Solving the word problem in real time

Derek F. Holt and Sarah Rees
Mathematics Institute, University of Warwick,
Coventry CV4 7AL, UK. dfh@maths.warwick.ac.uk
Department of Mathematics, University of Newcastle,
Newcastle NE1 7RU, UK. Sarah.Rees@ncl.ac.uk

11th January 2000

Abstract

This paper is devoted to the study of groups whose word problem can be solved by a Turing machine which operates in real-time. A recent result of the first author for word hyperbolic groups is extended to prove that under certain conditions the generalised Dehn algorithms of Cannon, Goodman and Shapiro, which clearly run in linear time, can be programmed on real-time Turing machines. It follows that word-hyperbolic groups, finitely generated nilpotent groups and geometrically finite hyperbolic groups all have real-time word problems.

AMS Subject Classification (1991): Primary 20F10, Secondary 68Q45, 20F32.

1 Introduction

The purpose of this paper is to study groups with real-time word problem, that is for which the word problem can be solved with a deterministic, multi-tape Turing machine which reads its input at constant speed and completes the processing of its input as it finishes reading it. Real-time algorithms necessarily run in linear time, but there exist linear time algorithms which cannot be programmed on a real-time Turing machine.

Word hyperbolic groups are well known to have a linear time solution to the word problem (due to Dehn [3]) and a result [6] by the first author shows that the Dehn algorithm can be programmed on a real-time Turing machine. A recent article [2] generalises Dehn's algorithm to define a wider class of linear time algorithms, and constructs generalised Dehn algorithms for finitely generated nilpotent groups and geometrically finite hyperbolic groups. In this paper we prove that, under certain conditions, the generalised Dehn algorithms of [2] can also be run on real-time Turing machines. We show that these conditions hold in word hyperbolic groups, finitely generated nilpotent groups and geometrically finite hyperbolic groups, and hence that all these groups have real-time word problems.

The terminology of the paper is explained in section 2. A few elementary results concerning the closure properties of the family of groups with real-time word problem are proved in section 3. In section 4, we show how a generalised Dehn algorithm may sometimes be translated onto a real-time Turing machine, while sections 5 and 6 are devoted to finitely generated nilpotent and geometrically finite hyperbolic groups.

2 Terminology

We assume throughout that all group generating sets are closed under inversion. Let G be a group with finite generating set X. A solution to the *word problem* for G over X is an algorithm which, given any word w over X as input, terminates either with the output $w =_G 1$ or the output $w \neq_G 1$.

Generalised Dehn algorithms to solve the word problem are defined in [2]. Such an algorithm R is defined by a finite set S of length reducing rewrite rules together with a set of instructions governing the order in which the rules should be applied. Each rule in S is of the form $u \to \rho(u)$, where both u and $\rho(u)$ are words over a finite alphabet A which includes X (possibly strictly), and $\rho(u)$ is shorter than u. At most one rule has left hand side u. Wherever a word w is to be rewritten which contains several left hand sides of rules as subwords, the rule which is applied by the algorithm is one which ends closest to the start of w; if there are several such, of those the rule with the longest left hand side is selected. (A rewriting algorithm with this property is called *incremental* in [2].) Any word which contains a left hand side (that is, is not (R)-*reduced*) is rewritten by the algorithm; the reduced word to which a word w is rewritten after a finite number of steps is denoted $R(w)$. It is required that, for any $w \in X^*$,

$$R(w) = \epsilon \iff w =_G 1$$

(where ϵ denotes the empty word). The incremental condition implies that $R(uv) = R(R(u)v))$, and hence that if $R(x) = R(y)$, for $x, y \in X^*$, then x and y represent the same element of G. Generalised Dehn algorithms run in linear time.

Word hyperbolic groups certainly have generalised Dehn algorithms; the standard Dehn algorithm is simply a generalised Dehn algorithm for which $A = X$. It is shown in [2] that nilpotent groups and geometrically finite hyperbolic groups also have generalised Dehn algorithms.

In this article, we are interested in groups G for which an algorithm can be found which solves the word problem for X on a real-time Turing machine. We say that such a group G has *real-time word problem* over X. We define a *real-time Turing machine* to be a deterministic Turing machine with finitely many doubly-infinite tapes, which completes the processing of its input as it finishes reading it. A single 'move' of the machine consists of a single operation on each of those tapes. During such an operation, the read-write head on a tape reads the symbol it is currently pointing at, may delete or write over that symbol, and in general may stay put, or move to the position either directly to the left or directly to the right of its current position. One tape is reserved for the input word, which is written from left to right with consecutive symbols in consecutive positions on the tape; the head pointing at that tape points initially at the leftmost symbol of the input, and always moves to the right after reading an input symbol.

In fact, we may slightly generalise the above definition to allow a real-time Turing machine to do a bounded number of operations on any one of its tapes other than the input tape for each input symbol read. Given such a machine, a Turing machine for which a move consists of a single operation on each tape can be constructed from it by increasing the number of tapes. Where convenient, we shall use this slightly more general definition below without further comment.

We have chosen not to allow our Turing machines to shift between a finite number of states, but rather to operate purely with tapes; the use of states can anyway be mimicked through the use of an additional tape.

We shall also be interested in a subclass of real-time algorithms, which we shall call *tidy real-time algorithms*, and which are solved on *tidy real-time Turing machines*. We define such a machine to be a Turing machine as above, but with a finite number of semi-infinite (rather than doubly-infinite) tapes, whose heads remain always at the rightmost end of the contents of the tapes (so that the tapes are basically stacks), and which halts with all its tapes other than the input tape empty when it accepts an input word.

It is well known that any Turing machine is equivalent to a machine with two stacks; a machine with $2n$ stacks is constructed from one with n tapes by splitting each tape into two portions at the tape-head. So the only restriction we have added above is that the tapes/stacks can be assumed empty whenever an input word is accepted. It is not clear to us how much of a restriction this is. We note that all the examples we know of groups with real-time word problem actually have word problem in tidy real-time. Further, we can prove that the family of tidy real-time groups is closed under free products, but we cannot prove that the same is true of real-time groups.

3 Closure operations

In this section we note a few basic properties of the family of groups with real-time word problem.

First we observe that having real-time or tidy real-time word problem is really a property of the group alone, and does not depend on the finite generating set.

Lemma 3.1 *If X and Y are two different finite generating sets for a finitely generated group G, then if G has real-time word problem over X it has real-time word problem over Y. Similarly, if G has tidy real-time word problem over X, it has tidy real-time word problem over Y.*

PROOF: These are special cases of a result in [5], namely that the above holds for any family $\mathcal{F}$ of formal languages which is closed under inverse homomorphism (i.e. for which, whenever $f : X_1^* \to X_2^*$ is a monoid homomorphism, and $L \subseteq X_2^*$, then if L is in $\mathcal{F}$, $f^{-1}(L)$ must also be in $\mathcal{F}$). That the family of real-time languages is closed under inverse homomorphisms is well known (see for instance [8]). A real-time Turing machine to accept a language L can be adapted to accept $f^{-1}(L)$ by using one extra tape to convert an input string $w = x_1 \ldots x_n$ over X symbol by symbol to the string $f(w)$, which is then read symbol by symbol into the Turing machine for L. (More precisely, the Turing machine for L reads and deletes the symbols of the converted string as they appear on the extra tape.) It is clear that if the machine to accept L is tidy, then the same is true of some machine which accepts $f^{-1}(L)$. Hence the family of tidy real-time languages is also closed under inverse homomorphisms. □

We define $\mathcal{RT}$ to be the family of groups with real-time word problem, and $\mathcal{TRT}$ to be the family of groups with tidy real-time word problem.

We note that

Proposition 3.2 **(a)** *Any finitely generated subgroup H of a group G in $\mathcal{RT}$ (resp. $\mathcal{TRT}$) is in $\mathcal{RT}$ (resp. $\mathcal{TRT}$).*

(b) *Any group K in which a group G in $\mathcal{RT}$ (resp. $\mathcal{TRT}$) has finite index is in $\mathcal{RT}$ (resp. $\mathcal{TRT}$).*

(c) *If N is a finite normal subgroup of a group G in $\mathcal{RT}$ (resp. $\mathcal{TRT}$), and $Q = G/N$, then Q is in $\mathcal{RT}$ (resp. $\mathcal{TRT}$).*

(d) *A direct product of two groups in $\mathcal{RT}$ (resp. $\mathcal{TRT}$) is in $\mathcal{RT}$ (resp. $\mathcal{TRT}$).*

(e) *A free product of two groups in $\mathcal{TRT}$ is in $\mathcal{TRT}$.*

PROOF: To prove (a), note that if X is a generating set for G and Y one for H, then $X \cup Y$ is also a generating set for G. Hence the word problem for H over Y can be solved using the algorithm for G over $X \cup Y$.

To prove (b), choose a generating set X for K, let $T = \{t_1 = 1, t_2, \dots t_k\}$ be a finite transversal for G in K, and let Y be a set of symbols representing the full set of Schreier generators (i.e. all elements in G of the form txt'^{-1}, for $x \in X$, $t, t' \in T$). We can solve the word problem for K by adding one tape to the real-time Turing machine which solves the word problem for G. Using the extra tape, a word w over X is rewritten symbol by symbol (by applying rules of the form $tx = yt'$) to a word $w't$, where w' is a word over Y of the same length as w, and $t \in T$. At the same time, w is read (and deleted) as input by the real-time Turing machine that solves the word problem for G over Y. The word w represents the identity if and only if both w' does and $t = 1$.

To prove (c), let X be a generating set for G, and Y the image of that set under the natural homomorphism from G to Q. Suppose that we have a real-time Turing machine which solves the word problem for G. We simply use $|N|$ copies of that machine to solve the word problem for Q. We can lift any word w over Y to a word w' of the same length over X which maps to it, and express each of the elements of N as a word n_i over X. Then we solve the word problem for each word $w'n_i$ separately. The word w represents the identity in Q precisely if one such $w'n_i$ represents the identity in G.

To prove (d), we assume that we have disjoint generating sets for the two factors. We solve the word problem for the direct product by combining the two real-time Turing machines which solve the word problem for the direct factors. Let $w = w_{11}w_{21}w_{12}w_{22}\dots w_{1k}w_{2k}$ be an input word, where for each i, j, w_{ij} is a subword written over the generators of the i-th factor. The combined machine passes an input symbol to the first machine if it is a generator for the first factor, and otherwise to the second Turing machine. Hence the first Turing machine receives the word $w_{11}w_{12}\dots w_{1k}$ as input and the second Turing machine the word $w_{21}\dots w_{2k}$. The combined machine reports that a word w represents the identity if the machines for both factors report that the portions they process represent the identity.

To prove (e), again we assume that we have disjoint generating sets for the two factors, and solve the word problem for the free product by combining the two tidy real-time Turing machines for the two free factors. But here the situation is a little more complicated than for the direct product. Let $w = w_{11}w_{21}w_{12}w_{22}\ldots w_{1k}w_{2k}$ be an input word, where for each i, j, w_{ij} is a subword written over the generators for the i-th factor. Then clearly w represents the identity precisely if at least one factor w_{ij} represents the identity and further the word formed from w by deleting w_{ij} represents the identity. Since this can arise without each individual factor representing the identity, by cancellation between factors either side of an identity factor, we need the combined Turing machine to be in the same configuration after reading an identity factor w_{ij} as immediately before. Provided the component Turing machines are tidy, this is ensured. □

Remark 3.3. By a similar argument to the proof of part (c) of the above theorem, we can show that if H is any finite subgroup of a group G in $\mathcal{RT}$ or $\mathcal{TRT}$, then we can construct a real-time Turing machine (tidy in the second case) that determines whether a word in the generators of G represents an element of H and, if so, which element. By using such machines we can generalise part (e) of the theorem and prove that a free product with finite amalgamated subgroups of two groups in $\mathcal{TRT}$ is in $\mathcal{TRT}$. Similarly, it can be proved that if the group G in $\mathcal{TRT}$ has two isomorphic finite subgroups with associated isomorphism f, then the HNN-extension $G*_f$ is in $\mathcal{TRT}$. We omit the details of these proofs.

Remark 3.4. We have not been able to prove the converse of part (c) of the above theorem - does Q in $\mathcal{RT}$ imply G in $\mathcal{RT}$? This can however be proved when G is a split extension of N by Q.

4 A tidy real-time translation of a generalised Dehn algorithm

Below we describe how a tidy real-time Turing machine may be constructed to run a generalised Dehn algorithm R provided that a certain relationship holds between the length of the reduction $R(w)$ of a word w in the group generators and the geodesic length of w (that is, the length of the shortest word in the group generators which is equal to it in the group).

We shall call the non-zero function $f : \mathbb{N} \to \mathbb{N}$ such that $f(n)$ is the minimum geodesic length of any word $w \in X^*$ for which $R(w)$ has length n the *geodesic lower bound* for the Dehn algorithm R.

This section is devoted to the proof of the following.

Theorem 4.1 *Let G be a group with a generalised Dehn algorithm R over X. Suppose that R has geodesic lower bound f which is linear or above. Then G has a generalised Dehn algorithm over X which can be run on a tidy real-time Turing machine.*

The basic difficulty in translating the algorithm R onto a real-time Turing machine is in dealing with backtracking. Any time a reduction rule is applied to reduce a word, it is necessary afterwards to rescan a portion of the word starting just before the point at which the substitution took place, since a reducible subword may have been introduced which consists of some symbols before the substitution point and some of the substituted symbols. A single reduction may provoke several backtracks. As we backtrack we need to stack the backtracked symbols, and to continue to read from the input tape while we process the stacked symbols. We need to control the speed at which we process stacked input in relation to our reading speed so that we do not run out of symbols to read from the input tape prematurely during processing. The function f controls the total extent of backtracking, in relation to unread input from a word representing the identity, and so it is the properties of f which allow us to relate our processing and reading speeds so that the algorithm works in real-time.

Proof: Let L be the maximal length of a left hand side of a rule in R.

We construct a Turing machine with 5 tapes to run the algorithm. One of these is the input tape, containing the input word w. In addition to the input tape we have 4 work tapes, all 4 of which are empty at the start. Tape 1 is used for output. At any stage (except at the point immediately after a symbol has been read which provokes a reduction), the word on tape 1 is reduced according to R. Tape 2 is used for stacking strings which are taken off tape 1 when backtracking is necessary. Tapes 3 and 4 are used for queueing unprocessed input while strings stored on tape 2 are being processed. We need two such tapes because we need to be able to continue to read and queue from the input tape while previously queued input is being transferred onto tape 2 for processing.

We assume that we can do a bounded number, k, of operations on each of the 4 tapes for each symbol read from the input tape; to model this on a machine where only one operation is allowed on each tape per symbol read we simply need to increase the number of work tapes.

We assume further that we can always see the last L (or fewer) symbols on tape 1, in order to see if that string contains the left hand side of a reduction rule as a suffix. (We can mimic this on a Turing machine in which only one symbol can be seen through the use of additional tapes.)

The algorithm alternates between two stages. Initially it is in stage 1; during this stage symbols are read directly from the input tape onto tape 1. When a reduction forces backtracking, symbols from tape 1 need to be stacked, and the algorithm enters stage 2, during which stacked symbols are processed and symbols from the input tape are queued. The algorithm returns to stage 1 when all the stacked and queued symbols have been processed.

More precisely, we describe the algorithm as follows:-

Stage 1 Read symbols g one by one from the input tape onto tape 1 until either

(a) a left hand side ug is seen at the right hand end of tape 1 or

(b) the end of the input is reached.

If case (a) if the left hand side is just the single symbol g (reducing to the trivial word), or if the left hand side occupies the whole of tape 1, delete ug from tape 1, replace it by the right hand side $\rho(ug)$ of the rule, and continue reading from the input tape. Otherwise, delete ug from tape 1, write the reverse of $\rho(ug)$ onto tape 2, designate tape 3 as the queue and enter stage 2.

If case (b), return $w =_G 1$ if tape 1 is empty, and $w \neq_G 1$ otherwise.

Stage 2 Read symbols g one by one from the tape 2 onto tape 1. At the same time read from the input tape onto the queue. Read k symbols from tape 2 for every symbol read from the input onto the queue. Continue in this way until either

(a) a left hand side ug is seen at the right hand end of tape 1 or

(b) the end of tape 2 is reached, but there are still items on the queue or

(c) the end of tape 2 is reached and there are no items on the queue or

(d) the end of the input is reached.

In case (a) if the left hand side is just the single symbol g, or if the left hand side occupies the whole of tape 1, delete ug from tape 1 and replace it by $\rho(ug)$. Otherwise delete ug from tape 1 and write the reverse of $\rho(ug)$ onto tape 2. Then continue reading from tape 2.

In case (b), transfer the queued input (reversed) onto tape 2, while continuing to read slowly from the input onto the fourth of the tapes, which now becomes the queue. If the last symbol is read from the input tape before this process is complete, return $w \neq_G 1$. Otherwise continue reading from tape 2.

In case (c), return to stage 1.

In case (d), return $w \neq_G 1$.

What can cause this algorithm to fail to work? This algorithm will give a wrong answer if, for some input word w with $w =_G 1$, it runs out of symbols to read from the input tape while there is still unprocessed data on one of tapes 2,3 or 4, i.e. during stage 2. We need to verify that the condition that f is at least linear prevents this happening with a suitably large k.

We divide into a number of phases each period of time spent in stage 2. The first phase starts just before stage 2 is entered from stage 1 (as the substitution is made which causes stage 2 to be entered), and ends when tape 2 is next empty. Each subsequent phase starts with tape 2 empty and finishes when it is next empty. At the end of the last phase, we expect the algorithm to either terminate with a correct answer or return to stage 1.

We suppose that immediately before reading the symbol g that provokes the first substitution, the reduced word w_0 is on tape 1. We let l_0 be the geodesic length of w_0. Then $l_0 \geq f(|w_0|)$.

We define s_i to be the total number of substitutions made during the i-th phase (including those which do not provoke a backtrack), and r_i to be the number of symbols read from the input during that period. We define r_0 to be 1 since just before the beginning of the first phase a single symbol g is adjoined to tape 1. We note that s_1 includes the very first substitution of phase 1 which provokes entry into stage 2. Observe that since all substitutions are length reducing, the total number of substitutions made during the first i phases must be at most the length of the total word being reduced by those substitutions. The word being reduced is formed by appending $r_0 + \ldots r_{i-1}$ generators onto the end of w_0. Hence

$$s_1 + \ldots + s_i \leq |w_0| + r_0 + \ldots + r_{i-1}. \tag{1}$$

We avoid running out of input during or at the end of phase i with some $w =_G 1$ provided that

$$r_0 + \ldots + r_i < l_0, \quad \forall i \geq 0. \quad (*)$$

Our aim is to show that provided that f is linear or above we can select k so that $(*)$ holds.

First observe that, for some constant c, we can bound by $cs_i + 2r_{i-1}$ the number of moves during phase i on any one of tape 1, tape 2 and whichever of tapes 3 and 4 we are currently reading from (the 'read-queue'). For tape 2, we observe that r_{i-1} symbols are written onto the tape (after being read from the read-queue), and each of these subsequently read off the tape again, while in addition at most s_i substitutions are made on that tape, each at the cost of a bounded number (dependent on L) of operations on the tape, giving a bound of precisely the above form. For tape 1, again, each of the s_i substitutions has a bounded cost, and during phase i at most r_{i-1} symbols are written one by one to the tape. Finally, during the phase, r_{i-1} symbols are read off the read-queue. Hence on all three tapes the number of moves is bounded by $cs_i + 2r_{i-1}$. Since k operations on each of those tapes are performed for each of the r_i symbols read from the input, it follows that

$$kr_i \leq cs_i + 2r_{i-1}.$$

Summing this gives

$$k(r_1 + \ldots + r_i) \leq c(s_1 + \ldots + s_i) + 2(r_0 + \ldots + r_{i-1}). \qquad (2)$$

We shall now prove by induction on i that $(*)$ holds, provided that f is at least linear.

Note that the case $i = 0$ holds trivially since $r_0 = 1$ and we never enter into stage 2 with w_0 of length 0 or 1. So now suppose that for some i,

$$r_0 + \ldots + r_{i-1} < l_0.$$

Then, combining this with the inequalities (1) and (2), we deduce that

$$r_1 + \ldots + r_i < \frac{c}{k}(|w_0| + l_0) + \frac{2}{k}l_0.$$

Hence the condition

$$r_0 + \ldots + r_i < l_0$$

is implied provided that

$$1 + \frac{c}{k}(|w_0| + l_0) + \frac{2}{k}l_0 \leq l_0$$

or equivalently

$$|w_0| \leq \frac{(k-2-c)l_0 - k}{c}. \quad (\dagger)$$

So provided $(\dagger)$ holds, induction now shows that we have the condition $(*)$ that we need. Since $l_0 \geq f(|w_0|)$ the condition $(\dagger)$ is implied by the condition

$$n + \frac{k}{c} \leq \frac{k-2-c}{c}f(n). \quad (\dagger\dagger)$$

For $(\dagger\dagger)$ to hold we need f to be linear or superlinear. Suppose then that $f(n) \geq n/d$, for some integer d. Then k can certainly easily be chosen to make (††) hold for larger values of n, but smaller values of n (up to d) can cause problems, since for those values of n we could have $f(n) = 1$.

Since we never enter stage 2 with $|w_0| = 1$, we need not consider the case $n = 1$. We can exclude the possibility that $2 \leq |w_0| \leq d$ by adding a rule to R of the form $u \to h$ for each word $u = R(w)$ of length at most d for which $w \in X^*$ has geodesic length 1. This has the effect of modifying f so that $f(n) > 1$ for all $n > 1$. With the modified rewrite system and new f, it is easy to find a constant k to make (††) hold for all $n > 1$, and hence to ensure that (†) holds.

In this case, we have described a tidy real-time algorithm. □

It follows as a simple corollary from the above result that word hyperbolic groups have tidy real-time word problem. For a finite collection of length reducing rules can be used to reduce any word to one for which any subword of bounded length is a geodesic. In a word hyperbolic group such a word is quasigeodesic, and hence at most a constant factor longer than an actual geodesic. Hence such a set of rules gives a generalised Dehn algorithm with linear geodesic lower bound f. This result is proved in [6].

The remainder of the paper will be devoted to proving that finitely generated nilpotent groups and geometrically finite hyperbolic groups have tidy real-time word problem. We shall deduce each result as a consequence of the above, by showing that an appropriate function f is associated with the generalised Dehn algorithm described in [2].

5 Nilpotent and virtually nilpotent groups

It is proved in [2, Theorem 4.8] that any finitely generated virtually nilpotent group has a generalised Dehn algorithm. This is seen by embedding the torsion free nilpotent subgroup of finite index in the group $U_N(\mathbb{Z})$ of upper unitriangular integer matrices, and applying a generalised Dehn's algorithm for that group, and then using the fact, proved in [2, Theorem 2.10] that a group of which a finite index subgroup has a generalised Dehn's algorithm has one itself.

In order to explore the existence of an appropriate geodesic lower bound f, we need to give some of the details of the algorithm in [2] for $G = U_N(\mathbb{Z})$. We suppose that a finite generating set X has been fixed. The algorithm exploits the existence of an *expanding homomorphism* for $G = U_N(\mathbb{Z})$, defined to be an endomorphism ϕ from $U_N(\mathbb{Z})$ onto a finite index subgroup for which $l_X(\phi(g)) \geq M l_X(g)$, for all $g \in G$, and some fixed M (where $l_X(g)$ is defined to be the geodesic length of g over X). It is shown in [2] that $U_N(\mathbb{Z})$ has such an endomorphism for any $M > 0$. We define $A = X \cup \{\tau, \tau^{-1}\}$, write $\phi(g)$ as $\tau g \tau^{-1}$ and so use ϕ to give a shorter expression for elements in $\phi(G)$, or indeed in $\phi^k(G)$, than is given by words over X. Using length reducing rewrite rules, any word over X can be reduced to one of the form

$$\tau^m g_m \tau^{-1} \ldots \tau^{-1} g_1 \tau^{-1} g_0,$$

where the words g_i lie in a fixed transversal of $\phi(G)$ in G and therefore have bounded length, and g_m is non-trivial for $m > 0$. It is proved in [2, Lemma 3.4] that the geodesic length of a non-trivial word of the above type is bounded below by $(M/3)^m$. From this we can easily deduce that the geodesic lower bound f is bounded below by a function of the form Ca^n, for constants C, a with $a > 1$, because the length n of the reduced word is at most a constant multiple of m.

The generalised Dehn algorithm of $U_N(\mathbb{Z})$ and associated exponential function f are certainly inherited by any finitely generated subgroup of $U_N(\mathbb{Z})$, and hence by any torsion-free finitely generated nilpotent group.

The above and the results of section 3 give us immediately the following consequence of theorem 4.1.

Theorem 5.1 *Any finitely generated nilpotent (or even virtually nilpotent) group has tidy real-time word problem.*

Looking ahead, however, we need a little bit more. For the purposes of the next section on geometrically finite hyperbolic groups we need to prove the following slightly stronger result about the generalised Dehn algorithm for virtually nilpotent groups, from which the above theorem can also be deduced.

Lemma 5.2 *If G is any finitely generated virtually nilpotent group, then G has a generalised Dehn algorithm whose associated geodesic lower bound is at least exponential.*

PROOF: Let G be virtually nilpotent, and H a torsion-free nilpotent subgroup of G of finite index, embedding in $U = U_N(\mathbb{Z})$, for some N. Let $\{1\} \cup T$ be a transversal for H in G, let X be a generating set for G and let $X' = X \cup T$. Let Y be a generating set for H that contains all non-identity elements of H of the form $xx'x''$, $xx'x''t^{-1}$, xx', or $xx't^{-1}$ for $x, x', x'' \in X \cup T$ and $t \in T$. Let Z be a generating set for U which contains Y. Let f_U be the geodesic lower bound associated with the generalised Dehn algorithm R_U for U over Z described in [2]; then f_U is at least exponential.

H inherits a generalised Dehn algorithm over Y from U simply by embedding as a subgroup, since any word over Y is also a word over Z. We call that inherited algorithm R_H, and the associated function f_H. Then, like f_U, f_H has an exponential lower bound. For if w is a word over Y, reducing using R_U (and hence R_H) to w' of length n, then the geodesic length of w over Z is at least $f_U(n)$, and hence so is the geodesic length of w over the subset Y of Z.

A generalised Dehn algorithm R_G for G over X' is described in [2, Theorem 2.10]. Since X is already a generating set for G, we can equally well regard that same algorithm as a generalised Dehn algorithm for G over X. We define S to be the set of all valid rules with left hand side $xx'x''$ and right hand side one of y, yt, t or ϵ, together with all valid rules of the form $xx' \to y$, $xx' \to \epsilon$ or $x \to \epsilon$, for $x, x', x'' \in X \cup T$, $y \in Y$, $t \in T$. Certainly S is length reducing. Any word $w = x_1x_2 \ldots x_r$ over X rewrites using S to a word of the form uv, where $u \in Y^*$, and v is a word of length at most 2, either trivial, a single element of either X or T, or a product of an element of either X or T by an element of X. If w represents the identity then it must reduce to a word over Y. We construct a generalised Dehn algorithm R_G for G over X by combining the rules of S with the rules of the generalised Dehn algorithm R_H for H.

We want now to compute the geodesic lower bound, f_G, for R_G.

The word $w = x_1 \ldots x_r$ over X above is reduced by R_G to a word $w' = u'v$, where u' is the reduction according to R_H of the word u over Y, and v is as above. So

$$|u'| \leq |w'| \leq |u'| + 2.$$

The geodesic length of u over Y is then at least $f_H(|u'|)$. Now let $x'_1 \ldots x'_q$ be a geodesic word over X representing w. Rearranging the equation $uv =_G x'_1 \ldots x'_q$ gives an equation $ut =_G x'_1 \ldots x'_q v'$, where $t \in T \cup \{1\}$, and v' is a word over X of length 0,1 or 2. Applying Reidemeister-Schreier rewriting to the right hand side of that equation yields a product $u''t'$, where u'' is a word of length at most $q+2$ over Y, and $t' \in T \cup \{1\}$. We deduce that $t = t'$ and $u =_H u''$. Hence u'' has length at least $f_H(|u'|)$, and so $q+2 \geq f_H(|u'|)$, $q \geq f_H(|u'|) - 2$.

Now we know that $f_H(n)$ is at least exponential. So suppose that $f_H(n) \geq Ca^n$ for all n, and positive constants C, a with $a > 1$. Then

$$q \geq f_H(|u'|) - 2 \geq Ca^{|u'|} - 2 \geq Ca^{(|w'|-2)} - 2 = \frac{C}{a^2}a^{|w'|} - 2.$$

Hence we can find a positive constant A such that

$$f_G(n) \geq Aa^n$$

i.e. we have an exponential lower bound on $f_G(n)$. (Here, in order to deal with small values of n, we need to use the fact that $f_G(n) > 0$ for all $n > 0$, which is a consequence of the fact that only words over X representing the identity are reduced by R to the trivial word.) □

6 Geometrically finite groups

This section is devoted to the proof of

Theorem 6.1 *Geometrically finite hyperbolic groups have real-time word problem.*

We use the definition of a geometrically finite hyperbolic group found in [7], also used in [4, 2]. Such a group G acts as a discrete group of isometries on n-dimensional hyperbolic space $\mathbb{H}^n$, for some n, and satisfies certain extra conditions. The fundamental groups of finite volume hyperbolic manifolds or orbifolds are examples, more generally so is any G whose action admits a convex, finite sided polyhedral fundamental domain. A precise definition can be given in terms of the maximal parabolic subgroups of G (each of which consists of all elements fixing a particular point on the boundary of $\mathbb{H}^n$, and with no other fixed point) and the limit set $\Lambda(G)$ of G (the set of accumulation points on the boundary of an orbit of the action of G on $\mathbb{H}^n$). A maximal parabolic subgroup is virtually abelian and fixes a family of spheres of infinite radius, centred on the fixed boundary point, known as horospheres. G is defined to be geometrically finite if a disjoint set $\mathcal{B}$ of G-equivariant horospheres can be found, such that G acts cocompactly on the intersection of $\mathbb{H}^n \setminus \cup_{B\in\mathcal{B}} B$ with the convex hull of $\Lambda(G)$; we call this intersection X. There are many equivalent definitions of geometrical finiteness (examined by Bowditch in [1]). The one above is convenient for us to work with since we shall need to examine the action of G on X, and the embedding of the Cayley graph of G in X.

Theorem 6.1 will follow immediately from theorem 4.1 once we have proved the following lemma.

Lemma 6.2 *If G is any finitely generated geometrically finite hyperbolic group, then G has a generalised Dehn algorithm with an associated geodesic lower bound which is at least linear.*

We shall use the generalised Dehn algorithm described in [2], and need only prove that it has an appropriate geodesic lower bound.

Before giving details of that generalised Dehn algorithm, we give some more details of the space X on which G acts. This description is based on one in [2], and our notation is taken from there. Further details are given in [7]. The Cayley graph Γ of G embeds in X quasi-isometrically, via a $(\lambda, 0)$-quasi-isometry, for some λ (that is, such that the embedding distorts the length of any path in Γ by a factor between λ and $1/\lambda$).

X inherits the hyperbolic metric from $\mathbb{H}^n$ locally on its interior. Our proof of the theorem will hinge on a comparison of the lengths of geodesics in X (X*-geodesics*) and hyperbolic geodesics in the larger space $\mathbb{H}^n$. In comparing them we shall need to look also at paths in X which we shall call (as in [2]) *rough geodesics*. We construct the rough geodesic γ_R between two points p and q of X by taking the hyperbolic geodesic γ_H between p and q in the larger space $\mathbb{H}^n$, and then replacing any section of γ_H which passes through a horoball by a shortest path around the boundary of the horoball.

We note here a few technical results about the relationship between distances in $\mathbb{H}^n$ and in X, which will be used in the proof of the theorem.

Lemma 6.3 *If p and p' are points on the surface of a horoball B, the length l_h of the shortest path on the surface of the horoball joining p to p' is related to the length l_H of the hyperbolic geodesic joining p to p' by the rule*

$$\frac{l_h}{2} = \sinh \frac{l_H}{2}.$$

PROOF: Where p_0 is the point of the boundary of $\mathbb{H}^n$ at which B is tangential, both the shortest path from p to p' on the surface of the horoball and the hyperbolic geodesic between p and p' are contained in the (hyperbolic) plane spanned by p, p' and p_0. Hence we may consider this problem within the hyperbolic plane. We use the upper half-space model, and apply Möbius transformations as necessary so that p_0 is transformed to the point at infinity, p to the point $(a, 1)$, p' to the point $(-a, 1)$ and the intersection of B with the plane to the region above the Euclidean line $y = 1$. The distance l_h along the surface of B from p to p' is then the integral of $1/y$ from p to p' along the line $y = 1$, and hence

$$l_h = 2a.$$

The length l_H of the hyperbolic geodesic from p to p' is the integral of $1/y$ along the (Euclidean) circle centred on $(0, 0)$ with (Euclidean) radius $\sqrt{1 + a^2}$. The points of this circle have coordinates (x, y), where $y^2 = 1 + a^2 - x^2$, and x varies between $-a$ and a. Then $2y\frac{dy}{dx} = -2x$, so

$$ds = \sqrt{1 + \left(\frac{dy}{dx}\right)^2} dx = \sqrt{1 + \frac{x^2}{y^2}} dx = \sqrt{1 + a^2}\frac{dx}{y}.$$

Hence

$$\begin{aligned}
l_H &= \int_{x=-a}^{a} \frac{ds}{y} = \int_{-a}^{a} \frac{\sqrt{1+a^2}}{y^2} dx \\
&= \int_{-a}^{a} \frac{\sqrt{1+a^2}}{1+a^2-x^2} dx = \left[\tanh^{-1} \frac{x}{\sqrt{1+a^2}}\right]_{-a}^{a} \\
&= 2\tanh^{-1} \frac{a}{\sqrt{1+a^2}} = 2\sinh^{-1} a = 2\sinh^{-1} \frac{l_h}{2}.
\end{aligned}$$

□

Corollary 6.4 *The distance in X between two points joined by a hyperbolic geodesic of length d is at most*

$$2\sinh \frac{d}{2}.$$

PROOF: The distance in X between two points p and q is at most the length of the rough geodesic between them. Where γ is the hyperbolic geodesic and γ' the rough geodesic, for some set of points $p_0 = p, p_1, \ldots, p_n = q$ on γ, the path γ' is formed by replacing the sections of γ between alternate successive pairs of those points by paths around horoballs. Where $d_i = d_\gamma(p_i, p_{i+1})$, and $d_{i'} = d_{\gamma'}(p_i, p_{i+1})$, we have either $d_{i'} = d_i$ (where the path between p_i and p_{i+1} is a hyperbolic geodesic) or $d_{i'} = 2\sinh d_i/2$ (where the path between p_i and p_{i+1} is on the surface of a horoball). Using the facts that (a) $x \leq \sinh x$ for all $x \geq 0$ and (b) $\sinh(x+y) \geq \sinh x + \sinh y$ for all $x, y \geq 0$, we see that the length of γ', computed as the sum of the $d_{i'}$, is at most $2\sinh(d/2)$. □

Lemma 6.5 *Let B be a horoball, let q be a point inside it, and let γ be a hyperbolic geodesic through q. Suppose that the hyperbolic distance from q to the boundary of B is R. Then the hyperbolic distance along γ to the boundary of B is at most* $\cosh^{-1} e^R$.

PROOF: We use the upper half plane model of $\mathbb{H}^n$, and may assume that B is the horoball with boundary the horizontal line $y = 1$ in that model, γ the hyperbolic geodesic cutting it at $p = (a, 1)$ and $p' = (-a, 1)$ (the Euclidean semi- circle with centre $(0, 0)$ and Euclidean radius $r = \sqrt{1+a^2}$), and q the point $(\sqrt{r^2-b^2}, b)$ (where $a > 0$, and $1 \leq b \leq r$). The nearest point to q on the surface of B is then the point $(\sqrt{r^2-b^2}, 1)$, at hyperbolic distance $\int_1^b \frac{dy}{y} = \log b$ from q. Hence

$$b = e^R.$$

Now the distance from q to p is the integral around the semi-circle

$$\begin{aligned} R' &= \int_{x=\sqrt{r^2-b^2}}^{x=\sqrt{r^2-1}} \frac{ds}{y} = \int_{\sqrt{r^2-b^2}}^{\sqrt{r^2-1}} \frac{r\,dx}{r^2-x^2} \\ &= \left[\tanh^{-1}\frac{x}{r}\right]_{\sqrt{r^2-b^2}}^{\sqrt{r^2-1}} \\ &= \tanh^{-1}\frac{\sqrt{r^2-1}}{r} - \tanh^{-1}\frac{\sqrt{r^2-b^2}}{r} = \cosh^{-1} r - \cosh^{-1}\frac{r}{b}. \end{aligned}$$

Now for fixed b, $\cosh^{-1} r - \cosh^{-1}\frac{r}{b}$ decreases with r, and so takes its maximal value when $r = b$. Hence

$$R' \leq \cosh^{-1} b - \cosh^{-1} 1 = \cosh^{-1} e^R.$$

□

We are now in a position to embark on the proof of lemma 6.2, and the remainder of this section is devoted to that proof.

We start with some details of the generalised Dehn algorithm described in [2] for a geometrically finite group G. That algorithm operates with a generating set X which contains a mixture of parabolic and non-parabolic elements, such that for every conjugacy class of parabolic subgroups some representative of the class is generated by a subset of X. When a geodesic word over X is traced out in the embedded Cayley graph Γ, any portion which begins and ends sufficiently close to a boundary horosphere consists of parabolic generators (in the same conjugacy class as the elements which fix that horosphere), apart from possibly a bounded number of generators at either end. A subword representing an element within a parabolic subgroup, which is necessarily a product of generators for that subgroup, is called a parabolic subpath, and labels a subpath close to a horosphere.

In describing the generalised Dehn algorithm for G, [2] defines a D-good path in Γ (for D a positive integer) to be a path for which any parabolic subpath is a Cayley graph geodesic, and any one of the subpaths remaining when maximal parabolic subpaths of length $D/4$ or more are deleted is D-locally geodesic (that is, all its subpaths of length at most D are geodesic). We call a maximal parabolic subpath of length at least $D/4$ a long parabolic segment, and any one of the subwords remaining when such segments are deleted a long parabolic free (or LPF) segment. The algorithm reduces any word w to a word representing the same group element which is formed from a D-good path (for some fixed D) by replacing any maximal parabolic subpath by its reduction according to the generalised Dehn algorithm for a virtually nilpotent (in this case, virtually abelian) group, described in section 5.

In order to prove that G has real-time word problem, we need to verify an appropriate relationship between the length of a word w which is reduced according to the generalised Dehn algorithm and the length of a geodesic word in Γ which represents w. We shall show that (provided that D is chosen large enough) the geodesic length of w is bounded below by a constant multiple of the length of w.

In fact it is enough to establish a relationship between the length of a D-good path in Γ and a geodesic in X joining the ends of that path. For since the length of words reduced according to the generalised Dehn algorithm for nilpotent groups is never more than a constant times the geodesic length of word in the original group generators which they represent (since the geodesic lower bound for a nilpotent group is at least exponential), the reduced words in such a group are never more than a constant longer than D-good words. The same statement is true in virtually abelian groups, by lemma 5.2. Moreover, since Γ embeds via a $(\lambda, 0)$-quasi-isometry in X, the ratio of the length of a Cayley graph geodesic to an X-geodesic between the same two points is sandwiched between two constants.

So let γ be a D-good path from p_b to p_e, containing m long parabolic segments, and let γ_X be the geodesic in the space X which joins p_b to p_e. Our aim is to show that, when D is large enough, γ is at most a constant factor times as long as γ_X, and the remainder of this section is devoted to proving this fact.

We define three further paths, γ_Q, γ_R and γ_H to help us prove this.

We obtain γ_Q from γ as follows. On γ there are points at the beginning and end of long parabolic segments. Let $p_{i,b}$ and $p_{i,e}$ be the points at the beginning and end of the i-th long parabolic segment. Define $q_{i,b}$ to be the point on the horosphere corresponding to the i-th long parabolic segment which is closest to $p_{i,b}$, and define $q_{i,e}$ similarly with respect to $p_{i,e}$. Now define for each $q_{i,j}$ a further point $r_{i,j}$, within bounded distance k_1 of $q_{i,j}$ such that, where γ_Q is the path in $\mathbb{H}^n$ which is formed by concatenating hyperbolic geodesics from p_b to $r_{1,b}$, $r_{1,b}$ to $r_{1,e}$, $r_{1,e}$ to $r_{2,b}$, …, and finally $r_{m,e}$ to p_e, the sections of γ_Q which replace LPF segments of γ meet the horosphere at the points $r_{i,e}$ and $r_{i+1,b}$ at right-angles. According to [2] (in the proof of lemma 5.6), provided that D is sufficiently large, γ_Q is a quasigeodesic in $\mathbb{H}^n$.

γ_H is the geodesic in $\mathbb{H}^n$ which joins the endpoints of γ, i.e. p_b to p_e. We know that γ_H must fellow travel with γ_Q, since γ_Q is a quasigeodesic in $\mathbb{H}^n$ and hence (provided D is big enough, so that the long parabolic segments of γ are sufficiently long) that γ_H must pass through the m horoballs which are determined by the long parabolic segments of γ. Suppose that γ_H enters the i-th such horoball at $s_{i,b}$ and leaves it at $s_{i,e}$.

Now define γ_R to be the rough geodesic of X corresponding to γ_H. According to [2], γ_R asynchronously fellow travels with γ_X.

Define points $t_{i,j}$ to be the points on γ_X which are associated with the points $s_{i,j}$ as fellow travellers.

We claim:

Lemma 6.6 *Each $p_{i,j}$ is within a bounded distance of $q_{i,j}$, each $q_{i,j}$ within a bounded distance of $r_{i,j}$, each $r_{i,j}$ within a bounded distance of $s_{i,j}$, and each $s_{i,j}$ within a bounded distance of $t_{i,j}$.*

PROOF: The first two statements are proved in [2] (in the proof of lemma 5.5), and $t_{i,j}$ was especially selected to force the fourth statement. It remains to prove the third.

Let B be the horosphere in question. Let Y be the space $\mathbb{H}^n \setminus B$. Clearly Y contains X, and distances in Y are no longer than distances in X.

We know that γ_Q is a quasi-geodesic and so fellow travels with γ_H. Given a point $s = s_{i,j}$ on γ_H, let r' be the point on γ_Q fellow travelling with it. Let s' be the point on γ_H fellow travelling with $r = r_{i,j}$. Then r is at hyperbolic distance at most k from s' and s at hyperbolic distance at most k from r'. We want to show that s is also at a bounded distance from the point r. There are two cases to consider, namely (1) where r' is inside the horoball and s' outside, and (2) where r' is outside the horoball and s' inside.

For (1), let d be the distance from r to s along the surface of the horosphere (this is the distance in both X and in Y). The hyperbolic geodesic between r and s then has length $2\sinh^{-1}(d/2)$, by lemma 6.3. Hence

$$\begin{aligned} d_H(r,r') &\leq d_H(r,s) + d_H(s,r') \leq 2\sinh^{-1}(d/2) + k \\ d_H(s,s') &\leq d_H(s',r) + d_H(r,r') + d_H(r',s) \\ &\leq k + 2\sinh^{-1}(d/2) + k + k \leq 3k + 2\sinh^{-1}(d/2). \end{aligned}$$

Using the triangle inequality in Y, we see that

$$d \leq d_Y(r,s') + d_Y(s',s).$$

Since s' and s are joined in Y by a hyperbolic geodesic,

$$d_Y(s',s) = d_H(s',s) \leq 3k + 2\sinh^{-1}(d/2).$$

Further

$$d_Y(r,s') \le d_X(r,s') \le 2\sinh\left(d_H(r,s')/2\right) \le 2\sinh\left(k/2\right).$$

so

$$d \le 2\sinh\left(k/2\right) + d(s',s) \le 2\sinh\left(k/2\right) + 3k + 2\sinh^{-1}\left(d/2\right).$$

Hence d is bounded by a function of the constant k.

For (2), since the path along γ_Q from r' to r meets the horoball at a right angle, we see that r is the closest point on the horoball to r', and so is at least as close to r' as s is, and therefore $d_H(r',r) \le k$. Hence, using theorem 6.4,

$$d_X(s,r) \le d_X(s,r') + d_X(r',r) \le 4\sinh\left(k/2\right).$$

□

It follows from lemma 6.6 that each $p_{i,j}$ is distance at most R from the corresponding $t_{i,j}$, for some constant R. We shall compare the lengths of γ_X and γ by comparing the lengths of corresponding segments. We compare the length of a segment of γ between successive points $p_{i,j}$ with the length of a segment of γ_X between successive points $t_{i,j}$. A segment of γ between successive points $p_{i,j}$ is either long parabolic or long parabolic free.

We shall need to estimate the lengths of these segments.

LP segments are quasigeodesic and hence any such is at most a bounded factor longer than the X-geodesic path joining its ends.

We claim that the same is true of LPF segments.

Lemma 6.7 *Provided that D is chosen large enough, each LPF segment of γ is at most a bounded factor longer than the X-geodesic path joining its ends.*

PROOF: Let δ be an LPF segment of γ, joining a point p to a point q. Let δ_X be the X-geodesic path joining p and q. Let δ_H be the $\mathbb{H}^n$-geodesic path joining p and q. Let δ_R be the rough geodesic joining p and q.

We can find points $p = p_0, \ldots, p_m = q$ on δ, each of which is not in a parabolic segment, and such that for any i, p_i and p_{i+1} are at distance (measured along δ) between $D/4$ and $D/2$ apart (except that p_{m-1} and $p_m = q$ could be closer). From [2] (in the proof of lemma 5.5) we see that each p_i (for $1 \le i \le m-1$) is distance at most R_1 (where R_1 is a constant not depending on D) from a point q_i on δ_H. The point q_i might be in a horoball B, but if so, since p_i is not in a horoball, q_i is at most hyperbolic distance R_1 from the boundary of B, and hence the the distance along δ_H from q_i to the boundary of B is at most $\cosh^{-1} e^{R_1}$. So we can find a point r_i on δ_R at distance at most $\cosh^{-1} e^{R_1}$ from q_i and hence at distance at most $R_1 + \cosh^{-1} e^{R_1}$ from p_i. Since δ_X and δ_R fellow travel, there is a point s_i on δ_X at hyperbolic distance at most k from r_i, and hence at hyperbolic distance at most $k' = k + R_1 + \cosh^{-1} e^{R_1}$ from p_i. The distance in X between s_i and p_i is then at most $k'' = 2\sinh(k'/2)$. Now the length in Γ of δ is the sum of the distances in the Cayley graph between successive p_i's, which as at most a constant factor A more than the distance in X, and hence is at most A times $l_X(\delta) + 2(m-1)k''$. Since $l_\Gamma(\delta) \geq (m-1)D/4$, we see that

$$l_\Gamma(\delta) \leq A(l_X(\delta) + \frac{8k''}{D} l_\Gamma(\delta))$$

and hence (provided that $D > 8k''$) that

$$l_\Gamma(\delta) \leq \frac{AD}{D - 8k''} l_X(\delta).$$

□

Now γ divides up into $2m+1$ segments between successive points from the set

$$\{p_b, p_{1,b}, p_{1,e}, \ldots, p_{m,b}, p_{m,e}, p_e\}$$

Call those segments $\gamma^1, \ldots, \gamma^{2m+1}$ (following along γ from p_b to p_e in that order). Similarly, call the $2m+1$ segments of γ_X between successive points from the set

$$\{p_b, t_{1,b}, t_{1,e}, \ldots, t_{m,b}, t_{m,e}, p_e\}.$$

$\gamma_X^1, \ldots, \gamma_X^{2m+1}$. For each i, denote by $\hat{\gamma}^i$ the X-geodesic which joins the two ends of γ^i. By the above, for some constant K,

$$l(\gamma^i) \leq K l(\hat{\gamma}^i).$$

Further, using the triangle inequality, we see that for all $i \neq 1, 2m+1$,

$$l(\hat{\gamma}^i) \leq 2R + l(\gamma_X^i)$$

and that

$$l(\hat{\gamma}^1) \leq R + l(\gamma_X^1), \quad l(\hat{\gamma}^{2m+1}) \leq R + l(\gamma_X^{2m+1}).$$

Combining the above we deduce that

$$l(\gamma) \leq K(4mR + l(\gamma_X)).$$

Now since γ contains m long parabolic segments and each of those has length at least $D/4$, we also see that

$$l(\gamma) \geq mD/4,$$

so

$$mD/4 \leq 4KmR + Kl(\gamma_X), \quad m(D - 16KR)/4 \leq Kl(\gamma_X),$$

and hence for large enough D ($> 16KR$),

$$m \leq 4Kl(\gamma_X)/(D - 16KR)$$

yielding

$$l(\gamma) \leq (16 + D - 16R)Kl(\gamma_X)/(D - 16KR).$$

This establishes the promised relationship between $l(\gamma)$ and $l(\gamma_X)$ which completes the proof of theorem 6.1.

References

[1] B.H. Bowditch, Geometrical finiteness for hyperbolic groups, J. Functional Analysis 113 (1993), 245–317.

[2] Cannon, Goodman and Shapiro, Dehn's algorithm for non-hyperbolic groups, preprint.

[3] Max Dehn, Über die Topologie des dreidimensionalen Raumes, Math. Ann. 69 (1910) 137–168.

[4] David B. A. Epstein J.W. Cannon, D. F. Holt, S. Levy, M. S. Paterson and W. Thurston, *Word processing in groups*, Jones and Bartlett, 1992.

[5] Thomas Herbst and Richard M. Thomas, Group presentations, formal languages and characterizations of one-counter groups, Theor. Comp. Sci. 112 (1993) 187–213.

[6] Derek F. Holt, Word-hyperbolic groups have real-time word problem, Internat. J. Algebra Comput., to appear.

[7] W.D. Neumann and M. Shapiro, Automatic structures, rational growth and geometrically finite hyperbolic groups, Inventiones Mathematicae 120 (1995), 259–287.

[8] Arnold L. Rosenberg, Real-time definable languages, J. Assoc. Comput. Mach. 14 (1967) 645–662.