Groups-Complexity-Cryptology
Volume 1 (2009), No. 2, 261–273

Space Complexity and Word Problems of Groups

Stephen R. Lakin
Division of Mathematics and Statistics, Faculty of Advanced Technology, University of Glamorgan, Pontypridd CF37 1DL, UK

Richard M. Thomas
Department of Computer Science, University of Leicester, Leicester LE1 7RH, UK

Dedicated to Benjamin Fine on the occasion of his 60th birthday.

Received: July 1, 2009
Revised manuscript received: November 10, 2009

This paper is concerned with the question of determining which groups have their word problems lying in a given complexity class. Our main results give sufficient conditions for the existence of groups whose word problem is contained in some specified space complexity class but is not contained in some other given space complexity class.

1. Introduction

In this section we will give an overview of the paper; precise definitions of the terms used here will be found in Sections 2, 3 and 4. Our starting point is that there are several intriguing links between group theory and formal language theory. One of the many interesting such connections is the consideration of which groups G have a word problem lying in a particular class $\mathcal{F}$ of formal languages, and this will be the main theme of this paper; see [7, 8, 10, 23] (for example) for a survey of some of these issues.

One somewhat surprising feature of this is that we can find examples of naturally occurring classes of languages $\mathcal{F}_1$ and $\mathcal{F}_2$ such that $\mathcal{F}_1 \subset \mathcal{F}_2$ but such that the classes of groups whose word problems lie in $\mathcal{F}_1$ and $\mathcal{F}_2$ coincide. Examples of this include the case where $\mathcal{F}_1$ is the class of deterministic context-free languages and $\mathcal{F}_2$ is the class of context-free languages; this follows from the beautiful classification of such groups by Muller and Schupp [17, 18] which shows that a finitely generated group has a context-free or deterministic context-free word problem if and only if it has a free subgroup of finite index. Indeed, this is still the case if we replace the class of deterministic context-free languages by the class of NTS languages [2].

In fact, if we assume that our class of languages is a cone (i.e. a class of languages which is closed under homomorphism, inverse homomorphism and intersection

ISSN 1867-1144 / $ 2.50

with regular languages) contained in the class of context-free languages, then there are only three classes of groups that occur, the finite groups (corresponding to the regular languages), the virtually cyclic groups (corresponding to the one-counter languages) and the virtually free groups (corresponding to the context-free languages); see [9]. Recall that a group is said to be *virtually* $\mathcal{P}$ if it has a subgroup of finite index with property $\mathcal{P}$.

The general point here is that there are some significant restrictions imposed on a language by insisting that it is the word problem of a group; see [22], for example, for a necessary and sufficient set of conditions for a language to be such a word problem. A consequence of this is that some distinct classes of languages $\mathcal{F}_1$ and $\mathcal{F}_2$ with $\mathcal{F}_1 \subset \mathcal{F}_2$ coincide when restricted to word problems of groups. In some sense, this phenomenon appears to be more prevalent the simpler the classes of languages are. The aim of this paper is to provide some sufficient conditions on such classes $\mathcal{F}_1$ and $\mathcal{F}_2$ to ensure that this situation does not occur, i.e. that there really are groups whose word problem lies in $\mathcal{F}_2$ but not in $\mathcal{F}_1$.

In our case $\mathcal{F}_1$ and $\mathcal{F}_2$ will be space complexity classes and the groups we construct will all have decidable word problem. It is well known that a finitely generated group can have undecidable word problem, even if we add the extra assumption that the group is finitely presented [4, 19]. When we consider decidable word problems, it is interesting that, even if we make some quite strong restrictions on the structure of a language, it can still be undecidable whether or not a language is the word problem of a group. For example, there is no algorithm that, given a context-free grammar Γ, will decide whether or not the language $L(\Gamma)$ generated by Γ is the word problem of a group. This fact may well have been noted elsewhere but we will give a outline here as to why this is the case.

It is well known that, the problem of determining, given an arbitrary context-free grammar $\Gamma = (N, \Sigma, P, S)$ as input, whether or not $L(\Gamma) = \Sigma^*$ is undecidable (see Theorem 8.11 of [11] for example). Suppose that we could decide, given such a context-free grammar Γ, whether or not $L(\Gamma)$ was the word problem of a group. Since Σ^* is the word problem of a group (namely the trivial group), knowing that $L(\Gamma)$ was not such a word problem would immediately give that $L(\Gamma) \neq \Sigma^*$. On the other hand, if $L(\Gamma)$ were the word problem of a group G, then we could write down a finite presentation of G from the context-free grammar Γ using the Hotz group (see [5, 6, 12] for example). Now $L(\Gamma) = \Sigma^*$ if and only if G is trivial, and we can decide the triviality of G given a presentation for G if we know (as we do here) that G is virtually free (see [14] for example). This gives a contradiction to our assumption that we could decide whether or not $L(\Gamma)$ was the word problem of a group.

Let us return to the matter in hand. After introducing our terms and notation in Sections 2, 3 and 4, we set up the machinery we need to tackle this problem in Section 5. Our main tool is small cancellation theory and, in particular, Greendlinger's Lemma (Theorem 5.3). Our main result then appears in Section 6: we show how to construct groups whose word problem lies in $\mathcal{F}_2$ but not $\mathcal{F}_1$ where

$\mathcal{F}_1$ and $\mathcal{F}_2$ are both space complexity classes (see Theorems 6.4 and 6.5). Our results are stated in terms of space complexity but the techniques demonstrated in this paper would allow one to prove similar results for time complexity.

2. Complexity classes

Let us begin by setting up the notation we will use, and giving the necessary definitions and preliminary results. In this section we talk about complexity classes; for a general account of these issues, see (for example) [11, 20].

An *alphabet* is a finite set X and a *language* is some subset of X^*, the set of all finite strings over X. A language may or may not include the empty word, which we will denote by ϵ. A *class* of languages is simply a collection of languages (usually satisfying some given property).

We will take the *Turing machine* as our model of computation. A *nondeterministic Turing machine* has a finite set Q of *states*, a finite set Σ of *input symbols* and a finite set Γ of *work-tape symbols*. We think of the input α as being a word in Σ^* recorded on an *input tape* bounded by special symbols $\triangleright$ and $\triangleleft$ (which are not elements of Σ); for convenience, we let $\overline{\Sigma}$ denote the set $\Sigma \cup \{\triangleright, \triangleleft\}$. The set Γ contains the special symbols $\triangleright$ and $\sqcup$; we have one or more work-tapes, each containing $\triangleright$ in the leftmost cell followed by symbols from $\Gamma - \{\triangleright\}$. The symbol $\sqcup$ represents a blank cell.

If we have n work-tapes, then we have a transition relation

$$\tau \subseteq Q \times \overline{\Sigma} \times \Gamma^n \times Q \times \{L, R, N\} \times \Gamma^n \times \{L, R, N\}^n.$$

A transition $(q, a, g_1, \ldots, g_n, r, D, h_1, \ldots, h_n, D_1, \ldots, D_n)$ means that, if we are in state q reading a on the input tape and $g_1, \ldots, g_n$ on the n work-tapes, then we can move to state r, move in direction D on the input tape, replace g_i by h_i on the i^{th} work-tape and move in direction D_i on the i^{th} work-tape (where $1 \leqslant i \leqslant n$). Not too surprisingly, L stands for "left", R for "right" and N for "no move".

On the input tape, we cannot move left when the head is positioned over $\triangleright$ or right when the head is positioned over $\triangleleft$. Note that, whilst we can otherwise move freely over the input tape, we cannot alter the symbols written there. We cannot move left on the-work tape when reading $\triangleright$ and we insist that $\triangleright$ appears in the left-hand cell of the work-tape and nowhere else.

If τ is a partial function from

$$Q \times \overline{\Sigma} \times \Gamma^n \text{ to } Q \times \{L, R, N\} \times \Gamma^n \times \{L, R, N\}^n,$$

then our Turing machine is *deterministic*. In a deterministic Turing machine there is only (at most) one possible computation path for a particular input whereas, in a nondeterministic Turing machine, there may be several such paths.

We will be concerned with the case where a Turing machine represents a decision procedure for a language. In addition to the situation described above, we have

three designated distinct states s, h_Y and h_N (the *start state* the *accept state* and the *reject state* respectively). There is no transition defined when we reach either h_Y or h_N.

Let us first consider a deterministic Turing machine M. If M is set up in the start state s with input α (and only $\triangleright$ initially on each work tape), then α is said to be *accepted* if M reaches the accept state h_Y and *rejected* if M reaches the reject state h_N (we insist that, for any input, one of these two states is eventually reached). The *language* $L(M)$ of M is the set of all words accepted by M.

With a nondeterministic Turing machine, we accept the input if some computation path leads to h_Y and reject the input if every computation path leads to h_N. It is a standard fact that, if there is a nondeterministic Turing machine deciding a language, then there is a deterministic Turing machine deciding that language.

If f is a function from $\mathbb{N}$ to $\mathbb{N}$, then we say that a language L lies in $\mathbf{DSPACE}(f(n))$ if L is decided by a deterministic Turing machine that uses at most $f(n)$ squares on its work tapes when deciding a word of length n. We similarly have $\mathbf{NSPACE}(f(n))$, where we have a nondeterministic Turing machine in place of a deterministic one.

If $\mathcal{F}$ is a set of functions then we will use the notation $\mathbf{NSPACE}(\mathcal{F})$ and $\mathbf{DSPACE}(\mathcal{F})$ to denote the respective space complexity classes, where a language L lies in $\mathbf{NSPACE}(\mathcal{F})$ if and only if L lies in $\mathbf{NSPACE}(f(n))$ for some $f(n) \in \mathcal{F}$, and similarly for $\mathbf{DSPACE}(\mathcal{F})$.

As well as deciding languages, Turing machines can also compute functions. In this case we have a single halt state and also an *output tape* which we can write to (but where the symbols are not erased once written). We say that the Turing machine computes the function f if, for any input α, we have that the output tape contains $f(\alpha)$ when we reach the halt state.

Suppose that $\mathcal{F}$ is a class of *proper complexity functions*, i.e. a class of functions $f : \mathbb{N} \to \mathbb{N}$ which are nondecreasing ($f(n) \leqslant f(n+1)$ for all $n \in \mathbb{N}$) and such that there is a Turing machine M_f with output such that, on an input x of length n, M_f outputs a word $\sqcup^{f(n)}$ of length $f(n)$, using $\mathcal{O}(f(n))$ space (apart from the input) and halting after $\mathcal{O}(n+f(n))$ steps. (Note that the convention throughout this paper will be that $\mathbb{N}$ contains 0.) The only condition we impose on our classes of functions will be as follows:

Property 2.1. $\mathcal{F}$ is a class of proper complexity functions satisfying:

(i) $f(n) \geqslant n$ for every $f(n) \in \mathcal{F}$ and all $n \in \mathbb{N}$;
(ii) if $f(n) \in \mathcal{F}$, if $a \in \mathbb{N} - \{0\}$ and $b \in \mathbb{N}$, and if $g(n)$ is the function defined by $g(n) = f(an + b)$ for all $n \in \mathbb{N}$, then $g(n) \in \mathcal{F}$.

For example, the class of non-constant polynomial functions with coefficients in $\mathbb{N}$ satisfies this condition, as does the subclass of all functions $f(n)$ of the form $c_1 n + c_2$ (where $c_1 \in \mathbb{N} - \{0\}$ and $c_2 \in \mathbb{N}$ are constants). □

3. Group presentations

We now turn our attention to groups and group presentations; for further information, the reader is referred to [13, 15] (for example).

Suppose that G is a group and X is an alphabet, and we have a mapping $\theta : X \to G$; then θ extends naturally to a homomorphism $\varphi : X^* \to G$ where we take $(x_1 \dots x_n)\varphi$ to be $(x_1\theta) \dots (x_n\theta)$. If this homomorphism φ is surjective then we say that X *generates* G, and we call X a *generating set* for G. In this way each word in X^* represents a unique element of G. We sometimes need to be careful and distinguish equality of words in X^* from equality in G; in this context, if u and u are elements of X^*, we will write $u \equiv v$ if u and v are identical as words and $u = v$ if u and v represent the same element of G (i.e. if $u\varphi = v\varphi$).

Note that, when we say "generating set" here, we are really referring to a *monoid generating set*, i.e. a set of elements that generates the group G as a monoid. If such a monoid generating set X is a disjoint union $Y \cup Y^{-1}$, where $Y^{-1} = \{y^{-1} : y \in Y\}$ is a set in a (1-1) correspondence with Y and $y^{-1}\theta = (y\theta)^{-1}$, then we say that Y is a *group generating set* for G. We will mainly be considering group generating sets in this paper.

Suppose that X is a monoid generating set for a group G and let $\varphi : X^* \to G$ be the natural homomorphism as above; then the *word problem* of G with respect to X is defined to be

$$W_X(G) = 1\varphi^{-1},$$

that is the set of words in X^* which represent the identity element of G. Equivalently, we can think of the word problem as being the question as to whether or not a given word in X^* represents the identity element. In the case of a group generating set Y, our homomorphism maps $(Y \cup Y^{-1})^*$ onto G as above.

The main question we shall consider in this paper is the following: what sort of conditions on classes of formal languages $\mathcal{F}_1$ and $\mathcal{F}_2$ would guarantee that there is a group with word problem lying in $\mathcal{F}_2$ but not in $\mathcal{F}_1$? Note that, whilst it would appear that whether or not the word problem for a group lies in a class $\mathcal{F}$ depends on the choice of the generating set X, it is well known that this is not the case if $\mathcal{F}$ is closed under inverse homomorphism (see [10] for example).

Our construction relies heavily on small cancellation theory; see [15] for example. Suppose we have the free group F on the set $Y = \{y_1, \dots, y_m\}$. Let R be a subset of $(Y \cup Y^{-1})^*$, where R satisfies the following conditions:

(i) every element u of R is cyclically reduced (i.e. u is reduced and is not of the form $y\alpha y^{-1}$ for any $y \in Y \cup Y^{-1}$ and any $\alpha \in (Y \cup Y^{-1})^*$);
(ii) for every u in R, u^{-1} also lies in R;
(iii) for every u in R, every cyclic permutation of u also lies in R.

When we talk about u^{-1} for a word u over $Y \cup Y^{-1}$, we are performing the obvious construction: we replace each symbol y in u by the corresponding inverse symbol y^{-1} and then reverse the word. Such a set R as described here is said to

be *symmetrized*. Given that the words in R are all reduced, we can think of R as being a subset of the free group F as opposed to a subset of $(Y \cup Y^{-1})^*$, as distinct elements of R will represent distinct elements of F. Note that a cyclic permutation of a reduced word u is a cyclically reduced conjugate of u in F.

Suppose that $u_1 \equiv st_1$ and $u_2 \equiv st_2$ are distinct elements of R; then s is said to be a *piece* (relative to R). Note that, since s is cancelled in the product ${u_1}^{-1}u_2$, and R is symmetrized, a piece is simply a subword of an element of R which is cancelled in the product of two non-inverse elements of R. We now recall the definition of the small cancellation condition $C'(\tau)$:

Definition 3.1. Suppose that R is a symmetrized subset of a free group F as above and $\tau > 0$; then R satisfies the *small cancellation condition* $C'(\tau)$ if, for any $u \equiv st$ in R where s is a piece, we have that $|s| < \tau|u|$. □

For a word u of F, we use the notation $u > cR$ to mean that there is some $r \in R$ with $r \equiv uv$ and $|u| > c|r|$. We will be particularly interested here in groups given by a presentation $G = \langle X : R \rangle$ where R satisfies $C'(\frac{1}{6})$; these are known as *sixth-groups*.

4. String rewriting systems

We now recall some properties of "string rewriting systems"; for a general account of such systems, see [3] (for example).

Given a finite alphabet X, a *(string) rewriting system* $\Re$ over X is a set of rules of the form $u \to v$, where $u, v \in X^*$. We define the *reduction relation* $\Longrightarrow_{\Re}^*$ to be the reflexive transitive closure of the relation $\Longrightarrow_{\Re}$, where $xuy \Longrightarrow_{\Re} xvy$ if $x, y \in X^*$ and $(u \to v) \in \Re$.

A rewriting system $\Re$ is said to be *confluent* if, whenever we have words u, v_1 and v_2 in X^* with $u \Longrightarrow_{\Re}^* v_1$ and $u \Longrightarrow_{\Re}^* v_2$, then there exists $w \in X^*$ such that $v_1 \Longrightarrow_{\Re}^* w$ and $v_2 \Longrightarrow_{\Re}^* w$. A rewriting system $\Re$ is said to be *Noetherian* if there is no infinite chain

$$u_1 \Longrightarrow_{\Re} u_2 \Longrightarrow_{\Re} u_3 \Longrightarrow_{\Re} \ \ldots\ldots$$

A rewriting system which is both confluent and Noetherian is said to be *complete*. A word which does not contain the left-hand side of a rule in $\Re$ is said to be *irreducible*. If we have a complete rewriting system $\Re$, then, for each $u \in X^*$, there is a unique irreducible $w \in X^*$ with $u \Longrightarrow_{\Re}^* w$ we refer to w as the *normal form* of u.

We are sometimes satisfied with a rewriting system that is "confluent on a language L". In this situation we have a rewriting system $\Re$ over a set X and a language $L \subseteq X^*$ which is preserved by $\Re$ (i.e. if $u \in L$ and $u \Longrightarrow_{\Re} v$ then $v \in L$). We say that $\Re$ is *confluent on* L if, whenever we have $u, v_1, v_2 \in L$ with $u \Longrightarrow_{\Re}^* v_1$ and $u \Longrightarrow_{\Re}^* v_2$, then there exists $w \in L$ such that $v_1 \Longrightarrow_{\Re}^* w$ and $v_2 \Longrightarrow_{\Re}^* w$. In this case (assuming that the system $\Re$ is Noetherian), elements of L will have normal forms but elements of $X^* - L$ may not.

This is relevant (for example) for rewriting systems for groups where L is the word problem of the group; if the rewriting system is confluent on L but not necessarily elsewhere, then we say that the rewriting system is $[\epsilon]$-*confluent*. The Dehn algorithm in a hyperbolic group is an example of such a system (see [1] for example); this is related to the techniques described later in this paper (though the groups we describe here are not necessarily finitely presented). Other such systems of interest are the so called *special rewriting systems* where each rule is of the form $\alpha \to \epsilon$; see [16, 21] for example.

5. Preliminary results

We will now establish some preliminary results that will allow us to make our construction.

Proposition 5.1. *Let $X = \{a_1, \ldots, a_r\}$ be a finite alphabet and suppose that θ is a permutation of X such that $\theta^2 = 1$. For a word $u \equiv a_{i_1} a_{i_2} \ldots a_{i_m}$ in X^*, we define $u\overline{\theta}$ to be $(a_{i_m}\theta)(a_{i_{m-1}}\theta)\ldots(a_{i_1}\theta)$.*

Suppose further that $\mathcal{F}$ is a class of functions satisfying Property 2.1 *and that $K \subseteq X^*$ is such that $K \in \mathbf{NSPACE}(\mathcal{F})$. Then, defining S to be the set of rewrite rules*

$$S = \{a(a\theta) \to \epsilon, (a\theta)a \to \epsilon : a \in X\} \cup \{v \to w\overline{\theta} : vw \in K, |v| > |w|\},$$

we have that $L = \{u \in X^* : u \Longrightarrow_S^* \epsilon\} \in \mathbf{NSPACE}(\mathcal{F})$.

Proof. Take some word $u \in X^*$. If $u \in L$ then some sequence of reductions (using the rules in S) leads to the empty word. We begin with u and our nondeterministic algorithm works as follows. At each stage, we choose some reduction to make: we either remove a subword of the form $a(a\theta)$ or $(a\theta)a$ or else we perform the following:

(i) choose some subword v of u;
(ii) choose some word w with $|w| < |v|$;
(iii) verify that the word vw lies in K;
(iv) replace v by $w\overline{\theta}$.

We continue performing reductions until we either reduce to the empty word ϵ or else there are no more possible reductions we can make. Note that this algorithm is clearly terminating since every reduction strictly reduces the length of the word and hence the number of possible steps is bounded by the length of the original word. Also note that there are only a finite number of choices at any point, since there are a finite number of subwords of any word and a finite number of possible words of length strictly less than a given word.

It is clear from the definition that, if $u \in L$, then some sequence of choices leads to the empty word, and hence we can accept u; if $u \notin L$ then no such sequence of choices leads to the empty word and so we must reject u. Hence this algorithm certainly decides L. It remains to consider the space bounds of this algorithm.

Now $K \in \mathbf{NSPACE}(\mathcal{F})$, and so K is decidable in nondeterministic space $f(m)$ for some $f(m) \in \mathcal{F}$. If $|u| = n$, v is a subword of u and $vw \in K$ (as per our rules) then, since $|vw| < 2n$ (as v has length at most n and $|w| < |v|$), verifying membership of K can be done in space at most $f(2n) \in \mathcal{F}$.

Every possible rewrite strictly reduces the length of the word, and so it is clear that we can perform any rewrite in linear space. Choosing a subword and some shorter word can also obviously be done in linear space. Hence, since $\mathcal{F}$ satisfies Property 2.1, it is immediate that L lies in $\mathbf{NSPACE}(\mathcal{F})$ as required. □

This result concerns nondeterministic languages. In fact, with a suitable condition on the rewrite rules, we can prove a similar result for deterministic languages (which is the fact we will actually be using in this paper):

Proposition 5.2. *Suppose we have the situation in Proposition* 5.1*, but suppose that* $K \in \mathbf{DSPACE}(\mathcal{F})$ *and the set of rewrite rules* S *is confluent on* L*; then we have that* $L \in \mathbf{DSPACE}(\mathcal{F})$*.*

Proof. The basis of the algorithm is as in Proposition 5.1, but we need to make it deterministic. The crucial point to note is that, since our rewriting system always terminates, we have a Noetherian system which is confluent on L and hence every word in L has the unique normal form ϵ. Thus, if a word reduces to the empty word ϵ under some choice of sequences of rewrites, it does so under every possible sequence of rewrites.

The system need not be confluent on words outside L. The point is that a word $u \notin L$ must rewrite to some irreducible word v (as the algorithm is terminating) and v cannot be the empty word ϵ (or else u would lie in L). Since all words in L have the unique normal form ϵ, once we reach a non-empty irreducible word, we know that the original word u cannot lie in L.

Hence we can use some deterministic procedure to choose which rewrite to perform at each stage on a word u. Our algorithm works as follows. At each stage, we remove subwords of the form $a(a\theta)$ or $(a\theta)a$, and then:

(i) consider every possible subword v of u in turn;
(ii) for each subword v, consider every possible word w with $|w| < |v|$;
(iii) if we find v, w such that $vw \in K$, replace v by $w\theta$ and start the algorithm again.

This algorithm allows us to continue performing deterministic rewrites until we terminate either with the empty word ϵ (in which case $u \in L$), or with some non-empty word (in which case $u \notin L$); so it clearly recognises L. As in the proof of Proposition 5.1 it is clear that L lies in $\mathbf{DSPACE}(\mathcal{F})$. □

As we mentioned above, the main tool used in our construction will be small cancellation theory. Suppose, as before, that we have a free group F with generating set X, and that R is a symmetrized subset of F. The following well-known result

(*Greendlinger's lemma for sixth groups*) is of fundamental importance here; see [15] for example.

Theorem 5.3. *Let F be a free group with generating set X, R be a symmetrized subset of F, and suppose that $G = F/N$ where N is the normal closure of R in F. Let $u \in N$ be a non-trivial, cyclically reduced word and suppose that R satisfies $C'(\frac{1}{6})$. Then either $u \in R$ or else some cyclic permutation of u contains one of the following:*

(*i*) *two disjoint subwords, each $> 5R/6$;*
(*ii*) *three disjoint subwords, each $> 2R/3$;*
(*iii*) *four disjoint subwords, two $> 2R/3$ and two $> R/2$;*
(*iv*) *five disjoint subwords, four $> R/2$ and one $> 2R/3$;*
(*v*) *six disjoint subwords, each $> R/2$.*

This has the following consequence:

Proposition 5.4. *We take F, X, R and N as in Theorem* 5.3*, with R a decidable set satisfying $C'(\frac{1}{6})$. Suppose that $u \in N$. Then we can reduce u to ϵ with a finite sequence of steps of the form*

(*i*) *removing a subword of the form $x^{-1}x$ or xx^{-1} with $x \in X$;*
(*ii*) *replacing a subword v by w^{-1} where $|w| < |v|$ and $vw \in R$.*

Proof. Let u be a (word representing a) non-trivial element of N; we can assume that u is cyclically reduced (if u is if the form $w^{-1}vw$ for some words v and w and if we can reduce v to ϵ, then we can reduce u to ϵ by following up with rules of type (*i*) applied to the resulting word $w^{-1}w$). Hence, from Theorem 5.3, we either have $u \in R$ (in which case we replace u by ϵ as an application of the second rule and we are done) or else u contains more than half of a relator in R. This second possibility follows from each of the cases in Theorem 5.3, since there are at least two disjoint subwords in every case, and then, even without any cyclic permutation of the word, there must exist at least one suitable subword in our word.

In this second case we again apply the second rule; we can then apply the first rule repeatedly to obtain a reduced word. This reduced word still lies in N (since it still represents the same element of the group G, namely the identity) and is strictly shorter than u, and hence, proceeding inductively, we must eventually terminate with ϵ.

It is clear that this algorithm will take only a finite number of steps since it is length-reducing. □

Now suppose that we have an alphabet $X = \{x_1, \ldots, x_k\}$ and suppose that $J \subseteq X^*$ where $J \in \mathbf{DSPACE}(\mathcal{F})$, for some class $\mathcal{F}$ of functions satisfying Property 2.1. Let

$$A = \{x_1, \ldots, x_k, a_1, \ldots, a_{12}\}, \tag{1}$$

where $a_i \notin X$ for each i, and define R' to be

$$\{a_1 u a_2 u \dots a_{12} u : u \in J\}.$$

Then we may define

$$R = \{v : v \text{ is a cyclically reduced word formed from } R' \text{ by taking the closure under inverses and cyclic permutations}\}. \quad (2)$$

Let $\Sigma = A \cup A^{-1}$. Then we may define a permutation $\theta : \Sigma \to \Sigma$ where θ sends each symbol to its corresponding inverse symbol; clearly $\theta^2 = 1$.

Proposition 5.5. *R is symmetrized and satisfies $C'(\frac{1}{6})$.*

Proof. It is clear from the definition that R is symmetrized. Note that a maximal piece p of R is of the form va_iw or $w^{-1}{a_i}^{-1}v^{-1}$ where $1 \leqslant i \leqslant 12$, v is a suffix to a word u in J and w a prefix to the same word u. Hence, for any piece p, we certainly have that $|p| \leqslant 1 + 2|u|$.

Suppose a piece p occurs in a cyclic permutation of a word $z \equiv a_1 u \dots a_{12} u$ or its inverse. Then we have

$$|p| \leqslant 1 + 2|u| < \frac{1}{6}(12 + 12|u|) = \frac{1}{6}|z|.$$

Hence any piece of R is of length strictly less than a sixth of any word of which it is a part, and hence R satisfies $C'(\frac{1}{6})$ as required. □

6. The main result

Now consider the group G with presentation $\langle A : R \rangle$ with A and R defined as in (1) and (2) above, and let W be the word problem of this group. We keep this convention throughout Propositions 6.1, 6.2 and 6.3.

Proposition 6.1. *$W \in \boldsymbol{DSPACE}(\mathcal{F})$.*

Proof. By Propositions 5.4 and 5.5, a word in W can be reduced to the empty word by a sequence of rewrites either of the form $xx^{-1} \to \epsilon$, $x^{-1}x \to \epsilon$, or else $v \to w^{-1}$ where $|w| < |v|$ and $vw \in R$; these rewrites are precisely the rules given in Propositions 5.1 and 5.2 (note that, since R is symmetrized, the rewrite rules specified there are sufficient) and hence W is equivalent to the language L defined therein. The only issue we need to resolve is the question of whether these rewrite rules are confluent on W.

This is a simple matter of induction. There are no rewrites to perform on the empty word, and so the base step is trivial. For the inductive step, take some non-empty word $u \in W$. Whichever rewrite rule we choose, we either remove a word equivalent to the identity, or else replace a word by a shorter word representing the same element of the group; hence our new word u' represents the same element

of the group as u and therefore lies in W. By induction, u' has the unique normal form ϵ, and hence, whichever sequence of choices of rewrites we make on u, we always terminate with ϵ; thus our rewrite system is confluent on W.

Hence, by Proposition 5.2, we immediately deduce that $W \in \mathbf{DSPACE}(\mathcal{F})$ as required. □

Persisting with our notation, we note the following results:

Proposition 6.2. *If $u \in A^*$, then $u \in J$ if and only if $a_1 u \dots a_{12} u \in W$.*

Proof. If $u \in J$ then, by definition of R, we have that

$$a_1 u \dots a_{12} u \in R \subseteq W.$$

Conversely, suppose that $v \equiv a_1 u \dots a_{12} u \in W$. From Proposition 5.4, we note that we can reduce any word in W to ϵ by a sequence of moves where we either remove trivial subwords of the form xx^{-1} and $x^{-1}x$ to freely reduce v, or else we replace a subword y by z^{-1} where $|z| < |y|$ and $yz \in R$. The important point is that, when we perform a replacement like this, we must have $|y| > \frac{1}{2}|yz|$. Hence, if v is reduced, then it must contain at least half of some element of R, which immediately forces $u \in J$. □

Proposition 6.3. *Suppose there is an algorithm to decide membership of W in space bound $g(n)$; then there is an algorithm to decide membership of J in space bound $g(12n + 12)$.*

Proof. This follows immediately from the previous result, since to check whether or not a word u of length n lies in J, we simply test the word $a_1 u a_2 u \dots a_{12} u$ for membership of W, and this word is of length $12n + 12$. □

The usefulness of our construction becomes more apparent when one considers hierarchies of complexity functions and classes of languages solvable within some space bound but not another.

Theorem 6.4. *Suppose there exists a language J contained in $\mathbf{DSPACE}(\mathcal{G})$ but not contained in $\mathbf{DSPACE}(\mathcal{F})$, where $\mathcal{F}$ and $\mathcal{G}$ satisfy Property 2.1. Then there exists a group whose word problem is contained in $\mathbf{DSPACE}(\mathcal{G})$ but not contained in $\mathbf{DSPACE}(\mathcal{F})$.*

Proof. We can follow the construction above to construct a group with word problem solvable in $\mathbf{DSPACE}(\mathcal{G})$. Then, from Proposition 6.3, this group cannot have word problem lying in $\mathbf{DSPACE}(\mathcal{F})$, otherwise J would be decidable in this space bound too. □

As a result we se that, if there is a hierarchy of space complexity classes, then we have a similar hierarchy of groups. For example, if $\mathcal{F}$ is the class of functions of the form $an + b$ and if $\mathcal{G}$ is the class of functions of the form $an^2 + bn + c$, then

it is well known that $\mathbf{DSPACE}(\mathcal{G})$ strictly contains $\mathbf{DSPACE}(\mathcal{F})$ (see [20] for example), and hence, by Theorem 6.4, there exists a group whose word problem is decidable in deterministic quadratic space but not in deterministic linear space. Indeed, by the Space Hierarchy Theorem, $\mathbf{DSPACE}(f(n))$ is a proper subset of $\mathbf{DSPACE}(f(n)\log(f(n)))$ for any proper complexity function $f(n)$; so (for example) $\mathbf{DSPACE}(\mathcal{H})$ strictly contains $\mathbf{DSPACE}(\mathcal{F})$ where $\mathcal{F}$ is (as above) the class of functions of the form $an + b$ and $\mathcal{H}$ is the class of functions of the form $(an + b)\log(an + b)$.

In an entirely analagous way, we can prove;

Theorem 6.5. *Suppose there exists a language J contained in* $\mathbf{NSPACE}(\mathcal{G})$ *but not contained in* $\mathbf{NSPACE}(\mathcal{F})$*, where $\mathcal{F}$ and $\mathcal{G}$ satisfy Property* 2.1*. Then there exists a group whose word problem is contained in* $\mathbf{NSPACE}(\mathcal{G})$ *but not contained in* $\mathbf{NSPACE}(\mathcal{F})$*.*

Acknowledgements. The authors would like to thank Volodya Shavrukov for helpful conversations related to the work presented in this paper. The constructive comments of the referee (including the suggestion to say something about the decidability question for a context-free language to be the word problem of a group) were appreciated and helped us to improve the presentation of the paper. The authors would also like to thank Darja Chebotareva and Hilary Craig for all their help and encouragement.

References

[1] J. M Alonso, T. Brady, D. Cooper, V. Ferlini, M. Lustig, M. Michalik, M. Shapiro, H. Short: Notes on word hyperbolic groups, in: Group Theory from a Geometric Viewpoint (Trieste, 1990), E. Ghys, A. Haefliger, A. Verjovsky (eds.), World Scientific, Singapore (1991) 3–63.

[2] J. M. Autebert, L. Boasson, G. Sénizergues: Groups and NTS languages, J. Comput. Syst. Sci. 35 (1987) 243–267.

[3] R. V. Book, F. Otto: String Rewriting Systems, Springer, New York (1993).

[4] W. W. Boone: The word problem, Ann. Math. 70 (1959) 207–265.

[5] V. Diekert: Investigations on Hotz groups for arbitrary grammars, Acta Inf. 22 (1986) 679–698.

[6] C. Frougny, J. Sakarovitch, E. Valkema: On the Hotz group of a context-free grammar, Acta Inf. 18 (1982) 109–115.

[7] R. H. Gilman: Formal languages and infinite groups, in: Geometric and Computational Perspectives on Infinite Groups (Minneapolis, 1994; Princeton, 1994), G. Baumslag et al. (ed.), DIMACS, Ser. Discrete Math. Theor. Comput. Sci. 25, American Mathematical Society, Providence (1996) 27–51.

[8] R. H. Gilman: Formal languages and their application to combinatorial group theory, in: Groups, Languages, Algorithms (Baltimore, 2003), A. V. Borovik (ed.), Contemporary Mathematics 378, American Mathematical Society, Providence (2005) 1–36.

[9] T. Herbst: On a subclass of context-free groups, RAIRO, Inform. Théor. Appl. 25 (1991) 255–272.

[10] T. Herbst, R. M. Thomas: Group presentations, formal languages and characterizations of one-counter groups, Theor. Comput. Sci. 112 (1993) 187–213.

[11] J. E. Hopcroft, J. D. Ullman: Introduction to Automata Theory, Languages, and Computation, Addison-Wesley, Reading (1979).

[12] G. Hotz: Eine neue Invariante für kontextfreie Sprachen, Theor. Comput. Sci. 11 (1980) 107–116.

[13] D. L. Johnson: Presentations of Groups, 2nd Ed., London Math. Soc. Student Texts 15, Cambridge University Press, Cambridge (1997).

[14] S. Krstić: Actions of finite groups on graphs and related automorphisms of free groups, J. Algebra 124 (1989) 119–138.

[15] R. C. Lyndon, P. E. Schupp: Combinatorial Group Theory, Ergebnisse der Mathematik und ihrer Grenzgebiete 89, Springer, Berlin (1977).

[16] K. Madlener, F. Otto: About the descriptive power of certain classes of finite string-rewriting systems, Theor. Comput. Sci. 67 (1989) 143–172.

[17] D. E. Muller, P. E. Schupp: Groups, the theory of ends, and context-free languages, J. Comput. Syst. Sci. 26 (1983) 295–310.

[18] D. E. Muller, P. E. Schupp: The theory of ends, pushdown automata and second-order logic, Theor. Comput. Sci. 37 (1985) 51–75.

[19] P. S. Novikov: On the algorithmic unsolvability of the word problem in group theory, Tr. Mat. Inst. Steklova 44 (1955).

[20] C. H. Papadimitriou: Computational Complexity, Addison-Wesley, Amsterdam (1995).

[21] D. W. Parkes, V. Yu. Shavrukov, R. M. Thomas: Monoid presentations of groups by finite special string-rewriting systems, RAIRO, Inform. Théor. Appl. 38 (2004) 245–256.

[22] D. W. Parkes, R. M. Thomas: Groups with context-free reduced word problem, Commun. Algebra 30 (2002) 3143–3156.

[23] I. A. Stewart, R. M. Thomas: Formal languages and the word problem for groups, in: C. M. Campbell et al. (ed.), Groups St. Andrews 1997 in Bath (Bath, 1997), Vol. 2, Lond. Math. Soc. Lect. Note Ser. 261, Cambridge University Press, Cambridge (1999) 689–700.