Communications in Algebra

Formal Languages and Groups as Memory

Mark Kambites [a]

[a] School of Mathematics , University of Manchester , Manchester, England, UK
Published online: 09 Oct 2009.

PLEASE SCROLL DOWN FOR ARTICLE

Taylor & Francis
Taylor & Francis Group

FORMAL LANGUAGES AND GROUPS AS MEMORY

Mark Kambites

School of Mathematics, University of Manchester,
Manchester, England, UK

*We present an exposition of the theory of **M**-automata and **G**-automata, or finite automata augmented with a multiply-only register storing an element of a given monoid or group. Included are a number of new results of a foundational nature. We illustrate our techniques with a group-theoretic interpretation and proof of a key theorem of Chomsky and Schützenberger from formal language theory.*

1. INTRODUCTION

In recent years, both computer scientists and pure mathematicians have become increasingly interested in the class of *M-automata*, or finite state automata augmented with a memory register which stores at any given time an element of a given monoid M. The register is initialised with the identity element of the monoid; while reading an input word the automaton can modify the register contents by multiplying by elements of the monoid. A word is accepted by the automaton if, having read the entire word, the automaton reaches a final state, with the register returned to the identity element.

Such automata have arisen repeatedly, both explicitly and implicitly, in the theory of computation. For example, the blind n-counter machines studied by Greibach (1978) are simply $\mathbb{Z}^n$-automata. Related examples have also been studied by Ibarra et al. (1976). *M*-automata are also equivalent to a class of regulated grammars, known as *regular valence grammars* (Fernau and Stiebe, 2002). More recently, there has been increasing interest in this idea from pure mathematicians, especially in the case that the register monoid is a group. An area of lasting interest in combinatorial group theory is the connection between structural properties of infinite discrete groups, and language theoretic properties of their *word problems*. Results of Gilman and Shapiro (1998), of Elston and Ostheimer (2004), and of

Kambites (2006) have demonstrated that M-automata can play a useful role in this area. Moreover, recent research of Kambites et al. (2007) has established a connection between the theory of G-automata and the *rational subset problem* for groups, allowing language-theoretic results to be applied to decision problems in group theory.

One aim of the present article is to provide a self-contained introduction to the theory of this important area, in a form intelligible both to pure mathematicians and to computer scientists. In doing so, we aim to make explicit and transparent the connections between the computational and the algebraic approaches to M-automata and G-automata. This is especially important since, to date, group theorists, semigroup theorists, and computer scientists have all worked on this topic, often unaware of crucial results from the other discipline. Another objective is to establish a number of new results of a foundational nature.

In addition to this introduction, this article comprises three sections. Section 2 briefly recalls some necessary preliminaries, and gives suggested references to more detailed treatments. Section 3 introduces M-automata and the families of languages they define. We also explain their relationship to the theory of rational transductions, and study the connection between algebraic properties of monoids and closure properties of the language classes they define. In Section 4, we proceed to show how M-automata techniques can be used to obtain a group-theoretic interpretation and proof of one of the most important results in formal language theory, namely, the Chomsky–Schützenberger theorem characterising context-free languages as the images under rational transductions of 2-sided Dyck languages (Chomsky and Schützenberger, 1963).

2. PRELIMINARIES

In this section, we introduce the basic definitions which will be required in this article. We begin with a very brief introduction to formal languages and automata; a more comprehensive exposition can be found in any of the numerous texts on the subject, such as Hopcroft and Ullman (1969). We assume a familiarity with some basic definitions from algebra, such as semigroups and monoids (Howie, 1995), groups, generating sets, and presentations (Lyndon and Schupp, 1977; Magnus et al., 1976). More specialized notions from algebra will be defined as and when they are needed.

Let Σ be a finite set of symbols, called an *alphabet*. A *word* over Σ is a finite sequence of zero or more symbols from Σ; the unique *empty word* of length zero is denoted ϵ. The set of all words over Σ forms a monoid under the operation of *concatenation*; this is called the *free monoid* on Σ and denoted Σ^*. A *language* over Σ is a set of words over Σ, that is, a subset of the free monoid Σ^*.

A *finite automaton* over a monoid M is a finite directed graph, possibly with loops and multiple edges, with each edge labelled by element of M, together with a designated *initial vertex* and a set of designated *terminal vertices*. The vertices and edges of an automaton are often called *states* and *transitions*, respectively. The labelling of edges extends naturally to a labelling of (directed) paths by elements of M. The *subset accepted* or *recognised* by the automaton is the set of all elements of M which label paths between the initial vertex and some terminal vertex. A subset

recognised by some automaton is called a *rational subset of M*. Notice that finitely generated submonoids of M are examples of rational subsets.

Of particular interest is the case where $M = \Sigma^*$ is a free monoid on an alphabet Σ, so that the automaton accepts a set of words over Σ, that is, a language over Σ. A language accepted by such an automaton is called a *rational language* or a *regular language*. The formal study of languages in general, and of regular languages in particular, is of fundamental importance in theoretical computer science, and increasingly also in combinatorial algebra.

Another interesting case is that where $M = \Sigma^* \times \Omega^*$ is a direct product of free monoids. Such an automaton is called a *finite transducer* from Σ^* to Ω^*; it recognises a relation, termed a *rational transduction*. Relations between free monoids, and rational transductions in particular, are a powerful tool for studying relationships between languages. If $R \subseteq \Sigma^* \times \Omega^*$ then we say that the *image* of a language $L \subseteq \Sigma^*$ under R is the language of all words $v \in \Omega^*$ such that $(u, v) \in R$ for some $u \in L$. We say that a language K is a *rational transduction* of a language L if K is the image of L under some rational transduction. For a detailed exposition of the theory of rational transductions, see Berstel (1979).

3. *M*-AUTOMATA AND *G*-AUTOMATA

In this section, we introduce the definitions and some basic properties of M-automata and G-automata. Let M be a monoid with identity 1 and Σ a finite alphabet. An *M-automaton over Σ* is a finite automaton over the direct product monoid $M \times \Sigma^*$. For simplicity, we assume that the edges are labelled by elements of $M \times (\Sigma \cup \{\epsilon\})$. We identify the free monoid Σ^* with its natural embedding into $M \times \Sigma^*$ as $\{1\} \times \Sigma^*$; thus, a word $w \in \Sigma^*$ is *accepted* by the automaton if there is a path from the initial vertex to a terminal vertex labelled $(1, w) \in M \times \Sigma^*$. The *language accepted* by the automaton is the set of all words in Σ^* accepted by the automaton; it is the intersection of the *subset* accepted with the embedded copy $\{1\} \times \Sigma^*$ of Σ^*. We denote by $\mathbb{F}(M)$ the family of all languages accepted by M-automata.

From a mathematical perspective, then, the theory of M-automata can be viewed as an attempt to understand certain properties of a space (the free monoid Σ^*) by embedding it into a larger space ($M \times \Sigma^*$) with more structure; in this sense, it is very roughly analoguous to the embedding of the real numbers into the complex numbers. From a computational perspective, an M-automaton can be thought of as a (nondeterministic) finite automaton augmented with an extra memory register, which stores at any point an element of the monoid M. The register is initialised with the identity element of the monoid, and at each stage in its operation, the automaton can modify the contents of the register by multiplication on the right by some element of the monoid M. Of course, it can also leave the register unchanged, simply by multiplying by the identity. The automaton cannot read the register during operation, but the contents act as an extra barrier to acceptance—a word is accepted only if reading it can result in reaching a final state in which the register value has returned to the identity.

As an example, recall that a *blind n-counter automaton* is a finite automaton augmented with n registers, each of which stores a single integer value (Greibach, 1978). The registers can be incremented and decremented but not read; they are

initialised to zero, and a word is accepted exactly if, when reading it, the automaton can reach a final state with all registers returned to zero. In view of the discussion above, it is clear that a blind n-counter automaton is essentially the same thing as a $\mathbb{Z}^n$-automaton, where $\mathbb{Z}$ denotes the group of integers under addition, that is, the infinite cyclic group.

Notice that we have not required that the register monoid be finitely generated. However, the following elementary observation will often allow us to restrict attention to the case in which it is.

Proposition 1. *Let M be a monoid, and suppose that L is accepted by an M-automaton. Then there exists a finitely generated submonoid N of M such that L is accepted by a N-automaton.*

Proof. Since an M-automaton has finitely many edges, only finitely many elements of M can feature as the left-hand component of edge labels in a given automaton. Clearly, the register can only ever hold values in the submonoid N of M generated by these elements, so it suffices to view the automaton as an N-automaton. $\square$

Notwithstanding Proposition 1, it is occasionally useful to consider the *class* of all M-automata where M is not finitely generated, since the corresponding class of languages (the union of the classes corresponding to the finitely generated submonoids of M) may not be defined by a single finitely generated monoid. See, for example, Gilman and Shapiro (1998, Theorem 6.2), for an application of this approach.

If M is a monoid generated by a set X, we say that the *identity language $W_X(M)$* of M with respect to X is the set of all words over X representing the identity. In the case M is a group, the identity language is traditionally called the *word problem* of the group; this terminology is justified by the fact that the membership problem of this language is algorithmically equivalent to the problem of deciding whether two given words represent the same element of the group, that is, to the word problem in the sense of universal algebra. In a general monoid there is no such equivalence, and so the term word problem is less appropriate.

The following simple observation has been made by several authors (see, for example, Gilman and Shapiro, 1998), but apparently overlooked by a number of others. It allows us to apply many standard results from the theory of formal languages to the study of M-automata.

Proposition 2. *Let L be a language and M a finitely generated monoid. Then the following are equivalent*:

(i) *L is accepted by an M-automaton*;
(ii) *L is a rational transduction of the identity language of M with respect to some finite generating set*;
(iii) *L is a rational transduction of the identity language of M with respect to every finite generating set.*

Proof. We begin by proving the equivalence of (i) and (ii). First suppose (i) holds, and let A be an M-automaton accepting L. Let S be the (necessarily finite) set of

elements of M which occur on the left-hand-side of edge labels in A. Extend S to a finite generating set X for M. Now A can be viewed as a finite transducer from X^* to Σ^*. It follows easily from the relevant definitions that the image of $W_X(M)$ is exactly the language L, so that (ii) holds.

Conversely, suppose (ii) holds. Then there is a finite generating set X for M and a finite transducer A from X^* to Σ^* such that L is the image of $W_X(M)$ under A. We obtain from A an M-automaton, by replacing each edge label (w, x) with (m, x) where m is the element of M represented by $w \in X^*$. Again, it follows easily from the definitions that the resulting M-automaton accepts exactly the language L.

Since the monoid is assumed to be finitely generated, it is immediate that (iii) implies (ii). It remains only to show that (ii) implies (iii). Let Y and X be finite generating sets for a monoid M. For each symbol $a \in X$, choose a word $w_a \in Y^*$ representing the same element of M. Now let R be the submonoid of $Y^* \times X^*$ generated by the (finitely many) pairs of the form (w_a, a). R is a finitely generated submonoid, and hence also, by our observations above, a rational transduction. Now if L is the image of $W_X(M)$ under a rational transduction S then the relational composition

$$R \circ S = \{(u, w) \mid (u, v) \in R, (v, w) \in S \text{ for some } v \in X^*\}$$

is a rational transduction (Berstel, 1979, Theorem II.4.4), and it is easily verified that L is the image of $W_Y(M)$ under $R \circ S$. □

Proposition 2 tells us that the theory of M-automata can be viewed as a special case of the well-established field of rational transductions. Indeed, we can easily and profitably translate a large body of existing theory concerning rational transductions into the M-automaton setting. For two main reasons, however, the study of M-automata retains a distinct flavour, and remains of interest in its own right. Firstly, the structure of the register monoid can be used to prove interesting things about the accepting power of M-automata. Secondly, M-automata can be used to gain insight into computational and language-theoretic aspects of monoids. Both of these factors have special weight in the case that the register monoid is a group, with all the extra structure that entails.

The following result, which has been observed independently by several authors (Corson, 2005; Elder, 2007), is an immediate corollary of Proposition 2 together with the fact that rational transductions are closed under composition (Berstel, 1979, Theorem III.4.4).

Corollary 3. *Let M and N be finitely generated monoids. Then the identity language of N is accepted by an M-automaton if and only if every language accepted by an N-automaton is accepted by an M-automaton.*

We end this section with a discussion of closure properties of language families of the form $\mathbb{F}(M)$. First, we observe that every language class of the form $\mathbb{F}(M)$ is easily seen to be closed under finite union, as a simple consequence of nondeterminism. On the other hand, a class $\mathbb{F}(M)$ need not be closed under intersection. The following theorem will allow us to provide a straightforward characterisation of when such a class is intersection-closed. The converse part was

essentially proved by Mitrana and Stiebe (2001) in the case that the register monoids are groups. Recall that a morphism between free monoids is called *alphabetic* if it maps each letter of the domain alphabet to either a single letter or the empty word; one language is said to be *an alphabetic morphism of* another if the former is the image of the latter under an alphabetic morphism of free monoids.

Theorem 4. *Let $M_1, M_2, \ldots M_n$ be monoids. Then a language is accepted by an $(M_1 \times \cdots \times M_n)$-automaton if and only if it is an alphabetic morphism of a language of the form $L_1 \cap \cdots \cap L_n$ where each L_i is accepted by an M_i-automaton.*

Proof. Suppose first that $L \subseteq \Sigma^*$ is accepted by an $(M_1 \times \cdots \times M_n)$-automaton. It follows easily from Proposition 1 that L is accepted by an $(M'_1 \times \cdots \times M'_n)$-automaton where each M'_i is a finitely generated submonoid of M_i. For each M'_i choose a finite generating set X_i and let $X = X_1 \cup \cdots \cup X_n$ so that X is a finite generating set for $M'_1 \times \cdots \times M'_n$. Now by Proposition 2, L is the image of the identity language $W = W_X(M'_1 \times \cdots \times M'_n)$ under some rational transduction $\rho \subseteq X^* \times \Sigma^*$.

   We claim that W can be written as

$$W = K_1 \cap \cdots \cap K_n,$$

where each $K_i \in X^*$ lies in $\mathbb{F}(M'_i)$. Let K_i be the set of all words $w \in X^*$ such that w represents a tuple in $M'_1 \times \cdots \times M'_n$ with the identity element 1 in the ith position. It is readily verified that K_i is accepted by an M'_i-automaton with a single (initial and terminal) state q, and a loop at q labelled (g, x) whenever the letter $x \in X$ represents a tuple with g in the ith position. Now a word w represents the identity of $M'_1 \times \cdots \times M'_n$ if and only if w lies in K_i for all i, so that W is the intersection of the K_i as required.

   Now by Berstel (1979, Theorem III.4.1), there exists an alphabet Z, a regular language $R \subseteq Z^*$ and two alphabetic morphisms $\alpha : Z^* \to X^*$ and $\beta : Z^* \to \Sigma^*$ such that

$$L = W\rho = (W\alpha^{-1} \cap R)\beta = ((K_1 \cap \cdots \cap K_n)\alpha^{-1} \cap R)\beta.$$

It is easy to check that inverse morphisms distribute over intersection, so that

$$(K_1 \cap \cdots \cap K_n)\alpha^{-1} = K_1\alpha^{-1} \cap \cdots \cap K_n\alpha^{-1}.$$

Thus we obtain

$$L = ((K_1\alpha^{-1} \cap \cdots \cap K_n\alpha^{-1}) \cap R)\beta = ((K_1\alpha^{-1} \cap R) \cap \cdots \cap (K_n\alpha^{-1} \cap R))\beta.$$

Each $\mathbb{F}(M'_i)$ is closed under rational transductions, and hence also under inverse morphisms and intersection with regular languages. It follows that

$$(K_i\alpha^{-1} \cap R) \in \mathbb{F}(M'_i) \subseteq \mathbb{F}(M_i)$$

for each i, so setting $L_i = (K_i\alpha^{-1} \cap R)$ completes the proof of the direct implication.

For the converse, it is easy to check that the graph of an alphabetic morphism between free monoids is a rational transduction; it follows that the family $\mathbb{F}(M_1 \times \cdots \times M_n)$ is closed under alphabetic morphisms. Thus, it suffices to suppose $L = L_1 \cap \cdots \cap L_n$ where each L_i is in $\mathbb{F}(M_i)$, and show that $L \in \mathbb{F}(M_1 \times \cdots \times M_n)$. For this, it is clearly sufficient to prove the case $n = 2$ and then apply induction. Suppose, then, that $L_1, L_2 \subseteq \Sigma^*$ and that each L_i is accepted and by an M_i-automaton A_i with state set Q_i. Recall that the edges of each A_i are labelled by elements of $M_i' \times (\Sigma \cup \{\epsilon\})$. We define an $(M_1 \times M_2)$-automaton A with:

- state set $Q_1 \times Q_2$;
- an edge from (s, u) to (t, v) labelled $((x, y), a)$ whenever A_1 has an edge from s to t labelled (x, a) and A_2 has an edge from u to v labelled (y, a) for some letter $a \in \Sigma$;
- an edge from (s, u) to (t, u) labelled $((x, 1), \epsilon)$ whenever A_1 has an edge from s to t labelled (x, ϵ);
- an edge from (s, u) to (s, v) labelled $((1, y), \epsilon)$ whenever A_2 has an edge from u to v labelled (y, ϵ);
- start state (p_0, q_0) where p_0 and q_0 are the start states of A_1 and A_2 respectively; and
- final states (p, q) such that p and q are final states of A_1 and A_2 respectively.

It is an easy exercise to verify that A accepts exactly the intersection $L = L_1 \cap L_2$. $\square$

Returning to our example, an elementary application of Theorem 4 is the following characterisation of the classes of languages accepted by blind n-counter automata.

Corollary 5. *A language is recognised by a blind n-counter automaton if and only if it is an alphabetic morphism of the intersection of n languages recognised by blind 1-counter automata.*

We also obtain a characterisation of those monoids M for which $\mathbb{F}(M)$ is closed under intersection.

Corollary 6. *Let M be a finitely generated monoid. Then $\mathbb{F}(M)$ is closed under finite intersection if and only if there exists an M-automaton accepting the identity language of $M \times M$.*

Proof. If there exists an M-automaton accepting the identity language of $M \times M$ then by Corollary 3, every language recognised by an $(M \times M)$-automaton is recognised by an M-automaton. But by Theorem 4, this means that every intersection of two M-automaton languages is recognised by an M-automaton. It follows by induction that $\mathbb{F}(M)$ is closed under finite intersection.

Conversely, the identity language of $M \times M$ is certainly recognised by an $(M \times M)$-automaton, and hence by Theorem 4 is an alphabetic morphism of the intersection of two languages in $\mathbb{F}(M)$. Thus, if the latter is intersection-closed then it contains the identity language of $M \times M$, as required. $\square$

Corollary 6 has a particularly interesting interpretation in the case that the monoid M is a free group. We shall see in Section 4 below that a language is context-free if and only if it is recognised by a free group automaton. A well known theorem of Muller and Schupp (1983), combined with a subsequent result of Dunwoody (1985), tells us that a finitely generated group has context-free word problem if and only if it is *virtually free*, that is, has a free subgroup of finite index. It follows that the fact that context-free languages are not intersection closed can be viewed as a manifestation of the fact that a direct product of virtually free groups is not, in general, virtually free.

4. FREE GROUPS AND CONTEXT-FREE LANGUAGES

In this section we consider M-automata where M is drawn from two particularly significant classes of monoids; namely, polycyclic monoids and free groups. From the perspective of algebra, these may be considered the motivating examples for the subject. We observe that an important theorem of Chomsky and Schützenberger (1963) has a natural interpretation in terms of M-automata, and show how M-automata techniques can be used to provide an algebraic and automata-theoretic proof of the theorem.

We begin by recalling a basic definition from automata theory. Let X be a finite alphabet. A *pushdown store* or *stack* with alphabet X is a storage device which stores, at any one time, a finite but unbounded sequence of symbols from X. The basic operations permitted are appending a new symbol to the right-hand end of the sequence ("pushing" a symbol), removing the rightmost symbol from the sequence ("popping") and reading the rightmost symbol on the stack.

The possible configurations of a pushdown store are naturally modelled by elements of the free monoid X^*. The operations of pushing and popping can be modelled by partial functions, defined upon subsets of X^*. Specifically, for each symbol $x \in X$, we define a function

$$P_x : X^* \to X^* \qquad w \mapsto wx,$$

which models the operation pushing the symbol x onto the stack. The corresponding operation of popping x can only be performed when the stack is in certain configurations—namely, when it has an x as the rightmost symbol—and so is modelled by a partial function.

$$Q_x : X^*x \to X^* \qquad wx \mapsto w.$$

The set of functions

$$\{P_x, Q_x \mid x \in X\}$$

generates a submonoid of the monoid of all partial functions on X^*, under the natural operation of composition. This monoid, which was first explicitly studied by Nivat and Perrot (1970), is called the *polycyclic monoid on X* and denoted $P(X)$. The elements of $P(X)$ encapsulate the various sequences of operations which can be performed upon a pushdown store with alphabet X. The *rank* of $P(X)$ is defined to

be the size $|X|$ of the alphabet X; a polycyclic monoid is uniquely determined (up to isomorphism) by its rank. Polycyclic monoids also arise naturally in the structural theory of semigroups; of particular importance is that of rank 1, which is known as the *bicyclic monoid*. For more general information see Lawson (1998, Section 9.3).

There is a natural embedding of the free monoid X^* into the polycyclic monoid $P(X)$, which takes each symbol x to P_x. With this in mind, we shall identify P_x with x itself. The element Q_x is an *inverse* to P_x in the sense of inverse semigroup theory (Lawson, 1998); hence, we shall denote it x^{-1}. Thus, the monoid $P(X)$ is simply generated by the set

$$\overline{X} = \{x, x^{-1} \mid x \in X\}$$

and we shall have no further need of the notation P_x and Q_x.

The inversion operation extends to the whole of $P(X)$, by defining $(x^{-1})^{-1} = x$ for each $x \in X$, and $(x_1 \ldots x_n)^{-1} = x_n^{-1} \ldots x_1^{-1}$ for $x_1, \ldots, x_n \in \overline{X}$. If x and y are distinct elements of X then the product xy^{-1} is the *empty partial function* and forms a *zero* in the semigroup $P(X)$.

We shall define a *pushdown automaton* to be a polycyclic monoid automaton. The equivalence of this definition to the standard one is straightforward and well-documented, for example in Gilman and Shapiro (1998). For those familiar with the usual definition, we remark that the top symbol on the stack corresponds to the rightmost position of the word, and that the automaton accepts with empty stack.

The attentive reader may have noticed that our polycyclic monoid model of the pushdown store does not provide explicitly for "reading" the contents of the stack. However, a polycyclic monoid automaton can use nondeterminism to test the rightmost stack symbol, by attempting to pop every possible symbol and moving to different states depending upon which succeeds; all but one attempt will result in the register containing a zero value, which effectively constitutes failure. This is an example of a more general phenomenon, in which the apparent blindness of an M-automaton can be overcome by the use of nondeterminism.

The languages accepted by pushdown automata are called *context-free*. The class of context-free languages, which also admits an equivalent definition in terms of generating grammars (Berstel, 1979), is one of the most important languages classes in computer science.

We recall also a key notion from combinatorial group theory. Recall that the *free group* on the alphabet X is the group defined by the monoid presentation

$$\langle \overline{X} \mid xx^{-1} = x^{-1}x = 1 \text{ for all } x \in X \rangle.$$

Generators from X we shall call *positive generators*, while those from $\overline{X} \backslash X$ are *negative generators*. Free groups are of central importance in combinatorial and geometric group theory; see Lyndon and Schupp (1977) or Magnus et al. (1976) for a detailed introduction.

The identity languages of the free group $F(X)$ and the polycyclic monoid $P(X)$, with respect to the standard generating set $\overline{X}$, are well known in formal language theory. They are called the *2-sided Dyck language* (or just *Dyck language*) on $\overline{X}$ and the *1-sided Dyck language* (or *restricted Dyck language* or *semi-Dyck language*) on $\overline{X}$, respectively. The former consists of all words over $\overline{X}$ which can be reduced to the

empty word by successive deletion of factors of the form xx^{-1} or $x^{-1}x$ where $x \in X$. The latter contains all words which can be reduced to the empty word by deleting only factors of the form xx^{-1} with $x \in X$. In particular, we see that the latter is (strictly) contained in the former. Thus, any word over $\overline{X}$ representing the identity in $P(X)$ also represents the identity in $F(X)$; the converse does not hold.

A well known theorem of Chomsky and Schützenberger states that the context-free languages are exactly the rational transductions of 1-sided Dyck languages, and of 2-sided Dyck languages (Chomsky and Schützenberger, 1963, Proposition 2). By Proposition 2, this result has the following interpretation in the M-automaton setting.

Theorem 7 (Chomsky and Schützenberger, 1963). *Let L be a language. Then the following are equivalent*:

(i) *L is context-free*;
(ii) *L is accepted by a polycyclic monoid [of rank 2] automaton*;
(iii) *L is accepted by a free group [of rank 2] automaton.*

We have already remarked that the equivalence of (i) and (ii) is the usual equivalence of context-free grammars and pushdown automata, and so is well known. It is also straightforward to show that a pushdown automaton can simulate a free group automaton, so that (iii) implies (ii); this is left as an exercise for the interested reader. The restriction to polycyclic monoids (respectively, free groups) of rank 2 is a simple consequence of the well known fact that every polycyclic monoid (free group) of countable rank embeds into the polycyclic monoid (free group) of rank 2. What remains, which is the real burden of the proof, is to show that (i) and/or (ii) implies (iii).

The original proof of Chomsky and Schützenberger starts with a context-free grammar, and produces from it an appropriate rational transduction; an example of this approach can be found in Berstel (1979). A direct group-theoretic proof of this result was claimed by Dassow and Mitrana (2000); however, their construction was fundamentally flawed (Corson, 2005). A correct algebraic proof has recently been provided by Corson (2005), who exhibited a free group automaton accepting the identity language of a polycyclic monoid automaton. The authors of both Dassow and Mitrana (2000) and Corson (2005) appear to have overlooked the equivalence of the statement to the theorem of Chomsky and Schützenberger.

Theorem 7 is quite surprising, in view of our comments above regarding the method used by a polycyclic monoid automaton to read the rightmost symbol of the stack. A polycyclic monoid automaton apparently makes fundamental use of its ability to "fail," by reaching a zero configuration of the register monoid. Since a free group has no zero, a free group automaton seems to have no such capability, and appears to be "blind" in a much more fundamental way. However, it transpires that a carefully constructed interplay between the finite state control and the group register can achieve the desired "failing" effect.

We remark also upon an interesting corollary to Theorem 7. It is well known that every recursively enumerable language is a homomorphic image (and hence a rational transduction) of the intersection of two context-free languages. Hence,

combining Theorems 4 and 7, we immediately obtain the following result, which was first observed in the group case by Mitrana and Stiebe (2001).

Theorem 8. *Let M be a free group of rank 2 or more, or a polycyclic monoid of rank 2 or more. Then $\mathbb{F}(M \times M)$ is the class of all recursively enumerable languages.*

In the rest of this section, we present an alternative group- and automata-theoretic proof of Theorem 7. In particular, we show explicitly how a free group automaton can simulate the operation of a pushdown automaton. In the process, we also obtain some technical results relating polycyclic monoids to free groups, which may be of independent interest. We begin by introducing a construction of a free group automaton from a polycyclic monoid automaton, that is, a pushdown automaton.

Suppose $L \subseteq \Sigma^*$ is the language accepted by a $P(X)$-automaton A with state set Q, that is, by a pushdown automaton with state set Q and stack alphabet X. Let # be a new symbol not in X, and let $X^{\#}$ denote the alphabet $X \cup \{\#\}$. We construct from A an new finite automaton A' with edges labelled by elements of $(\overline{X^{\#}}^* \times \Sigma^*)$. It has:

- state set $Q' = Q_- \cup Q_+$ where

$$Q_+ = \{q_+ \mid q \in Q\} \text{ and } Q_- = \{q_- \mid q \in Q\}$$

  are disjoint sets in bijective correspondence with Q;
- start state q_+ where q is the start state of A;
- final states of the form q_- where q is a final state of A;
- an edge from p_+ to q_+ labelled $(x\#, w)$ whenever A has an edge from p to q labelled (x, w) with x a positive generator.
- an edge from p_- to q_+ labelled $(x'\#, w)$ whenever A has an edge from p to q labelled by (x', w) with x' a negative generator;
- an edge from p_+ to q_+ labelled (ϵ, w) whenever A has an edge from p to q labelled (ϵ, w);
- for each $q \in Q$, an edge from q_+ to q_- labelled (ϵ, ϵ); and
- for each $q \in Q$, a loop at state q_- labelled $(\#^{-1}, \epsilon)$.

The automaton A' can be interpreted either as a $P(X^{\#})$-automaton or as an $F(X^{\#})$-automaton; it transpires that the language accepted is the same for each choice. Figure 1 illustrates a $P(\{x\})$-automaton accepting the 1-sided Dyck language on the alphabet $\{a, b\}$, together with the automaton constructed from it by the procedure above.

In general, we make the following claim.

Theorem 9. *The free group automaton A' and the polycyclic monoid automaton A' both accept exactly the language L.*

The rest of this section is devoted to the proof of Theorem 9. We shall need a number of preliminary definitions and results.
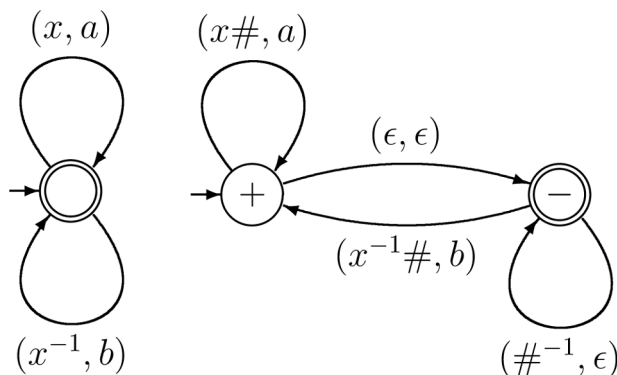
Figure 1 A pushdown automaton (left) and a free group or pushdown automaton (right), both accepting the 1-sided Dyck language on a two-letter alphabet.

Definition 10. Let $w_1, \ldots, w_n \in \overline{X}$ and $w = w_1 \ldots w_n \in \overline{X}^*$. Let # be a new symbol not in $\overline{X}$. A *permissible padding* of w is a word of the form

$$x_1 x_2 \ldots x_n (\#^{-1})^k$$

where $k \in \mathbb{N}_0$ and for each $i \in \{1 \ldots n\}$, we have

$$x_i = \begin{cases} (\#^{-1})^m w_i \# \text{ for some } m \in \mathbb{N}^0 & \text{if } w_i \text{ is a negative generator;} \\ w_i \# & \text{if } w_i \text{ is a positive generator.} \end{cases}$$

Thus, a permissible padding of w is obtained by inserting the symbol # after every generator in w, and zero or more $\#^{-1}$s before each *negative* generator, and at the end of the word.

The following lemma connects the above definition to our free group automaton construction; it can be routinely verified.

Lemma 11. *The automaton A' accepts (x, w) if and only if there exists a word $y \in \overline{X}^*$ such that x is a permissible padding of y, and (y, w) is accepted by A.*

We shall also use the following straightforward lemma concerning words representing the identity in the free group.

Lemma 12. *Let $w \in \overline{X}^*$ be a word representing the identity in a free group $F(X)$, and suppose $w = uxv$ where u, v are words and $x \in \overline{X}$. Then either u has a suffix $x^{-1}e$ where e represents the identity, or v has a prefix ex^{-1} where e represents the identity.*

Proof. We have seen that any word in the 2-sided Dyck language, that is, any word representing the identity in the free group, can be reduced to the empty word by successively removing factors of the form xx^{-1} and $x^{-1}x$ where $x \in X$. Such a reduction process for w must eventually bring the given occurrence of the generator x next to some occurrence x^{-1}, by deleting the letters between them.

But the product of these letters must be a factor representing the identity; setting e equal to this factor will give an appropriate factorisation of either u or v (depending upon whether the given occurrence of x^{-1} occurs before or after that of x). $\qquad\square$

Recall that an element of the free group $F(X)$ is called *positive* if it can be written as a product of one or more positive generators. The following definition facilitates a geometric interpretation of the positive elements.

Definition 13. Let $w \in \overline{X}^*$, and let $x \in F(X)$. We say that x is a *minimum* of w, if:

(i) w has a prefix representing x; and
(ii) no prefix of w which represents x is immediately followed by a negative generator.

Proposition 14. *Let w be a word representing the identity in $F(X)$. Then the following are equivalent*:

 (i) *w represents the identity in the polycyclic monoid $P(X)$*;
(ii) *every prefix of w represents a positive or identity element*;
(iii) *the only minimum of w is the identity of $F(X)$*.

Proof. The equivalence of (i) and (ii) is well known, and easily deduced from the definitions.

Suppose now that (ii) holds, that is, that every prefix of w represents a positive or identity element. It is easily seen that the identity is a minimum of w. Moreover, if x is a nonidentity element represented by a prefix of w then consider the longest prefix of w representing x. Considering the path traced through the Cayley graph of $F(X)$, and recalling that the latter is a tree, it is clear that the letter following this prefix must be a negative generator. Thus, x cannot be a minimum of w, and so (iii) is satisfied.

Conversely, suppose that (ii) does not hold, that is, that some prefix of w represents a nonpositive, nonidentity element. Suppose further for a contradiction that (iii) holds, that is, that the identity is the only minimum of w. Let e be a nonpositive, nonidentity element represented by a prefix of w. Since e is not a minimum for w, there is a prefix u of w representing e, which is followed by a negative generator x_1^{-1}. But now ux_1^{-1} represents another nonpositive, nonidentity element. Continuing in this way, we obtain an infinite sequence of prefixes of u, which must clearly all represent distinct elements. Since w is a finite word, this gives the required contradiction. $\qquad\square$

Lemma 15. *Let $w \in \overline{X^\#}^*$ be a word representing the identity in $P(X^\#)$ and suppose $w = uv$. Then there exists a factorisation $v = st$ such that*:

 (i) *either $t = \epsilon$ or t begins with a negative generator; and*
(ii) *$u\#s\#^{-1}t$ also represents the identity in $P(X^\#)$*.

Proof. If u represents the identity, then v also represents the identity, so it suffices to take $s = v$ and $t = \epsilon$.

Assume now that u does not represent the identity, and consider the path traced through the Cayley graph of the free group $F(X^\#)$ when starting from the identity and reading w. Since w represents the identity in $P(X^\#)$, it also represents the identity in $F(X^\#)$, so having reached the element represented by u, this path must return to the identity. Since the Cayley graph is a tree, the path must either leave in the direction of the identity, in which case we take $s = \epsilon$, or leave away from the identity and then return to the element represented by u having read a word s, before leaving in the direction of the identity. By Proposition 14, u represents a positive element, so "in the direction of the identity" means following a negative generator. Defining t to be such that $v = st$, it is now clear that s and the corresponding t have the desired properties.   □

The following proposition, which may also be of interest in its own right, is the main step in the proof.

Proposition 16.   *Let $w \in \overline{X}^*$. Then the following are equivalent*:

(i) *w represents 1 in the polycyclic monoid $P(X)$;*
(ii) *w admits a permissible padding which represents 1 in the polycyclic monoid $P(X^\#)$;*
(iii) *w admits a permissible padding which represents 1 in the free group $F(X^\#)$.*

Proof.   First suppose (i) holds. By repeated application of Lemma 15, we can insert the symbol # between every pair of generators and the symbol $\#^{-1}$ in appropriate places, so as to obtain a permissible padding of w which represents 1 in $P(X^\#)$. Thus, (ii) holds.

Clearly, every word representing the identity in $P(X^\#)$ also represents the identity in $F(X^\#)$, so that (ii) implies (iii).

Finally, suppose (iii) holds, and let w' be a permissible padding of w which represents 1 in $F(X^\#)$. Suppose for a contradiction that w does not represent 1 in $P(X)$. Certainly, since w' represents 1 in $F(X^\#)$, we must have that w represents 1 in $F(X)$. So by Proposition 14, w contains a minimum which is not the identity. Let u be the shortest prefix of w representing this minimum, and write $w = uv$.

It follows that we can write $w' = u'\#v'$ where $u'\#$ and v' are paddings of u and v, respectively. Certainly, since u is the shortest prefix representing the given element, u has no nonempty suffix representing the identity in $F(X)$. It follows that u' also has no nonempty suffix representing the identity. Since u' cannot end in $\#^{-1}$, it follows by Lemma 12 that we can write $v' = e'\#^{-1}q'$ where e' represents the identity.

Let q and e be the words over $\overline{X}^*$ obtained by deleting all occurences of the letters # and $\#^{-1}$ from q' and e' respectively. Since $w' = u'\#e'\#^{-1}q'$ is a permissible padding of w, it follows that $q = \epsilon$ or q begins with a negative letter. But we have $w = uv = ueq$ where e represents the identity. If $q = \epsilon$ then u must represent the identity, which is a contradiction. On the other hand, if q begins with a negative letter, then this contradicts the assumption that u is a minimum of w.   □

We are now ready to complete the proof of Theorem 9.

Proof.   Suppose a word w is accepted by the pushdown automaton A. Then by definition, there exists a word $x \in \overline{X}^*$ such that x represents the identity in $P(X)$,

and (x, w) is accepted by A when viewed as a usual finite automaton. Now by Proposition 16, x admits a permissible padding y which represents 1 in the polycyclic monoid $P(X^{\#})$, and hence also in the free group $F(X^{\#})$. Now by Lemma 11, (y, w) is accepted by A' as a finite automaton over $F(X^{\#}) \times \Sigma^*$ and over $P(X^{\#}) \times \Sigma^*$. Hence, w is accepted by A' as both a free group automaton and a polycyclic monoid automaton.

Conversely, if w is accepted by A' as a free group automaton [polycyclic monoid automaton], then by definition there exists a word y such that (y, w) is accepted by A' as an automaton over $F(X^{\#}) \times \Sigma^*$ [respectively, $P(X^{\#}) \times \Sigma^*$] and y represents 1 in the free group [polycyclic monoid]. Now by Lemma 11, y is a permissible padding of some word x, such that (x, w) is accepted by A viewed as a finite automaton over $P(X) \times \Sigma^*$. But by Proposition 16, x represents 1 in the polycyclic monoid $P(X)$, so that w is accepted by A, as required. $\qquad\square$

ACKNOWLEDGMENTS

REFERENCES

Berstel, J. (1979). *Transductions and Context-Free Languages*. Teubner: Informatik.

Chomsky, N., Schützenberger, M. P. (1963). The algebraic theory of context-free languages. In: *Computer Programming and Formal Systems*. Amsterdam: North-Holland, pp. 118–161.

Corson, J. M. (2005). Extended finite automata and word problems. *Internat. J. Algebra Comput.* 15(3):455–466,

Dassow, J., Mitrana, V. (2000). Finite automata over free groups. *Internat. J. Algebra Comput.* 10(6):725–737.

Dunwoody, M. J. (1985). The accessibility of finitely presented groups. *Invent. Math.* 81(3):449–457.

Elder, M. (2007). *G*-automata, counter languages and the Chomsky hierarchy. Groups St. Andrews 2005. Vol. 1. London Math. Soc. Lecture Note Ser., 339. Cambridge: Cambridge Univ. Press, pp. 313–318.

Elston, G. Z., Ostheimer, G. (2004). On groups whose word problem is solved by a counter automaton. *Theoret. Comput. Sci.* 320(2–3):175–185.

Fernau, H., Stiebe, R. (2002). Sequential grammars and automata with valences. *Theoret. Comput. Sci.* 276(1–2):377–405.

Gilman, R. H., Shapiro, M. (1998). On groups whose word problem is solved by a nested stack automaton. arXiv:math.GR/9812028 (1998).

Greibach, S. A. (1978). Remarks on blind and partially blind one-way multicounter machines. *Theoret. Comput. Sci.* 7(3):311–324.

Hopcroft, J. E., Ullman, J. D. (1969). *Formal Languages and their Relation to Automata*. Reading, Mass.-London-Don Mills, Ont: Addison-Wesley.

Howie, J. M. (1995). *Fundamentals of Semigroup Theory*. New York: Clarendon Press.

Ibarra, O. H., Sahni, S. K., Kim, C. E. (1976). Finite automata with multiplication. *Theoret. Comput. Sci.* 2:271–294.

Kambites, M. (2006). Word problems recognisable by deterministic blind monoid automata. *Theoret. Comput. Sci.* 362(1):232–237.

Kambites, M., Silva, P. V., Steinberg, B. (2007). On the rational subset problem for groups. *J. Algebra* 309:622–639.

Lawson, M. V. (1998). *Inverse Semigroups: The Theory of Partial Symmetries*. River Edge, NJ: World Scientific.

Lyndon, R. C., Schupp, P. E. (1977). *Combinatorial Group Theory*. London-New York: Springer-Verlag.

Magnus, W., Karrass, A., Solitar, D. (1976). *Combinatorial Group Theory*. New York: Dover Publications Inc., revised edition.

Mitrana, V., Stiebe, R. (2001). Extended finite automata over groups. *Discrete Appl. Math.* 108(3):287–300.

Muller, D. E., Schupp, P. E. (1983). Groups, the theory of ends, and context-free languages. *J. Comput. System Sci.* 26(3):295–310.

Nivat, M., Perrot, J.-F. (1970). Une généralisation du monoïde bicyclique. *C. R. Acad. Sci. Paris Sér. A-B* 271:A824–A827.