Algorithms and geometry for graph products of groups

Susan Hermiller*
Department of Mathematics, University of Melbourne,
Parkville, Victoria 3052, Australia; and
Department of Mathematical Sciences, New Mexico State University,
Las Cruces, NM 88003
smh@mundoe.maths.mu.oz.au

John Meier**
Department of Mathematics, Lafayette College, Easton, PA 18042
meierj@lafvax.lafayette.edu

1. Introduction

Recent work of Gromov, Epstein, Cannon, Thurston and many others has generated strong interest in the geometric and algorithmic structure of finitely generated infinite groups. (See [16],[17] and [14].) Many of these structures are preserved by taking graph products.

The graph product of groups (not to be confused with the fundamental group of a graph of groups) is a product mixing direct and free products. Whether the product between two groups in the graph product is free or direct is determined by a simplicial graph. Given a simplicial graph we say that two vertices are *adjacent* if they are joined by a single edge.

Definition. Given a finite simplicial graph $\mathcal{G}$ with a group (or monoid) attached to each vertex, the associated *graph product* is the group (monoid) generated by each of the vertex groups (monoids) with the added relations that elements of distinct adjacent vertex groups commute.

Graph products were defined by Green [15], and have also been studied by Chiswell [8], [9], [10]. Graph products are a generalization of "semifree groups"

* Research supported in part by NSF grants DMS 9022140 at MSRI and INT 9223826
** Research supported in part by a Lafayette College CASR grant

or "graph groups", which are a special case in which each vertex group is a free group of rank one; these groups have been studied by Droms, Servatius, VanWyk, and others (see [12],[13],[23], [25], and the references cited there). Another special case which has generated interest is the graph product of free monoids of rank one, also known as "free partially commuting monoids" ([4], [11]). These monoids have application in computer science to the study of concurrent processes.

Because the graph product involves creating direct products, it is impossible for the graph product of infinite groups which are negatively curved in the sense of Gromov to be negatively curved. However, direct products are not inconsistent with "nonpositive" curvature. The notion of nonpositive curvature in groups has been captured by Alonso and Bridson in [2] using geometric constraints on "bicombings" of groups. We show that such structure is preserved by taking graph products.

THEOREM A. *The graph product of finitely many semihyperbolic groups is semihyperbolic.*

In [14] various algorithmic structures were introduced for finitely generated groups, specifically the idea of an "automatic group" and the derivative notions of biautomatic, asynchronous automatic, and asynchronous biautomatic groups. Examples of automatic groups are finitely generated free groups, free abelian groups, Coxeter groups [6], and Artin groups of finite [7] and extra-large [22] type. It is known that the free product or direct product of finitely many automatic groups is automatic. The algorithmic structure of the graph product is inherited from the algorithmic structure of its vertex groups.

THEOREM B. *The graph product of finitely many automatic groups is automatic.*

This result is also true if the word "automatic" in theorem B is replaced by any of the related notions: biautomatic, asynchronous automatic or asynchronous biautomatic.

Another algorithmic structure applied to groups is a complete rewriting system, which was developed for application to automated theorem proving. Finite complete rewriting systems differ from the other constructions in this paper in that they apply to monoids as well as groups, and they are dependent upon the generating set [19]. The generators used to construct rewriting systems for graph products are different from those used to construct automatic structures. Examples of groups which admit finite complete rewriting systems include finitely generated free and free abelian groups, surface groups, and many Coxeter groups [18].

THEOREM C. *The graph product of finitely many groups (or monoids) which admit a complete rewriting system admits a canonical complete rewriting system. If the rewriting systems for the vertex groups (or monoids) are finite or regular, then the system for the graph product is also.*

Semihyperbolic structures, automatic structures, and finite complete rewriting systems all give an effective procedure for determining if words in the generators of a group are in a prescribed "normal form". All of these groups have solvable word problem and are of homological type FP_∞.

In section 2 we quickly recall important definitions, but a reader unfamiliar with automatic groups is advised to first read [14]. In section 3 we show that "combings" for graph products can be built out of the combings available for the vertex groups, and that the geometric structure of the vertex combings is preserved. In section 4 we prove theorem A, essentially just extending the ideas from section 3. In section 5 we prove theorem B, and section 6 contains a description of how to extend theorem B to the derivative notions of biautomatic groups, etc. In section 7 we construct another set of normal forms for graph products and prove theorem C.

We note that VanWyk [25] has independently proven theorems B and C in the special case of a graph group, using somewhat different arguments.

2. Definitions and Background

A. Combings

Given a set of monoid generators S for a group Γ, there is a natural map from the free monoid on S onto Γ, $S^* \xrightarrow{\pi} \Gamma$. A set of *normal forms* is a subset $N \subset S^*$ which bijects under π onto Γ. The idea of a "combing", as defined by Alonso, is essentially the geometric analogue of a set of normal forms.

Let Γ be a finitely generated group, and let S be a finite set of generators for Γ, closed under inverses. (That is, $S = S^{-1}$, or S is *symmetric*.) We denote the Cayley graph of Γ with respect to S by $\mathcal{C}(\Gamma, S)$. This is simply the graph whose vertices correspond to elements in the group; any two vertices are joined by an edge if their corresponding group elements only differ by right multiplication by a single generator. Different choices of generating sets do yield different Cayley graphs, but these differences are not significant to our results.

We think of the Cayley graph as a metric graph, where each edge is isometric to the unit interval, and the distance between any two points in the graph is the minimal length among all paths joining the two points. With this convention we can say that the length of any element $\gamma \in \Gamma$, $|\gamma|$, is simply the distance in $\mathcal{C}(\Gamma, S)$

from the identity to γ. (We do not always need to assume that the generating set is closed under inverses, especially when we discuss finite complete rewriting systems. We make this assumption primarily so that the "word length metric" on the group and the "Cayley graph metric" are equivalent.)

For a word $\omega \in S^*$, let $\epsilon(\omega)$ denote the vertex of the Cayley graph corresponding to the element of Γ represented by ω. Given any word $\omega = s_1 s_2 ... s_m \in S^*$, there is an eventually constant path $[0,\infty) \xrightarrow{p_\omega} \mathcal{C}(\Gamma, S)$, given by defining $n \mapsto \epsilon(s_1 s_2 ... s_n)$ for $n \leq m$, and $n \mapsto \epsilon(\omega)$ for $n \geq m$, for all natural numbers n, with the map having constant speed over the interval $[n, n+1]$.

Definition. Given a set of normal forms, the associated set of eventually constant paths is called a *combing.* Let $\sigma_\gamma(t)$ denote the path in $\mathcal{C}(\Gamma, S)$ corresponding to the normal form for the element $\gamma \in \Gamma$.

B. The Geometry of Combings

There are at least two important geometric constraints for combings, controlling the length and controlling the distance between combing paths. (See [5].) One way to control the length is to require that the combing paths be somewhat efficient, that is, the combing paths do not depart too much from paths of minimal length.

Definition. A combing is *geodesic* if for all $\gamma \in \Gamma$ the combing path $\sigma_\gamma(t)$ has minimal length among all paths joining the identity to γ.

Definition. Let T_γ be the time at which the path $\sigma_\gamma(t)$ becomes constant. Then the combing σ is *(λ, ϵ)-quasigeodesic* if there exist constants $\lambda \geq 1$ and $\epsilon \geq 0$ such that for all γ and for all $s, t \leq T_\gamma$,

$$\text{(QG)} \qquad \frac{1}{\lambda}|t-s| - \epsilon \leq d_{\mathcal{C}}(\sigma_\gamma(t), \sigma_\gamma(s)) \leq \lambda|t-s| + \epsilon.$$

Here $d_{\mathcal{C}}$ denotes the distance in the Cayley graph. This is just a restriction of the idea of a quasigeodesic map between metric spaces. (See [16],[17].)

Definition. The *width* of a combing σ is a function measuring how far combing paths can separate. Specifically,

$$W(n) = \sup_{\gamma \in \Gamma, s \in S} \{ d_{\mathcal{C}}(\sigma_\gamma(t), \sigma_{\gamma s}(t)) \mid t \in [0,\infty), |\gamma|, |\gamma s| \leq n \}.$$

A combing is *bounded* if there exists a constant $B > 0$ such that the width function satisfies $W(n) \leq B$ for all n. Weaker bounds have also yielded surprising results about the complexity of the word problem; for instance, groups admitting

combings whose width functions are bounded by $W(n) \leq n-1$ satisfy a quadratic exponential isoperimetric inequality [5].

What is referred to as a combing in [14] would be referred to here as a quasigeodesic bounded combing. Any automatic group admits such a combing. Groups admitting such combings, irrespective of their algorithmic structure, have also been studied ([1], [24]).

C. Semihyperbolic Groups

A semihyperbolic group is, intuitively, a group which "looks like" the fundamental group of a compact nonpositively curved Riemannian manifold. For details and the basic theorems about semihyperbolic groups, see [2].

Let $\mathcal{P}$ be the set of all eventually constant paths starting and stopping at vertices of the Cayley graph $\mathcal{C}(\Gamma, S)$, for a group Γ with finite generating set S. We assume that the path is parameterized by arc length. If $\iota(p)$ and $\tau(p)$ denote the initial and terminal vertices of such a path, there is a natural map $\mathcal{P} \xrightarrow{\epsilon} \Gamma \oplus \Gamma$ given by $\epsilon(p) = (\iota(p), \tau(p))$.

Definition. A *bicombing* is a section $\Gamma \oplus \Gamma \xrightarrow{\sigma} \mathcal{P}$ of the endpoints map ϵ. Denote the chosen path between vertices x and y by $\sigma_{(x,y)}$.

A bicombing is *equivariant* if $\gamma \cdot \sigma_{(x,y)}(t) = \sigma_{(\gamma \cdot x, \gamma \cdot y)}(t)$ for all γ, x, and y in Γ, and for all t.

A bicombing is *bounded* if there exists a constant $B > 0$ such that the following inequality holds.

$$d_{\mathcal{C}}(\sigma_{(x,y)}(t), \sigma_{(z,w)}(t)) \leq B \tag{B}$$

for all t, whenever $d_{\mathcal{C}}(x, z) \leq 1$ and $d_{\mathcal{C}}(y, w) \leq 1$.

Definition. A finitely generated group with a finite, symmetric set of generators is *semihyperbolic* if it admits a quasigeodesic, bounded, equivariant bicombing. (This definition does not actually depend on the choice of symmetric generating set.)

D. Automatic Structures

A reader unfamiliar with automatic groups is advised to read [14]. We use the same terminology as in [14] and collect relevant definitions and theorems in this section for easy reference.

A finite state automaton is a simple machine which can read a word from a chosen finite alphabet $\mathcal{A}$. The finite state automaton either accepts or rejects the word and the set of accepted words is called a *regular language.*

A finite state automaton can be most easily understood as a finite, directed, and labeled CW-graph. This graph will have a chosen vertex, the *initial state*, and all of the vertices are designated as either *accept states* or *non-accept states*. Going out of every vertex there is exactly one edge for every element in the alphabet $\mathcal{A}$.

Any word $\omega \in \mathcal{A}^*$ defines a path in the finite state automaton, starting at the initial state and then first traveling along the edge labeled by the first letter in ω, then following the edge labeled by the second letter in ω, etc. If the path ends at an accept state, then the word is accepted by the finite state automaton and is part of the corresponding regular language. If not, then the word is rejected.

A state is called a *fail state* in some finite state automaton if it is a non-accept state from which all exiting edges are loops.

The following definition is proven to be equivalent to the standard definition of an automatic group in [14]. It is this alternative definition with which we will be working.

Definition. Let Γ be a group and let S be a finite, symmetric set of generators for Γ. Then Γ is *automatic* if and only if Γ admits a bounded combing such that the set of normal forms associated to the combing is the regular language of a finite state automaton with alphabet S.

Remark: Because of the constraints imposed by the combing paths being accepted by a finite state automaton, these combings will not only be bounded, they will also be quasigeodesic [14].

E. Rewriting Systems

Let Λ be a finite set (called an *alphabet*) and let Λ^* be the free monoid on Λ as before; the empty word will be represented by 1. A *rewriting system* on Λ^* is a subset $R \subseteq \Lambda^* \times \Lambda^*$. An element $(u, v) \in R$, also written $u \to v$, is called a *rule* of R. The idea is that a rewriting system is an algorithm for substituting the right hand side of a rule whenever the left hand side appears in a word. Given a rewriting system R, write $x \to y$ for $x, y \in \Lambda^*$ if $x = uv_1w$, $y = uv_2w$ and $(v_1, v_2) \in R$. Write $x \xrightarrow{+} y$ if $x \to x_1 \to x_2 \to \ldots \to y$ for some finite chain of arrows, and $x \xrightarrow{*} y$ if $x = y$ or $x \xrightarrow{+} y$. An element x of Λ^* is *irreducible* with respect to R if there is no possible rewriting (or *reduction*) $x \to y$; otherwise x is called *reducible*. (Λ, R) is a rewriting system for a monoid M if

$$\langle \Lambda \mid v_1 = v_2 \text{ if } (v_1, v_2) \in R \rangle$$

is a presentation for M. A rewriting system for a group G is a rewriting system for G as a monoid; in particular, the alphabet must generate G as a monoid.

The rewriting system R is *Noetherian* if there is no infinite chain of rewritings $x \to x_1 \to x_2 \to \dots$ for any word $x \in \Lambda^*$. Another way to view this is in terms of orderings. A partial *well-founded* ordering on Λ^* is an ordering such that for any subset $\Xi \subseteq \Lambda^*$, there is an element $v \in \Xi$ which is minimal with respect to the ordering; that is, there is no other word $w \in \Xi$ with $v > w$. Using the axiom of choice, this is equivalent to saying that for any $x \in \Lambda^*$, there is no infinite descending chain of words $x > x_1 > x_2 > ....$ The partial ordering on Λ^* given by $x > y$ if $x \xrightarrow{+} y$ is well-founded if and only if the rewriting system is Noetherian.

R is defined to be *confluent* if whenever $x \xrightarrow{*} y_1$ and $x \xrightarrow{*} y_2$, there is a z so that $y_1 \xrightarrow{*} z$ and $y_2 \xrightarrow{*} z$. In the course of studying graph products, the confluence of many examples of rewriting systems has been checked using the software package RRL [20]. R is *complete* if R is Noetherian and confluent; a complete rewriting system for a group is also known as a complete presentation. A rewriting system is *regular* if $\{u \mid (u, v) \in R\}$ is a regular language. Standard properties of regular languages can be used to show that R is regular if and only if

$$\{w \in \Lambda^* \mid w \text{ is irreducible}\}$$

is a regular language. Finally, a rewriting system is *finite* if R is a finite set; since a finite set of words is regular, the irreducible words form a regular language in this case, also.

A theorem of M. H. A. Newman [21] states that a Noetherian rewriting system is complete if and only if there is exactly one irreducible word representing each element of the monoid it presents. For a complete rewriting system, then, the irreducible words are a set of normal forms, which is closed under taking subwords. In the case of a finite or regular complete presentation, this is a regular language of normal forms. The system also gives an algorithm for determining the normal form associated to any word, and hence an easily computable solution to the word problem.

3. Combining Combings

Given a graph product Γ of a finite set of groups G_v, if there are already normal forms prescribed for each vertex group (with the identity having normal form the empty word), there is a set of normal forms for Γ which restricts to the given normal forms for the vertex groups. In this section we show how to construct such normal forms for the graph product.

Convention. We will always be assuming in what follows that the normal form for the identity is the empty word. This is not restrictive in any of the classes of groups which we consider. For semihyperbolic groups this would only require

a (possibly) larger bound on the bicombing. For automatic groups, replacing the normal form for the identity element by the empty word gives a new set of normal forms which is again a regular language, so as before this only requires a (possibly) larger bound on the width of the combing.

Because the groups associated to adjacent vertices commute in the graph product, in order to define a normal form we will need to totally order the vertices of the underlying graph $\mathcal{G}$. To avoid double subscripts we label the vertices of $\mathcal{G}$ by letters, $v_A, v_B, ..., v_Z$, where the vertices are ordered lexicographically. The associated vertex groups will be denoted $G_A, G_B, ..., G_Z$, and their generating sets will be denoted $\mathcal{A}, \mathcal{B}, ..., \mathcal{Z}$, respectively. The union of the generating sets for the vertices is Σ, and it generates the graph product.

We will use the capital letters I, J, and K to stand for arbitrary letters (analogous to i, j and k standing for arbitrary integers); so, for instance, G_I will denote some vertex group.

DEFINITION. Given a word ω in Σ^*, a subword ω' of ω is a *local word* if it is written in letters coming from a single vertex group and there is no longer subword of ω, containing ω', written in letters coming from a single vertex group. The *type* of a local word is the label of the corresponding vertex.

Any word in Σ^* can be expressed as a product of local words. The *type* of a word $\omega \in \Sigma^*$ is the string of types of its local words. For example, if $\omega = \omega_A\omega_B\omega_C\omega_A$ where each ω_I is a word in the chosen generators of G_I, then the type of ω is $ABCA$.

DEFINITION. The *global length* of a word ω is the number of local words into which it can be decomposed.

If $\omega = ...\omega_I\omega_J...$, where v_I and v_J are adjacent, then the word $\omega' = ...\omega_J\omega_I...$, obtained by shuffling the local words ω_I and ω_J, will evaluate to the same element in the graph product Γ. Given any word in Σ, we may freely shuffle any adjacent local words whose corresponding vertices are adjacent in the graph $\mathcal{G}$, and the new word will evaluate to the same group element as the original word. If, in the process of shuffling, two local words of the same type become adjacent, then they can be amalgamated into a single local word. Remove this word if it represents the identity.

As the shuffling and amalgamating process continues, eventually the global length of the word will become stable; that is, it will become impossible to shuffle together two local words of the same type. After the global length has stabilized, chose a word among all possible shuffles whose type is minimal with respect to the lexicographic order.

This process of taking a word, shuffling, amalgamating, and eventually finding a smallest representative with respect to the ShortLex ordering of types, we call *pruning*. A *pruning of* ω is then a word obtained from ω by the pruning process.

Notice that the pruning process will not necessarily yield a unique word. Let G be the direct sum of two infinite cyclic groups with generators $a, \tilde{a}$ and $b, \tilde{b}$ (where there are two generators for each cyclic group since the generating sets are assumed to be symmetric). Then the word $a\tilde{a}\tilde{b}a$ can be pruned to both $a\tilde{a}a\tilde{b}$ and $a\tilde{b}$ by the method outlined above. This non-uniqueness is not very significant, since by the propositions below, if we take any pruning of a word and replace each local word by its normal form, we will get a unique word.

The following proposition is essentially proved in [15].

PROPOSITION 3.1. *Let ω and ω' be two words in Σ^* which evaluate to the same element in the graph product Γ. Let T and T' be the types of prunings of ω and ω' respectively. Then $T = T'$.* □

DEFINITION. Given a set of normal forms for the local words, a word ω in Σ^* is *proper* if it satisfies the following local condition and obstruction condition:

(L) Each local word is in its prescribed normal form.

(O) If $\omega = ...\omega_I...\omega_J...$ with $I \geq J$ and v_I adjacent to v_J in $\mathcal{G}$, then there is a local word ω_K such that $\omega = ...\omega_I...\omega_K..\omega_J...$, where v_K and v_J are non-adjacent in $\mathcal{G}$.

Proper words are exactly the words which are prunings with each local word in normal form.

PROPOSITION 3.2. *Let Γ be the graph product of a finite set of groups, all with prescribed normal forms. Then the set of proper words in Σ^* is a set of normal forms for Γ.*

Proof. Suppose ω and ω' are distinct proper words representing the same element of the graph product. Since any pruning of a proper word yields a word of the same type as the proper word, it follows that by Proposition 3.1, ω and ω' must have the same type. So at least one local word in ω must differ from the corresponding local word in ω'. Write $\omega = \omega_i\omega_J\omega_t$ and $\omega' = \omega_i\omega'_J\omega'_t$, where ω_J and ω'_J are corresponding local words in normal form which differ. Let $\tilde{\omega}$ denote the normal form for $\omega_J^{-1}\omega'_J$. Then the words ω_t and $\tilde{\omega}\omega'_t$ are proper words representing the same element in the graph product, but the types of these words differ, contradicting Proposition 3.1. □

These normal forms for the graph product preserve most of the geometric structure of the combings associated to the normal forms for the vertex groups.

PROPOSITION 3.3. *Let Γ be the graph product of a finite set of groups, all having geodesic combings. Then the combing of Γ associated to the proper words is geodesic.*

Proof. Take any minimal length expression, ω, in the generating set Σ for an element $\gamma \in \Gamma$. By pruning ω, we get a word ω' which also represents γ. Since the pruning process does not increase length, ω' has the same length as the geodesic word ω; hence it is geodesic. This means that each local word in ω' must be geodesic in its corresponding vertex group. Replace any local word which is not in the normal form of the associated vertex group by the normal form for that group element; call this new word ω''. Since the local combings are geodesic, this will not increase the length of the word. It follows that ω'' is the proper word representing γ and is of minimal possible length. □

If the generating sets of each vertex group are totally ordered, then there is a total ordering on Σ given by requiring that any generator for G_I is less than any generator for G_J if $I < J$. Using this ordering, the following corollary is immediate.

COROLLARY 3.4. *If the combing associated to each vertex group is the combing whose elements are minimal with respect to the ShortLex ordering, then the corresponding combing by proper words for the graph product is the combing whose elements are minimal with respect to the ShortLex ordering, where the lexicographic ordering on the generators is described above.* □

PROPOSITION 3.5. *Let Γ be the graph product of a finite set of groups, all having quasigeodesic combings. Then the combing of Γ associated to the proper words is quasigeodesic.*

Proof. For each vertex v_I, assume the combing of G_I is (λ_I, ϵ_I)-quasigeodesic. Let M be the largest value among $\{\lambda_I\}$ and let E be the largest among $\{\epsilon_I\}$. Let $\Lambda = M + EM$. This choice of Λ will insure the following inequalities.

$$\text{(UB)} \qquad \Lambda d \geq \lambda_I d + \epsilon_I \qquad \text{(for any } d \geq 1\text{)}$$

$$\text{(LB)} \qquad \frac{1}{\Lambda} d \leq \frac{1}{\lambda_I} d - \epsilon_I \qquad \text{(whenever } \frac{1}{\lambda_I} d - \epsilon_I \geq 1\text{)}$$

We will show that the combing given by proper words is $(\Lambda, 2E+1)$-quasigeodesic.

Pick any two points on a path associated to a proper word ω. Using the "1" in $2E+1$, we can assume these points are vertices in the Cayley graph of Γ. The

choice of the two vertices describes a subword $\omega' = \omega_1\omega_2...\omega_n$ of ω, each local word in the subword being quasigeodesic.

Let λ_i and ϵ_i be the quasigeodesic constants for the local word ω_i. Let t_i be the length of the word ω_i, and let d_i be the actual distance between the endpoints of ω_i. Being a subword of a proper word, ω' has the same type as a proper word (and would be proper if the initial and final local words were in their prescribed normal forms). The length of the combing path for ω' is then $T = \sum t_i$, and by the argument for Proposition 3.3, the distance between the endpoints of ω' is $D = \sum d_i$. The quasigeodesic condition (QG) then takes the form

$$(*) \qquad \frac{1}{\Lambda}T - (2E+1) \leq D \leq \Lambda T + (2E+1),$$

which we will establish by finding inequalities for each of the local words and then adding these together.

Case 1) $2 \leq i \leq n-1$

Any local word ω_i is in normal form, so it does not represent the identity. The distance between its endpoints is then nontrivial, so $d_i \geq 1$. Because the word is quasigeodesic we get the inequality $\frac{1}{\lambda_i}t_i - \epsilon_i \leq d_i \leq \lambda_i t_i + \epsilon_i$. Since we know that $d_i \geq 1$, it follows that

$$\max\{\frac{1}{\lambda_i}t_i - \epsilon_i, 1\} \leq d_i \leq \lambda_i t_i + \epsilon_i \ .$$

Since $t_i \geq d_i \geq 1$ we may apply inequality (UB) to the right term, and because we have introduced the maximum in the left hand term, we may apply the inequality (LB), to get the following inequalities.

$$(**) \qquad \frac{1}{\Lambda}t_i \leq d_i \leq \Lambda t_i$$

Case 2) $i = 1, n$

Since ω_i for $i = 1, n$ may only be a proper subword of a local word in ω, it is possible that ω_i does represent the identity. In this case $d_i = 0$ so the argument of case 1) does not apply. However, ω_i is still a quasigeodesic, hence it satisfies $\frac{1}{\lambda_i}t_i - \epsilon_i \leq d_i \leq \lambda_i t_i + \epsilon_i$. So by the choice of Λ and E we get the following inequalities.

$$(***) \qquad \frac{1}{\Lambda}t_i - E \leq d_i \leq \Lambda t_i + E$$

for $i = 1, n$.

If we sum the inequalities (**) for interior local words of ω' and (***) for ω_1 and ω_n, (and we recall that we moved our original initial and terminal points of the arc to vertices in the Cayley graph), we establish the inequalities (*). □

PROPOSITION 3.6. *Let Γ be the graph product of a finite set of groups $\{G_I\}$, with (λ_I, ϵ_I)-quasigeodesic combings $\{\sigma_I\}$ having width functions $\{W_I\}$. Then the function $W(n) = \max\{\lambda_I(W_I(n) + 1 + \epsilon_I)\} + 1$ is a width function for the quasigeodesic combing σ of Γ obtained from the σ_I's.*

Notice that it follows immediately from the proposition that quasigeodesic bounded combings for the vertex groups yield a quasigeodesic bounded combing for the graph product.

Proof. To find the width function for the combing σ, we need to compare the combing paths for an arbitrary element $\gamma \in \Gamma$ and an element $\gamma \cdot s$ where $s \in \Sigma$. Let ω be the proper word corresponding to γ. Pruning the word ωs, one gets a word of the same type as the proper word expression for $\gamma \cdot s$. We can therefore shuffle the local word "s" as far left as it can go until it is either adjacent to a local word of the same type, at which point we can replace the two local words by the normal form expression for their product, or it encounters a local word with which it cannot commute, in which case we can shuffle "s" back to the right until the word type is in lexicographic order.

We can divide the word ω into three parts, $\omega = \omega_i\omega_s\omega_t$, so that one pruning of ωs gives the word $\omega_i\omega_s s\omega_t$ or $\omega_i\omega_t$ (if "$\omega_s s$" gives the identity). Here ω_i, the initial subword, could be a word of almost any type, ω_s is a local word of the same type as s, and the terminal subword ω_t is composed only of local words which commute with s. (Note that any of these subwords could be the empty word.) Thus the proper word expression for $\gamma \cdot s$ is $\omega_i\omega'_s\omega_t$, where ω'_s is the normal form for the vertex group element corresponding to $\omega_s s$.

If it does occur that the words ω_s and s cancel, then the proper word is $\omega_i\omega_t$. To check that this is proper, note that s commutes with ω_t; hence any local word in ω_t can be shuffled past ω_s in the proper word ω. Thus $\omega_i\omega_t$ is of minimal global length, and by essentially the same argument, it is lexicographically first among all possible shuffles.

We may ignore the initial strings and assume that the proper word expressions for γ and $\gamma \cdot s$ are of the form $\omega_s\omega_t$ and $\omega'_s\omega_t$ respectively. Further, since s commutes with all the local words in ω_t, the following picture can be immersed in the Cayley graph of Γ.

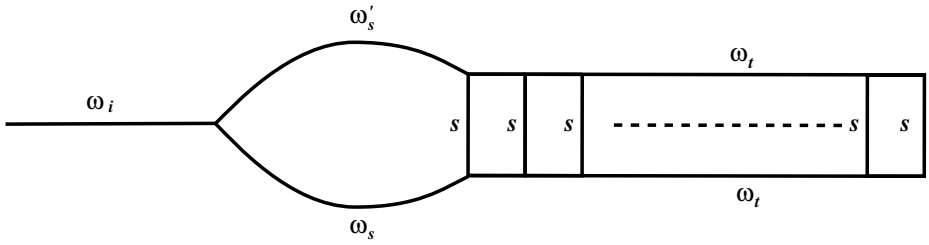

Fig. 1

Because ω_s and ω'_s are both local words, it follows that the distance between these two combing paths is less than or equal to $W_I(|\gamma|)$, where s is a generator in the vertex group G_I. The quasigeodesic property gives more information about the local behavior of these combing paths. If one combing path becomes constant before the other, they can be separated by a distance of at most $W_I(|\gamma|)$; at that point the longer combing path is within a distance $W_I(|\gamma|)+1$ of its endpoint. Because the combing paths are quasigeodesic, it follows that the longer combing path must reach its endpoint within a time $\lambda_I(W_I(|\gamma|)+1+\epsilon_I)$.

At some time t one of the combing paths $\omega_s\omega_t$ and $\omega'_s\omega_t$ will begin to move along the word ω_t. It follows that the combing path for the other word will have at most $\lambda_I(W_I(|\gamma|)+1+\epsilon_I)$ letters to traverse before it too begins to run along the word ω_t. Therefore the distance between the two points on the combing path will then be at most $\lambda_I(W_I(|\gamma|)+1+\epsilon_I)+1$. □

Corollary 3.7. *If Γ is the graph product of a finite set of groups admitting quasigeodesic bounded combings, then Γ also admits a quasigeodesic bounded combing.* □

4. Proof of Theorem A

Theorem A. *The graph product of finitely many semihyperbolic groups is semihyperbolic.*

Proof. Let Γ be a graph product of semihyperbolic groups. Using the proper words of the previous section, Γ admits a combing coming from the combings of the vertex groups. This can be turned into an equivariant bicombing by taking the path from γ_1 to γ_2 to be the translate under γ_1 of the combing path from 1 to $\gamma_1^{-1}\gamma_2$.

The argument that the bicombing is quasigeodesic is the same as the argument in the combing case. Similarly, the argument in §3 shows that the bicombing paths satisfy inequality (B) (from §2.C) if they start at the same vertex and become eventually constant at vertices a distance at most 1 apart. So to establish

theorem A it remains to show that any two bicombing paths which start a distance 1 apart and end at the same vertex are a bounded distance apart. The argument will be very similar to the argument given in the combings case of §3.

Assume without loss of generality that one initial vertex is the identity; hence the other is s for some generator s, and the terminal vertex is γ. Let ω be the proper word for γ. The bicombing path from s to γ will then follow the edges with the same labels as the proper word corresponding to $s^{-1}\gamma$. Hence this word can be obtained from the word $s^{-1}\omega$ by pruning (and possibly replacing a local word by its normal form).

Shuffle s^{-1} to the right until either s^{-1} is adjacent to a local word of the same type, in which case the product should be replaced with the normal form, or a local word with which s^{-1} cannot commute, in which case s^{-1} will get shuffled back into lexicographic order.

As we did before, ω can be expressed as the product of three subwords (some subwords being possibly empty), $\omega = \omega_i\omega_s\omega_t$, so that one pruning of $s^{-1}\omega$ gives the word $\omega_i s^{-1}\omega_s\omega_t$ or $\omega_i\omega_t$, where s^{-1} commutes with the local words in the initial subword ω_i, ω_s is of the same type as s^{-1}, and the terminal subword ω_t could be almost any word. Thus the proper word for $s^{-1}\cdot\gamma$ is $\omega_i\omega'_s\omega_t$ where ω'_s is the normal form for the group element corresponding to the local word $s^{-1}\omega_s$.

The following picture immerses into the Cayley graph.

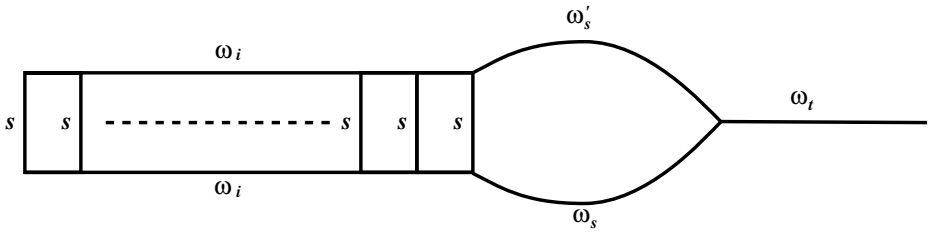

Fig. 2

The boundedness of the bicombing follows by essentially the same argument as is in the proof of Proposition 3.6. □

5. Proof of Theorem B

Theorem B. *The graph product of finitely many automatic groups is automatic.*

Proof. Fix automatic structures with associated automata $\mathcal{F}_I$ for the vertex groups G_I. (We think of these finite state automata as directed CW-graphs.) The $\mathcal{F}_I$'s will be used to create a larger directed graph which will be the finite state automaton admitting the set of proper words as its regular language. This

will give a bounded combing of the graph product whose combing paths come from the regular language of a finite state automaton; hence the graph product of automatic groups will be automatic.

To aid in the exposition, we first create an oriented graph, the *admissible graph*, onto which our finite state automaton graph will project. This admissible graph will serve as a guide for the allowed types of proper words. That is, the type of any proper word can be read off by an orientation preserving path on the admissible graph, and any string of types which occurs from following an orientation preserving path is the type of some proper word.

The admissible graph is not introduced solely as an aid in the exposition. It is essentially a finite state automaton accepting the proper words for the "right angled Coxeter group" [8], i.e. the graph product of cyclic groups of order two, based on the same graph $\mathcal{G}$. It is also a simple matter to convert the admissible graph to a finite state automaton which accepts the ShortLex normal forms for the graph product of free monoids of rank one (the free partially commuting monoid) with the underlying graph $\mathcal{G}$.

Let ω be a proper word. By property (O), if the type of ω is $...JI...$ and $J > I$, the vertices v_I and v_J in the underlying graph $\mathcal{G}$ are non-adjacent. It is possible to codify what types of strings are admissible as the types of proper words, by looking at the types of subwords which occur when the type of the entire word is not strictly increasing.

DEFINITION. A string of labels $IJK_1...K_n$ is an (I, J)-*admissible string* if

(i) $I > J$ and v_I and v_J are non-adjacent in the underlying graph $\mathcal{G}$;
(ii) the substring $JK_1...K_n$ is in strictly increasing order; and
(iii) if $K_i \leq I$, then the vertex v_i corresponding to the type K_i is not adjacent to all of the vertices $\{v_I, v_J, v_1, ..., v_{i-1}\}$.

Corresponding to a (I, J)-admissible string there is an oriented simplicial arc, the (I, J)-*admissible arc*, beginning with an edge labeled J, followed by an edge labeled K_1, etc. The (I, J)-*admissible tree* is created by amalgamating the (I, J)-admissible arcs along their common initial segments.

Example. Let the underlying graph $\mathcal{G}$ be:

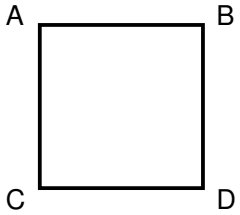

Fig. 3

Order the vertices by $A < B < C < D$. In this example, the only admissible type strings are $CA, CAC, CACD, CAD, DB$, and DBD. Hence the admissible trees would be:

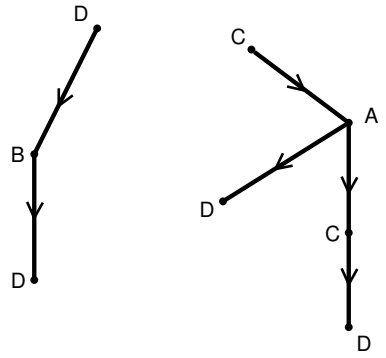

Fig. 4

The admissible graph will be built in stages.

Stage 1) Start with an initial vertex, together with a set of edges oriented away from the initial vertex, one edge for each type. The "type" of a vertex in the admissible graph will be the label of any edge flowing into it. (No vertex will have edges with different labels flowing into it.)

Stage 2) Because any increasing initial segment is of proper type, at the terminal vertex of an edge labeled I, attach edges going to all vertices of type J, where $J > I$. After this stage our graph for the example above will be:

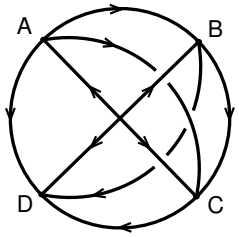

Fig. 5

Stage 3) A proper word does not have to have strictly increasing type, hence we add in the (I, J)-admissible trees to deal with the possible places where the type string of a proper word decreases. At the I vertex attach the (I, J)-admissible trees for all J. We will refer to these subtrees as the *hanging trees* of the admissible graph. At this stage our example admissible graph will be:

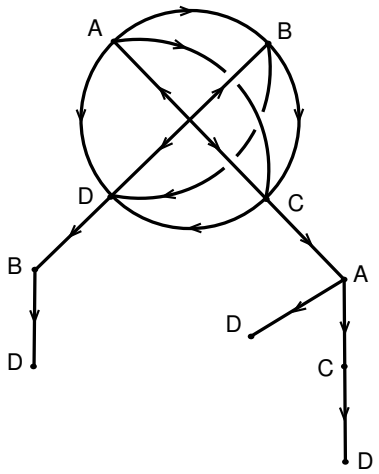

Fig. 6

Stage 4) It is possible for the type of a proper word to decrease more than once. Thus it is necessary to attach edges between certain vertices in the hanging trees. At each vertex of type I in a hanging tree we will add edges labeled J, for all $J < I$ such that v_J and v_I are non-adjacent in $\mathcal{G}$. This edge will connect the vertex of type I to the terminal vertex of the first edge of the (I, J)-hanging tree. Our admissible graph for the example above then is:

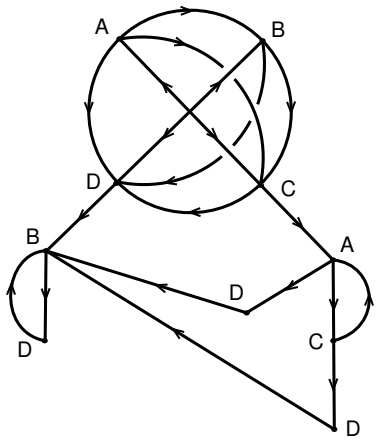

Fig. 7

Since cyclic groups of order two only have one nontrivial element, the type string for a proper word in a graph product of these groups is simply a transcription of the proper word; thus the admissible graph we have constructed will accept the proper words of the right-angled Coxeter group based on the graph $\mathcal{G}$.

Our admissible graph is not quite a finite state automaton admitting the proper words for the free partially commuting monoid based on $\mathcal{G}$. Since the monoid associated to each vertex is free on a single generator, in order to create

a finite state automaton admitting the proper words as its regular language we need to add loops labeled I at each vertex of type I.

To create a finite state automaton for the graph product, we base our construction on the admissible graph, adding in "local finite state automata" to insure that the local words are in their prescribed normal forms. The finite state automaton for the graph product will be built in stages analogous to the stages above.

Stage 0) For each finite state automaton $\mathcal{F}_I$ associated to a vertex group, add a fail state which receives edges from every non-accept state of $\mathcal{F}_I$. There should be an edge going to this fail state for each pair consisting of a non-accept state of $\mathcal{F}_I$ and a generator of any *other* vertex group. This will give a directed graph (which at the moment is not a finite state automaton) $\hat{\mathcal{F}}_I$.

This fail state is added since the local words should be in the regular language of the finite state automaton prescribed for the corresponding vertex group. Thus any word which contains a local word not in the regular language could not be a proper word.

If $a_i \in \mathcal{A}$, the generating set for G_A, we call the terminal vertex of the a_i-edge leaving the initial state of $\mathcal{F}_A$, the *a_i-vertex*. We similarly define the b_i-vertices and c_i-vertices, etc. The set of all such distinguished vertices we call *first states*.

Stage 1) Start with a copy of $\hat{\mathcal{F}}_I$ for each I, and identify all the initial vertices of the $\hat{\mathcal{F}}_I$'s. If the initial letter in a word is a_i, then the initial local word will be of type A, and the graph $\hat{\mathcal{F}}_A$ will test to see if the initial local word is in the prescribed normal form for the vertex group G_A.

If in each $\hat{\mathcal{F}}_I$ the edges attaching to the initial state are collapsed to a single edge, and the remaining vertices and edges are identified with the endpoint of this single edge, we get the subgraph of the admissible graph constructed in stage 1) of its construction.

Stage 2) For every edge from the I to the J vertex of the admissible graph which was added in the second stage of its construction, we will add edges between the graphs $\hat{\mathcal{F}}_I$ and $\hat{\mathcal{F}}_J$. There will be an edge between each pair consisting of an accept state of $\hat{\mathcal{F}}_I$ and a first state of $\hat{\mathcal{F}}_J$. It will be labeled by j_i if its terminal vertex is the j_i-vertex of $\hat{\mathcal{F}}_J$.

These edges are added because in any proper word ω, the local words will be in normal form, and hence end at an accept state of the corresponding finite state automaton before moving to the next local word. We will essentially repeat this process in the next two stages.

Stage 3) Make a copy of $\hat{\mathcal{F}}_J$ for every vertex of type J contained in a hanging tree in the admissible graph. Once again we will add edges between accept states and first states, this time based on the pattern of edges in the hanging trees.

Starting with the initial vertex of an (I,J)-hanging tree, add edges between each accept state of the graph $\hat{\mathcal{F}}_I$ and each first state of $\hat{\mathcal{F}}_J$. Repeat this process for all vertices attached to the J vertex of the hanging tree, and so on.

Once again, collapsing all the $\hat{\mathcal{F}}_I$ to points, and all the edges of the same type which run between the same copies of the $\hat{\mathcal{F}}_I$ to single edges, yields the subgraph of the admissible graph built in stage 3) of its construction.

Stage 4) For each edge added in stage 4) of the construction of the admissible graph, we will add edges between accept states and first states. Specifically, if an edge was attached from an I to a J vertex in the admissible graph, then edges are added between the accept states of the corresponding copy of $\hat{\mathcal{F}}_I$ to the first states of the corresponding copy of $\hat{\mathcal{F}}_J$.

Stage 5) After stage 4) not all vertices of our graph will have edges leaving them of all labels. However at this point there will be an oriented path corresponding to any proper word. Hence if any vertex is missing an edge corresponding to a particular generator, then add an edge labeled by that generator from the vertex to a fail state.

It is easy to check that this construction gives a finite state automaton accepting the proper words as its regular language. A word will not be accepted if its local words are not in prescribed form, or if it is not of an admissible type. □

Borrowing terminology from [14], we have the following corollary.

Corollary 5.1. *The graph product of weakly geodesic (ShortLex geodesic) automatic groups is weakly geodesic (ShortLex geodesic) automatic.*

Proof. These results follow from Corollaries 3.3 and 3.4. □

6. Alternative Algorithmic Structures

In this section we briefly indicate how to extend the ideas in §5 to the other standard algorithmic structures currently being discussed by group theorists, namely the asynchronous automatic, biautomatic and asynchronous biautomatic groups.

In an asynchronous automatic group, as opposed to an automatic group, the combing paths do not have to be of unit speed until they become constant, but rather they are allowed to pause for a moment before continuing on. Given this freedom it is even easier to establish that there is an "asynchronous bound"

on the distance between combing paths beginning at the identity and ending a distance one apart. Thus the graph product will inherit an asynchronously bounded regular combing from its vertex groups. By work in [3] , this shows that the graph product is an asynchronous automatic group.

A biautomatic group is similar to a semihyperbolic group in that there are combing paths given between any pair of vertices in the Cayley graph, and, in addition, the combing paths must be part of the regular language of a finite state automaton.

Definition. A finitely generated group Γ with a finite, symmetric set of monoid generators S is *biautomatic* if Γ admits an automatic structure with regular language $\mathcal{L}$ such that there is some other automatic structure for Γ admitting the set of formal inverses $\mathcal{L}^{-1}$ as its regular language.

Since formal inverses are obtained by inverting the order of a word, and then taking the inverses of each generator, the formal inverse of a proper word will be of minimal type with respect to a ShortLex ordering, using the ordering on vertices opposite from that used to determine $\mathcal{L}$. By the same methods described before, a finite state automaton can be constructed which admits these "reverse proper words" as its normal forms. Also using the same methods as before, the "reverse proper words" can be shown to give a bounded combing.

As an easy application, consider graph groups, which are the graph products of copies of Z. Since Z is biautomatic, the following result is immediate.

Proposition 6.1. *All graph groups are biautomatic.* □

This proposition has also been proven in [25], using a different combing.

7. Proof of Theorem C

Given a graph product of groups all of which admit finite complete rewriting systems, it is natural to try to find a finite complete presentation for the graph product in which the normal forms, or irreducible words, are the proper words defined in §3. However, this will not work in general.

Example. Let

$$\Gamma = \langle a, b, c, d, e \mid ab = ba,\ bc = cb,\ cd = dc,\ de = ed,\ ea = ae \rangle.$$

Then Γ is a graph product of free groups of rank one, generated by each of the five generators of Γ, with underlying graph a pentagon with edges giving the five commutation relations.

Each vertex group has a finite complete rewriting system of the form $\Lambda = \{i,\ \tilde{i}\}$ and $R = \{i\tilde{i} \to 1,\ \tilde{i}i \to 1\}$, where i represents one of the generators of Γ. Using the normal forms from these rewriting systems, and any ordering of the vertices, the corresponding set of proper words is not the set of normal forms of a finite complete rewriting system. For any ordering of the generators, there will be a sequence of three vertices in a row, reading in a clockwise or counterclockwise direction around the pentagon, in which the corresponding generators (without a tilde) are in increasing order. Suppose without loss of generality that $a < b < c$. Then the rewriting system must rewrite $ca^nb \xrightarrow{+} bca^n$ for any natural number n; this cannot be done with finitely many rules, all of which decrease the shortlex ordering. However, as will be shown in the proof of Theorem C, Γ does admit a finite complete rewriting system using another set of generators.

THEOREM C. *The graph product of finitely many groups (or monoids) which admit a complete rewriting system admits a canonical complete rewriting system. If the rewriting systems for the vertex groups (or monoids) are finite or regular, then the system for the graph product is also.*

Proof. The proof for a graph product of monoids is exactly the same as the proof for a graph product of groups; the proof given will be written in terms of groups.

Suppose that each vertex group G_I of a graph product Γ admits a complete rewriting system $(\mathcal{I}, R_I)$. Define a rewriting system for Γ as follows. For each vertex v_I, let 1_I represent the trivial word over the alphabet $\mathcal{I}$. The alphabet of the rewriting system is

$$\Lambda = \{\alpha = (\alpha_A, \alpha_B, ...\alpha_Z) \mid \alpha_I \in \mathcal{I} \cup \{1_I\} \text{ and } \{v_I \mid \alpha_I \neq 1_I\} \text{ is a nonempty set of vertices in a complete subgraph of } \mathcal{G}\}.$$

A complete subgraph of a graph $\mathcal{G}$ is a subgraph in which any two vertices are joined by an edge. In general a word in Λ^* will be denoted $\alpha_1...\alpha_n$ where each $\alpha_i = (\alpha_{iA}, ..., \alpha_{iZ})$. The rules of this system are given by

$R = \{1)\ (\alpha_A, ..., 1_I, ..., \alpha_Z)(\beta_A, ..., \beta_I, ..., \beta_Z) \to (\alpha_A, ..., \beta_I, ..., \alpha_Z)(\beta_A, ..., 1_I, ..., \beta_Z)$
if $\beta_I \neq 1_I$.

2) $\alpha_1\alpha_2...\alpha_m \to \beta_1\beta_2...\beta_n$ where:

i) $\alpha_{iJ} = \beta_{iJ}$ for all $1 \leq i \leq m$ and $J \neq I$.

ii) $\alpha_{1I}\alpha_{2I}...\alpha_{mI} \to \beta_{1I}\beta_{2I}...\beta_{kI}$ is a rule in R_I.

iii) If $k \leq m$, then $n = m$, and if $k < m$, $\beta_{iI} = 1_I$ for all $k < i \leq m$.

iv) If $k > m$, then $n = k$, and $\beta_{iJ} = 1_J$ for all $m < i \leq n$ and $J \neq I$. $\}$

These rules are presented in general form, assuming two conventions. The first is that if a symbol $(1_A, 1_B, ..., 1_Z)$ appears on the right hand side of a rule, it is replaced with the empty word 1_Λ of Λ^*. The second is that a rule occurs only if the letters exist; that is, if the letter $(\alpha_A, ..., \beta_I, ..., \alpha_Z)$ in rule 1) does not exist, then there is no rule.

Example. Suppose $\mathcal{G}$ is a graph with three vertices v_A, v_B, and v_C, and one edge joining v_A with v_B. Suppose also that the rewriting systems for the groups associated to the vertices are given by the following.

$$\begin{array}{lll} \mathcal{A} = \{a, \tilde{a}\}, & R_A = \{a^3 \to 1, & \tilde{a} \to a^2 \ \} \\ \mathcal{B} = \{b\}, & R_B = \{b^2 \to 1\} & \\ \mathcal{C} = \{c\}, & R_C = \{c^4 \to 1\} & \end{array}$$

Then the rewriting system for the graph product of these groups has alphabet

$$\Lambda = \{(a,1,1), (\tilde{a},1,1), (1,b,1), (1,1,c), (a,b,1), (\tilde{a},b,1)\}.$$

The set of rules for this example is

$$\begin{array}{lll} R = \{1) & (a,1,1)(1,b,1) \to (a,b,1) & (\tilde{a},1,1)(1,b,1) \to (\tilde{a},b,1) \\ & (a,1,1)(a,b,1) \to (a,b,1)(a,1,1) & (\tilde{a},1,1)(a,b,1) \to (\tilde{a},b,1)(a,1,1) \\ & (a,1,1)(\tilde{a},b,1) \to (a,b,1)(\tilde{a},1,1) & (\tilde{a},1,1)(\tilde{a},b,1) \to (\tilde{a},b,1)(\tilde{a},1,1) \\ & (1,b,1)(a,1,1) \to (a,b,1) & (1,b,1)(\tilde{a},1,1) \to (\tilde{a},b,1) \\ & (1,b,1)(a,b,1) \to (a,b,1)(1,b,1) & (1,b,1)(\tilde{a},b,1) \to (\tilde{a},b,1)(1,b,1) \\ 2) & (a,1,1)(a,1,1)(a,1,1) \to 1 & (a,1,1)(a,1,1)(a,b,1) \to (1,b,1) \\ & (a,1,1)(a,b,1)(a,1,1) \to (1,b,1) & (a,b,1)(a,1,1)(a,1,1) \to (1,b,1) \\ & (a,1,1)(a,b,1)(a,b,1) \to (1,b,1)(1,b,1) & \\ & (a,b,1)(a,1,1)(a,b,1) \to (1,b,1)(1,b,1) & \\ & (a,b,1)(a,b,1)(a,1,1) \to (1,b,1)(1,b,1) & \\ & (a,b,1)(a,b,1)(a,b,1) \to (1,b,1)(1,b,1)(1,b,1) & \\ & (\tilde{a},1,1) \to (a,1,1)(a,1,1) & (\tilde{a},b,1) \to (a,b,1)(a,1,1) \\ & (1,b,1)(1,b,1) \to 1 & (1,b,1)(a,b,1) \to (a,1,1) \\ & (a,b,1)(1,b,1) \to (a,1,1) & (a,b,1)(a,b,1) \to (a,1,1)(a,1,1) \\ & (1,b,1)(\tilde{a},b,1) \to (\tilde{a},1,1) & (\tilde{a},b,1)(1,b,1) \to (\tilde{a},1,1) \\ & (\tilde{a},b,1)(\tilde{a},b,1) \to (\tilde{a},1,1)(\tilde{a},1,1) & (a,b,1)(\tilde{a},b,1) \to (a,1,1)(\tilde{a},1,1) \\ & (\tilde{a},b,1)(a,b,1) \to (\tilde{a},1,1)(a,1,1) & (1,1,c)(1,1,c)(1,1,c)(1,1,c) \to 1\}. \end{array}$$

LEMMA 7.1. *The rewriting system (Λ, R) is a rewriting system for the graph product.*

Proof. Let M be the monoid presented by the rewriting system (Λ, R). Recall that Σ is the generating set for the graph product given by the union of

the generating sets for each vertex group. Define a map $\theta : \Sigma^* \to \Lambda^*$, by sending a generator $i_I \in \mathcal{I} \subseteq \Sigma$ to $(1_A, ..., i_I, ..., 1_Z)$, and let $\Sigma' = \theta(\Sigma)$. Rule 2) shows that all of the relations of the vertex groups G_I are satisfied by the generators in Σ'. Rule 1) implies the commutation relations of the graph product. So θ gives a map from the graph product Γ to M. Rule 1) also shows that the generators in Λ not in Σ' can be expressed as products of the generators in Σ'. The remaining occurrences of rules 1) and 2) are all implied by the relations of the graph product. Hence the monoid M presented by (Λ, R) is precisely the underlying monoid of the graph product Γ. □

LEMMA 7.2. *(Λ, R) is a Noetherian rewriting system.*

Proof. We will show that the rewriting system is Noetherian by showing that the ordering on Λ^* given by $v > w$ if $v \xrightarrow{+} w$ is well-founded. Each of the sets $\mathcal{I}$ has a well-founded ordering $>_I$ on it defined by the complete rewriting system on G_I. For each vertex v_I, define a map $\phi_I : \Lambda^* \to \mathcal{I}^*$ by mapping a word $\phi_I : \alpha_1...\alpha_m \mapsto \alpha_{1I}...\alpha_{mI}$, where symbols $\alpha_{iI} = 1_I$ represent the empty word. Let $\phi : \Lambda^* \to \mathcal{A}^* \times \mathcal{B}^* \times ... \times \mathcal{Z}^*$ be the map $\phi = (\phi_A, ..., \phi_Z)$. Define an ordering $>_\pi$ on $\mathcal{A}^* \times \mathcal{B}^* \times ... \times \mathcal{Z}^*$ by setting $(v_A, ..., v_Z) >_\pi (w_A, ..., w_Z)$ if $v_I \geq_I w_I$ for all I, with at least one of the inequalities being strict. This product of well-founded orderings is again a well-founded ordering.

Let $>_{lex}$ be a partial ShortLex ordering on Λ^* defined by the following partial ordering on Λ: $\alpha >_{lex} \beta$ if $|\{v_I \mid \alpha_I \neq 1_I\}| < |\{v_I \mid \beta_I \neq 1_I\}|$. (This definition is "contravariant" in order to be consistent with the "left greedy" rules of type 1).) This gives a well-founded ordering on Λ^*.

Suppose $v = \alpha_1...\alpha_m \to w = \beta_1...\beta_n$ is a rule of the rewriting system R. If it is of type 1), then $\phi_I(v) = \phi_I(w)$ for all I, and $v >_{lex} w$. If it is of type 2), then $\phi_J(v) = \phi_J(w)$ for all $J \neq I$, and $\phi_I(v) >_I \phi_I(w)$. It follows that for any sequence of rewritings $x \xrightarrow{+} y$, $\phi(x) \geq_\pi \phi(y)$, and if $\phi(x) = \phi(y)$, then $x >_{lex} y$.

To finish the proof, let Ξ be any subset of Λ^*. Let

$$\Xi_1 = \{w \in \Xi \mid w \text{ is minimal with respect to } >_\pi\};$$

this is a nonempty set since $>_\pi$ is a well-founded ordering. Let

$$\Xi_2 = \{w \in \Xi_1 \mid w \text{ is minimal with respect to } >_{lex}\};$$

again this is nonempty. Hence there is a nonempty subset, $\Xi_2 \subset \Xi$, whose elements are minimal with respect to the ordering on Λ^* defined by $x > y$ if $x \xrightarrow{+} y$, using the rewriting system R. Therefore, the system is Noetherian. □

LEMMA 7.3. (Λ, R) *is a complete presentation for* Γ.

Proof. We must show that the rewriting system is confluent. To do this, we will apply Newman's theorem; instead of proving confluence directly, we will prove that the set of irreducible words is a set of normal forms. The irreducible words over the rewriting system (Λ, R) are words of the form $\alpha_1\alpha_2...\alpha_m$ which satisfy local and obstruction conditions similar to those in §3.

(L) If $\alpha_{iJ} \neq 1_J$ for all $k \leq i \leq l$, then $\alpha_{kJ}...\alpha_{lJ}$ is an irreducible word for the rewriting system admitted by G_I.

(O) If $\alpha_{iJ} = 1_J$ and $\alpha_{(i+1)J} \neq 1_J$, then there is a vertex v_K which is not adjacent to v_J such that $\alpha_{iK} \neq 1_K$.

In the proof of Lemma 7.1, a map $\theta : \Sigma^* \to \Lambda^*$ was constructed. Suppose $\omega \in \Sigma^*$ is a proper word representing an element γ of the graph product. Applications of rule 1) of the rewriting system R to $\theta(\omega)$ essentially shuffle the letters in ω. Applying this rule as many times as possible, in any order, gives a unique word in Λ^*. Since ω is proper, rule 2) can not be applied at any stage of this shuffling, so this resulting word is irreducible; call this word $\iota(\omega)$. Note that $\iota(\omega)$ also represents γ in the graph product.

Define a map $\psi : \Lambda^* \to \Sigma^*$ on generators by sending $\psi : \alpha \mapsto \alpha_A\alpha_B...\alpha_Z$. Suppose $\alpha_1\alpha_2...\alpha_m \in \Lambda^*$ is an irreducible word. The pruning process applied to $\psi(\alpha_1\alpha_2...\alpha_m)$ again simply shuffles letters, in essence having the opposite effect that rule 1) has. Every subword of an irreducible local word is also irreducible, and the only irreducible representative of the trivial element is the trivial word for the vertex group rewriting systems. So in pruning $\psi(\alpha_1\alpha_2...\alpha_m)$, no representative of the identity is removed; therefore the pruning process in this case yields a unique word. Let $\nu(\alpha_1\alpha_2...\alpha_m)$ denote the pruning of $\psi(\alpha_1\alpha_2...\alpha_m)$. The characterization of irreducible words above implies that $\nu(\alpha_1\alpha_2...\alpha_m)$ is a proper word, representing the same element of the graph product as $\alpha_1\alpha_2...\alpha_m$.

The map $\iota \circ \nu$ is the identity map on the set of all irreducible words in Λ^*, and $\nu \circ \iota$ is the identity map on the set of all proper words in Σ^*, so there is a one-to-one correspondence between proper words and irreducible words, with corresponding words representing the same graph product elements. Therefore, the set of irreducible words also gives a normal form for the graph product. □

LEMMA 7.4. *If the complete rewriting systems for the vertex groups are regular (resp. finite), then so is* (Λ, R) .

Proof. It follows immediately from the definition of (Λ, R) that this rewriting system is finite if all of the rewriting systems $(\mathcal{I}, R_I)$ are finite.

Suppose that for each vertex v_I, $\mathcal{F}_I$ is a finite state automaton which accepts the regular language of irreducible words of a regular complete rewriting system for G_I. Since the set of irreducible words of $(\mathcal{I}, R_I)$ is closed under subwords, we may assume that $\mathcal{F}_I$ has only one non-accept state, which is a fail state and is not the initial state. As before, $\mathcal{F}_I$ will be considered as a finite directed graph. Let Q_I be the set of vertices, with initial state p_I and accept states P_I. For each vertex $q_I \in Q_I$ and each letter $\alpha_I \in \mathcal{I}$, let $\delta_I(q_I, \alpha_I)$ be the terminal vertex of the edge labeled α_I out of vertex q_I.

Define a finite state automaton $\mathcal{F}$ over the alphabet Λ whose vertex set is $Q = Q_A \times Q_B \times ... \times Q_Z \times \Lambda \cup \{F\}$. F will be a fail state, i.e. a non-accept state for which every outgoing edge is a loop. The initial state of $\mathcal{F}$ will be $(p_A, p_B, ..., p_Z, 1)$. We attach edges according to the following criteria.

a) Suppose $\alpha, \beta \in \Lambda$ and the following two conditions hold.

i) $\beta\alpha$ is irreducible with respect to rule 1) of the rewriting system R.

ii) $\delta_I(q_I, \alpha_I) \in P_I$ is one of the accept states of $\mathcal{F}_I$, for all I.

Then there is an edge labeled α whose initial vertex is

$$q = (q_A, q_B, ..., q_Z, \beta)$$

and whose terminal vertex is

$$\delta(q, \alpha) = (\delta_A(q_A, \alpha_A), \delta_B(q_B, \alpha_B), ..., \delta_Z(q_Z, \alpha_Z), \alpha),$$

where we denote $\delta_I(q_I, 1_I) = p_I$.

b) If at least one of the conditions i) and ii) does not hold, then $\delta(q, \alpha) = F$.

Finally, the accept states of $\mathcal{F}$ are defined to be the states in $P_A \times P_B \times ... \times P_Z \times \Lambda$. The finite state automaton $\mathcal{F}$ has as its language exactly the words in Λ^* which are irreducible with respect to R, so (Λ, R) is a regular complete presentation. □

Lemmas 7.1 through 7.4 complete the proof of Theorem C. □

Although it is natural to work with the set of generators used in sections three through six when constructing automatic structures, and intuitively easier, it is also possible to use the generators from the proof of Theorem C. Given automatic structures for the vertex groups, the images $\iota(\omega) \in \Lambda^*$ of proper words ω give normal forms for the graph product, as in the proof of Lemma 7.3. We

will continue to refer to these words as irreducible words, although they may not come from a rewriting system. Lemma 7.4 shows that these irreducible words give a regular language of normal forms.

In order to outline the proof that the combing defined by the irreducible words is bounded, the path of an irreducible word $\alpha_1...\alpha_n$ must be compared to the path of the irreducible word for the element $\alpha_1...\alpha_n\beta$, where $\beta \in \Lambda$. We can first first reduce to the case in which $\beta = (1_A, ..., \beta_I, ..., 1_Z)$ has only one nontrivial component; if in this case the two paths must stay within a bounded distance B_I, then in the general case they must stay within $B_A + ... + B_Z$. In order to find the normal form for $\alpha_1...\alpha_n\beta$ in this special case, the component β_I must be moved left as far as possible, as with rule 1) of the rewriting system in Theorem C, giving a new path which stays within a distance one of $\alpha_1...\alpha_n$. Finally, a word in the I component may have to be replaced by a normal form. If the width function for the automatic structure on the vertex group G_I is bounded by a constant C_I, then the normal form for $\alpha_1...\alpha_n\beta$ must stay within a distance $B_I = 1 + C_I$ of $\alpha_1...\alpha_n$. So the irreducible words also define a bounded combing, and an automatic structure for the graph product.

Acknowledgements

We thank Gary Gordon for creating the computer drawn figures used in this manuscript.

References

1. J. M. Alonso, Combings of groups, *in* "Algorithms and Classification in Combinatorial Group Theory," G. Baumslag and C. F. Miller III, eds., MSRI Publications 23, Springer, New York, 1992, 165-178.

2. J. M. Alonso and M. R. Bridson, Semihyperbolic groups, to appear, *Proc. London Math. Soc.*

3. G. Baumslag, S. M. Gersten, M. Shapiro, and H. Short, Automatic groups and amalgams, *J. Pure Appl. Algebra.* **76** (1991) 229-316.

4. R. V. Book and H.-N. Liu, Word problems and rewriting in a free partially commutative monoid, *Inf. Proc. Letters* **26** (1987/88) 29-33.

5. M. R. Bridson, On the geometry of normal forms in discrete groups, to appear, *Proc. London Math. Soc.*

6. B. Brink and R. Howlett, A finiteness property of Coxeter groups, *Math. Ann.* **296** (1993) 179-190.

7. R. M. Charney, Artin groups of finite type are biautomatic, *Math. Ann.* **292** (1992), 671-683.

8. I. M. Chiswell, Right-angled Coxeter groups, *in* "Low Dimensional Topology and Kleinian Groups," D. B. A. Epstein ed., London Math. Soc. Lecture Note Series 112, Cambridge University Press, Cambridge, 1986, 297-304.

9. I. M. Chiswell, The Euler characteristic of graph products of Coxeter groups, *in* "Discrete Groups and Geometry," W. J. Harvey and C. Machachlan, eds., London Math. Soc. Lecture Note Series 173, Cambridge University Press, Cambridge, 1992, 36-46.

10. I. M. Chiswell, The growth series of a graph product, preprint, Queen Mary and Westfield College, 1992.

11. V. Diekert, On Knuth-Bendix completion for concurrent processes, *Theoret. Comput. Sci.* **66** (1989), 117-136.

12. C. G. Droms, Graph groups, coherence, and three-manifolds, *J. Algebra* **102** (1987), 484-489.

13. C. G. Droms, Subgroups of graph groups, *J. Algebra* **110** (1987), 519-522.

14. D. B. A. Epstein, J. W. Cannon, D. F. Holt, S. V. F. Levy, M. S. Patterson, and W. P. Thurston, "Word Processing in Groups," Jones and Bartlett, Boston, 1992.

15. E. R. Green, Graph products of groups, Thesis, The University of Leeds, (1990).

16. M. Gromov, Hyperbolic groups, *in* "Essays in Group Theory" (S. M. Gersten, ed.), MSRI Publications 8, Springer, New York, 1987, 75-264.

17. M. Gromov, Asymptotic invariants of infinite groups, to appear *in* "Proceedings of Conference on Geometric Group Theory at the Isle of Thorns, July 1991," Cambridge University Press, Cambridge.

18. S. M. Hermiller, Rewriting systems for Coxeter groups, to appear, *J. Pure Appl. Algebra*.

19. M. Jantzen, "Confluent String Rewriting," EACTS monographs on Theoretical Computer Science 14, Springer, Berlin, 1988.

20. D. Kapur and H. Zhang, An overview of Rewrite Rule Laboratory (RRL), *in* "Rewriting Techniques and Applications" (N. Dershowitz, ed.), Lecture Notes in Computer Science 355, Springer, Berlin, 1989, 559-563.

21. M. H. A. Newman, On theories with a combinatorial definition of 'equivalence', *Ann. of Math.* **43** (1943), 223-243.

22. D. E. Peifer, Artin groups of extra-large type are automatic, Thesis, University of Illinois, 1992.

23. H. Servatius, Automorphisms of graph groups, *J. Algebra* **126** (1989), 34-60.

24. H. Short, Groups and combings, preprint, ENS Lyon, 1990.

25. L. A. VanWyk, Graph groups are biautomatic, to appear, *J. Pure Appl. Algebra*.