6 respectively. It is intriguing that we have a connection between word problems of groups and natural formal language conditions such as deletion closure as studied in [15].

Having established the decidability of these conditions for the classes of languages we are considering we turn our attention to the question of deciding their conjunction, i.e. that of deciding whether a given language is the word problem of a group. Whilst this is easily seen to be decidable for the regular languages we build on the work in [17] to show that this is undecidable for one-counter languages (and hence for context-free languages as well); see Theorem 28. However, we know that any context-free language that is the word problem of a group is deterministic context-free and we show that the problem of deciding whether a deterministic context-free language is the word problem of a group is actually decidable (see Theorem 31).

2 Background from Formal Language Theory

Throughout this paper we will be discussing regular and context-free languages accepted by finite automata and pushdown automata respectively. We will also be discussing *one-counter languages* which are those languages accepted by a *one-counter automaton*, i.e. a pushdown automaton where we have only a single stack symbol (apart from a symbol marking the bottom of the stack); these automata are nondeterministic and accept by final state. We will use some standard definitions and properties of classes of languages (such as their closure properties under certain operations and decidability results); see [2,11,14] for example.

In particular, it is well known (see [11] for example) that one cannot decide whether or not a context-free language $L \subseteq \Sigma^*$ is equal to Σ^* (the so called *universe problem*). In fact, this problem remains undecidable if one restricts oneself to the certain subsets of the context-free languages such as the one-counter languages [13]. We will need a slight strengthening of this fact where we restrict to the case where the alphabet has size 2:

Theorem 1. *The following decision problem is undecidable:*
Input: *a one-counter automaton M with input alphabet Ω of size* 2.
Output: *"yes" if $L(M) = \Omega^*$; "no" otherwise.*

Proof. We use a standard technique to show that, if we had an algorithm $\mathfrak{A}$ solving this decision problem, then we would have an algorithm solving the universe problem for alphabets of arbitrary size. So suppose that we have such an algorithm $\mathfrak{A}$. Let $\Sigma = \{x_1, x_2, \ldots, x_n\}$ be an arbitrary alphabet and let $\Omega = \{a, b\}$. Let $M = (Q, \Sigma, \Gamma, \tau, s, A)$ be a one-counter automaton and let $L = L(M)$. We want to determine whether or not $L(M) = \Sigma^*$.

Define $\varphi : \Sigma^* \to \Omega^*$ by $x_1 \mapsto ab$, $x_2 \mapsto a^2b$, $\ldots$, $x_n \mapsto a^n b$, and let $K = \Sigma^*\varphi$. Since K is regular, $R = \Omega^* - K$ is regular. Since φ is injective we have that $L = \Sigma^*$ if and only if $L\varphi = \Sigma^*\varphi = K$ which equivalent to saying that $L\varphi \cup R = K \cup R = \Omega^*$. Since $L\varphi \cup R$ is a one-counter language we may use algorithm $\mathfrak{A}$ to decide this problem, and so we could determine whether or not $L(M) = \Sigma^*$, a contradiction. □

Remark 2. In Theorem 1 all we have used about the family $\mathcal{F}$ of one-counter languages are the facts that the universe problem is undecidable for $\mathcal{F}$ and that $\mathcal{F}$ is closed under homomorphism and union with regular languages; so Theorem 1 applies to any such family of languages. □

In this paper we will also need the idea of the *prefix closure* of a language $L \subseteq \Sigma^*$ which is defined to be:

$$\text{prefix}(L) = \{\alpha \in \Sigma^* : \alpha\beta \in L \text{ for some } \beta \in \Sigma^*\}.$$

In the case where $\text{prefix}(L) = \Sigma^*$, we say that L has the *universal prefix closure property.*

We also say that a language $L \subseteq \Sigma^*$ is *deletion closed* if it satisfies the following condition:

$$\alpha, u, \beta \in \Sigma^*,\ \alpha u \beta \in L,\ u \in L \implies \alpha\beta \in L.$$

3 Background from Group Theory

If G is a group and Σ is a finite set of symbols such that there is a surjective (monoid) homomorphism $\varphi : \Sigma^* \to G$, then we say that Σ is a *generating set* for G. Note that Σ is a *monoid generating set* for G as opposed to a *group generating set*; in the latter case, we have a set of symbols X and then let $\Sigma = X \cup X^{-1}$ where X^{-1} is a set of symbols in a (1-1) correspondence with X (and where we insist that $x^{-1}\varphi = (x\varphi)^{-1}$). In either case the *word problem* of the group G is the set of all words in Σ^* that represent the identity element of G.

A *presentation* for a group G is an expression of the form $\langle A : R \rangle$ where A is a generating set for G and R is a set of relations of the form $\alpha = \beta$. If A is a monoid generating set for G and $R \subseteq A^* \times A^*$, we have a *monoid presentation* for G and, if A is a group generating set for G, $\Sigma = A \cup A^{-1}$ as above, and $R \subseteq \Sigma^* \times \Sigma^*$ we have a *group presentation* for G. In each case the set R must be a set of *defining relations* for G: if $\approx$ is the congruence generated by R (together with all pairs of the form $(x^{-1}x, \epsilon)$ or (xx^{-1}, ϵ) with $x \in X$ in the case of a group generating set, where ϵ denotes the empty word), then G is isomorphic to $\Sigma^*/\approx$, i.e. $\alpha \approx \beta$ if and only if $\alpha\varphi = \beta\varphi$. The free group on a set X has the (group) presentation $\langle X : \emptyset \rangle$.

If P is any property of groups, then we say that a group is *virtually P* if it has a subgroup of finite index with property P. We will need the following fact:

Theorem 3. *Given a finite group presentation $\wp = \langle X : R \rangle$ and the promise that the group G presented by $\wp$ is virtually free, triviality of G is decidable.*

Proof. We start two processes running. The first enumerates the consequences of R, terminating if all the pairs (x, ϵ) with $x \in X$ have been output; this terminates if G is trivial. The second enumerates all the subgroups of finite index (one can do this for any finitely presented group; see [16] for example) and terminates if it finds a proper subgroup (any non-trivial virtually free group must possess such a subgroup). Eventually one of these two processes must terminate. □

4 Characterizing Word Problems

When examining groups based on their word problem as a formal language it is quite common to classify groups based on what class of languages their word problem lies in. However, there is no guarantee that the word problem will lie in the same class $\mathcal{F}$ of languages for different generating sets. The following result (see [8]) shows that, under certain mild assumptions on $\mathcal{F}$, this is not a problem:

Theorem 4. *If a class of languages $\mathcal{F}$ is closed under inverse homomorphism and the word problem of a group G lies in $\mathcal{F}$ with respect to some finite generating set then the word problem of G will lie in $\mathcal{F}$ for all finite generating sets.*

Anisimov [1] classified the groups with a regular word problem:

Theorem 5. *A finitely generated group has a regular word problem if and only if it is a finite group.*

Further work was done by Muller and Schupp [18] which, along with a result of Dunwoody [4], characterised the groups with a context-free word problem:

Theorem 6. *A finitely generated group G has a context-free word problem if and only if it is a virtually free group.*

One might ask what other families of languages $\mathcal{F}$ contained in the context-free languages give rise to interesting classes of groups. Herbst [7] showed that, if $\mathcal{F}$ satisfies certain natural closure conditions, then there are not many possibilities:

Theorem 7. *If $\mathcal{F}$ is a subset of the context-free languages closed under homomorphism, inverse homomorphism and intersection with regular languages then the class of finitely generated groups whose word problem lies in $\mathcal{F}$ is 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.*

In the light of Theorems 5 and 6 it is natural to ask which groups have a one-counter word problem. Herbst characterised these groups in [7] (see also [10]):

Theorem 8. *A finitely generated group G has a one-counter word problem if and only if it is a virtually cyclic group.*

Given Theorem 7 it is natural to ask if one can decide if a language lying in one of these three families of languages is a word problem of a group and we answer this question in Section 7.

The following characterisation of word problems of groups was given in [19]:

Theorem 9. *A language L over an alphabet Σ is the word problem of a group with generating set Σ if and only if L satisfies the following two conditions:*
W1 for all $\alpha \in \Sigma^*$ there exists $\beta \in \Sigma^*$ such that $\alpha\beta \in L$;
W2 $\alpha u \beta \in L, u \in L \Rightarrow \alpha\beta \in L$.

Condition W1 says that the prefix closure prefix(L) of L is Σ^*, i.e. that L has the universal prefix closure property, whereas condition W2 says that the language L is deletion closed. It is natural to ask, for the families of languages in Theorem 7, whether one can decide whether a language in that family satisfies these conditions and we will answer these questions in Sections 5 and 6 respectively.

5 Universal Prefix Closure and Decidability

In this section we investigate the decidability of the question prefix$(L) = \Sigma^*$. It is clear this is decidable if L is specified by means of a finite automaton:

Proposition 10. *The following problem is decidable:*
Input: a finite automaton $N = (Q, \Sigma, \tau, s, A)$.
Output: "yes" if prefix$(L(N)) = \Sigma^*$; "no" otherwise.

For example, given a finite state automaton N we construct the minimal deterministic automaton M accepting $L(N)$. As M has no unreachable states, prefix$(L(N)) = \Sigma^*$ if and only if M has no fail states, which is clearly decidable.

Remark 11. We are only interested in decidability questions in this paper and not with computational complexity. In our one non-trivial result about decidability (see Theorem 31) we do not have an easily computable time complexity (see Remark 32). For more information about the complexity of problems related to that described in Proposition 10 see [20] for example. □

When we consider the corresponding problem for one-counter languages we need the idea of a *counter machine*. There are several ways of describing these machines and we give one possibility here.

A counter machine M (as distinct from a one-counter automaton) is a two-tape machine. The first tape is the input tape; it is read only and the head can only move to the right. The second tape is a stack: whenever we move left, M erases the symbol it moved away from. There is only one stack symbol, a say. Intuitively M can only store a natural number (so that we can think of M as having an input tape and a counter). As we will see, the stack is never empty.

More formally, a *counter machine* is a sextuple $M = (Q, \Sigma, a, \delta, q_0, q_f)$ where Q is a finite set of states containing two distinguished states, q_0, the start state, and q_f, the final state. The input alphabet Σ is a finite set of symbols such that $a \notin \Sigma$. A *configuration* of M is a word of the form qa^n where $q \in Q$ and $n > 0$ (where the current state is q and the current stack contents are a^n).

We take C to be $\{1, 2, 3, 5, 7, \frac{1}{2}, \frac{1}{3}, \frac{1}{5}, \frac{1}{7}\}$; there is no particular significance in our choice of 2, 3, 5 and 7, in that any four pair-wise coprime natural numbers would suffice. The transition relation δ is a function from $(Q - \{q_f\}) \times \Sigma \times C$ to $(Q - \{q_0\}) \times (Q - \{q_0\})$; the fact that δ is a function means that M is deterministic. M starts with just a on its stack (i.e. with the counter set to 1) and must set its counter to 1 again before entering q_f.

A move $(p, b, x, q, r) \in \delta$ is interpreted as follows. If M is in state p reading an input b and if the result of multiplying the current value n of the counter (i.e. we have a^n on the stack) by the value x is an integer, then we set the counter to xn and move to state q; if xn is not an integer then the counter remains set at n and M moves to state r. We write $pa^n \vdash qa^{xn}$ or $pa^n \vdash ra^n$ as appropriate.

Given a Turing Machine, one can effectively construct a counter machine accepting the same language (see [11] for example). We now turn to the computations of a counter machine:

Definition 12. *Let M be a counter machine. A* valid computation *of M is a word $C_0C_1 \dots C_n \in Q \cup \{a\}^*$ such that the C_i are configurations of M and*

$$C_0 = q_0a \vdash C_1 \vdash \dots \vdash C_{n-1} \vdash C_n = q_fa.$$

An invalid computation *is a word in $(Q \cup \{a\})^*$ which is not a valid computation.*

In any valid computation of M, any configuration qa^n will have $n = 2^b3^c5^d7^e$ for some $b, c, d, e \geqslant 1$. Multiplying by 2, 3, 5 or 7 increases b, c, d or e by 1 and multiplying by $\frac{1}{2}$, $\frac{1}{3}$, $\frac{1}{5}$ or $\frac{1}{7}$ (if possible) decreases b, c, d or e by 1; so we effectively have four counters each of which can be increased or decreased. The fact that we can only multiply by x if nx is an integer is effectively saying that we can test each counter individually for zero (for example, if $n = 2^b 3^c 5^d 7^e$ and we want to multiply by $\frac{1}{2}$, then we must have that $b > 0$).

Our aim is to show that the problem of deciding whether a language has the universal prefix property is undecidable for one-counter languages. In order to do this we need to relate the set of invalid computations of M to a one-counter language (i.e. a language accepted by a one-counter automaton as defined above). There have been other similar approaches to such problems (see [22] for example).

Proposition 13. *If $M = (Q, \Sigma, a, \delta, q_0, q_f)$ is a counter machine then the following language is a one-counter language:*

$K = \{qa^n pa^j$: *the following conditions hold:*
if (q, b, k, p, r) is a quintuple of δ and kn is an integer then $kn \neq j$;
if (q, b, k, p, r) is a quintuple of δ and kn is not an integer then $j \neq n\}$

Proof. Since δ is a finite set of quintuples (q, b, k, p, r) and the one-counter languages are closed under union, it is sufficient to show that the language

$$\{qa^n pa^j : (kn \in \mathbb{Z} \Rightarrow kn \neq j) \wedge (kn \notin \mathbb{Z} \Rightarrow n \neq j)\}$$

is a one-counter language for any fixed quintuple (q, b, k, p, r). Now the condition

$$(kn \in \mathbb{Z} \Rightarrow kn \neq j) \wedge (kn \notin \mathbb{Z} \Rightarrow n \neq j)$$

is equivalent to

$$(kn \in \mathbb{Z} \wedge kn \neq j) \vee (kn \notin \mathbb{Z} \wedge n \neq j),$$

and (using again the fact that the one-counter languages are closed under union) we only need to show that the languages

$$\{qa^n pa^j : kn \in \mathbb{Z} \wedge kn \neq j\}, \quad \{qa^n pa^j : kn \notin \mathbb{Z} \wedge n \neq j\}$$

are both one-counter languages.

If $k \in \mathbb{N}$ then the condition $kn \in \mathbb{N}$ is automatically satisfied; if $k \notin \mathbb{N}$, then the condition $kn \in \mathbb{N}$ is equivalent to n mod $\frac{1}{k}$ being zero which we may check in the states of the machine. As far as $kn \neq j$ or $n \neq j$ is concerned this can be easily verified for any fixed k using the stack and the result follows. □

We now use Proposition 13 to prove the following result:

Theorem 14. *The following problem is undecidable:*
Input: *a one-counter automaton $N = (Q, \Sigma, \Gamma, \delta, s, A)$.*
Output: *"yes" if prefix$(L(N)) = \Sigma^*$; "no" otherwise.*

Proof. Let $M = (Q, \Sigma, a, \delta, q_0, q_f)$ be a counter machine and $\Gamma = Q \cup \{a\}$. The unique halting configuration of M is $q_f a$ and the following language over Γ

$K = \{qa^n pa^j$: the following conditions hold:
if (q, b, k, p, r) is a quintuple of δ and kn is an integer then $kn \neq j$;
if (q, b, k, p, r) is a quintuple of δ and kn is not an integer then $j \neq n\}$

is a one-counter language by Proposition 13. We consider the following languages:

(i) $L_1 = \Gamma^* - q_0 a\Gamma^*$. This is the set of all words in Γ^* which don't start with the unique initial configuration of M.
(ii) $L_2 = \Gamma^* - \Gamma^* q_f a\Gamma^*$. This is the set of all words not containing the unique halting configuration of M.
(iii) $L_3 = L_2 QQ\Gamma^* \cup a\Gamma^*$. This is the set of all words which are badly formed (as a sequence of configurations of M) before the halting configuration of M appears (if it appears).
(iv) $L_4 = L_2 K\Gamma^*$. This is the set of words which contain two successive configurations where the second does not follow from the first before the halting configuration of M appears (if it appears).

L_1, L_2 and L_3 are regular and L_4 is a one-counter language since the one-counter languages are closed under concatenation. If $L = L_1 \cup L_2 \cup L_3 \cup L_4$ we see that L is a one-counter language as the one-counter languages are closed under union.

L is the set of all invalid computations α of M such that no prefix of α is a valid computation; so L is prefix-closed. We have $L = \Gamma^*$ precisely when M does not accept any input and so deciding if $L = \Gamma^*$ is equivalent to deciding if $L(M) = \emptyset$; as counter machines are Turing complete this is undecidable.

If one could decide, given a one-counter automaton $N = (Q, \Sigma, \Omega, \delta, s, A)$, if prefix$(L(N)) = \Sigma^*$, then one could decide if $L = \Gamma^*$. However, L is prefix-closed; so its prefix closure is equal to Γ^* precisely when L itself is equal to Γ^*. We have just pointed out that determining whether or not $L = \Gamma^*$ is undecidable. □

Remark 15. Given Theorem 14 it is immediate that the problem of deciding whether prefix$(L(N)) = \Sigma^*$ is undecidable for pushdown automata. Given our interest in word problems of groups, we have focussed on one-counter and context-free languages in this paper, but the argument used in Theorem 14 would easily apply to other classes of languages as well □

6 Deletion Closed Languages and Decidability

We now investigate the decidability of the question as to whether a language L is deletion closed. As with Proposition 10, we note that this is easily seen to be decidable if L is specified by means of a finite automaton:

Proposition 16. *The following problem is decidable:*
Input: *a finite automaton* $N = (Q, \Sigma, \tau.s, A)$.
Output: *"yes" if* $L(N)$ *is deletion closed; "no" otherwise.*

Proof. Given N we construct the minimal automaton P accepting $L = L(N)$ and then calculate the syntactic monoid M of L as the transition monoid of P.

Let φ be the natural map from Σ^* to M, so that $L = S\varphi^{-1}$ for some $S \subseteq M$. For each element $x \in M$ we can test whether or not $x\varphi^{-1} \in L$ (this is independent of the choice of $x\varphi^{-1}$), and so we may determine S. Since the condition that L is deletion closed, i.e. that $\alpha u\beta \in L, u \in L \Rightarrow \alpha\beta \in L$, is equivalent to

$$\alpha u\beta \in S, u \in S \Rightarrow \alpha\beta \in S,$$

and since the latter is clearly decidable (as S is a subset of a finite monoid M), we can decide whether or not L is deletion closed. □

It is clear that the technique used in the proof of Proposition 16 will apply to many other properties of regular languages.

We will use the following result from [9], known as *Higman's Lemma*, in what follows:

Theorem 17. *The set of finite words over a finite alphabet, as partially ordered by the subsequence relation, is well-quasi-ordered. This, in particular, implies that there does not exist an infinite sequence where the elements of the sequence are all pairwise incomparable or, equivalently, any set containing only pairwise incomparable finite words is finite.*

We will write $\alpha \prec \beta$ if α can be obtained by deleting some symbols in β, i.e. if α is a proper subsequence of β; if α is a (not necessarily proper) subsequence of β, we will write $\alpha \preceq \beta$. The following may be of some independent interest:

Proposition 18. *A language $L \subseteq \Sigma^*$ which is deletion closed and contains Σ is regular.*

Proof. Given that L is deletion closed and contains Σ, deleting any symbols from a word in L always results in another word in L; so, if $\alpha \in L$ and $\beta \prec \alpha$, then $\beta \in L$. If $L = \Sigma^*$ then the result is clearly true; so we will assume that $L \neq \Sigma^*$ in what follows.

First, consider words $\beta \notin L$ such that $\alpha \prec \beta \Rightarrow \alpha \in L$ (such words are guaranteed to exist since $\emptyset \neq L \neq \Sigma^*$). Given two such words γ and β we must have that $\gamma \not\prec \beta$ and that $\beta \not\prec \gamma$; so, by Theorem 17, the set U of all such words must be finite.

Consider the language $V = \{\alpha \in \Sigma^* : \exists\ \beta \in U$ such that $\beta \preceq \alpha\}$. This language is regular (as all we are doing is checking that a subsequence lies in a finite set).

If $\alpha \in V$ then there exists $\beta \in U$ such that $\beta \preceq \alpha$ and so $\alpha \notin L$ (as $\beta \notin L$).

Conversely, if $\alpha \notin L$, then choose β minimal such that $\beta \preceq \alpha$ and $\beta \notin L$. If $\gamma \prec \beta$, then $\gamma \in L$ by the minimality of β; so $\beta \in U$ and hence $\alpha \in V$.

Given this we see that $\Sigma^* - L = V$ is regular and hence L is regular. □

Remark 19. The hypothesis that L contains Σ in Proposition 18 is necessary; for example, the language $\{a^n b^n : n \geqslant 0\}$ is deletion closed but not regular. Indeed, since the word problem of any finitely generated group is deletion closed and there are finitely generated groups with unsolvable word problem, there exist deletion closed languages that are not even recursively enumerable. □

Recall that a language $L \subseteq \Sigma^*$ is said to be *bounded* if there exist non-empty words $w_1, w_2, \ldots, w_k$ in Σ^* such that $L \subseteq w_1^* w_2^* \ldots w_k^*$. It is known that the problem of deciding, given a context-free grammar G, if $L(G)$ is bounded is decidable (see Theorem 5.5.2 in [6] for example). Given this, we can now prove:

Theorem 20. *The following problem is undecidable:*
Input: a one-counter automaton M.
Output: "yes" if $L = L(M)$ is deletion closed; "no" otherwise.

Proof. We show that, if we could solve this decision problem, then we could solve the universe problem for alphabets of size 2, contradicting Theorem 1.

Let us assume that we have an algorithm determining whether or not $L(M)$ is deletion closed for a one-counter automaton $M = (Q, \Sigma, \Gamma, \tau, s, A)$ where $\Sigma = \{a, b\}$. First we note that, if the language $L = L(M)$ is not deletion closed, then it cannot be Σ^*. Our next observation is that, if L does not contain Σ (which we can test as membership is decidable for one-counter languages), then it also cannot be Σ^*; so in these cases we simply output "no" and terminate.

Now assume that L is deletion closed and contains Σ. Each non-empty word α in L is uniquely expressible in the form $a^{n(1)}b^{m(1)} \ldots a^{n(\ell)}b^{m(\ell)}$ for some $\ell \geqslant 1$ where $n(1) \geqslant 0$, $n(i) > 0$ for $i > 1$, $m(i) > 0$ for $i < \ell$ and $m(\ell) \geqslant 0$; let us call this the *standard decomposition* for α and, given such a decomposition, let $\|\alpha\|$ denote ℓ. One of the following two possibilities must occur:

(i) there is a bound on $\|\alpha\|$ for $\alpha \in L$, i.e. there exists $k > 0$ such that every word of L has a standard decomposition $a^{n(1)}b^{m(1)} \ldots a^{n(\ell)}b^{m(\ell)}$ with $\ell \leqslant k$;
(ii) there is no such bound on $\|\alpha\|$ and so, for any k, we have a word of the form $a^{n(1)}b^{m(1)} \ldots a^{n(\ell)}b^{m(\ell)}$ in L with $\ell \geqslant k$. Given this, for every $k > 0$ there exists $\beta \in L$ such that $(ab)^k$ is a subsequence of β.

If possibility (i) occurs then L is bounded and $L \neq \Sigma^*$. If possibility (ii) occurs then, as every word in Σ^* is a subsequence of $(ab)^k$ for some k and L is deletion closed, we must have that $L = \Sigma^*$ in this case (and so L is not bounded).

So we test if L is bounded. If L is bounded then it is not Σ^* and we output "no" and, if L is not bounded, then $L = \Sigma^*$ and we output 'yes'. This gives us our contradiction. □

Remark 21. Given Theorem 20, it is immediate that the problem of deciding whether or not $L(M)$ is deletion closed is undecidable for pushdown automata. Given our interest in word problems of groups, we have focussed on one-counter and context-free languages in this paper, but the argument used in Theorem 20 would apply to other classes of languages as well.

If one were only interested in context-free languages then there are other approaches simpler than the one we have presented here. For example the property of a language being deletion closed distinguishes Σ^* from $\Sigma^* - \{w\}$ for any word w and one can build undecidability proofs from this based on the invalid computations of a Turing machine (see [11] for example). □

7 Word Problems and Decidability

We now turn our attention to word problems, i.e. those languages satisfying both the conditions W1 and W2 in Theorem 9. Given Theorem 9, together with Propositions 10 and 16, we immediately have the following result:

Proposition 22. *The following decision problem is decidable:*
Input: *a finite automaton N.*
Output: *"yes" if $L(N)$ is the word problem of a group; "no" otherwise.*

When we come to the one-counter languages, however, this problem becomes undecidable. This was shown for context-free languages in [17] and the argument used there extends to one-counter languages as well. This result does not follow immediately from Theorems 14 and 20; it is possible to have two undecidable problems whose conjunction is decidable. In order to prove the undecidability of this problem we need the concept of a *Hotz group* from [12]:

Definition 23. *The* Hotz group *$H(G)$ of a grammar $G = (V, \Sigma, P, S)$ is the group with presentation $\langle V \cup \Sigma : \{\alpha = \beta : (\alpha \to \beta) \in P\}\rangle$.*

Hotz showed that the group $H(G)$ for a reduced context-free grammar G depends only on $L(G)$. We also need the idea of a *collapsing group*:

Definition 24. *The* collapsing group *$C(L)$ of a language $L \subseteq \Sigma^*$ is the group with presentation $\langle \Sigma : \{\alpha = \beta : \alpha, \beta \in L\}\rangle$.*

The following connection between these two concepts will play a central role in what follows:

Definition 25. *A language $L \subseteq \Sigma^*$ is called a* language with Hotz isomorphism *if there exists a reduced grammar $G = (V, \Sigma, P, S)$ with $L = L(G)$ such that the collapsing group of L is isomorphic to $H(G)$.*

It is known [5] that all context-free languages are languages with Hotz isomorphism. In fact it is shown in [3] that:

Theorem 26. *A language $L \subseteq \Sigma^*$ is a language with Hotz isomorphism if and only if the collapsing group $C(L)$ is finitely presentable.*

Remark 27. The collapsing group $C(L)$ of a language $L \subseteq \Sigma^*$, where the empty word ϵ lies in L, will have every word in L representing the identity element of $C(L)$ but it may have other words representing the identity element as well.

Let $\wp$ denote the presentation $\langle \Sigma : \{\alpha = 1 : \alpha \in L\}\rangle$. If L is the word problem of some group K, then $\wp$ is a presentation for K and so K is isomorphic to $C(L)$. If L is not the word problem of a group then the word problem of the group with presentation $\wp$ must contain L as a proper subset.

In particular, if L is a context-free language which is the word problem of a group K, then $C(L)$ is isomorphic to K and we may obtain a finite presentation for K using the facts that K is isomorphic to $H(G)$ (where G is a context-free grammar generating L) and that the definition of $H(G)$ in Definition 23 is via a finite presentation. □

We are now in a position to prove our undecidability result:

Theorem 28. *The following decision problem is undecidable:*
Input: *a one-counter automaton $N = (Q, \Sigma, \Gamma, \tau, s, A)$.*
Output: *"yes" if $L(N)$ is the word problem of a group; "no" otherwise.*

Proof. Suppose we had an algorithm $\mathfrak{A}$ which could decide, given a one-counter automaton N, whether or not $L = L(N)$ were the word problem of some group G. We will show that one could then decide whether or not $L = \Sigma^*$ which is a contradiction by Theorem 1.

Since Σ^* is the word problem of the group $\{1\}$, if $\mathfrak{A}$ outputs "no", then we have that $L \neq \Sigma^*$. On the other hand, since L is context-free, if $\mathfrak{A}$ outputs "yes", then we know that the corresponding group G has a context-free word problem, and so G is virtually free by Theorem 6. We can now obtain a finite presentation $\wp$ for G as in Remark 27 and then use the presentation $\wp$ to test G for triviality as in Theorem 3. Since G is trivial if and only if $L = \Sigma^*$ we now have an algorithm for determining whether or not $L = \Sigma^*$, a contradiction. □

Remark 29. Since every one-counter language is context-free, in that a one-counter automaton is a special case of a pushdown automaton, it immediately follows from Theorem 28 that there is no algorithm to decide whether or not $L(M)$ is the word problem of a group for a pushdown automaton M (as proved in [17]). The proof given in Theorem 28 will, in fact, work for any family $\mathcal{F}$ of context-free languages where the universe problem is undecidable (provided that $\mathcal{F}$ is specified in such a way that a finite presentation for the Hotz group of any language L in $\mathcal{F}$ can be effectively determined). □

8 Deterministic Context-Free Languages

We saw in Theorem 6 that a group has a context-free word problem if and only if it is virtually free. It is not hard to show that the word problem of a virtually free group is deterministic context-free. So we have the following immediate consequence of Theorem 6:

Theorem 30. *If a group G has a context-free word problem, then it has a deterministic context-free word problem.*

However, despite the fact that it is undecidable whether or not a context-free language is the word problem of a group, this problem becomes decidable if the language is deterministic context-free and is given by a deterministic pushdown automaton:

Theorem 31. *The following decision problem is decidable:*
Input: *a deterministic pushdown automaton $M = (Q, \Sigma, \Gamma, \tau, s, A)$.*
Output: *"yes" if $L(M)$ is the word problem of a group; "no" otherwise.*

Proof. If $\epsilon \notin L = L(M)$ then L is not the word problem of a group; so we check first that $\epsilon \in L$ (outputting "no" if that is not the case); we will assume that $\epsilon \in L$ in what follows.

We convert our deterministic pushdown automaton to a reduced context-free grammar Γ such that $L(\Gamma) = L$ and then use the Hotz group construction in Remark 27 to write down a finite presentation $\wp$ of the group $G = H(\Gamma)$.

As in Remark 27, if L is the word problem of a group, then it must be the word problem of G. If W is the word problem of G with respect to the generating set Σ, then the question is whether $L = W$ (in which case L is the word problem of a group) or $L \subset W$ (in which case L is not the word problem of a group).

If L is the word problem of G then, as L is context-free, G must be virtually free. With this is mind, we start a process which we will refer to as Process 1.

Process 1 enumerates the finite-index subgroups of G and enumerates all presentations of the finite-index subgroups, checking each such presentation it generates to see if it is a natural presentation of a free group (i.e. a presentation with no relations). This is a semi-decision process; if G is virtually free, then we will eventually find such a presentation and so know that G is virtually free, but, if G is not virtually free, then this process will not terminate.

At the same time we start Process 2. Process 2 takes the finite presentation $\wp$ and enumerates the words in Σ^* representing the identity element, checking each one it generates for membership of L. If Process 2 ever finds a word which is trivial in the group G but not a member of L then we terminate all the running processes and output "no". (If L were the word problem of a group then it has to be the word problem of G, in which case no word trivial in G could lie outside L.) Process 2 is also a semi-decision process; we continue enumerating words whilst we do not have an output of "no".

Eventually one of these two processes must terminate. If it is Process 1 then we know that the group G is virtually free and we start Process 3. Process 3 uses the presentation $\wp$ of G and its finite-index free subgroup to construct a deterministic pushdown automaton N which accepts the word problem of G; we can then test N for equivalence with M by the theorem of Sénizergues in [21]. We halt all the processes and output the result of the equivalence test as our final output. Note that, if we reach Process 3, then Process 3 will always terminate.

Eventually either Process 2 terminates (and we output "no") or else Process 1 (and therefore Process 3) terminates. Thus we have an algorithm which outputs "yes" if $L(M)$ is the word problem of a group and outputs "no" if it is not, as required. $\square$

Remark 32. As the reader will see, the use of the theorem of Sénizergues concerning the decidability of the equivalence problem for deterministic pushdown automata is a critical component of the proof of Theorem 31. We are also using procedures such as the enumeration of finite-index subgroups of a group searching for one that is a free group. As it is (in general) undecidable as to whether or not a finitely presented group is virtually free, this procedure will not necessarily terminate, and our proof relies on the fact that we can run this in parallel with another semi-decision procedure and that, given our situation, one of these two procedures must terminate. Given that the two procedures we have used will not have computable time complexity in general, we are not claiming any degree of efficiency for this decision procedure, merely that the problem is decidable. $\square$

Acknowledgements. The authors would like to thank the referees for their careful reading of the paper and for their constructive suggestions. The second author would also like to thank Hilary Craig for all her help and encouragement.

References

1. Anisimov, V.A.: The group languages. Kibernetika 4, 18–24 (1971)
2. Berstel, J.: Transductions and Context-free Languages. Teubner (1979)
3. Diekert, V., Möbus, A.: Hotz-isomorphism theorems in formal language theory. Informatique Théorique et Applications 23, 29–43 (1989)
4. Dunwoody, M.J.: The accessibility of finitely presented groups. Inventiones Mathematicae 81, 449–457 (1985)
5. Frougny, C., Sakarovitch, J., Valkema, E.: On the Hotz-group of a context-free grammar. Acta Mathematica 18, 109–115 (1982)
6. Ginsburg, S.: The Mathematical Theory of Context-free Languages. McGraw-Hill (1966)
7. Herbst, T.: On a subclass of context-free groups. Informatique Théorique et Applications 25, 255–272 (1991)
8. Herbst, T., Thomas, R.M.: Group presentations, formal languages and characterizations of one-counter groups. Theoretical Computer Science 112, 187–213 (1993)
9. Higman, G.: Ordering by divisibility in abstract algebras. Proceedings of the London Mathematical Society 2, 326–336 (1952)
10. 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, 197–209 (2008)
11. Hopcroft, J.E., Ullman, J.D.: Introduction to Automata Theory, Languages and Computation. Addison-Wesley (1979)
12. Hotz, G.: Eine neue invariante für kontext-freie sprachen. Theoretical Computer Science 11, 107–116 (1980)
13. Ibarra, O.H.: Restricted one-counter machines with undecidable universe problems. Mathematical Systems Theory 13, 181–186 (1979)
14. Ito, M.: Algebraic Theory of Automata & Languages. World Scientific Press (2004)
15. Ito, M., Kari, L., Thierrin, G.: Insertion and deletion closure of languages. Theoretical Computer Science 183, 3–19 (1997)
16. Johnson, D.L.: Presentations of Groups. Cambridge University Press (1990)
17. Lakin, S.R., Thomas, R.M.: Space complexity and word problems of groups. Groups-Complexity-Cryptology 1, 261–273 (2009)
18. Muller, D.E., Schupp, P.E.: Groups, the theory of ends, and context-free languages. Journal of Computer and System Sciences 26, 295–310 (1983)
19. Parkes, D.W., Thomas, R.M.: Groups with context-free reduced word problem. Communications in Algebra 30, 3143–3156 (2002)
20. Rampersad, N., Shallit, J., Xu, Z.: The computational complexity of universality problems for prefixes, suffixes, factors, and subwords of regular languages. Fundamenta Informaticae 116, 223–236 (2012)
21. Sénizergues, G.: $L(A) = L(B)$? Decidability results from complete formal systems. Theoretical Computer Science 251, 1–166 (2001)
22. Valk, R., Vidal-Naquet, G.: Petri nets and regular languages. Journal of Computer and System Sciences 23, 299–325 (1981)