Applications of Rewriting Systems and Gröbner Bases to Computing Kan Extensions and Identities Among Relations

Thesis submitted to the University of Wales in support of
the application for the degree of PhilosophiæDoctor

by

A. Heyworth

supervised by

Prof. R. Brown and Dr. C. D. Wensley

October 1998

KEY WORDS: Presentation, Congruence, Category, Kan Extension, Rewrite System, Gröbner Basis, Normal Form, Automaton, Regular Expression, Peiffer Relation, Module, Crossed Module, Identities Among Relations, Covering Groupoid, Contracting Homotopy, Crossed Resolution.

A. Heyworth,
School of Mathematics,
University of Wales,
Bangor,
Gwynedd LL57 1UT.

Contents

Summary

This thesis concentrates on the development and application of Gröbner bases methods to a range of combinatorial problems (involving groups, semigroups, categories, category actions, algebras and K-categories) and the use of rewriting for calculating Kan extensions.
The first chapter gives a short introduction to presentations, rewrite systems, and completion.

Chapter Two contains the most important result, which is the application of Knuth-Bendix procedures to Kan extensions, showing how rewriting provides a useful method for attempting to solve a variety of combinatorial problems which can be phrased in terms of Kan extensions. A GAP3 program for Kan extensions is included in the appendix.

Chapter Three shows that the standard Knuth-Bendix algorithm is step-for-step a special case of Buchberger's algorithm. The one-sided cases and higher dimensions are considered, and the relations between these are made precise. The standard noncommutative Gröbner basis calculation may be expressed as a Kan extension over modules. A noncommutative Gröbner bases program (in GAP3) has been written.

Chapter Four relates rewrite systems, Gröbner bases and automata. Automata which only accept irreducibles, and automata which output reduced forms are discussed for presentations of Kan extensions. Reduction machines for rewrite systems are identified with standard output automata and the reduction machines devised for algebras are expressed as Petri nets.

Chapter Five uses the completion of a group rewriting system to algorithmically determine a contracting homotopy necessary in order to compute the set of generators for the module of identities among relations using the covering groupoid methods devised by Brown and Razak Salleh [17]. (The resulting algorithm has been implemented in GAP3). Reducing the resulting set of submodule generators is identified as a Gröbner basis problem.

Acknowledgements

I would first like to express my deepest appreciation to my parents.

I would also like to thank the School of Mathematics at the University of Wales in Bangor for giving me the opportunity to do the PhD and a friendly environment in which to work. I am particularly grateful to Ronnie Brown and Chris Wensley for their joint supervision and encouragement and to Tim Porter and Larry Lambe for their additional advice and inspirations.

I am grateful to many true and kind friends who have encouraged me. My wonderful brother Ben. My 'little sisters' Angie and Nergiz; My good friend Tanveer; The lovely people at Barnardo's – especially Siän and Yvonne; My neighbours in Rachub – especially Emma H and family; Helen, Emma M and Val who made University a nicer place to be.

There is so much to learn . . .

Chapter 1

Introduction

1.1 Presentations

1.1.1 Background

A computational problem in group theory typically begins "Given a group G, determine...". Methods of solution of the problem depend on the way the information about G is given. The study of groups given by presentations is called combinatorial group theory. Study of other algebraic objects (for example categories) through presentations may be called combinatorics. This section is an attempt to outline a little of the (controversial) history of and motivation for the study of groups and in particular the use of group presentations.

The origins of group theory might go back to 1600 BC. Stone tablets remain as evidence that the Babylonians knew how to solve quadratic equations (though they had no algebraic notation). The solution (by radicals) of a cubic equation was not discovered until the 16th century, and published simultaneously with the method for solving quartics (by reducing to a cubic). Mathematicians such as Euler and Lagrange worked on the problem, and in 1824 Abel proved that there was no general solution by radicals of a quintic equation. Work began on determining whether a given quintic was soluble, and it is from Évariste Galois's paper "On the Conditions of Solubility of Equations by Radicals" (submitted and rejected in 1831) that group theory really began. (That is not to say that group theoretic ideas did not exist before Galois (according to [73], they did) and a number of results were obtained before the definition of an abstract group reached its final form.) The first formal development of group theory followed Galois's ideas and was limited almost entirely to finite groups. The idea of an abstract infinite group is included in Arthur Cayley's work (1854, 1878) on group axioms, but was not pursued at that time. Finitely generated groups were defined by Dyck in 1882, and it is (disputedly) here that the first definition of a presentation by generators and relations was given.

Studying groups became important; groups of transformations came from symmetries and congruences in Euclidean Geometry, (semigroups come from partial symmetries) automorphism groups were used in Klein's "Erlangen Programme", cyclic groups came from numbers and modular arithmetic and more groups from Gauss's composition of binary quadratic forms (groupoids from Brandt's generalisation of this problem). Abstract finite groups were defined by Weber in 1882, and it was in 1893 that he published what we recognise as the modern definition of an arbitrary abstract group.

A major stimulus to the study of infinite discrete groups, however, was the development of topology. In 1895 Poincaré introduced the notion of a fundamental group $\Pi_1(X, a)$ of closed paths of a space X from a point a. The properties of the fundamental group of a topological space correspond to some properties of the space. Interest in classifying the topological spaces generated interest in fundamental groups. In 1911 Max Dehn, a student of Hilbert's, wrote a paper [31] which dealt with presentations of fundamental groups of closed, orientable surfaces, for which he formulated three fundamental decision problems: the word problem, the conjugacy problem, and the isomorphism problem. It is thought that

by this time the idea of trying to determine properties of a group given by a finite presentation was already familiar. Anyway, some consider the problems to be part of what became known as "Hilbert's Programme". Nielsen was also an important influence: his work led naturally to the study of groups presented through generators and relators.
There are certain advantages of presentations as a method for studying groups, or indeed other algebraic structures (monoids, categories, algebras). One advantage is that a presentation is compact as compared to (say) a Cayley table. An efficient presentation describes the group with the minimal amount of information. By now there is a lot of theoretical machinery for working with presentations, this may be called computational group theory (or computational category theory, etc), which really began with Turing and Newman's work at the end of World War II. Modern work in computational group theory may be found in Charles Sims's recent book [73], and a lot of work developing computer programs for group theoretic computations continues at Warwick (KBMAG), St Andrews (GAP) and Sydney (MAGMA) to name a few. The area has also broadened, problems with monoids are more widely researched and now categories are coming into the picture. Computational category theory is one relatively new field of computer algebra which has considerable prospects.

Rewriting systems are sets of directed equations or rules which are useful in computations. Rewrite rules specify the repeated replacement of subterms of a given formula with equivalent terms. Rewriting theory was introduced as a method of solving the word problem. The original word problem was expressed by Axel Thue in 1914:
"Suppose one has a set of objects, and a set of transformations (rules) that when applied to these objects yield objects in the same set. Given two objects x and y in the set, can x be transformed into y, or is there perhaps a third object z such that both x and y can be transformed into z?".

Thue established some preliminary results about strings of symbols (i.e. elements of a free monoid) and suggested that the approach might extend to more structured combinatorial objects (at about this time Dehn was working on the beginnings of combinatorial group theory). Thue wanted to develop a "calculus" to decide the word problem, that is a set of procedures or algorithms that could be applied to the given objects to obtain the correct answer. He wanted a general algorithm to solve the word problem in a variety of different settings.

Apparently Thue's work was disregarded until the 1930's when logicians were seeking formal definitions of concepts like "algorithm" and "effective procedure". In the mid 1950's and 60's notions of semi-Thue systems became important in mathematical linguistics. Work on formal language theory used semi-Thue systems as mathematical models for phrase-structure grammars. At the same time technology was improving to the extent where mathematicians began to consider mechanical theorem proving, and in the 1960's automated deduction quickly developed. As a form of computer program, rewriting systems made their debut in 1967 in a paper by Gorn. A particularly influential role was played by a paper written by Knuth and Bendix in 1970 [48]. They described an automatic procedure for solving word problems in abstract algebras.

In the 1970's term-rewriting systems took an important role in the study of automated deduction, which was still a rapidly developing area. However, it was not really until the 1980's that Thue systems became popular. A book which contains the most fundamental results of the 1980's is [7]. Since then, rewriting systems have continued to be of increasing interest, being investigated for different properties and applied to a widening range of areas. The computational aspect is particularly important. Many modern programs for symbolic manipulation continue to use rewrite rules in an ad hoc manner, and there is now much work on the more formal use of rewriting systems in programming (in particular see [42][43][73]).

1.1.2 Monoid and Group Presentations

It is assumed that the reader is familiar with monoids and groups. The terms and definitions for presentations are given in the following paragraphs to fix the notation.

Let X be a set. The **free semigroup** $X^\dagger$ on X consists of all nonempty sequences (strings) of elements of X. Composition is defined by concatenation of the strings. The **free monoid** PX (sometimes denoted X^*) on X consists of all strings of elements of X, including the empty string. Composition is defined by string concatenation with the empty string acting as identity.

A **set of relations** R for a monoid generated by X is a subset of $PX \times PX$. A **congruence** $=_S$ on a monoid A is an equivalence relation on A such that, for all $u,v \in A$, if $l =_S r$ then $ulv =_S urv$. The **congruence** $=_R$ **generated by** R **on** PX, where R is a set of relations, is given by $x =_R y$ if and only if there is a system of equations

$$\begin{array}{rcl} x & = & u_1 l_1 v_1 \\ u_1 r_1 v_1 & = & u_2 l_2 v_2 \\ \dots & \dots & \dots \\ u_n r_n v_n & = & y \end{array}$$

where either (l_i, r_i) or $(r_i, l_i) \in R$ for $i = 1, \dots, n$, $n \geq 1$. This is equal to the smallest equivalence relation on PX containing R such that for all $u,v \in PX$ $x =_R y \Rightarrow uxv =_R uyv$ [30]. If A is a monoid and $=_S$ a congruence on A then the **factor monoid** $A/=_S$ is the monoid whose elements are the congruence classes of $=_S$ on A and whose composition is induced by that on A. The congruence class of an element $a \in A$ with respect to S will be denoted $[a]_S$.

A **monoid presentation** is a pair $mon\langle X|R\rangle$, where X is a set and $R \subseteq PX \times PX$ is a set of relations. The monoid it presents is the factor monoid $PX/=_R$. We say $mon\langle X|R\rangle$ is a monoid presentation of M if $M \cong PX/=_R$. The **free group** on X is the group $F(X)$ with monoid presentation $mon\langle \bar{X}|R_0\rangle$ where $\bar{X} := \{x^+, x^- : x \in X\}$ and $R_0 := \{(x^+x^-, id), (x^-x^+, id) : x \in X\}$. A **group presentation** is a pair $grp\langle X|R\rangle$ where X is a set and $R \subseteq F(X)$ (the group *relators*). The group it presents is defined as the monoid that is presented by $mon\langle \bar{X}|\bar{R}\rangle$ where $\bar{R} := R_0 \cup \{(r, id) : r \in R\}$. (To verify that this is a group note that any element has the form $[x_1{}^{\varepsilon_1} \dots x_n{}^{\varepsilon_n}]_{\bar{R}}$ where $x_1, \dots, x_n \in X$, $\varepsilon_1, \dots, \varepsilon_n \in \{+, -\}$ and so has inverse $[x_n{}^{-\varepsilon_n} \dots x_1{}^{-\varepsilon_1}]_{R'}$ where $-(+) := -, -(-) := +$.)

A monoid is **finitely presented** if it has a presentation $mon\langle X \,|\, R\rangle$ where X and R are finite sets (similarly for groups). Monoid presentations are often used to give all the information about the monoid in a compact form. The main question, given a monoid presentation, is known as the word problem. The **word problem for a monoid presentation** $mon\langle X|R\rangle$ is as follows:

INPUT:	$u, v \in PX$	(two elements in the free monoid),
QUESTION:	$u =_R v$?	(do they represent the same element in the monoid presented?)

Rewriting systems (defined later) are one method of tackling this problem (another being the Todd-Coxeter procedure). However, as is well known, rewriting cannot solve the problem in general but only when the rewriting system can be *completed* (defined later). Fortunately there are a large number of interesting examples (all finite monoids, all abelian monoids - see later) for which rewriting systems are completable.

1.1.3 Category and Groupoid Presentations

It is assumed that the reader is familiar with the general concepts of category, functor and natural transformation. The following paragraphs fix the notation used and define presentations of categories and groupoids and the associated word problem.

A **directed graph** Γ consists of a set of objects $\mathrm{Ob}\Gamma$, a set of arrows $\mathrm{Arr}\Gamma$ and two functions $src, tgt : \mathrm{Arr}\Gamma \to \mathrm{Ob}\Gamma$. (Throughout the text, unless otherwise specified, "graph" should be taken to mean such a directed graph. If a graph has only one object this will be denoted $\bullet$.) A **morphism of graphs** $F : \Gamma \to \Delta$ consists of functions $\mathrm{Ob}F : \mathrm{Ob}\Gamma \to \mathrm{Ob}\Delta$, $\mathrm{Arr}F : \mathrm{Arr}\Gamma \to \mathrm{Arr}\Delta$ such that $src \circ \mathrm{Arr}F = \mathrm{Ob}F \circ src$ and $tgt \circ \mathrm{Arr}F = \mathrm{Ob}F \circ tgt$. This gives the category DirG of directed graphs.

The forgetful functor $U : \mathsf{Cat} \to \mathsf{DirG}$ from the category of small categories to directed Graphs has a left adjoint which we write P, the **free category** on a graph. It is realised in the usual way: if Γ is a graph then $\mathrm{Ob}P\Gamma := \mathrm{Ob}\Gamma$, and the non-identity arrows $P\Gamma(A_1, A_2)$ consist of all paths $a_1 \cdots a_n$, i.e. sequences $a_1, \ldots, a_n \in \Gamma$ such that $tgt(a_i) = src(a_{i+1})$ for $i = 1, \ldots, n-1,\ n \geq 1$. The identity arrows are such that for all objects A of the free category $\mathit{id}_A a = a$ for any path a with source A and $c\,\mathit{id}_A = c$ for any path c with target A. Composition is defined by concatenation. Thus if Γ has one object then $P\Gamma$ can be identified with the free monoid on $\mathrm{Arr}\Gamma$.

A set of **relations** R for a category A is a subset of $\mathrm{Arr}\mathsf{A} \times \mathrm{Arr}\mathsf{A}$, every relation $(l, r) \in R$ must satisfy $src(l) = src(r),\ tgt(l) = tgt(r)$. A **congruence** $=_S$ on a category A is an equivalence relation on the set $\mathrm{Arr}\mathsf{A}$ which satisfies $l =_S r \Rightarrow src(l) = src(r), tgt(l) = tgt(r)$ and for all $u, v \in \mathrm{Arr}\mathsf{A}$, if $l =_S r$ then $ulv =_S urv$ when these products are defined. The **congruence** $=_R$ **generated by** R **on** $P\Gamma$, where R is a set of relations, is given by $x =_R y$ if there is a system of equations

$$\begin{array}{rcl} x & = & u_1 l_1 v_1 \\ u_1 r_1 v_1 & = & u_2 l_2 v_2 \\ \cdots & \cdots & \cdots \\ u_n r_n v_n & = & y \end{array}$$

where either (l_i, r_i) or $(r_i, l_i) \in R$ for $i = 1, .., n$ and the products $u_i l_i v_i$ and $u_i r_i v_i$ are defined. If A is a category and $=_S$ is a congruence on A then the **factor category** $\mathsf{A}/ =_S$ is the category whose objects are $\mathrm{Ob}\mathsf{A}$ and whose arrows are the congruence classes with respect to $=_S$ of $\mathrm{Arr}\mathsf{A}$ with composition induced by that on A. The congruence class of an arrow $a \in \mathsf{A}$ with respect to S will be denoted $[a]_S$. Congruent arrows have the same sources and targets as each other, so src, tgt are preserved.

A **category presentation** is a pair $cat\langle\Gamma|R\rangle$, where Γ is a graph and $R \subset \mathrm{Arr}P\Gamma \times \mathrm{Arr}P\Gamma$ is a set of relations. The category it presents is the factor category $P\Gamma/ =_R$. We say that $cat\langle\Gamma|R\rangle$ is a category presentation for C if $\mathsf{C} \cong P\Gamma/ =_R$.

The **free groupoid** on Γ is denoted $F(\Gamma)$. It is defined to be the free category $P\bar{\Gamma}$ factored by the relations R_0 where $\mathrm{Ob}\bar{\Gamma} := \mathrm{Ob}\Gamma$, $\mathrm{Arr}\bar{\Gamma} := \{a^+, a^- : a \in \mathrm{Arr}\Gamma\}$ with $src(a^+) = tgt(a^-) = src(a)$ and $tgt(a^+) = src(a^-) = tgt(a)$ and $R_0 := \{(a^+a^-, \mathit{id}_{src(a)}), (a^-a^+, \mathit{id}_{tgt(a)}) : a \in \mathrm{Arr}\Gamma\}$. A **groupoid presentation** is a pair $gpd\langle\Gamma|R\rangle$ where Γ is a graph and R is a subset of the disjoint union of the vertex groups of $F(X)$. The groupoid it presents is defined as the category that is presented by $cat\langle\bar{\Gamma}|\bar{R}\rangle$ where $\bar{\Gamma}$ and R_0 are as above and $\bar{R} := R_0 \cup \{(r, \mathit{id}_{src(r)} : r \in R\}$. (To verify that this is a groupoid note that any element has the form $[a_1{}^{\varepsilon_1} \cdots a_n{}^{\varepsilon_n}]_{\bar{R}}$ where $a_1, \ldots, a_n \in \Gamma$, $\varepsilon_1, \ldots, \varepsilon_n \in \{+, -\}$ and so has inverse $[a_n{}^{-\varepsilon_n} .. a_1{}^{-\varepsilon_1}]_{\bar{R}}$ where $-(+) := -, -(-) := +$.)

Some motivation for considering groupoid presentations is given by the fact that a presentation $grp\langle X|R\rangle$ of a group G lifts to a presentation $gpd\langle\widetilde{X}|\widetilde{R}\rangle$ of the *covering groupoid* of the Cayley graph $\widetilde{X}$ of the group G [40]. In detail: let $\theta : F(X) \to G$ be the quotient map, and let $\mathrm{Ob}\widetilde{X} = \{g : g \in G\}$, $\mathrm{Arr}\widetilde{X} = \{[g, x] : g \in G, x \in X\}$ where $src([g, x]) := g$, $tgt([g, x]) := g\theta(x)$, and $\widetilde{R} = G \times R$. (This is referred to in detail in Chapter 5). A monoid (or group) can be regarded as a category (or groupoid) with one object. Let $mon\langle X|R\rangle$ present a monoid M. Then the presentation $cat\langle \Gamma X|R\rangle$, where ΓX is the one object graph and $\mathrm{Arr}\Gamma X := X$, is a category presentation for the monoid M.

A category C is **finitely presented** if it has a presentation $cat\langle\Gamma|R\rangle$ where $\mathrm{Ob}\Gamma, \mathrm{Arr}\Gamma$ and R are finite sets. The **word problem for a category presentation** $cat\langle\Gamma|R\rangle$ is as follows:

INPUT:	$u, v \in \mathrm{Arr}(P\Gamma)$	(two arrows in the free category),
QUESTION:	$u =_R v$?	(do they represent the same element in the category presented?)

Terminology: The trivial category, with category presentation $cat\langle\bullet|\rangle$ has only one object $\bullet$ and one arrow – the identity $id_\bullet$. The null functor maps a category to the trivial category, by mapping all the objects to $\bullet$ and the arrows to $id_\bullet$. The hom-set of all arrows between two particular objects A and B of a category P will be denoted $\mathsf{P}(A, B)$.

1.2 Abstract Reduction Relations

We recall the definitions of reduction relations on abstract sets and some of their properties. This is a brief exposition of the introductory material in [7], the results stated are proved there. These results will be generalised to P-sets, where P is a category, in Section 2.4

Let T be a set. A **reduction relation** $\rightarrow$ on a set T is a subset of $T \times T$. We write $l \rightarrow r$ when (l, r) is an element (rule) of $\rightarrow$. The pair $(T, \rightarrow)$ will be called a **reduction system**. **Reduction** is the name given to the procedure of applying rules to a given term to obtain another term i.e. we "reduce t_1 to t_2 in one step" if (t_1, t_2) is an element of the reduction relation. An element t_1 of T is said to be **reducible** if there is another element t_2 of T such that $t_1 \rightarrow t_2$, otherwise it is **irreducible**. The reflexive, transitive closure of a reduction relation $\rightarrow$ is denoted $\overset{*}{\rightarrow}$ i.e. if $t_1 \rightarrow t_2 \rightarrow \cdots \rightarrow t_n$ then we write $t_1 \overset{*}{\rightarrow} t_n$.
The reflexive, symmetric, transitive closure of $\rightarrow$ is denoted $\overset{*}{\leftrightarrow}$ This is the smallest equivalence relation on T that contains $\rightarrow$. The equivalence class of an element t of T under $\overset{*}{\leftrightarrow}$ will be denoted $[t]$.

The **word problem** for a reduction system $(T, \rightarrow)$ is:

INPUT:	$t_1, t_2 \in T$	(two elements of T).
QUESTION:	$t_1 \overset{*}{\leftrightarrow}_R t_2$	(are they equivalent under $\overset{*}{\leftrightarrow}_R$)?

Let $\rightarrow$ be a reduction relation on a set T. A **normal form** for an element $t \in T$ is an irreducible element $t_N \in T$ such that $t \overset{*}{\leftrightarrow} t_N$. A **set of unique normal forms** is a subset of T which contains exactly one normal form for each equivalence class of T with respect to $\overset{*}{\leftrightarrow}$. A **unique normal form function** is a function $N : T \rightarrow T$ whose image is a set of unique normal forms. One approach to solving the word problem is to attempt to choose a set of unique normal forms as representatives of the classes of the equivalence relation. Given any pair of elements, if their normal forms can be computed, it can be seen that the elements are equivalent if and only if their normal forms are equal.

The definitions above indicate that if the irreducible elements are to be unique normal forms we require exactly one irreducible in each equivalence class. Further, if reduction is to be the unique normal form function then we should be able to obtain the normal form of any element by a finite sequence of reductions. We consider conditions that guarantee these properties. It is essential that equivalent elements reduce to the same irreducible. A reduction system $(T, \rightarrow)$ is **confluent**, if for all terms $t, u_1, u_2 \in T$ such that $t \overset{*}{\rightarrow} u_1$ and $t \overset{*}{\rightarrow} u_2$ there exists an element $v \in T$ such that $u_1 \overset{*}{\rightarrow} v$ and $u_2 \overset{*}{\rightarrow} v$. The following picture illustrates the confluence condition.

$$\begin{tikzcd}
 & t \arrow[dl, "*"'] \arrow[dr, "*"] & \\
u_1 \arrow[dr, "*"'] & & u_2 \arrow[dl, "*"] \\
 & v &
\end{tikzcd}$$

The following facts may be found in [7].

Fact 1.2.1 *If a reduction system $(T, \rightarrow)$ is confluent then for each $t \in T$, $[t]$ has at most one normal form.*

We require that the irreducibles be obtainable by a finite sequence of reductions. A reduction system $(T, \rightarrow)$ is **Noetherian** (or terminating) if there is no infinite sequence $t_1, t_2, \ldots \in T$ such that for all $i \in \mathbb{N}$, $t_i \rightarrow t_{i+1}$. A reduction system $(T, \rightarrow)$ is **locally confluent** if for all elements $t, u_1, u_2 \in T$ such that $t \rightarrow u_1$ and $t \rightarrow u_2$ there exists a term $v \in T$ such that $u_1 \overset{*}{\rightarrow} v$ and $u_2 \overset{*}{\rightarrow} v$.

Fact 1.2.2 *A Noetherian reduction system is confluent if it is locally confluent.*

Fact 1.2.3 *If a reduction system $(T, \rightarrow)$ is Noetherian then for every $t \in T$, $[t]$ has a normal form (not necessarily unique).*

A reduction system $(T, \rightarrow)$ is **complete** (or convergent) if it is confluent and $\rightarrow$ is Noetherian.

Fact 1.2.4 *Let $(T, \rightarrow)$ be a reduction system. If it is complete then for every $t \in T$, $[t]$ has a unique normal form.*

Some motivation for considering complete reduction systems is that they enable the solution of the word problem through a **normal form algorithm**. The normal forms are the irreducible elements (completeness ensures that there is exactly one irreducible in each equivalence class). The normal form function is repeated reduction (the Noetherian property ensures that the irreducible is reached in finitely many reductions). So: given two terms, we reduce them to irreducibles, the words are equivalent only if the irreducibles are equal.

Fact 1.2.5 *If a reduction system $(T, \rightarrow)$ is complete and T is finite, then the word problem for $(T, \rightarrow)$ is decidable.*

It is not in general possible to determine whether a finite reduction system is Noetherian, confluent or complete. However, if a finite system is known to be Noetherian, we can determine whether or not it is complete. Non-confluence occurs when different rules apply to the same term, giving different reduced terms. A **critical pair** is a pair (u_1, u_2) where there exists a term $t \in T$ such that $t \rightarrow u_1$ and $t \rightarrow u_2$. A critical pair (u_1, u_2) is said to **resolve** if there exists a term $v \in T$ such that $u_1 \overset{*}{\rightarrow} v$ and $u_2 \overset{*}{\rightarrow} v$.

Fact 1.2.6 *Let $(T, \rightarrow)$ be a reduction system. Let $N : T \rightarrow T$ be the normal form function where $N(s)$ is the irreducible form of s with respect to $\rightarrow$. If for all $t \rightarrow s_1, t \rightarrow s_2$, $N(s_1) = N(s_2)$ then $(T, \rightarrow)$ is complete.*

A Noetherian system may sometimes be made confluent by adding in extra rules (the unresolvable critical pairs). This procedure will be discussed in the next chapter in the particular setting with which we are concerned.

Chapter 2

Using Rewriting to Compute Kan Extensions of Actions

This chapter defines rewriting procedures for terms $x|w$ where x is an element of a set and w is a word. Two kinds of rewriting are involved here. The first is the familiar $x|ulv \to x|urv$. The second is given by an action of certain words on elements, so allowing rewriting $x|F(a)v \to x \cdot a|v$. Further, the elements x and $x \cdot a$ are allowed to belong to different sets. The natural setting for this rewriting is a "presentation" $kan\langle\Gamma|\Delta|RelB|X|F\rangle$ where Γ, Δ are (directed) graphs and $X : \Gamma \to \mathsf{Sets}$ and $F : \Gamma \to P\Delta$ are graph morphisms to the category of sets, and the free category on Δ respectively, and $RelB$ is a set of relations on $P\Delta$. The main result defines rewriting procedures on the P-set

$$T := \bigsqcup_{B \in \mathrm{Ob}\Delta} \bigsqcup_{A \in \mathrm{Ob}\Gamma} XA \times \mathsf{P}(FA, B) \tag{2.1}$$

in order to attempt the computation of Kan extensions of actions of categories given by presentations (see section 5).

So the power of rewriting theory may now be brought to bear on a much wider range of combinatorial enumeration problems. Traditionally rewriting is used for solving the word problem for monoids. It may now also be used in the specification of

i) equivalence classes and equivariant equivalence classes,

ii) arrows of a category or groupoid,

iii) action of a group on the cosets given by a subgroup,

iv) right congruence classes given by a relation on a monoid,

v) orbits of an action of a group or monoid.

vi) conjugacy classes of a group,

vii) coequalisers, pushouts and colimits of sets,

viii) induced permutation representations of a group or monoid.

and many others.

2.1 Kan Extensions of Actions

The concept of the Kan extension of an action will be central to this chapter. It will therefore be defined here with some familiar examples to motivate the construction listed afterwards. There are two types of Kan extension (the details are in Chapter 10 of [51]) known as right and left. Which type is right and which left varies according to authors' chosen conventions. In this text only one type is used (left according to [25], right according to other authors) and to save conflict it will be referred to simply as "the Kan extension" - it is the colimit one, so there is an argument for calling it a co-Kan, and the other one simply Kan, but we shall not presume to do that here.

Let A be a category. A **category action** X of A is a functor $X : \mathsf{A} \to \mathsf{Sets}$. This means that for every object A there is a set XA and the arrows of A act on the elements of the sets associated to their sources to return elements of the sets associated to their targets. So if a_1 is an arrow in $\mathsf{A}(A_1, A_2)$ then XA_1 and XA_2 are sets and $Xa_1 : XA_1 \to XA_2$ is a function where $Xa_1(x)$ is denoted $x \cdot a_1$. Furthermore, if $a_2 \in \mathsf{A}(A_2, A_3)$ is another arrow then $(x \cdot a_1) \cdot a_2 = x.(a_1 a_2)$ so the action preserves the composition. This is equivalent to the fact that $Xa_2(Xa_1(x)) = X(a_1 a_2)(x)$ i.e. X is a functor. Also $F(\mathit{id}_A) = \mathit{id}_{FA}$ so $x \cdot \mathit{id} = x$ when defined.

Given the category A and the action defined by X, let B be a second category and let $F : \mathsf{A} \to \mathsf{B}$ be a functor. Then an **extension of the action** X **along** F is a pair (K, ε) where $K : \mathsf{B} \to \mathsf{Sets}$ is a functor and $\varepsilon : X \to F \circ K$ is a natural transformation. This means that K is a category action of B and ε makes sure that the action defined is an extension with respect to F of the action already defined on A. So ε is a collection of functions, one for each object of A, such that $\varepsilon_{src(a)}(Xa)$ and $K(F(a))$ have the same action on elements of $K(F(src(a))$.

The **Kan extension of the action** X **along** F is an extension of the action (K, ε) with the universal property that for any other extension of the action (K', ε') there exists a unique natural transformation $\alpha : K \to K'$ such that $\varepsilon' = \varepsilon \circ \alpha$. Here K may thought of as the universal extension of the action of A to an action of B.

Kan Extension

$$\begin{tikzcd}
\mathsf{A} \arrow[rr, "F"] \arrow[dr, "X"'] & {} \arrow[d, phantom, "\varepsilon \Rightarrow"] & \mathsf{B} \arrow[dl, dashed, "K"] \\
& \mathsf{Sets} &
\end{tikzcd}$$

Universal Property of Kan Extension

$$\begin{tikzcd}
\mathsf{A} \arrow[rr, "F"] \arrow[dr, "X"'] & {} \arrow[d, phantom, "\varepsilon' \Rightarrow"] & \mathsf{B} \arrow[dl, dashed, "K'"] & & \mathsf{A} \arrow[rr, "F"] \arrow[dr, "X"'] & {} \arrow[d, phantom, "\varepsilon \Rightarrow"] & \mathsf{B} \arrow[dl, dashed, "K"] \arrow[dl, dashed, bend left=80, "K'"] \arrow[dl, phantom, bend left=40, "\alpha\Rightarrow"] \\
& \mathsf{Sets} & & = & & \mathsf{Sets} &
\end{tikzcd}$$

2.2 Examples

Some familiar problems will now be expressed in terms of Kan extensions. This is not a claim that these problems can always be computed, it merely demonstrates that they are all special cases of the general problem of computing a Kan extension. MacLane wrote that "the notion of Kan extensions subsumes all the other fundamental concepts of category theory" in section 10.7 of [51] (entitled "All Concepts are Kan Extensions"). This list helps to illustrate his statement. Throughout these examples we use the same notation as the definition, so the pair (K, ε) is the Kan extension of the action X of A along the functor F to B. By a monoid (or group) "considered as a category" we mean the one object category with arrows corresponding to the monoid elements and composition defined by composition in the monoid.

1) Groups and Monoids

Let B be a monoid regarded as a category. Let A be the trivial category, acting trivially on a one point set $X\bullet$, and let $F : \mathsf{A} \to \mathsf{B}$ be the inclusion map. Then the set $K\bullet$ is bijective with the set of elements of the monoid and the right action of the arrows of B is right multiplication by the monoid elements. The natural transformation maps the unique element of $X\bullet$ to the element of $K\bullet$ representing the monoid identity.

2) Groupoids and Categories

Let B be a category. Let A be the (discrete) category of objects of B with identity arrows only. Let X define the trivial action of A on a collection of one point sets $\sqcup_A XA$ (one for each object $A \in \mathrm{Ob}\mathsf{A}$), and let $F : \mathsf{A} \to \mathsf{B}$ be the inclusion map. Then the set KB for $B \in \mathsf{B}$ is isomorphic to the set of arrows of B with target B and the right action of the arrows of B is defined by right composition. The natural transformation maps the unique element of a set XA to the representative identity arrow for the object FA for every $A \in \mathsf{A}$.

3) Cosets, and Congruences on Monoids

Let B be a group considered as a category, and let A be a subgroup of B, with inclusion F. Let X map the object of A to a one point set. The set $K\bullet$ represents the (right) cosets of A in B, with the right action of any group element b of $\mathrm{Arr}\mathsf{B}$ taking the representative of the coset Hg to the representative of the coset Hgb. The left cosets can be similarly represented, defining the right action K by a left action on the cosets. The natural transformation picks out the representative for the subgroup H.
Alternatively, let B be a monoid considered as a category and A be generated by arrows which map under F to a set of generators for a right congruence. Then the set $K\bullet$ represents the congruence classes, the action of any monoid element b of $\mathrm{Arr}\mathsf{B}$ taking the representative (in $K\bullet$) of the class $[m]$ to the representative of the class $[mb]$. The natural transformation picks out the representative for the class $[id]$. (As above, left congruence classes may also be expressed in terms of a Kan extension.)

4) Orbits of Group Actions

Let A be a group thought of as a category and let X define the action of the group on a set $X\bullet$. Let B be the trivial category and let F be the null functor. Then the set $K\bullet$ is a set of representatives of the distinct orbits of the action and the action of B on $K\bullet$ is trivial. The natural transformation ε maps any element of the set $X\bullet$ to its orbit representative in B.

5) Colimits in Sets

Let A be any category and let B be the trivial category, with F being the null functor and X being a functor to sets. Then the Kan extension corresponds to the colimit of (the diagram) $X : \mathsf{A} \to \mathsf{Sets}$; $K\bullet$ is the colimit object, and ε defines the colimit functions from each set XA to $K\bullet$. Examples of this are when A has two objects A_1 and A_2, and two non-identity arrows $a_1, a_2 : A_1 \to A_2$, (*coequaliser* of the functions Xa_1 and Xa_2 in Sets); A has three objects A_1, A_2 and A_3 and two arrows $a_1 : A_1 \to A_2$ and $a_2 : A_1 \to A_3$ (*pushout* of the functions Xa_1 and Xa_2 in Sets).

6) Induced Permutation Representations

Let A and B be groups thought of as categories, F being a group morphism and X being a right action of the group A on the set $X\bullet$. The Kan extension of the action along F is known as the action of B

induced from that of A by F (sometimes written $F_*(X)$). There are simple methods of constructing the set $K\bullet$ when A and B are groups, but this is more difficult for monoids.

This last example is very close to the full definition of a Kan extension. A Kan extension *is* the action of the category B induced from the action of A by F together with ε which shows how to get from the A-action to the B-action. The point of the other examples is to show that Kan extensions can be used as a method of representing a variety of situations.

2.3 Presentations of Kan Extensions of Actions

The problem that has been introduced is that of "computing a Kan extension". In order to keep the analogy with computation and rewriting for presentations of monoids we propose the following definition of a presentation of a Kan extension. This formalises ideas used in [26].
First, we define 'Kan extension data'.

Definition 2.3.1 *A* ***Kan extension data*** (X', F') *consists of small categories* A, B *and functors* $X' : \mathsf{A} \to \mathsf{Sets}$ *and* $F' : \mathsf{A} \to \mathsf{B}$*.*

Definition 2.3.2 *A* ***Kan extension presentation*** *is a quintuple* $\mathcal{P} := kan\langle \Gamma | \Delta | RelB | X | F \rangle$ *where*

i) Γ *and* Δ *are graphs,*

ii) $cat\langle \Delta | RelB \rangle$ *is a category presentation,*

iii) $X : \Gamma \to U\mathsf{Sets}$ *is a graph morphism,*

iv) $F : \Gamma \to UP\Delta$ *is a graph morphism.*

$\mathcal{P}$ ***presents the Kan extension data*** (X', F') *where* $X' : \mathsf{A} \to \mathsf{Sets}$ *and* $F' : \mathsf{A} \to \mathsf{B}$ *if*

i) Γ *is a generating graph for* A *and* $X : \Gamma \to \mathsf{Sets}$ *is the restriction of* $X' : \mathsf{A} \to \mathsf{Sets}$*,*

ii) $cat\langle \Delta | RelB \rangle$ *is a category presentation of* B*,*

iii) $F : \Gamma \to P\Delta$ *induces* $F' : \mathsf{A} \to \mathsf{B}$*.*

We also say $\mathcal{P}$ ***presents*** *the Kan extension* (K, ε) *of the Kan extension data* (X', F')*. The presentation is* ***finite*** *if* Γ*,* Δ *and* $RelB$ *are finite.*

Remark 2.3.3 The fact that X, F induce X', F' implies extra conditions on X, F in relation to A and B. In practice we need only the values of X', F' on Γ. This is analogous to the fact that for coset enumeration of a subgroup H of G where G has presentation $grp\langle \Delta | R \rangle$ we need only that H is generated by certain words in the set Δ.

2.4 P-sets

In this section we extend some of the usual concepts and terminology of rewriting in order to apply them to the new situation.

Definition 2.4.1 *For a category* P*, a* P***-set*** *is a set* T *together with a function* $\tau : T \to \mathrm{Ob}\mathsf{P}$ *and a partial action* $\cdot$ *of the arrows of* P *on* T*. The action* $t \cdot p$ *is defined for* $t \in T$*,* $p \in \mathrm{Arr}\mathsf{P}$ *when* $\tau(t) = src(p)$ *and satisfies*

$$i)\ \tau(t \cdot p) = tgt(p),$$

Further, for all $t \in T$, $p, q \in \mathrm{Arr}\mathsf{P}$ such that $(t \cdot p) \cdot q$ is defined the following properties hold

$$ii)\, t \cdot id_{\tau(t)} = t,$$
$$iii)\, (t \cdot p) \cdot q = t \cdot (pq).$$

Definition 2.4.2 *A* ***reduction relation*** *on a* P*-set* T *is a relation* $\to$ *on* T *such that for all* $t_1, t_2 \in T$, $t_1 \to t_2$ *implies* $\tau(t_1) = \tau(t_2)$.

Definition 2.4.3 *A reduction relation* $\to$ *on the P-set* T *is* ***admissible*** *if for all* $t_1, t_2 \in T$, $t_1 \to t_2$ *implies* $t_1 \cdot q \to t_2 \cdot q$ *for all* $q \in \mathrm{Arr}\mathsf{P}$ *such that* $src(q) = \tau(t_1)$.

For the rest of this chapter we assume that $\mathcal{P} = kan\langle\Gamma|\Delta|RelB|X|F\rangle$ is a presentation of a Kan extension. The following definitions will be used throughout. Let P denote the free category $P\Delta$. Then define

$$T := \bigsqcup_{B \in \mathrm{Ob}\Delta} \bigsqcup_{A \in \mathrm{Ob}\Gamma} XA \times \mathsf{P}(FA, B) \tag{2.2}$$

It is convenient to write an element (x, p) of $XA \times \mathsf{P}(FA, B)$ as $x|p$, a kind of "tagged word" – with x being the tag and p the word. The function $\tau : T \to \mathrm{Ob}\mathsf{P}$ is defined by

$$\tau(x|p) := tgt(p) \text{ for } x|p \in T.$$

The action of P on T is given by right multiplication

$$x|p \cdot q := x|pq \text{ for } x|p \in T,\ q \in \mathrm{Arr}\mathsf{P} \text{ when } src(q) = \tau(x|p).$$

It is routine to verify that $\tau(x|p \cdot q) = tgt(q)$ and $(x|p \cdot q) \cdot r = (x|p) \cdot (qr)$, whenever these terms are defined, hence proving the following lemma.

Lemma 2.4.4 *T is a* P*-set.*

Now we define some 'rewriting procedures' which require two types of rule.

The first type is the 'ε-rules' $R_\varepsilon \subseteq T \times T$. They are to ensure that the action is an extension of the action of A – this is the requirement for $\varepsilon : X \to KF$ to be a natural transformation. For each arrow $a : A_1 \to A_2$ in Γ we get a set of ε-rules. In this set there is one rule for each element x of XA_1. Formally

$$R_\varepsilon := \{(x|Fa, x \cdot a|id_{FA_2})|x \in XA_1, a \in \Gamma(A_1, A_2), A_1, A_2 \in \mathrm{Ob}\Gamma\}. \tag{2.3}$$

The other type is the 'K-rules' $R_K \subseteq \mathrm{Arr}\mathsf{P} \times \mathrm{Arr}\mathsf{P}$. They are to ensure that the action preserves the structure of B – this is the requirement for K to be a functor/category action. These are simply the relations (l, r) of B, formally:

$$R_K := RelB. \tag{2.4}$$

Now define $R_{init} := (R_\varepsilon, R_K)$. This we call the **initial rewrite system** that results from the presentation. A **rewrite system** for a Kan presentation $\mathcal{P}$ is a pair R of sets R_T, R_P where $R_T \subseteq T \times T$ and $R_P \subseteq \mathrm{Arr}\mathsf{P} \times \mathrm{Arr}\mathsf{P}$ such that for all $(s, u) \in R_T$, $\tau(s) = \tau(u)$ and for all $(l, r) \in R_P$, $src(l) = src(r)$ and $tgt(l) = tgt(r)$.

Definition 2.4.5 *The* ***reduction relation generated by*** *a rewrite system* $R = (R_T, R_P)$ *on the* P*-set* T *is defined as* $t_1 \to_R t_2$ *if and only if one of the following is true:*

i) There exist $(s,u) \in R_T, q \in \mathrm{Arr}\mathsf{P}$ *such that* $t_1 = s \cdot q$ *and* $t_2 := u \cdot q$.

ii) There exist $(l,r) \in R_P$, $s \in T$, $q \in \mathrm{Arr}\mathsf{P}$ *such that* $t_1 = s \cdot lq$ *and* $t_2 = s \cdot rq$.

Then we say t_1 ***reduces*** *to* t_2 *by the rule* (s,u) *or by* (l,r) *respectively.*

Note that $\to_R$ is an admissible reduction relation on T – the proof of this is part of the next lemma. The relation $\overset{*}{\to}_R$ is the reflexive, transitive closure of $\to_R$, and $\overset{*}{\leftrightarrow}_R$ is the reflexive, symmetric, transitive closure of $\to_R$.

Remark 2.4.6 Essentially, the rules of R_P are two-sided and apply to any substring to the right of the separator $|$. This distinguishes them from the one- sided rules of R_T. The one-sided rules are not simply 'tagged rewrite rules' (tags being the part to the left of $|$) because the tags are being rewritten.

Lemma 2.4.7 *Let* R *be a rewrite system on a* P*-set* T. *Then* $\overset{*}{\leftrightarrow}_R$ *is an admissible equivalence relation on the* P*-set* T.

Proof By definition $\overset{*}{\leftrightarrow}_R$ is symmetric, reflexive and transitive. Now let $t_1, t_2 \in T$ be such that $t_1 \to_R t_2$ and let $v \in \mathrm{Arr}\mathsf{P}$. be such that $src(v) = \tau(t_1)$. Then there are two possibilities. For the first case suppose (i) there exist $(s,u) \in R_T, q \in \mathrm{Arr}\mathsf{P}$ such that $t_1 = s \cdot q$ and $t_2 = u \cdot q$. Then it follows that $t_1 \cdot v = s \cdot qv$ and $t_2 \cdot v = u \cdot qv$, (by P-set properties). For the second case suppose (ii) there exist $s \in T$, $(l_1, r_1) \in R_P$, $q \in \mathrm{Arr}\mathsf{P}$ such that $t_1 = s \cdot lq$ and $t_2 = s \cdot rq$. Then it follows that $t_1 \cdot v = s \cdot lqv$ and $t_2 \cdot v = s \cdot rqv$. In either case $t_1 \cdot v \to_R t_2 \cdot v$ by the definition of $\to_R$. Therefore $\to_R$ is admissible, and hence $\overset{*}{\leftrightarrow}_R$ is admissible. □

Notation: the equivalence class of $t \in T$ under $\overset{*}{\leftrightarrow}_R$ will be denoted $[t]$.

A Kan extension (K, ε) is given by a set KB for each $B \in \mathrm{Ob}\Delta$ and a function $Kb : KB_1 \to KB_2$ for each $b : B_1 \to B_2 \in \mathsf{B}$, (defining the functor K) together with a function $\varepsilon_A : XA \to KFA$ for each $A \in \mathrm{Ob}\mathsf{A}$ (the natural transformation). This information can be given in four parts: the set $\sqcup KB$, a function $\bar{\tau} : \sqcup KB \to \mathrm{Ob}\mathsf{B}$, a partial function (action) $\sqcup KB \times \mathrm{Arr}\mathsf{P} \to \sqcup KB$ and a function $\varepsilon : \sqcup XA \to \sqcup KB$. Here $\sqcup KB$ and $\sqcup XA$ (by a small abuse of notation) are the disjoint unions of the sets KB, XA over $\mathrm{Ob}\mathsf{B}$, $\mathrm{Ob}\mathsf{A}$ respectively; $\bar{\tau}(z) = B$ for $z \in KB$ and if $src(p) = B$ for $p \in \mathrm{Arr}\mathsf{P}$ then $z \cdot p$ is defined.

Theorem 2.4.8 *Let* $\mathcal{P} = kan\langle \Gamma | \Delta | RelB | XF \rangle$ *be a Kan extension presentation, and let* P, T, $R_{init} = (R_\varepsilon, R_K)$ *be defined as above. Then the Kan extension* (K, ε) *presented by* $\mathcal{P}$ *is given by the following data:*

i) the set $\sqcup KB = T / \overset{*}{\leftrightarrow}_R$,

ii) the function $\bar{\tau} : \sqcup KB \to \mathrm{Ob}\mathsf{B}$ *induced by* $\tau : T \to \mathrm{Ob}\mathsf{P}$,

iii) the action of B *on* $\sqcup KB$ *induced by the action of* P *on* T,

iv) the natural transformation ε *determined by* $x \mapsto [x | id_{FA}]$ *for* $x \in XA$, $A \in \mathrm{Ob}\mathsf{A}$.

Proof The initial rules R on T generate a reduction relation $\to$ on T. Let $\overset{*}{\leftrightarrow}$ denote the reflexive, symmetric, transitive closure of $\to$.
Claim $\overset{*}{\leftrightarrow}$ preserves the function τ.
Proof Let $[x|p]$ denote the class of elements equivalent under $\overset{*}{\leftrightarrow}$ to $x|p \in T$. We prove that $\leftrightarrow$, the symmetric closure of $\to$ preserves τ. Let $t_1, t_2 \in T$ so that $t_1 \leftrightarrow t_2$. ¿From the definition of $\to$ there are two possible situations. For the first case suppose that there exist $(s_1, s_2) \in R_\varepsilon$ such that $t_1 = s_1 \cdot p$ and $t_2 = s_2 \cdot p$ for some $p \in \mathrm{Arr}\mathsf{P}$. Clearly $\tau(t_1) = \tau(t_2)$. For the other case suppose that there exist

$(l,r) \in R_K$ such that $t_1 = s \cdot (lp)$ and $t_2 = s \cdot (rp)$ for some $s \in T$, $p \in \mathrm{Arr}\mathsf{P}$. Again, it is clear that $\tau(t_1) = \tau(t_2)$. Hence $\bar{\tau} : T/\overset{*}{\leftrightarrow}_R \to \mathrm{Ob}\mathsf{P}$ is well-defined by $\bar{\tau}[t] = \tau(t)$. □

Claim $T/\overset{*}{\leftrightarrow}$ is a B-set.

Proof First we prove that B acts on the equivalence classes of T with respect to $\overset{*}{\leftrightarrow}$. An arrow of B is an equivalence class $[p]$ of arrows of P with respect to $RelB$. It is required to prove that $[t] \cdot p := [t \cdot p]$ is a well defined action of P on $T/\overset{*}{\leftrightarrow}$ such that $[t] \cdot p = [t] \cdot q$ for all $p =_{RelB} q$. Let $t \in T, p \in \mathrm{Arr}\mathsf{P}$ be such that $\tau[t] = src[p]$ i.e. $\tau(t) = src(p)$. Then $t \cdot p$ is defined. Suppose $s \overset{*}{\leftrightarrow} t$. Then $[s \cdot p] = [t \cdot p]$ since $s \cdot p \overset{*}{\leftrightarrow} t \cdot p$, whenever $s \cdot p, t \cdot p$ are defined. Suppose $p =_{RelB} q$. Then $[t \cdot p] = [t \cdot q]$ since $t \cdot p \overset{*}{\leftrightarrow}_{R_K} t \cdot q$, whenever $t \cdot p, t \cdot q$ are defined and $\overset{*}{\leftrightarrow}_{RelB}$ is contained in $\overset{*}{\leftrightarrow}$. Therefore P acts on $T/\overset{*}{\leftrightarrow}$ and this action preserves the relations of B and so defines an action of B on $T/\overset{*}{\leftrightarrow}$. Furthermore $\bar{\tau}([t] \cdot p) = \bar{\tau}[t \cdot p] = tgt(p)$ and if $q \in \mathsf{P}$ such that $src(q) = tgt(p)$ then $([t] \cdot p) \cdot q = [(t \cdot p) \cdot q] = [t \cdot (pq)] = [t] \cdot pq$. □

The Kan extension may now be defined. For $B \in \mathrm{Ob}\mathsf{B}$ define

$$KB := \{[x|p] : \bar{\tau}[x|p] = B\}. \qquad (2.5)$$

For $b : B_1 \to B_2$ in B define

$$Kb : KB_1 \to KB_2 : [t] \mapsto [t \cdot p] \text{ for } [t] \in KB_1 \text{ where } p \in [b]. \qquad (2.6)$$

It is now routine to verify, since $p_1 =_{RelB} p_2$ implies $t \cdot p_1 \overset{*}{\leftrightarrow}_R t \cdot p_2$, for all t where $tcdotp_1$ is defined, that this definition of the action is a functor $K : \mathsf{B} \to \mathsf{Sets}$. Then define

$$\varepsilon : X \to KF : x \mapsto [x|id_{FA}] \text{ for } x \in XA, A \in \mathrm{Ob}\mathsf{A}. \qquad (2.7)$$

It is straightforward to verify that this is a natural transformation since $x|id_{FA_1} \cdot Fa \overset{*}{\leftrightarrow}_R x \cdot a|id_{FA_2}$ for all $x \in XA_1$, $a : A_1 \to A_2 \in \mathrm{Ob}\mathsf{A}$.

Therefore (K, ε) is an extension of the action X of A. The proof of the universal property of the extension is as follows. Let $K' : \mathsf{B} \to \mathsf{Sets}$ be a functor and $\varepsilon' : X \to K'F$ be a natural transformation. Then there is a unique natural transformation $\alpha : K \to K'$, defined by

$$\alpha_B[x|p] = K'(f)(\varepsilon'_A(x)) \text{ for } [x|p] \in KB,$$

which clearly satisfies $\varepsilon \circ \alpha = \varepsilon'$. □

Remark 2.4.9 If the Kan extension presentation is finite then R is finite. The number of initial rules is by definition $(\Sigma_{a \in \mathrm{Arr}\Gamma} |Xsrc(a)|) + |Rel\mathsf{B}|$.

2.5 Rewriting Procedures for Kan Extensions

In the next section we will explain the completion process for the initial rewrite system. It is convenient for this procedure to have a notation for the implementation of the data structure for a *finite* presentation $\mathcal{P}$ of a Kan extension. This we do here.

2.5.1 Input Data

`ObA` This is a list of integers $[1, 2, \dots]$, where each entry i corresponds uniquely to an object A_i of Γ.

`ArrA` This is a list of pairs of integers $[[i_1, j_1], [i_2, j_2], \dots]$, one for each arrow $a_k : A_{i_k} \to A_{j_k}$ of $\mathrm{Arr}\Gamma$. The first element of each pair is the source of the arrow it represents, and the other entry is the target.

`ObB` Similarly to ObΓ, this is a list of integers representing the objects of Δ.

`ArrB` This is a list of triples $[[b_1,i_1,j_1],[b_2,i_2,j_2],\ldots]$, one triple for each arrow $b_k : B_{i_k} \to B_{j_k}$ of ArrΔ. The first entry of each triple is a label for the arrow (in GAP this is called a generator), and the other entries are integers representing the source and target respectively. Note that the arrows of Γ did not have labels. The arrows of Δ will form parts of the terms of T whilst those of Γ do not, so this is why we have labels here and not before.

`RelB` This is a finite list of pairs of paths. Each path is represented by a finite list $[b_1,b_2,\ldots,b_n]$ of labels of composable arrows of ArrΔ. In GAP it is convenient to consider these lists as words $b_1 \cdots b_n$ in the generators that are labels for the arrows of Δ.

`FObA` This is a list of |ObΓ| integers. The kth entry represents the object of Δ which is the image of the object A_k under F.

`FArrA` This is a list of paths where the entry at the kth position is the path of P which is the image of the arrow a_k of Γ under F. The length of the list is |ArrΓ|.

`XObA` This is a list of lists of distinct (GAP) generators. There is one list of elements for each object in Γ. The list at position k represents the set which is the image of A_k under X.

`XArrA` This is a list of lists of generators. There is one list for each arrow a of Γ. It represents the image under the action Xa of the set $X(src(a))$. Suppose $a_k : A_{i_k} \to A_{j_k}$ is the arrow at entry k in ArrΓ, and $[x_1,x_2,\ldots,x_m]$ is the ith entry in XObΓ (the image set $X(A_i)$). Then the kth entry of XArrΓ is the list $[x_1 \cdot a, x_2 \cdot a, \ldots, x_m \cdot a]$ where $x_i \in X(A_j)$.

Note: All the above lists are finite since the Kan extension is finitely presented.

2.5.2 Initial Rules Procedure

The programmed function `InitialRules` extracts from the above data the initial rewrite system $R_{init} := (R_\varepsilon, R_K)$.

```
INPUT:     (ObA,ArrA,ObB,ArrB,RelB,FObA,FArrA,XObA,XArrA);
PROCEDURE: ans:=RelB;
           i:=1;
           while(i>Length(ArrA)) do
               a:=ArrA[i];                   ## arrow
               A:=a[1];                      ## source
               XA:=XObA[Position(ObA,A)];    ## set
               for j in [1..Length(XA)] do
                   x:=XA[j];                 ## element
                   xa:=XArrA[i][j];          ## element after action
                   Fa:=FArrA[i][j];          ## image of arrow
                   rule:=[[x,Fa],[xa]];      ## epsilon-rule
                   Add(ans,rule);
               od;
           i:=i+1;
           od;
OUTPUT:    R:=ans;                           ## initial rewrite system
```

We continue with the notation introduced so far, and apply the standard terminology of reduction relations to the reduction relation $\to_R$ on T.

2.5.3 Lists

In our GAP implementation terms of T are represented by words in generators, the generators may be thought of as labels, and the words as lists. The first entry in the list must be a label for an element of XA for some $A \in \mathrm{Ob}\Gamma$. The following entries will be labels for composable arrows of Δ, with the source of the first being FA. Formally:

Let L be the set of lists $l = [\mathtt{x},\mathtt{b1},\ldots,\mathtt{bn}]$, $n \geq 1$, such that $p = b_1 \cdots b_n$ is a reduced path (i.e. with no identity arrows) of P and $x|p \in T$ or $l = [x]$ and $x|id_{\tau(x)} \in T$. We will refer to $\mathtt{List}(t)$ as the unique list associated with the element $t \in T$. We will make use of the computer notation to extract particular elements of the list. So $t[1]$ means the first element x when $t = x|b_1 \cdots b_n$ and $t[2..5]$ is the sublist which is $[b_1, \ldots, b_4]$ in the example, which is an arrow in P. Also, $\mathtt{Length}(t)$ means the number of elements in the list t. A sublist of the list for a tagged string $t \in T$ will be referred to as a *part* of t.

2.5.4 Orderings

To work with a rewrite system R on T we will require certain concepts of order on T. We show how to use an ordering $>_X$ on $\sqcup XA$ together with an ordering $>_P$ on $\mathrm{Arr}\mathsf{P}$, these having certain properties, to construct an ordering $>_T$ on T with the properties needed for the rewriting procedures.

Definition 2.5.1 *A binary operation $>$ on the set is called a* ***strict partial ordering*** *if it is irreflexive, antisymmetric and transitive.*

Definition 2.5.2 *Let $>_X$ be a strict partial ordering on the set $\sqcup XA$. It is called a* ***total ordering*** *if for all $x, y \in \sqcup XA$ either $x >_X y$ or $y >_X x$ or else $x = y$.*

Definition 2.5.3 *Let $>_P$ be a strict partial ordering on* $\mathrm{Arr}\mathsf{P}$. *It is called a* ***total path ordering*** *if for all $p, q \in \mathrm{Arr}\mathsf{P}$ such that $src(p) = src(q)$ and $tgt(p) = tgt(q)$ either $p >_P q$ or $q >_P p$ or else $p = q$.*

Definition 2.5.4 *The ordering $>_P$ is* ***admissible on*** $\mathrm{Arr}\mathsf{P}$ *if $p >_P q \;\Rightarrow\; upv >_P uqv$ for all $u, v \in \mathrm{Arr}\mathsf{P}$ such that $upv, uqv \in \mathrm{Arr}\mathsf{P}$.*

Definition 2.5.5 *An ordering $>$ is* ***well-founded*** *on a set of elements if there is no infinite sequence $x_1 > x_2 > \cdots$. An ordering $>$ is a* ***well-ordering*** *on a structure if it is well-founded and a total ordering with respect to that structure.*

Lemma 2.5.6 *Let $>_X$ be a well-ordering on the finite set $\sqcup XA$ and let $>_P$ be an admissible well-ordering on P. For $t_1, t_2 \in T$ define $t_1 >_T t_2$ if*

$$\Leftarrow t_1[2..Length(t_1)] >_P t_2[2..Length(t_2)] \text{ or } t_1[2..Length(t_1)] = t_2[2..Length(t_2)] \text{ and } t_1[1] >_X t_2[1].$$

Then $>_T$ is an admissible well-ordering on the P-set T.

Proof It is straightforward to verify that irreflexivity, antisymmetry and transitivity of $>_X$ and $>_P$ imply those properties for $>_T$. The ordering $>_T$ is admissible on T because it is made compatible with the right action (defined by composition between arrows on P) by the admissibility of $_P$ on $\mathrm{Arr}\mathsf{P}$. The ordering is linear, since if $t_1, t_2 \in T$ such that neither $t_1 >_T t_2$ nor $t_2 >_T t_1$, it follows by the linearity of $>_X$ and linearity of $>_P$ on $\mathrm{Arr}\mathsf{P}$ that $t_1 = t_2$. That $>_T$ is well-founded is easily verified using the fact that any infinite sequence in terms of $>_T$ implies an infinite sequence in either $>_X$ or $>_P$ and $>_X$ and $>_P$ are both well-founded, so there are no such sequences. □

The last result shows that there is some scope for choosing different orderings on T. The actual choice is even wider than this but it is not relevant to discuss this here. We are not concerned here with considering ranges of possible orderings, but work with the one that is most straightforward to use. The ordering implemented is a variation on the above. It corresponds to the length-lexicographical ordering and is defined in the following way.

Definition 2.5.7 (Implemented Ordering) *Let $>_X$ be any linear order on (the finite set) $\sqcup XA$. Let $>_\Gamma$ be a linear ordering on (the finite set) $\mathrm{Arr}\Delta$. This induces an admissible ordering $>_P$ on $\mathrm{Arr}\mathsf{P}$ where $p >_P q$ if and only if $Length(p) > Length(q)$ or $Length(p) = Length(q)$ and there exists $k > 0$ such that $p[i] >_\Gamma q[i]$ for all $i < k$ and $p[k] = q[k]$. The ordering $>_T$ is then defined as follows: $t_1 >_T t_2$ if $Length(t_1) > Length(t_2)$ or if $Length(t_1) = Length(t_2)$ and $t_1[1] >_X t_2[1]$, or if $Length(t_1) = Length(t_2)$ and there exists $k \in [1..Length(t_1)]$ such that $t_1[i] = t_2[i]$ for all $i < k$ and $t_1[k] >_\Gamma t_2[k]$.*

Proposition 2.5.8 *The definitions above give an admissible, length-non-increasing well-order $>_T$ on the* P*-set T.*

Proof It is immediate from the definition that $>_T$ is length-non-increasing. It is straightforward to verify that $>_T$ is irreflexive, antisymmetric and transitive. It can also be seen that $>_T$ is linear (suppose neither $t_1 >_T t_2$ nor $t_2 >_T t_1$ then $t_1 = t_2$, by the definition, and linearity of $>_X$, $>_\Gamma$). It is clear from the definition that $>_T$ is admissible on the P-set T (if $t_1 >_T t_2$ then $t_1.p >_T t_2.p$). To prove that $>_T$ is well-founded on T, suppose that $t_1 >_T t_2 >_T t_3 >_T \cdots$ is an infinite sequence. Then for each $i > 0$ either $Length(t_i) > Length(t_{i+1})$ or if $Length(t_i) = Length(t_{i+1})$ and $t_i[1] >_X t_{i+1}[1]$, or if $Length(t_i) = Length(t_{i+1})$ and there exists $k \in [1..Length(t_i)]$ such that $t_i[j] = t_{i+1}[j]$ for all $j < k$ and $t_i[k] >_\Gamma t_{i+1}[k]$. This implies that there is an infinite sequence of type $n_1 > n_2 > n_3 > \cdots$ of positive integers from some finite n_1, or of type $x_1 >_X x_2 >_X x_3 > \cdots$ of elements of $\sqcup XA$ or else of type $p_1 >_\Gamma p_2 >_\Gamma p_3 >_\Gamma \cdots$ of arrows of Δ, none of which is possible as $>$, $>_X$, and $>_\Gamma$ are well-founded on $\mathbb{N}$, $\sqcup XA$ and $\mathrm{Arr}\Delta$ respectively. Hence $>_T$ is well-founded. □

Proposition 2.5.9 *Let $>_T$ be the order defined above. Then $p_1 >_P p_2 \Rightarrow s \cdot p_1 >_T s \cdot p_2$.*

Proof This follows immediately from the definition of $>_T$. □

Remark 2.5.10 The proposition can also be proved for the earlier definition of $>_T$ induced from $>_X$ and $>_P$.

2.5.5 Reduction

Now that we have defined an admissible well-ordering on T it is possible to discuss when a reduction relation generated by a rewrite system is compatible with this ordering.

Lemma 2.5.11 *Let R be a rewrite system on T. Orientate the rules of R so that for all (l,r) in R, if $l, r \in \mathrm{Arr}\mathsf{P}$ then $l >_P r$ and if $l, r \in T$ then $l >_T r$. Then the reduction relation $\to_R$ generated by R is compatible with $>_T$.*

Proof Let $t_1, t_2 \in T$ such that $t_1 \to_R t_2$. There are two cases to be considered 2.4.2. For the first case let $t_1 = s_1 \cdot p$, $t_2 = s_2 \cdot p$ for some $s_1, s_2 \in T$, $p \in \mathrm{Arr}\mathsf{P}$ such that $(s_1, s_2) \in R$. Then $s_1 >_T s_2$. It follows that $t_1 >_T t_2$ since $>_T$ is admissible on T. For the second case let $t_1 = s \cdot p_1 q$, $t_2 = s \cdot p_2 q$ for some $s \in T$, $p_1, p_2, q \in \mathrm{Arr}\mathsf{P}$ such that $(p_1, p_2) \in T$. Then $p_1 >_P p_2$ and so by Proposition 2.5.9 $s \cdot p_1 >_T s \cdot p_2$. Hence $t_1 >_T t_2$ by admissibility of $>_T$ on T. Therefore, in either case $t_1 >_T t_2$ so $\to_R$ is compatible with $>_T$. □

Remark 2.5.12 A reduction is the replacement of a part of a tagged string $x|p \in T$ according to a rule of R. Rules from R_T replace the tag $x|$ and part of the string p whilst rules from R_P replace substrings of p. The reduction relation $\rightarrow_R$ is the successive replacement of parts of a tagged string.

It is a standard result that if a reduction relation is compatible with an admissible well-ordering, then it is Noetherian. The next pseudo program shows the function Reduce which returns from a term $t \in T$ and a rewrite system $R \subseteq T \times T \sqcup \mathrm{ArrP} \times \mathrm{ArrP}$ a term $t_n \in [t]$ which is irreducible with respect to $\rightarrow_R$.

```
INPUT:(t,R);
PROCEDURE: new:=t; old:=[];
           while not(new=old) do
               old:=new;
               for rule in R do
                   lhs:=rule[1]; rhs:=rule[2];
                   if lhs is a sublist of new
                       replace lhs in new by rhs
                   fi;
               od;
           od;
OUTPUT: tn                # irreducible term in T #
```

2.5.6 Critical Pairs

We can now discuss what properties of R will make $\rightarrow_R$ a complete (i.e. Noetherian and confluent) reduction relation. By standard abuse of notation the rewrite system R will be called **complete** when $\rightarrow_R$ is complete. In this case $\overset{*}{\leftrightarrow}_R$ admits a normal form function.

Lemma 2.5.13 (Newman's Lemma) *A Noetherian reduction relation on a set is confluent if it is locally confluent [3].*

Hence, if R is compatible with an admissible well-ordering on T and $\rightarrow_R$ is locally confluent then $\rightarrow_R$ is complete. By orientating the pairs of R with respect to the chosen ordering $>_T$ on T, R is made to be Noetherian. The remaining problem is testing for local confluence of $\rightarrow_R$ and changing R in order to obtain an equivalent confluent reduction relation.

We will now explain the notion of critical pair for a rewrite system for T, extending the traditional notion to out situation. In particular the overlaps involve either just R_T, or just R_P or an interaction between R_T and R_P.

A term $crit \in T$ is called **critical** if it may be reduced by two or more different rules i.e. $crit \rightarrow_R crit1$, $crit \rightarrow_R crit2$ and $crit1 \neq crit2$. The pair $(crit1, crit2)$ resulting from two single-step reductions of the same term is called a **critical pair**. A critical pair for a reduction relation $\rightarrow_R$ is said to **resolve** if there exists a term res such that both $crit1$ and $crit2$ reduce to a common term res i.e. $crit1 \overset{*}{\rightarrow}_R res$, $crit2 \overset{*}{\rightarrow}_R res$.
We now define overlaps of rules for our type of rewrite system, and show how each kind results in a critical pair of the reduction relation. Let $R = (R_T, R_P)$ be a rewrite system, where $R_T \subseteq T \times T$ and $R_P \subseteq \mathrm{ArrP} \times \mathrm{ArrP}$.

Definition 2.5.14 *Let $(rule1, rule2)$ be a pair of rules of R such that $rule1$ and $rule2$ may both be applied to the same term $crit$ in such a way that there is a part of the term $crit$ that is affected by both* *the rules. When this occurs the rules are said to* ***overlap****. There are five types of overlap for this kind of rewrite system.*

Suppose $rule1, rule2 \in R_T$*. Put* $rule1 := (s_1, u_1)$*,* $rule2 := (s_2, u_2)$*. Then there is one type of overlap:*

$$i)\ s_1 = s_2 \cdot q \textit{ for some } q \in \mathrm{Arr}\mathsf{P}, \textit{ with resulting critical pair } (u_1, u_2 \cdot q).$$

Suppose $rule1, rule2 \in R_P$*. Put* $rule1 := (l_1, r_1)$*,* $rule2 := (l_2, r_2)$*. Then there are two possible types of overlap:*

$$ii)\ l_1 = pl_2q \textit{ for some } p, q \in \mathrm{Arr}\mathsf{P}, \textit{ with resulting critical pair } (r_1, pr_2q).$$
$$iii)\ l_1q = pl_2 \textit{ for some } p, q \in \mathrm{Arr}\mathsf{P}, \textit{ with resulting critical pair } (r_1q, pr_2).$$

Suppose $rule1 \in R_T$*,* $rule2 \in R_P$*. Put* $rule1 := (s_1, u_1)$*,* $rule2 := (l_1, r_1)$*. Then there are two possible types of overlap:*

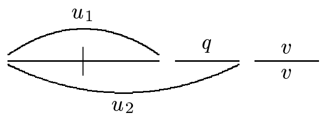
$$iv)\ s_1 \cdot q = s \cdot l_1 \textit{ for some } s \in T,\ q \in \mathrm{Arr}\mathsf{P}, \textit{ with resulting critical pair } (u_1 \cdot q, s \cdot r_1).$$
$$v)\ s_1 = s \cdot (l_1q) \textit{ for some } s \in T,\ q \in \mathrm{Arr}\mathsf{P}, \textit{ with resulting critical pair } (u_1, s \cdot r_1q).$$

One pair of rules may overlap in more than one way, giving more than one critical pair. For example the rules $(x|a^2ba, y|ba)$ and (a^2, b) overlap with critical term $x|a^2ba$ and critical pair $(y|ba, x|b^2a)$ and also with critical term $x|a^2ba^2$ and critical pair $(y|ba^2, x|a^2b^2)$.

Lemma 2.5.15 *Let* R *be a finite rewrite system on the* P*-set* T*. If* (t_1, t_2) *is a critical pair then either the pair resolves immediately or there is an overlap between two rules* $(rule1, rule2)$ *such that if the critical pair* $(crit1, crit2)$ *resulting from that overlap resolves then* (t_1, t_2) *resolves.*

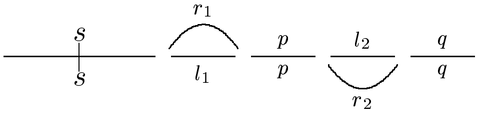
Proof Let (t_1, t_2) be a critical pair. Then there exists a critical term t and two rules $rule1$, $rule2$ such that t reduces to t_1 with respect to $rule1$ and to t_2 with respect to $rule2$. There are seven cases that must be considered.

Suppose $rule1 := (s_1, u_1), rule2 := (s_2, u_2) \in R_T$. Then the rules must overlap on t as shown:



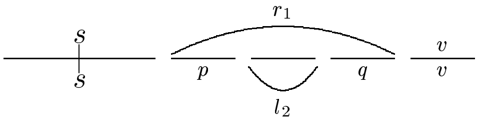
and there exist $q, v \in \mathrm{Arr}\mathsf{P}$ such that $t = s_1 \cdot qv = s_2 \cdot v$ and then $t_1 = u_1 \cdot qv$ and $t_2 = u_2 \cdot v$. The critical pair resulting from this overlap (i) is $(u_1 \cdot q, u_2)$ and if this resolves to a common term r then (t_1, t_2) resolves to $r \cdot v$.

Suppose $rule1 := (l_1, r_1)$, $rule2 := (l_2, r_2) \in R_P$. Then there are three possible ways in which the rules may apply to t. In the first case the rules do not overlap:

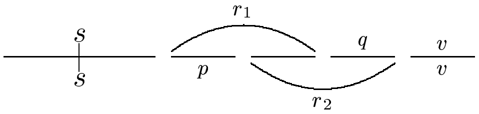


and there exist $s \in T$, $p, q \in \mathrm{Arr}\mathsf{P}$ such that $t = s \cdot l_1pl_2q$ and then $t_1 = s \cdot r_1pl_2q$ and $t_2 = s \cdot l_1pr_2q$. The pair (t_1, t_2) immediately resolves to $u \cdot r_1pr_2q$ by applying $rule2$ to t_1 and $rule1$ to t_2.

In the second case one rule is contained within the other:

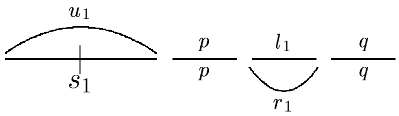


and there exist $s \in T$, $p, q, v \in \mathrm{ArrP}$ such that $t = s \cdot l_1 v = s \cdot p\, l_2 q v$ and then $t_1 = s \cdot r_1 v$ and $t_2 = s \cdot p r_2 q v$. The critical pair resulting from the overlap of the rules (ii) is $(r_1, p r_2 q)$ and if this resolves to a common term r then (t_1, t_2) resolves to $s \cdot r v$.
In the third case one part of the term is changed by both rules:

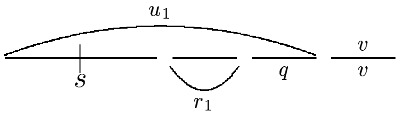


and there exist $s \in T$, $p, q, v \in \mathrm{ArrP}$ such that $t = s \cdot l_1 q v = s \cdot p l_2 v$ and then $t_1 = s \cdot r_1 q v$ and $t_2 = s \cdot p r_2 v$. The critical pair resulting from the overlap of the rules (iii) is $(r_1 q, p r_2)$ and if this resolves to a common term r then (t_1, t_2) resolves to $s \cdot r v$.

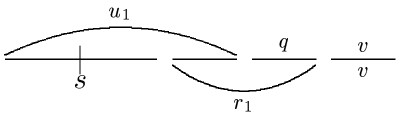
Suppose finally that $rule1 := (s_1, u_1) \in R_T$ and $rule2 := (l_1, r_1) \in R_P$. Then there are (again) three possible ways in which the rules may apply to t. In the first case the rules do not overlap:



and there exist $p, q \in \mathrm{ArrP}$ such that $t = s_1 \cdot p l_1 q$ and then $t_1 = u_1 \cdot p l_1 q$ and $t_2 = s_1 \cdot p r_1 q$. The pair (t_1, t_2) immediately resolves to $u_1 \cdot p r_1 q$ by applying $rule2$ to t_1 and $rule1$ to t_2.
In the second case one rule is contained within the other:



and there exist $s \in T$, $q, v \in \mathrm{ArrP}$ such that $t = s_1 v = s \cdot l_1 q v$ and then $t_1 = u_1 v$ and $t_2 = s r_1 q v$. The critical pair resulting from the overlap of the rules (iv) is $(u_1, s \cdot r_1 q)$ and if this resolves to a common term r then (t_1, t_2) resolves to $r \cdot v$.
In the third case there is one part of the term changed by both rules:



and there exist $s \in T$, $q, v \in \mathrm{ArrP}$ such that $t = s_1 \cdot q v = s \cdot l_1 v$ and then $t_1 = u_1 \cdot q v$ and $t_2 = s \cdot r_1 v$. The critical pair resulting from the overlap of the rules (v) is $(u_1 \cdot q, s \cdot r_1)$ and if this resolves to a common term r then (t_1, t_2) resolves to $r \cdot v$.

Thus we have considered all possible ways in which a term may be reduced by two different rules, and shown that resolution of the critical pair (when not immediate) depends upon the resolution of the critical pair resulting from a particular overlap of the rules. □

Corollary 2.5.16 *If all the overlaps between rules of a rewrite system R on T resolve then all the critical pairs for the reduction relation $\rightarrow_R$ resolve, and so $\rightarrow_R$ is confluent.*

Proof Immediate from the Lemma. □

Lemma 2.5.17 *All overlaps of a pair of rules of R can be found by looking for two types of overlap between the lists representing the left hand sides of rules.*

Proof Let $rule1 = (l_1, r_1)$ and $rule2 = (l_2, r_2)$ be a pair of rules. Recall that $\texttt{List}(t)$ is the representation of a term $t \in T$ as a list. The first type of list overlap occurs when $\texttt{List}(l_2)$ is a sublist of $\texttt{List}(l_1)$ (or vice-versa). This happens in cases (i), (ii) and (v). The second type of list overlap occurs when the end of $\texttt{List}(l_1)$ matches the beginning of $\texttt{List}(l_2)$ (or vice-versa). This happens in cases (iii) and (iv). □

The program for finding overlaps and the resulting critical pairs is called `CriticalPairs`. The outline of part of it is reproduced here: Let $rule1 := (l1, r1)$ and $rule2 := (l2, r2)$ be a pair of rules. The program compares $rule1$ with $rule2$ to look for overlaps. This part of the program shows how to determine whether $l1$ contains $l2$ or the beginning of $l1$ overlaps with the end of $l2$. To find other critical pairs the program can compare $rule2$ with $rule1$.

```
l1 := List(l1); len1 := Length(l1);
l2 := List(l2); len2 := Length(l2);

# Search for type 1 pairs  (l2 is contained in l1).
if len1 >= len2 then
    for i in [1..len1-len2] do
    if l1{[i..i+len2-1]} = l2 then
            if i=1 then   u := IdWord;
            else          u := Product( Sublist(l1,1,i-1) );
            if i+len2-1 = len1 then    v := IdWord;
            else          v := Product( Sublist(l1,i+len2,len1) );
            [ u*r2*v, r1 ]   ## critical pair found

# Search for type 2 pairs: (right of l1 overlaps the left of l2)
for i in [1..len1] do
    while not( i>len1 or i>len2 ) do
        if ( l1{[len1-i+1..len1]} = l2{[1..i]} ) then
            if i = len1 then  u := IdWord;
            else              u := Product( Sublist(l1,1,len1-i) );
            if i = len2 then  v := IdWord;
            else              v := Product( Sublist(l2,i+1,len2) );
            [ r1*v, u*r2 ]   ## critical pair found
```

It has now been proved that all the critical pairs of a finite rewrite system R on T can be listed. To test whether a critical pair resolves, each side of it is reduced using the function `Reduce`. If `Reduce` returns the same term for each side then the pair resolves.

2.5.7 Completion Procedure

We have shown how to (i) find overlaps between rules of R and (ii) test whether the resulting critical pairs resolve. Further we have shown that if all critical pairs for R resolve then $\rightarrow_R$ is confluent. We now show that critical pairs which do not resolve may be added to R without affecting the equivalence R defines on T.

Lemma 2.5.18 *Any critical pair $(crit1, crit2)$ of a rewrite system R may be added to the rewrite system without changing the equivalence relation $\overset{*}{\leftrightarrow}_R$.*

Proof This result is proved by considering any critical pair (t_1, t_2). By definition this pair is the result of two different single-step reductions being applied to a critical term t. Therefore $t \to_R t_1$ and $t \to_R t_2$. It is immediate that $t_1 \overset{*}{\leftrightarrow}_R t \overset{*}{\leftrightarrow}_R t_2$, and so adding (t_1, t_2) to R does not add anything to the equivalence relation $\overset{*}{\leftrightarrow}$. □

We have now set up and proved everything necessary for a variant of the Knuth-Bendix procedure, which will add rules to a rewrite system R resulting from a presentation of a Kan extension, to attempt to find an equivalent complete rewrite system. The benefit of such a system is that $\to_R$ then acts as a normal form function for $\overset{*}{\leftrightarrow}_R$ on T.

Theorem 2.5.19 *Let $\mathcal{P} = \langle \Gamma | \Delta | RelB | X | F \rangle$ be a finite presentation of a Kan extension (K, ε). Let $P := P\Delta$,*

$$T := \bigsqcup_{B \in \mathrm{Ob}\Delta} \bigsqcup_{A \in \mathrm{Ob}\Gamma} XA \times \mathsf{P}(FA, B),$$

and let $R = (R_\varepsilon, R_P)$ be the initial rewrite system for $\mathcal{P}$ on T. Let $>_T$ be an admissible well-ordering on T. Then there exists a procedure which, if it terminates, will return a rewrite system R^C which is complete with respect to $>_T$ such that the admissible equivalence relations $\overset{*}{\leftrightarrow}_{R^C}$ and $\overset{*}{\leftrightarrow}_R$ coincide.

Proof The procedure finds all critical pairs resulting from overlaps of rules of R. It attempts to resolve them. When they do not resolve it adds them to the system as new rules. Critical pairs of the new system are then examined. When all the critical pairs of a system resolve, then the procedure terminates, the final rewrite system R^C obtained is complete. This procedure has been verified in the preceding results of this section. □

```
INPUT: (R,>T);
PROCEDURE:  NEW:=R; OLD:=[];
            while not OLD=NEW do
                CRIT:=CriticalPairs(R)
                for crit in CRIT do
                    crit[1]:=Reduce(crit[1],R);
                    crit[2]:=Reduce(crit[2],R);
                    if crit[1]=crit[2] then Remove(CRIT,crit);
                    if crit[1]<crit[2] then crit:=(crit[2],crit[1]);
                od;
                Add(NEW,CRIT);
            od;
OUTPUT: NEW;  ## complete rewrite system.
```

The whole procedure, which takes as input the presentation of a Kan extension and yields as output a complete rewrite system with respect to the ordering $>_T$, when this can be found, has been implemented in GAP in the file *kan.g*. We will now briefly discuss how to interpret a complete rewrite system on T, supposing that the program has returned one.

2.6 Interpreting the Output

2.6.1 Finite Enumeration of the Kan Extension

When every set KB is finite we may catalogue the elements of all of the sets $\sqcup KB$ in stages. The first stage consists of all the elements $x|\mathit{id}_{FA}$ where $x \in XA$ for some $A \in \mathrm{Ob}\Gamma$. These elements are considered to have length zero. The next stage builds on the set of irreducible elements from the last block to construct elements of the form $x|b$ where $b : FA \to B$ for some $B \in \mathrm{Ob}\Delta$. This is effectively acting on the sets with the generating arrows to define new (irreducible) elements of length one. The next builds on the irreducibles from the last block by acting with the generators again. When all the elements of a block of elements of the same length are reducible then the enumeration terminates (any longer term will contain one of these terms and therefore be reducible). The set of irreducibles is a set of normal forms for $\sqcup KB$. The subsets KB of $\sqcup KB$ are determined by the function $\bar{\tau}$, i.e. if $x|b_1 \cdots b_n$ is a normal form in $\sqcup KB$ and $\tau(x|b_1 \cdots b_n) := tgt(b_n) = B_n$ then $x|b_1 \cdots b_n$ is a normal form in KB_n. Of course if one of the sets KB is infinite then this may prevent the enumeration of other finite sets KB_i. The same problem would obviously prevent a Todd-Coxeter completion. This cataloguing method only applies to finite Kan extensions. It has been implemented in the function *kan*, which currently has an enumeration limit of 1000 on $\sqcup KB$ set in the program. If this limit is exceeded, the program returns the completed rewrite system – provided the completion procedure terminates.

2.6.2 Regular Expression for the Kan Extension

Let R be a finite complete rewrite system on T for the Kan extension (K, ε). Then the theory of languages and regular expressions may be applied. The set of irreducibles in T is found after the construction of an automaton from the rewrite system and the derivation of a language from this automaton. Details of this method may be found in Chapter Four.

2.6.3 Iterated Kan Extensions

One of the pleasant features of this procedure is that the input and output are of similar form. The consequence of this is that if the extended action K has been defined on Δ then given a second functor $G' : \mathsf{B} \to \mathsf{C}$ and a presentation $cat\langle \Lambda | RelC \rangle$ for C it is straightforward to consider a presentation for the Kan extension data (K', G'). This new extension is in fact the Kan extension with data $(X', F' \circ G')$

Lemma 2.6.1 *Let $kan\langle \Gamma | \Delta | RelB | X | F \rangle$ be a presentation for a Kan extension (K, ε). Then let $cat\langle \Lambda | RelC \rangle$ present a category C and let $G' : \mathsf{B} \to \mathsf{C}$. Then the Kan extension presented by $kan\langle \Gamma | \Lambda | RelC | X | F \circ G | \rangle$ is equal to the Kan extension presented by $kan\langle \Delta | \Lambda | RelC | K | G \rangle$.*

Proof Let $kan\langle \Gamma | \Delta | RelB | X | F \rangle$ present the Kan extension data (X', F') for the Kan extension (K, ε). Let C be a category finitely presented by $cat\langle \Lambda | RelC \rangle$ and let $G' : \mathsf{B} \to \mathsf{C}$. Then $kan\langle \Delta | \Lambda | RelC | K | G \rangle$ presents the Kan extension data (K', G') for the Kan extension (L, η).
We require to prove that $(L, \varepsilon \circ \eta)$ is the Kan extension presented by $kan\langle \Gamma | \Lambda | RelC | X | F \circ G \rangle$ having data $(X', F' \circ G')$. It is clear that $(L, \epsilon \circ \eta)$ defines an extension of the action X along $F \circ G$ because L defines an action of C and $\varepsilon \circ \eta : X \to F \circ G \circ L$ is a natural transformation.
For the universal property, let (M, ν) be another extension of the action X along $F \circ G$. Then consider the pair $(G \circ M, \nu)$, it is an extension of X along F. Therefore there exists a unique natural transformation $\alpha : X \to F \circ G \circ M$ such that $\varepsilon \circ \alpha = \nu$ by universality of (K, ε). Now consider the pair (M, α), it is an extension of K along G. Therefore there exists a unique natural transformation $\beta : L \to M$ such that $\eta \circ \beta = \alpha$ by universality of (L, η). Therefore β is the unique natural transformation such that $\varepsilon \circ \eta \circ \beta = \nu$, which proves the universality of the extension $(L, \varepsilon \circ \eta)$. □

2.7 Example of the Rewriting Procedure for Kan Extensions

Let A and B be the categories generated by the graphs below, where B has the relation $b_1b_2b_3 = b_4$.

$$\begin{tikzcd} A_1 \arrow[r, bend left, "a_1"] & A_2 \arrow[l, bend left, "a_2"] \end{tikzcd} \qquad \begin{tikzcd} B_1 \arrow[loop left, "b_4"] \arrow[rr, "b_1"] \arrow[dr, bend right, "b_5"'] & & B_2 \arrow[dl, "b_2"] \\ & B_3 \arrow[ul, "b_3"'] & \end{tikzcd}$$

Let $X : \mathsf{A} \to \mathsf{Sets}$ be defined by $XA_1 = \{x_1, x_2, x_3\}$, $XA_2 = \{y_1, y_2\}$ with
$Xa_1 : XA_1 \to XA_2 : x_1 \mapsto y_1, x_2 \mapsto y_2, x_3 \mapsto y_1$,
$Xa_2 : XA_1 \to XA_2 : y_1 \mapsto x_1, y_2 \mapsto x_2$,
and let $F : \mathsf{A} \to \mathsf{B}$ be defined by $FA_1 = B_1$, $FA_2 = B_2$, $Fa_1 = b_1$ and $Fa_2 = b_3b_2$. The input to the computer program takes the following form. First we set up the variables:

```
gap> F := FreeGroup("b1","b2","b3","b4","b5","x1","x2","x3","y1","y2");;
gap> b1 := F.1;; b2 := F.2;; b3 := F.3;; b4 := F.4;; b5 := F.5;;
gap> x1 := F.6;; x2 := F.7;; x3 := F.8;; y1 := F.9;; y2 := F.10;;
```

Then we input the data:

```
gap> ObA := [1,2];;
gap> ArrA := [ [1,1], [2,2] ];;
gap> ObB := [1,2,3];;
gap> ArrB := [ [b1,1,2], [b2,2,3], [b3,3,1], [b4,1,1], [b5,1,3] ];;
gap> RelB := [ [b1*b2*b3,b4] ];;
gap> FObA := [1,2];;
gap> FArrA := [b1,b2*b3];;
gap> XObA := [ [x1,x2,x3], [y1,y2] ];;
gap> XArrA := [ [y1,y2,y1],[x1,x2] ];;
```

To combine all this data in one record do:

```
gap> KAN := rec( ObA:=ObA, ArrA:=ArrA,  ObB:=ObB, ArrB:=ArrB, RelB:=RelB,
                 FObA:=FObA, FArrA:=FArrA, XObA:=XObA, XArrA:=XArrA );;
```

To calculate the initial rules do

```
gap> IR := InitialRules( KAN );
```

The output will be

```
i= 1, XA= [ x1, x2, x3 ], Ax= x1, rule= [ x1*b1, y1 ]
i= 1, XA= [ x1, x2, x3 ], Ax= x2, rule= [ x2*b1, y2 ]
i= 1, XA= [ x1, x2, x3 ], Ax= x3, rule= [ x3*b1, y1 ]
i= 2, XA= [ y1, y2 ], Ax= y1, rule= [ y1*b2*b3, x1 ]
i= 2, XA= [ y1, y2 ], Ax= y2, rule= [ y2*b2*b3, x2 ]
[ [ b1*b2*b3, b4 ], [ x1*b1, y1 ], [ x2*b1, y2 ], [ x3*b1, y1 ],
  [ y1*b2*b3, x1 ], [ y2*b2*b3, x2 ] ]
```

This means that there are five initial ε-rules from: ($x_1|Fa_1, x_1.a_1|id_{FA_2}$), ($x_2|Fa_1, x_2.a_1|id_{FA_2}$), ($x_3|Fa_1, x_3.a_1|id_{FA_2}$), ($y_1|Fa_2, y_1.a_1|id_{FA_1}$), ($y_2|Fa_2, y_2.|a_1 1_{FA_1}$), i.e. $x_1|b_1 \to y_1|id_{B_2}$, $x_2|b_1 \to y_2|id_{B_2}$, $x_3|b_1 \to y_1|id_{B_2}$, $y_1|b_2b_3 \to x_1|id_{B_1}$, $y_2|b_2b_3 \to x_2|id_{B_1}$ and one initial K-rule: $b_1b_2b_3 \to b_4$. To attempt to complete the Kan extension presentation do:

```
gap> KB( IR );
```

The output is:

```
[ [ x1*b1, y1 ], [ x1*b4, x1 ], [ x2*b1, y2 ], [ x2*b4, x2 ], [ x3*b1, y1 ],
  [ x3*b4, x1 ], [ b1*b2*b3, b4 ], [ y1*b2*b3, x1 ], [ y2*b2*b3, x2 ] ]
```

In other words to complete the system we have to add the rules

$$x_1|b_4 \to x_1, \quad x_2|b_4 \to x_2, \text{ and } x_3|b_4 \to x_1.$$

The result of attempting to compute the sets by doing:

```
gap> Kan(KAN);
```

is a long list and then:

```
enumeration limit exceeded: complete rewrite system is:
[ [ x1*b1, y1 ], [ x1*b4, x1 ], [ x2*b1, y2 ], [ x2*b4, x2 ], [ x3*b1, y1 ],
  [ x3*b4, x1 ], [ b1*b2*b3, b4 ], [ y1*b2*b3, x1 ], [ y2*b2*b3, x2 ] ]
```

This means that the sets KB for B in B are too large (the limit set in the program is 1000). In fact this example is infinite. The complete rewrite system is output instead of the sets. We can in fact use this to obtain regular expressions for the sets. In this case the regular expressions are:

$$\begin{array}{lcl}
KB_1 & := & (x_1 + x_2 + x_3)|(b_5(b_3{b_4}^*b_5)^*b_3{b_4}^* + id_{B_1}). \\
KB_2 & := & (x_1 + x_2 + x_3)|b_5(b_3{b_4}^*b_5)^*b_3{b_4}^*(b_1) + (y_1 + y_2)|id_{B_2}. \\
KB_3 & := & (x_1 + x_2 + x_3)|b_5(b_3{b_4}^*b_5)^*(b_3{b_4}^*b_1b_2 + id_{B_3}) + (y_1 + y_2)|b_2.
\end{array}$$

The actions of the arrows are defined by concatenation followed by reduction. For example $x_1|b_5b_3b_4b_4b_5$ is an element of KB_3, so b_3 acts on it to give $x_1|b_5b_3b_4b_4b_5b_3$ which is irreducible, and an element of KB_1.

Details of how, in general, to obtain regular expressions will be given in Chapter Four.

2.8 Special Cases of the Kan Rewriting Procedure

2.8.1 Groups and Monoids

ORIGINAL PROBLEM: Given a monoid presentation $mon\langle\Sigma|Rel\rangle$, find a set of normal forms for the monoid presented.
KAN INPUT DATA: Let Γ be the graph with one object and no arrows. Let $X\bullet$ be a one point set. Let B be generated by the graph Δ with one object and arrows labelled by Σ, it has relations $Rel\mathsf{B}$ given by the monoid relations. The functor F maps the object of Γ to the object of Δ.
KAN EXTENSION: The Kan extension presented by $kan\langle\Gamma|\Delta|RelB|X|F\rangle$ is such that $K\bullet$ is a set of normal forms for the elements of the monoid, the arrows of B (elements of PX) act on the right of B by right multiplication. The natural transformation ε makes sure that the identity of B acts trivially and helps to define the normal form function. The normal form function is $w \mapsto \varepsilon_\bullet(1)\cdot(w) := Kw(\varepsilon_\bullet(1))$.

In this case the method of completion is the standard Knuth-Bendix procedure used for many years for working with monoid presentations of groups and monoids. This type of calculation is well documented.

2.8.2 Groupoids and Categories

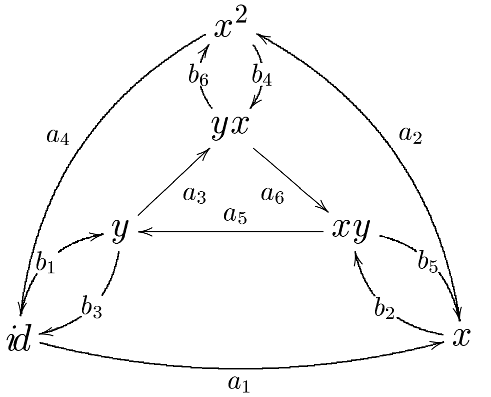
ORIGINAL PROBLEM: To specify a set of normal forms for the elements of a groupoid or category given by a finite category presentation $cat\langle \Lambda | Rel \rangle$.
KAN INPUT DATA: Let Γ be the discrete graph with no arrows and object set equal to $\mathrm{Ob}\Lambda$. Let XA be a distinct one object set for each $A \in \mathrm{Ob}\Gamma$. Let B be the category generated by $\Delta := \Lambda$ with relations $Rel\mathsf{B} := Rel$. Let F be defined by the identity map on the objects.
KAN EXTENSION: Then the Kan extension presented by $kan\langle \Gamma | \Delta | RelB | X | F \rangle$ is such that KB is a set of normal forms for the arrows of the category with target B, the arrows of B (elements of $P\Gamma$) act on the right of B by right multiplication. The natural transformation ε makes sure that the identities of B act trivially and helps to define the normal form function. The normal form function is $w \mapsto \varepsilon_A \cdot (w) := Kw(\varepsilon_A)$.

Example 2.8.1 Consider the group S_3 presented by $\langle x, y | x^3, y^2, xyxy \rangle$. The elements are
$\{id, x, y, x^2, xy, yx\}$. The covering groupoid is generated by the Cayley graph. The 12 generating arrows of the groupoid are $G \times X$:

$$\{[id, x], [x, x], [y, x], \dots, [yx, x], [id, y], [x, y], \dots, [yx, y]\}.$$

To make calculations clearer, we relabel them $\{a_1, a_2, a_3, \dots, a_6, b_1, b_2, \dots, b_6\}$.

The groupoid has 18 relators $G \times R$ – the boundaries of irreducible cycles of the graph. The cycles may be written $[id, x^3]$ and the corresponding boundary is $[id, x][x, x][x^2, x]$ i.e. $a_1a_2a_4$. For the category presentation of the group we could add in the inverses $\{A_1, A_2, \dots, A_6, B_1, B_2, \dots, B_6\}$ with the relators A_1a_1 and a_1A_1 etc and end up with a category presentation with 24 generators and the 42 relations. In this case however the groupoid is finite and so there is no need to do this. For example there would be no need for A_2 because $(a_2)^{-1} = a_4a_1$.

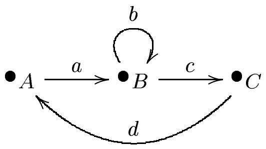


Now suppose the left hand sides of two rules overlap (for example $(a_1a_2a_4, id)$ and $(a_4b_1a_3b_6, id)$) in one of the two possible ways previously described then we have a critical pair $(b_1a_3b_6, a_1a_2)$). The following is GAP output of the completion of the rewrite system for the covering groupoid of our example:

```
gap> Rel;                         ## Input rewriting system:
[ [ a1*a2*a4, IdWord ], [ a2*a4*a1, IdWord ], [ a4*a1*a2, IdWord ],
  [ a3*a6*a5, IdWord ], [ a6*a5*a3, IdWord ], [ a5*a3*a6, IdWord ],
  [ b1*b3, IdWord ], [ b3*b1, IdWord ], [ b2*b5, IdWord ],
  [ b5*b2, IdWord ], [ b4*b6, IdWord ], [ b6*b4, IdWord ],
  [ a1*b2*a5*b3, IdWord ], [ a2*b4*a6*b5, IdWord ],
  [ a3*b6*a4*b1, IdWord ], [ a4*b1*a3*b6, IdWord ],
  [ a5*b3*a1*b2, IdWord ], [ a6*b5*a2*b4, IdWord ] ]
gap> KB( Rel );                    ## Completed rewriting system:
[ [ b1*b3, IdWord ], [ b2*b5, IdWord ], [ b3*b1, IdWord ],
  [ b4*b6, IdWord ], [ b5*b2, IdWord ], [ b6*b4, IdWord ],
  [ a1*a2*a4, IdWord ], [ a1*a2*b4, b1*a3 ], [ a1*b2*a5, b1 ],
  [ a2*a4*a1, IdWord ], [ a2*a4*b1, b2*a5 ], [ a2*b4*a6, b2 ],
  [ a3*a6*a5, IdWord ], [ a3*a6*b5, b3*a1 ], [ a3*b6*a4, b3 ],
  [ a4*a1*a2, IdWord ], [ a4*a1*b2, b4*a6 ], [ a4*b1*a3, b4 ],
  [ a5*a3*a6, IdWord ], [ a5*a3*b6, b5*a2 ], [ a5*b3*a1, b5 ],
  [ a6*a5*a3, IdWord ], [ a6*a5*b3, b6*a4 ], [ a6*b5*a2, b6 ],
  [ b1*a3*a6, a1*b2 ],  [ b1*a3*b6, a1*a2 ], [ b2*a5*a3, a2*b4 ],
  [ b2*a5*b3, a2*a4 ],  [ b3*a1*a2, a3*b6 ], [ b3*a1*b2, a3*a6 ],
  [ b4*a6*a5, a4*b1 ],  [ b4*a6*b5, a4*a1 ], [ b5*a2*a4, a5*b3 ],
  [ b5*a2*b4, a5*a3 ],  [ b6*a4*a1, a6*b5 ], [ b6*a4*b1, a6*a5 ] ]
```

It is possible from this to enumerate elements of the category. One method is to start with all the shortest arrows $(a_1, a_2, \ldots, b_6)$ and see which ones reduce and build inductively on the irreducible ones:
Firstly we have the six identity arrows id_{id}, id_x, id_y, id_{x^2}, id_{xy}, id_{yx}.
Then the generators a_1, a_2, a_3, a_4, a_5, a_6, b_1, b_2, b_3, b_4, b_5, b_6 are all irreducible.
Now consider paths of length 2:
a_1a_2, a_1b_2, a_2a_4, a_2b_4, a_3a_6, a_3b_6, a_4a_1, a_4b_1, a_5a_3, a_5b_3, a_6a_5, a_6b_5, b_1a_3, $b_1b_3 \to id_{id}$,
b_2a_5, $b_2b_5 \to id_x$, b_3a_1, $b_3b_1 \to id_y$, b_4a_6, $b_4b_6 \to id_{x^2}$, b_5a_2, $b_5b_2 \to id_{xy}$, b_6a_4, $b_6b_4 \to id_{yx}$.
Building on the irreducible paths we get the paths of length 3: $a_1a_2a_4 \to id_{id}$, $a_1a_2b_4 \to b_1a_3$,
$a_1b_2a_5 \to b_1$, $a_1b_2b_5 \to a_1$, $a_2a_4a_1 \to id_x, \ldots$
All of them are reducible, and so we can't build any longer paths; the covering groupoid has 30 morphisms and 6 identity arrows and is the tree groupoid with six objects.

Example 2.8.2 This is a basic example to show how it is possible to specify the arrows in an infinite small category with a finite complete presentation. Let C be the category generated by the following graph Γ



with the relations $b^2c = c$, $ab^2 = a$. This rewriting system is complete, and so we can determine whether two arrows in the free category $P\Gamma$ are equivalent in C. An automaton can be drawn (see chapter 3), and from this we can specify the language which is the set of normal forms. It is in fact

$$a(cd(acd) * ab + bcd(acd) * ab) + b^{\dagger} + cd(acd)^{*}ab + d(acd)^{*}ab$$

(and the three identity arrows) where $(acd)^*$ is used to denote the set of elements of $\{acd\}^*$ (similarly $b^\dagger$), so $d(acd)*$, for example, denotes the set $\{d, dacd, dacdacd, dacdacdacd, \ldots\}$, $+$ denotes the union and $-$ the difference of sets. This is the standard notation of languages and regular expressions.

2.8.3 Coset systems and Congruences

ORIGINAL PROBLEM: Given a finitely presented group G and a finitely generated subgroup H find a set of normal forms for the coset representatives of G with respect to H.
KAN INPUT DATA: Let Γ be the one object graph Γ with arrows labelled by the subgroup generators. Let $X\bullet$ be a one point set on which the arrows of Γ act trivially. Let B be the category generated by the

one object graph Δ with arrows labelled by the group generators, with the relations $Rel\mathsf{B}$ of B being the group relations. Let F be defined on Γ by inclusion of the subgroup elements to the group.
KAN EXTENSION: The Kan extension presented by $kan\langle\Gamma|\Delta|RelB|X|F\rangle$ is such that the set $K\bullet$ is a set of representatives for the cosets, Kb defines the action of the group on the cosets $Hg \mapsto Hgb$ and $\varepsilon_\bullet$ maps the single element of $X\bullet$ to the representative for H in $K\bullet$. Therefore it follows that the Kan extension defined is computable if and only if the coset system is computable.

In the monoidal case F is the inclusion of the submonoid A of the monoid B, and the action is trivial as before. The Kan extension of this action gives the quotient of B by the right congruence generated by A, namely the equivalence relation generated by $ab \sim b$ for all $a \in \mathsf{A}, b \in \mathsf{B}$, with the induced right action of B.

It is appropriate to give a calculated example here. The example is infinite so standard Todd-Coxeter methods will not terminate, but the Kan extension / rewriting procedures enable the complete specification of the coset system.

Example 2.8.3 Let B represent the infinite group presented by

$$grp\langle a, b, c \mid a^2b = ba, a^2c = ca, c^3b = abc, caca = b\rangle$$

and let A represent the subgroup generated by $\{c^2\}$.
We obtain one initial ε-rule (because A has one generating arrow) i.e. $H|c^2 \to H|id$.
We also have four initial K-rules corresponding to the relations of B:

$$a^2b \to ba,\ a^2c \to ca,\ c^3b \to abc,\ caca \to b.$$

Note: On completion of this rewriting system for the group, we find 24 rules and for all $n \in \mathbb{N}$ both a^n and c^n are irreducibles with respect to this system (one way to prove that the group is infinite).

The five rules are combined and an infinite complete system for the Kan extension of the action is easily found (using Knuth-Bendix with the length-lex order). The following is the GAP output of the set of 32 rules:

```
[ [ H*b, H*a ], [ H*a^2, H*a ], [ H*a*b, H*a ], [ H*c*a, H*a*c ],
  [ H*c*b, H*a*c ], [ H*c^2, H ], [ a^2*b, b*a ], [ a^2*c, c*a ],
  [ a*b^2, b^2 ], [ a*b*c, c*b ], [ a*c*b, c*b ], [ b*a^2, b*a ],
  [ b*a*b, b^2 ], [ b*a*c, c*b ], [ b^2*a, b^2 ], [ b*c*a, c*b ],
  [ b*c*b, b^2*c ], [ c*a*b, c*b ], [ c*b*a, c*b ], [ c*b^2, b^2*c ],
  [ c*b*c, b^2 ], [ c^2*b, b^2 ], [ H*a*c*a, H*a*c ], [ H*a*c^2, H*a ],
  [ b^4, b^2 ], [ b^3*c, c*b ], [ b^2*c^2, b^3 ], [ b*c^2*a, b^2 ],
  [ c*a*c*a, b ], [ c^2*a^2, b*a ], [ c^3*a, c*b ], [ c*a*c^2*a, c*b ] ]
```

Note that the rules without H i.e. the two-sided rules, constitute a complete rewriting system for the group. The set KB (recall that there is only one object B of B) is infinite. It is the set of (right) cosets of the subgroup in the group. Examples of these cosets include:

$$H, Ha, Hc, Ha^2, Hac, Ha^3, Ha^4, Ha^5, \ldots$$

A regular expression for the coset representatives is:

$$a^* + c + ac.$$

Alternatively consider the subgroup generated by b. Add the rule $Hb \to H$ and the complete system below is obtained:

```
[ [ H*a, H ], [ H*b, H ], [ H*c*a, H*c ], [ H*c*b, H*c ], [ H*c^2, H ],
  [ a^2*b, b*a ], [ a^2*c, c*a ], [ a*b^2, b^2 ], [ a*b*c, c*b ],
  [ a*c*b, c*b ], [ b*a^2, b*a ], [ b*a*b, b^2 ], [ b*a*c, c*b ],
  [ b^2*a, b^2 ], [ b*c*a, c*b ], [ b*c*b, b^2*c ], [ c*a*b, c*b ],
  [ c*b*a, c*b ], [ c*b^2, b^2*c ], [ c*b*c, b^2 ], [ c^2*b, b^2 ],
  [ b^4, b^2 ], [ b^3*c, c*b ], [ b^2*c^2, b^3 ], [ b*c^2*a, b^2 ],
  [ c*a*c*a, b ], [ c^2*a^2, b*a ], [ c^3*a, c*b ], [ c*a*c^2*a, c*b ] ]
```

Again, the two-sided rules are the rewriting system for the group. This time the subgroup has index 2, and the coset representatives are *id* and c.

2.8.4 Equivalence Relations and Equivariant Equivalence Relations

ORIGINAL PROBLEM: Given a set Ω and a relation Rel on Ω. Find a set of representatives for the equivalence classes of the set Ω under the equivalence relation generated by Rel.
KAN INPUT DATA: Let Γ be the graph with object set Ω and generating arrows $a : A_1 \to A_2$ if $(A_1, A_2) \in Rel$. Let $XA := \{A\}$ for all $A \in \Omega$. The arrows of Γ act according to the relation, so $src(a) \cdot a = tgt(a)$. Let Δ be the graph with one object and no arrows so that B is the trivial category with no relations. Let F be the null functor.
KAN EXTENSION: The Kan extension presented by $kan\langle\Gamma|\Delta|RelB|X|F\rangle$ is such that $K\bullet := \Omega/\overset{*}{\leftrightarrow}_{Rel}$ is a set of representatives for the equivalence classes of the set Ω under the equivalence relation generated by Rel.

Alternatively let Ω be a set with a group or monoid M acting on it. Let Rel be a relation on Ω. Define Γ to have object set Ω and generating arrows $a : A_1 \to A_2$ if $(A_1, A_2) \in Rel$ or if $A_1 \cdot m = A_2$ Again, $XA := \{A\}$ for $A \in \mathrm{Ob}\Gamma$ and the arrows act as in the case above. Let Δ be the one object graph with arrows labelled by generators of M and for B let $Rel\mathsf{B}$ be the set of monoid relations. Let F be the null functor. The Kan extension gives the action of M on the quotient of X by the M-equivariant equivalence relation generated by Rel. This example illustrates the advantage of working in categories, since this is a coproduct of categories which is a fairly simple construction.

2.8.5 Orbits of Actions

ORIGINAL PROBLEM: Given a group G which acts on a set Ω, find a set KB of representatives for the orbits of the action of A on Ω.
KAN INPUT DATA: Let Γ be the one object graph with arrows labelled by the generators of the group. Let $X\bullet := \Omega$. Let Δ be the one object, zero arrow graph generating the trivial category B with $Rel\mathsf{B}$ empty. Let F be the null functor.
KAN EXTENSION: The Kan extension presented by $kan\langle\Gamma|\Delta|RelB|X|F\rangle$ is such that $K\bullet$ is a set of representatives for the orbits of the action of the group on Ω.
We present a short example to demonstrate the procedure in this case.

Example 2.8.4 Let A be the symmetric group on three letters with presentation $mon\langle a, b | a^3, b^2, abab\rangle$ and let X be the set $\{v, w, x, y, z\}$. Let A act on X by giving a the effect of the permutation $(v\ w\ x)$ and b the effect of $(v\ w)(y\ z)$.

In this calculation we have a number of ε-rules and no K-rules. The ε-rules just list the action, namely (trivial actions omitted):

$$v \to w,\ w \to x,\ x \to v,\ v \to w,\ w \to v,\ y \to z,\ z \to y.$$

The system of rules is complete and reduces to $\{w \to v,\ x \to v,\ z \to y\}$. Enumeration is simple: $v,\ w \to v,\ x \to v,\ y,\ z \to y$, so there are two orbits of Ω represented by v and y.

This is a small example. With large examples the idea of having a minimal element (normal form) in each orbit to act as an anchor or point of comparison makes a lot of sense. This situation serves as another illustration of rewriting in the framework of a Kan extension, showing not only that rewriting gives a result, but that it is the procedure one uses naturally to do the calculation.

One variation of this is if Ω is the set of elements of the group and the action is conjugation: $x^a := a^{-1}xa$. Then the orbits are the *conjugacy classes* of the group.

Example 2.8.5 Consider the quarternion group, presented by $\langle a, b \mid a^4, b^4, abab^{-1}, a^2b^2\rangle$ and $\Omega = \{\mathit{id},\ a,\ b,\ a^2,\ ab,\ ba,\ a^3,\ a^2b\}$ – enumerating the elements of the group using the method described in Example 3. Construct the Kan extension as above, where the actions of a and b are by conjugation on elements of A.
There are 16 ε-rules which reduce to $\{a^3 \to a,\ a^2b \to b,\ ba \to ab\}$. The conjugacy classes are enumerated by applying these rules to the elements of A. The irreducibles are $\{\mathit{id},\ a,\ b,\ a^2,\ ab\}$, and these are representatives of the five conjugacy classes.

2.8.6 Colimits of Diagrams of Sets

ORIGINAL PROBLEM: Given a presentation of a category action $act\langle\Gamma|X\rangle$ find the colimit of the diagram in Sets on which the category action is defined.
KAN INPUT DATA: Let Γ and X be those given by the action presentation. Let Δ be the graph with one object and no arrows that generates the trivial category B with $Rel\mathsf{B}$ empty. Let F be the null functor.
KAN EXTENSION: The Kan extension presented by $kan\langle\Gamma|\Delta|RelB|X|F\rangle$ is such that $K\bullet$ is the colimit object, and ε is the set of colimit functions of the functor $X : \mathsf{A} \to \mathsf{Sets}$.

Particular examples of this are when A has two objects A_1 and A_2, and two non-identity arrows a_1 and a_2 from A_1 to A_2, and Xa_1 and Xa_2 are functions from the set XA_1 to the set XA_2 (*coequaliser* of a_1 and a_2 in Sets); A has three objects A_1, A_2 and A_3 and two non-identity arrows $a_1 : A_1 \to A_2$ and $a_2 : A_1 \to A_3$. XA_1, XA_2 and XA_2 are sets, and Xa_1 and Xa_2 are functions between these sets (*pushout* of a_1 and a_2 in Sets). The following example is included not as an illustration of rewriting but to show another situation where presentations of Kan extensions can be used to express a problem naturally.

Example 2.8.6 Suppose we have two sets $\{x_1, x_2, x_3\}$ and $\{y_1, y_2, y_3, y_4\}$, with two functions from the first to the second given by $(x_1 \mapsto y_1,\ x_2 \mapsto y_2,\ x_3 \mapsto y_3)$ and $(x_1 \mapsto y_1,\ x_2 \mapsto y_1,\ x_3 \mapsto y_3)$.
Then we can calculate the coequaliser. We have a number of ε-rules

$$y_1|\mathit{id}_\bullet \to x_1|\mathit{id}_\bullet,\ y_2|\mathit{id}_\bullet \to x_2|\mathit{id}_\bullet,\ y_3|\mathit{id}_\bullet \to x_3|\mathit{id}_\bullet,\ y_1|\mathit{id}_\bullet \to x_1|\mathit{id}_\bullet,\ y_2|\mathit{id}_\bullet \to x_1|\mathit{id}_\bullet,\ y_3|\mathit{id}_\bullet \to x_3|\mathit{id}_\bullet.$$

There is just one overlap, between $(y_2|\mathit{id}_\bullet \to x_1|\mathit{id}_\bullet)$ and $(y_2|\mathit{id}_\bullet \to x_2|\mathit{id}_\bullet)$: to resolve the critical pair we add the rule $(x_2|\mathit{id}_\bullet \to x_1|\mathit{id}_\bullet)$, and the system is complete:

$$\{y_1|\mathit{id}_\bullet \to x_1|\mathit{id}_\bullet,\ y_2|\mathit{id}_\bullet \to x_1|\mathit{id}_\bullet,\ y_3|\mathit{id}_\bullet \to x_3|\mathit{id}_\bullet,\ x_2|\mathit{id}_\bullet \to x_1|\mathit{id}_\bullet\}.$$

The elements of the set $K\bullet$ are easily enumerated:

$$x_1|\mathit{id}_\bullet,\ x_2|\mathit{id}_\bullet \to x_1|\mathit{id}_\bullet,\ x_3|\mathit{id}_\bullet,\ y_1|\mathit{id}_\bullet \to x_1|\mathit{id}_\bullet,\ y_2|\mathit{id}_\bullet \to x_1|\mathit{id}_\bullet,\ y_3|\mathit{id}_\bullet \to x_3|\mathit{id}_\bullet,\ y_4|\mathit{id}_\bullet.$$

So the coequalising set is

$$K\bullet = \{x_1|\mathit{id}_\bullet, x_3|\mathit{id}_\bullet, y_4|\mathit{id}_\bullet\},$$

and the coequaliser function to it from XA_2 is given by $y_i \mapsto y_i|\mathit{id}_\bullet$ for $i = 1, \ldots, 4$ followed by reduction defined by $\to$ to an element of $K\bullet$.

2.8.7 Induced Permutation Representations

Let A and B be groups and let $F : \mathsf{A} \to \mathsf{B}$ be a morphism of groups. Let A act on the set XA. The Kan extension of this action along F is known as the action of B *induced* from that of A by F, and is written $F_*(XA)$. It can be constructed simply as the set $X \times \mathsf{B}$ factored by the equivalence relation generated by $(xa, b) \sim (x, F(a)b)$ for all $x \in XA, a \in \mathsf{A}, b \in \mathsf{B}$. The natural transformation ε is given by $x \mapsto [x, 1]$, where $[x, b]$ denotes the equivalence class of (x, b) under the equivalence relation $\sim$. The morphism F can be factored as an epimorphism followed by a monomorphism, and there are other descriptions of $F_*(XA)$ in these cases, as follows.

Suppose first that F is an epimorphism with kernel N. Then we can take as a representative of $F_*(XA)$ the orbit set X/N with the induced action of B.

Suppose next that F is a monomorphism, which we suppose is an inclusion. Choose a set T of representatives of the right cosets of A in B, so that $1 \in T$. Then the induced representation can be taken to be $XA \times T$ with ε given by $x \mapsto (x, 1)$ and the action given by $(x, t)^b = (xa, u)$ where $t, u \in T, b \in \mathsf{B}, a \in \mathsf{A}$ and $tb = au$.

On the other hand, in practical cases, this factorisation of F may not be a convenient way of determining the induced representation. In the case A, B are monoids, so that XA is a transformation representation of A on the set XA, we have in general no convenient description of the induced transformation representation except by one form or another of the construction of the Kan extension.

Chapter 3

Noncommutative Gröbner Bases (over fields)

The results and methods that will be discussed in this chapter use, or are related to, the noncommutative version of Gröbner bases. The first section therefore contains a very brief introduction to the area of computer algebra known as Gröbner basis theory.

Section 2 describes explicitly the relation between Gröbner bases and string rewriting. It is well known [56] that Buchberger's algorithm can be applied to rewriting problems, a complete rewrite system being equivalent to a Gröbner basis, and observations have been made on the similarities between the Knuth-Bendix and Buchberger algorithms in this case. However, the exact relation of the algorithms (i.e. that Knuth-Bendix is a special case of Buchberger's algorithm) is not widely recognised. This section makes the correspondence explicit.

Section 3 builds on the results discussed previously. One-sided rewriting systems and their relations to the calculation of one-sided ideals are considered. The results are as follows:- A complete one-sided rewrite system for a right congruence on a semigroup S is equivalent to a Gröbner basis for a right ideal in an algebra $K[S]$ (to some extent, this is already known). In addition, the one-sided Knuth-Bendix algorithm is a special case of the Buchberger algorithm (new). The section concludes with an original application of the one-sided Buchberger algorithm to computing Green's relations for a couple of monoids. This method for computing Green's relations directly from a presentation has certain advantages of convenience and efficiency over conventional methods (using transformation representations), and can also deal with infinite problems.

Section 4 begins by showing how Gröbner basis techniques may be applied to K-category presentations. It then rounds off the chapter by placing the Gröbner basis techniques for noncommutative polynomial algebras in terms of Kan extensions. This begins to give a new perspective on noncommutative Gröbner bases and relates them more strongly to category theory.

3.1 Historical Introduction to Gröbner Bases

In 1926 Hermann posed a question [38] which has since arisen in different forms in various areas and has become known as the **ideal membership problem**. The problem is usually described in the following way. Let $X := \{x_1, \dots, x_n\}$ be a set of commuting variables and let K be a field. Then define $K[X]$ to be the polynomial ring, whose monomials are power products of the x_i and whose coefficients are from K. Given a set F of polynomials $f_1, \dots, f_k \in K[X]$ let $\langle F \rangle$ denote the ideal generated by F. Given another polynomial $f \in K[X]$, the problem is to determine whether f is a member of $\langle F \rangle$. This is equivalent to asking whether there are polynomials $h_1, \dots, h_k \in K[X]$ such that $f = h_1 f_1 + \cdots + h_k f_k$.

In 1965 Bruno Buchberger devised a solution [22] to this problem. His invention, Gröbner bases (named for his supervisor), are special generating sets for ideals in polynomial rings. Typically, one uses an ordering on the monomials of the polynomial ring $K[X]$ to work on a generating set F for the ideal $\langle F \rangle$, computing (using Buchberger's algorithm) a Gröbner basis for the ideal.

It took about ten years before the concept became known to research communities in Mathematics and Theoretical Computer Science. It is now well recognised at least that Gröbner basis techniques enable us to answer questions of algebraic interest:

i) The Ideal Description Problem: does every ideal $I \in K[X]$ have a finite generating set?

ii) The Ideal Membership Problem: does a particular polynomial f lie in an ideal $\langle F \rangle$?

iii) The Problem of Solving Polynomial Equations: find all common solutions in K^n of a system of polynomial equations in n variables.

iv) Equality problem: are two polynomials f, f' equal in the quotient ring $K[X]/\langle F \rangle$ (this is equivalent to asking whether $f - f'$ is a member of $\langle F \rangle$)?

v) Intersection Problem: What is the intersection in $K[X]$ of two ideals $\langle F \rangle$ and $\langle F' \rangle$?

The problem we will be concentrating on is the membership problem. The others are looked at in more detail in [29].

Gröbner basis theory has since become an important part of computational algebra; in the commutative case it is included in all major symbolic computation program systems and is applied in a wide variety of seemingly unrelated research areas. To name a few: applications have been found in robotics, computational geometry, statistical analysis and geometric theorem proving. Further applications to surface modelling and cryptography are under investigation. It is thought [9] "inevitable that like Galois theory, Buchberger theory will become a tool used by pure mathematicians in proofs". For the moment it is used to compute specific examples.

Since Buchberger introduced Gröbner bases for ideals in commutative polynomial rings over fields, a number of authors have extended and generalised the theory to other algebraic objects. In 1978 Bergman extended the notion of Gröbner bases to the case where the variables of X do not commute [5]. He also attempted to generalize Buchberger's algorithm for computing the bases, but this was much improved by F.Mora in 1986 [55] who gives a variant of the Buchberger's algorithm which is guaranteed to halt, returning a finite Gröbner basis of the finitely generated ideal (with respect to a fixed ordering) if and only if such a basis exists. This procedure is described and illustrated later, I have implemented it in GAP (the program is `grobner.g`).

The procedure has certain disadvantages compared with the commutative method:

i) Termination: the procedure will not necessarily terminate. When running the procedure, unless it actually does terminate, we cannot tell whether or not it is going to at some point. Some judgement must therefore be made, to say that if it has not completed after a certain amount of time or a certain number of passes in the program, it may be considered to have failed – this is referred to as "forcing termination".

ii) Orderings: the orderings used have to be more complicated, because the order of the generators is important. This means that most orderings will use the lexicographic order at some point.

iii) If an attempt is considered to have failed (see(i)), there is still the possibility that a different ordering may the procedure may be successful. There are infinitely many orderings on a free semigroup and so the procedure may in general be attempted infinitely many times before a Gröbner basis is found.

iv) In the commutative case there is a procedure known as the Gröbner walk, whereby you can convert a basis with respect to one order to a basis with respect to a new order [2]. This does not work in the noncommutative case because it may be that there is no Gröbner basis with respect to the second order.

In summary, one has an infinite number of orderings to try, and also a small problem of knowing when to stop trying one ordering and consider another. T. Mora points out that as there are infinitely many orders, the chances of finding the correct one in finitely many attempts could be zero. Hence the idea of trying many possible orderings in parallel: T. Mora was not put off by the idea of running many possible systems simultaneously, each new polynomial creating as many new systems as there are ways of choosing its leading term in a way compatible with the original system. This may be a nightmare computationally, but does produce an algorithm which is guaranteed to halt if and only if I has a finite Gröbner basis with respect to some ordering (which satisfies a certain property FDR). This method is theoretically powerful, being limited only by the ability to produce orderings satisfying the FDR property (there are infinitely many such orderings). However, we still have the problem that failure to terminate within a certain time proves nothing, and also that the computations get very big very quickly, and in terms of implementation in GAP, my program which attempts completion of a single system using one ordering can be quite slow enough...

The other method of extending Buchberger's theory was to keep the commuting variables the same and change the structure of the field of coefficients. In 1978 Zacharius considered commutative polynomial rings with coefficients in commutative, unital rings, satisfying some computability requirements [80]. More recently (1989) Möller worked on the same problem [54]. We find motivation for this direction in Chapter Five.
My main reference and starting point was T.Mora's paper [56]. Two useful introductory books are those by Cox, Little and O'Shea [29] and Adams and Lousannau [1].

Gröbner basis theory continues to develop and generates "increasing interest because of its usefulness in providing computational tools which are applicable to a wide range of problems in mathematics, science and engineering" [1]. A conference marking 33 years of Gröbner bases was recently held at R.I.S.C. in Linz, and the proceedings [23] contain papers on many different aspects of Gröbner bases (including a few on the noncommutative case) which are currently being researched. (There were plans to compile a database of all the Gröbner basis material, to be accessible through the R.I.S.C. internet site.)

3.1.1 Algebra Presentations

Let K be a field. A **K-algebra** is a set A with a unique element 0, two binary operations $+$ and $*$ and a scalar multiplication of elements of A by elements of K satisfying the following properties.

$$\begin{array}{llll}
\text{i)} & a+(b+c)=(a+b)+c, & \text{ii)} & a+0=0+a=a, \\
\text{iii)} & \exists -a\in A : a+(-a)=0, & \text{iv)} & a+b=b+a, \\
\text{v)} & k(a+b)=ka+kb, & \text{vi)} & (k+h)a=ka+ha, \\
\text{vii)} & (kh)a=k(ha), & \text{viii)} & (0)a=0, \\
\text{ix)} & a*(b*c)=(a*b)*c, & \text{x)} & a*(b+c)=(a*b)+(b*c), \\
\text{xi)} & (b+c)*d=(b*d)+(c*d), & \text{xii)} & (ka)*b=k(a*b)=a*(kb),
\end{array}$$

for all $k,h\in K$ and $a,b,c,d\in A$.

An **ideal** I in a K-algebra A is a sub-K-algebra I (closed under $+$, $-$, $*$ and scalar multiplication) such that for all $f \in I$ and $a, b \in A$, $a * f * b \in I$.

Recall that $X^\dagger$ denotes the free semigroup of all nonempty strings of elements on the set X. A set of relations on the free semigroup is a set $R \subseteq X^\dagger \times X^\dagger$. A set of relations R generates a congruence $=_R$ on the free semigroup. The factor semigroup $X^\dagger / =_R$ is the semigroup of congruence classes of $X^\dagger$ under $=_R$. A **semigroup presentation** is a pair $sgp\langle X|R\rangle$ where X is a set and R is a set of relations on $X^\dagger$. The semigroup it presents is the factor semigroup $X^\dagger / =_R$. This definition will be used in a later theorem.

Let S be a semigroup and let K be a field. The **free K-algebra** $K[S]$ on S consists of all the polynomials (formal sums) $k_1 m_1 + \cdots + k_t m_t$ where $k_1, \ldots, k_t \in K$, $m_1, \ldots, m_t \in S$. Addition of polynomials is defined using the formal sums ($+$ is commutative). The zero element of the algebra is denoted 0. Multiplication of polynomials is defined in the usual way: $\Sigma_i k_i m_i * \Sigma_j h_j n_j = \Sigma_{i,j} k_i h_j m_i n_j$. When the semigroup S has an identity element id the algebra $K[S]$ has a multiplicative identity also denoted id.

Let $K[S]$ be the free K-algebra on a semigroup S, where K is a field. Elements (polynomials) may be written $f = k_1 m_1 + \cdots + k_t m_t$ as a sum of **terms** $k_i m_i$, where the $m_i \in S$ are **monomials** and $k_i \in K$ are **coefficients**.

Let $F := \{f_1, \ldots, f_n\}$ be a set of polynomials in $K[S]$. The **ideal generated by** F is denoted $\langle F\rangle$ and defined to have as elements all sums of multiples of elements of F:

$$\langle F\rangle := \{p_1 f_1 q_1 + \cdots + p_n f_n q_n | p_i, q_i \in K[S]\}.$$

Given $K[S]$ and F the **ideal membership problem** is:

INPUT:	$f \in K[S]$	(a polynomial of the free algebra),
QUESTION:	$f \in \langle F\rangle$?	(is it in the ideal?)

i.e. are there polynomials $p_1, \ldots, p_n, q_1, \ldots, q_n \in K[S]$ so that $f = p_1 f_1 q_1 + \cdots + p_n f_n q_n$?

The ideal determines a congruence $=_F$ on $K[S]$ where

$$f =_F h \Leftrightarrow f - h \in \langle F\rangle.$$

The proof is straightforward (if $f =_F h$ then $p(f-h)q \in \langle F\rangle$ so $pfq =_F phq$ for all $p, q \in K[S]$).

The **factor algebra** $K[S]/ =_F$ is the algebra whose elements are congruence classes $[f]$ of elements of $K[S]$ with respect to $=_F$. Addition is defined by $[f] + [h] := [f + h]$ and multiplication by $[f][h] = [fh]$ for all $f, h \in K[S]$. Scalar multiplication is also preserved, so $k[f] = [kf]$, for $k \in K, f \in K[S]$. It can be verified that this is an algebra by checking the axioms i to xii.

A **K-algebra presentation** is a pair $alg\langle S|F\rangle$ where S is a semigroup and $F \subseteq K[S]$. The algebra A that it presents is the factor algebra $K[S]/ =_F$.

The **equality problem** for a K-algebra presentation $alg\langle S|F\rangle$ is as follows:

INPUT:	$f, h \in K[S]$	(two polynomials of $K[S]$).
QUESTION:	$f =_F h$?	(are they equivalent under $=_F$?)

This problem is the same as the problem of determining ideal membership of $f - h$ (by definition of $=_F$). Therefore the equality problem asks whether there are $p_1, \ldots, p_n, q_1, \ldots, q_n \in K[S]$ such that $f - h = p_1 f_1 q_1 + \cdots + p_n f_n q_1$. Recall that a set of normal forms for $=_F$ contains exactly one element from each congruence class and a normal form function $N : K[S] \to K[S]$ is such that $N(K[S])$ is a set of normal forms, and for all $p \in K[S], p =_F N(p)$.

The approach is to construct a reduction relation $\to_F$ on $K[S]$, that is compatible with a well-ordering (so $\to_F$ is Noetherian) and to attempt to make this reduction relation confluent by changing the set F that generates it without changing the congruence $\overset{*}{\leftrightarrow}_F$.
Let $>$ be an admissible well-ordering on the semigroup S (i.e. a well-ordering $>$ such that $m > n \Rightarrow umv > unv$ for all $u, v \in S$).

The **leading term** of a polynomial is the term with the largest monomial with respect to the chosen ordering on S. The **leading monomial** is the monomial of the leading term, and the **leading coefficient** is the coefficient of the leading term. We will assume all polynomials to be monic, as we are working over a field, and can therefore divide all polynomials by their leading coefficient. The function `LM` is used to extract the leading monomial of a polynomial. The **remainder `rem`** of a polynomial f satisfies

$$f = \mathtt{LM}(f) - \mathtt{rem}(f).$$

We use the notions of leading monomials to define a Noetherian reduction relation.

Definition 3.1.1 ***Reduction*** *of a polynomial $h = k_1m_1 + \cdots + k_tm_t$ where $k_1, \ldots, k_t \in K$ and $m_1, \ldots m_t \in S$ with respect to a basis $F = \{f_1, \ldots, f_n\}$ is possible if any of the monomials m_j of h is a multiple of a leading monomial of any $f_i \in F$. Suppose $f_i = l_i - r_i$ (leading monomial l_i and remainder r_i) and that m_j is a monomial in h such that $m = ul_iv$ for some $u, v \in S$. Then m reduces to ur_iv, and h reduces to $k_1m_1 + \cdots + k_jur_iv + \cdots + k_ts_t$ i.e.*

$$h \to h - k_ju(f_i)v.$$

If none of the leading terms of any of the polynomials in F is a subword of any of the monomials of h, then h is said to be ***irreducible****.*

Lemma 3.1.2 *The reduction relation $\to_F$ is Noetherian.*

Proof For a proof by contradiction, suppose that $\to_F$ is not Noetherian. Then there exists some infinite sequence of reductions $h_1 \to_F h_2 \to_F h_3 \to_F \cdots$. This implies that there is an infinite sequence of monomials $m_1 > m_2 > m_3 > \cdots$. This is not the case as $>$ is a well-ordering, therefore there is no infinite sequence of reductions and $\to_F$ is Noetherian. □

The reflexive, symmetric, transitive closure of $\to_F$ is denoted $\overset{*}{\leftrightarrow}_F$.

Lemma 3.1.3 *Let $F = \{f_1, \ldots, f_n\}$ be a basis for an ideal on the free K-algebra $K[S]$ on a semigroup S. Then $\overset{*}{\leftrightarrow}_F$ and $=_F$ coincide.*

Proof Suppose $f \overset{*}{\leftrightarrow}_F h$. Then $f - h = kulv - kurv = ku(l-r)v$ for some $k \in K,\ u, v \in S,\ p \in K[S]$ and $(l - r) \in F$. Therefore $f - h \in \langle F \rangle$. Hence $\overset{*}{\leftrightarrow}_F$ is contained in $=_F$.
For the converse, suppose $f =_F h$. Then by definition $f - h \in \langle F \rangle$. Therefore there exist $p_1, \ldots, p_n$, $q_1, \ldots, q_n \in K[S]$ such that $f - h = p_1f_1q_1 + \cdots + p_nf_nq_n$. Now we can write $p_if_iq_i = k_{1_i}u_{1_i}f_iv_{1_i} + \cdots + k_{t_i}u_{t_i}f_iv_{t_i}$ for some $k_{1_i}, \ldots, k_{t_i} \in K$, $u_{1_i}, \ldots, u_{t_i}$, $v_{1_i}, \ldots, v_{t_i} \in S$. Consider the sequence of one step reductions $f \to_F f - k_{1_1}u_{1_1}f_1v_{1_1} \to_F f - k_{1_1}u_{1_1}f_1v_{1_1} - k_{2_1}u_{2_1}f_1v_{2_1} \to_F \cdots$. The result will follow if $h \leftrightarrow_F h - k_ju_jf_jv_j$ for all $h \in K[S], u_j, v_j \in S, k_j \in K$. Now either h contains a term $k_ju_jl_jv_j$ where $l_f = \mathtt{LM}(f_j)$ in which case $h \to_F h - k_ju_jf_jv_j$ or else it does not, and $h - k_ju_jf_jv_j \to (h - k_ju_jf_jv_j) + k_ju_jf_jv_j = h$. Either way $h \leftrightarrow_F h - k_ju_jf_jv_j$, and so $=_F$ is contained in $\overset{*}{\leftrightarrow}_F$. Hence we have proved that $=_F$ coincides with $\overset{*}{\leftrightarrow}_F$. □

Definition 3.1.4 *A **Gröbner basis** G for an ideal I of $K[S]$ is a basis for I that generates a complete reduction relation $\to_F$ (with respect to an admissible well-ordering on S) on $K[S]$.*

Equivalent conditions for a basis being a Gröbner basis are that an element $h \in K[S]$ is an element of I if and only if it reduces to zero by $\to_G$. Or [55], a set $G \subseteq I$ of polynomials is a Gröbner basis for I if the ideal generated (in S) by the leading monomials of G is equal to the ideal generated by the leading monomials of I. A basis F is not a Gröbner basis with respect to an order $>$ if $\to$ is not locally confluent. (Local confluence and confluence are equivalent for a Noetherian reduction relation). If it is not confluent then there is a critical pair of polynomials, obtained by reducing one polynomial in two different ways.

Definition 3.1.5 *Let $K[S]$ be a K-algebra and let $F \subseteq K[S]$. An **S-polynomial** occurs when a polynomial $h \in K[S]$ may be reduced in two distinct ways $h \to_F h_1$ and $h \to_F h_2$, $h_1 \neq h_2$ for $h_1, h_2 \in K[S]$. The S-polynomial is defined to be the difference $h_1 - h_2$ between the reduced polynomials. When an S-polynomial can be reduced to zero we say that it can be **resolved**.*

If the distinct reductions apply to different terms of the polynomial then it is clear that further reduction will yield a polynomial res so that $h_1 \to_F res$ and $h_2 \to_F res$. In other words, the S-polynomial $h_1 - h_2$ can be resolved. Similarly, if the reductions apply to the same term but do not overlap, the S-polymomial will resolve. The interesting cases occur when the reductions overlap on a monomial. All these cases are multiples of the following situation.

Let $F := \{f_1, \ldots, f_n\} \subseteq K[S]$. A pair of polynomials f_i, f_j is said to have a **match** if their leading monomials overlap. Suppose f_i, f_j are a pair of polynomials whose leading monomials l_i, l_j overlap. Let r_i, r_j denote the remainders of f_i and f_j respectively. The overlap is one of four types: $u_i l_i v_i = l_j$, $l_i = u_j l_j v_j$, $u_i l_i = l_j v_j$ or $l_i v_i = u_j l_j$.

In any case it is possible to write $u_i l_i v_i = u_j l_j v_j$.where $u_i, u_j, v_i, v_j \in S + id$. The S-polynomial resulting from the overlap is $u_i f_i v_i - u_j f_j u_j$ which simplifies to $u_j r_j v_j - u_i r_i v_i$.
If all S-polynomials resulting from an overlaps of polynomials in F resolve, then F is a Gröbner basis. If an S-polynomial does not resolve then it can be added to F without changing $\langle F \rangle$. This is essentially Buchberger's algorithm; all the S-polynomials of a set of polynomials F are found, and are reduced as far as possible with respect to F. Any non-zero remainders are then added to F, and the process is repeated. The flow chart on the next page describes Buchberger's algorithm more precisely.

Example 3.1.6 The following example is an application of noncommutative Gröbner bases to the fourth Hecke algebra H_4. This problem (with other much more complex ones – for the string algebras) was kindly suggested to me by Bruce Westbury (Warwick) to test my computer program `grobner.g`. The algebra H_4 has presentation $\mathbb{Q}[\{e_1, e_2, e_3\}]/{=_P}$ where P is the set of polynomials

$$\{e_1e_1 - e_1,\ e_2e_2 - e_2,\ e_3e_3 - e_3,\ e_3e_1 - e_1e_3,\ e_2e_1e_2 - e_1e_2e_1 + \frac{2}{9}e_2 - \frac{2}{9}e_1,\ e_3e_2e_3 - e_2e_3e_2 + \frac{2}{9}e_3 - \frac{2}{9}e_2\}.$$

We apply the algorithm to P. The first overlap is between the lead monomials e_2e_2 and $e_2e_1e_2$. The unreduced S-polynomial resulting from the overlap in $e_2e_1e_2e_2$ is $(e_2e_1e_2) - (e_1e_2e_1e_2 - \frac{2}{9}e_2e_1 + \frac{2}{9}e_1e_2)$. The remainder of this polynomial modulo P is zero, so it is resolved. In the same way, the S-polynomials resulting from the overlaps in the words $e_2e_2e_1e_1, e_3e_2e_3e_3$ and $e_3e_3e_2e_3$ also resolve. The other overlap is between the leading monomials e_3e_1 and $e_3e_2e_3$. Reduction of $e_3e_2e_3e_1$ gives us an irreducible S-polynomial $(e_3e_2e_1e_3) - (e_2e_3e_2e_1 - \frac{2}{9}e_2e_1 + \frac{2}{9}e_1e_3)$ which we add to P. In fact P is now a Gröbner basis – any other S-polynomials reduce to zero.

This Hecke algebra has dimension 20, which we can prove by using the Gröbner basis to enumerate the irreducible monomials in a catalogue, much like the rewrite situation:

id,
$e_1,\ e_2,\ e_3,$
$e_1e_2,\ e_1e_3,\ e_2e_1,\ e_2e_3,\ e_3e_2,$
$e_1e_2e_1,\ e_1e_2e_3,\ e_1e_3e_2,\ e_2e_1e_3\ ,e_2e_3e_2,\ e_3e_2e_1,$
$e_1e_2e_1e_3,\ e_1e_2e_3e_2,\ e_1e_3e_2e_1,\ e_2e_1e_3e_2,\ e_2e_3e_2e_1.$

Any irreducible polynomial will be a sum of $\mathbb{Q}$-multiples of these monomials, any element of H_4 is representable by exactly one of these polynomials.

3.2 Gröbner Bases and Rewrite Systems

Similarities between the two critical pair completion methods (Knuth-Bendix and Buchberger's algorithm) have often been pointed out. Good (recent) references for this are [75, 70]. In particular it is well known that the commutative Buchberger algorithm may be applied to presentations of abelian groups to obtain a complete rewrite system. Possibly further similarities were not recognised earlier as noncommutative Gröbner bases were some time in developing. Teo Mora [56] recorded that a complete rewrite system for a semigroup S presented by $sgp\langle X|Rel\rangle$ is equivalent to a noncommutative Gröbner basis for the ideal specified by the congruence $=_R$ on $X^\dagger$ in the algebra $K[X^\dagger]$ where K is a field. The ideal is equivalent to S. In fact, we show that step for step, the algorithms in this case are equivalent, and so the Knuth-Bendix algorithm is a special case of Buchberger's algorithm. It is accepted that the work in this section may already be known in some form, though it seems standard to talk in terms of group and monoid rings and apply Buchberger's algorithm to solving the word problem in groups without recognising the restricted algorithm as the Knuth-Bendix algorithm – for recent examples see [8] [57] and [3]. The following lemma is a variation of a result of [56].

Lemma 3.2.1 *Let K be a field and let S be a semigroup with presentation $sgp\langle X|R\rangle$. Then the algebra $K[S]$ is isomorphic to the factor algebra $K[X^\dagger]/=_F$ where F is the basis $\{l_i - r_i) : (l_i, r_i) \in R\}$.*

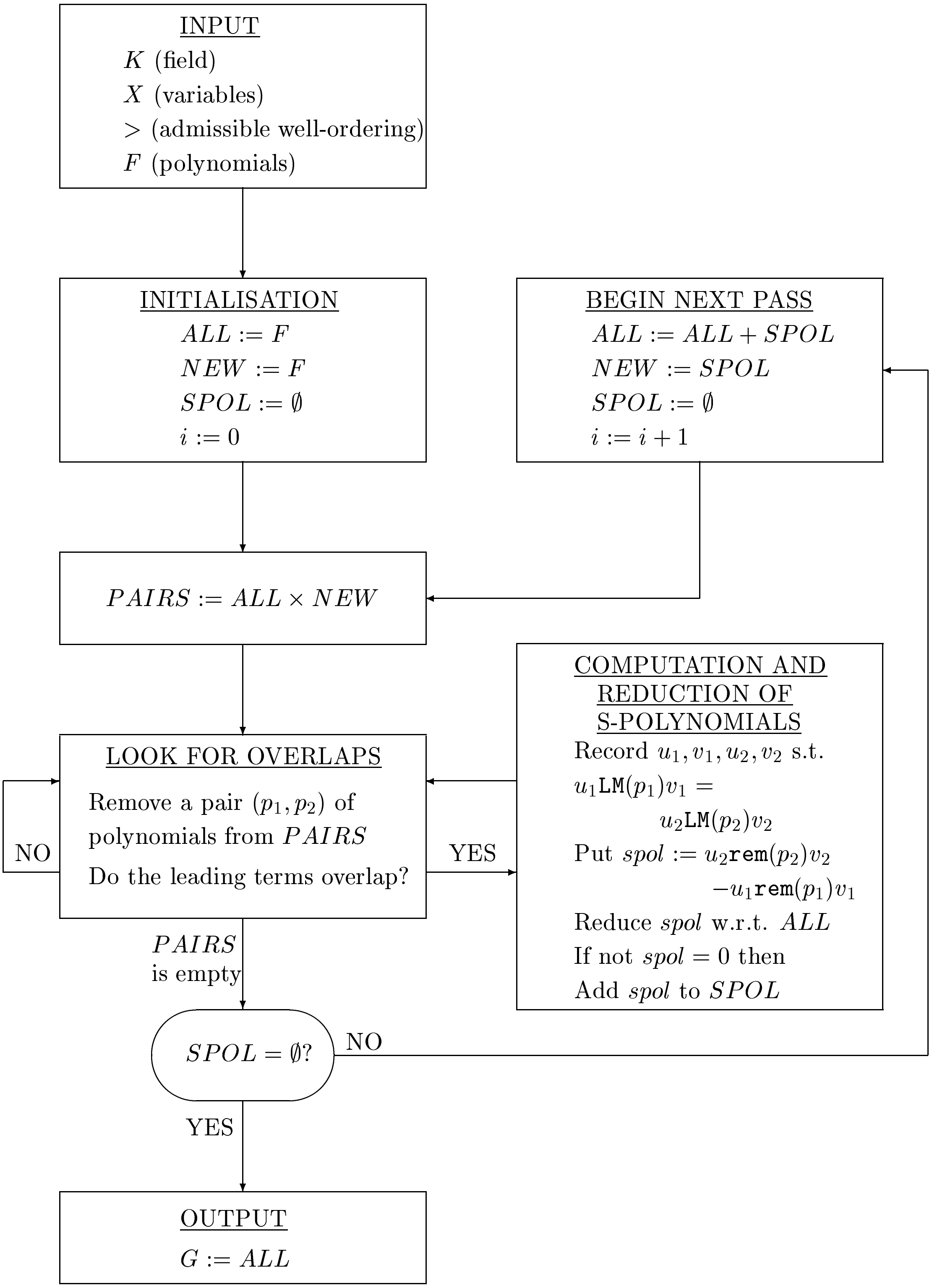
Proof Define $\phi : K[X^\dagger] \to K[S]$ by $\phi(k_1w_1 + \cdots + k_tw_t) := k_1[w_1]_R + \cdots + k_t[w_t]_R$ for $k_1, \ldots, k_t \in K$, $w_1, \ldots, w_t \in X^\dagger$. Define a homomorphism $\phi' : K[X^\dagger]/=_F \to K[S]$ by $\phi'([p]_F) := \phi(p)$. It is injective since $\phi'[p]_F = \phi[q]_F$ if and only if $[p]_F = [q]_F$ (using the definitions $\phi(p) = \phi(q) \Leftrightarrow p =_F q$). It is also surjective. Let $f \in K[S]$. Then $f = k_1m_1 + \cdots + k_tm_t$ for some $k_1, \ldots, k_t \in K$, $m_1, \ldots, m_t \in S$. Since S is presented by $sgp\langle X|R\rangle$ there exist $w_1, \ldots, w_t \in X^\dagger$ such that $[w_i]_R = m_i$ for $i = 1, \ldots, t$. Therefore let $p = k_1w_1 + \cdots + k_tw_t$. Clearly $p \in K[X^\dagger]$ and also $\phi'[p]_F = f$. Hence ϕ' is an isomorphism. □

Theorem 3.2.2 *Let K be a field, let S be a semigroup presented by $sgp\langle X|R\rangle$ and let A be the K-algebra presented by $alg\langle X|F\rangle$ where $F := \{l - r : (l, r) \in R\}$. Then the Knuth-Bendix critical pair completion procedure for R corresponds step-by-step to the noncommutative Buchberger algorithm for finding a Gröbner basis for the ideal generated by F.*

Proof Both the Knuth-Bendix and the Buchberger algorithm begin by specifying a monomial ordering on $X^\dagger$ which we denote $>$. Our proof considers the two procedures in turn, identifying the corresponding components by indexing them (i)-(xii).

In terms of rewriting we consider the *rewrite system* (i) R which consists of a *set of rules* (ii) of the form (l, r) orientated so that $l > r$. A *word* (iii) $w \in X^\dagger$ may be *reduced* (iv) with respect to R if it contains the *left hand side* (v) l of a rule (l, r) as a *subword* (vi) i.e. if $w = ulv$ for some $u, v \in X^*$. To reduce $w = ulv$ using the rule (l, r) we replace l by the *right hand side* (vii) r of the rule, and write $ulv \to_R urv$. The Knuth-Bendix algorithm looks for *overlaps between rules* (viii). Given a pair of rules (l_1, r_1), (l_2, r_2)

The Noncommutative Buchberger Algorithm


INPUT
K (field)
X (variables)
$>$ (admissible well-ordering)
F (polynomials)
INITIALISATION
$ALL := F$
$NEW := F$
$SPOL := \emptyset$
$i := 0$
BEGIN NEXT PASS
$ALL := ALL + SPOL$
$NEW := SPOL$
$SPOL := \emptyset$
$i := i + 1$
$PAIRS := ALL \times NEW$
COMPUTATION AND REDUCTION OF S-POLYNOMIALS
Record u_1, v_1, u_2, v_2 s.t.
$u_1 \mathtt{LM}(p_1) v_1 =$
$u_2 \mathtt{LM}(p_2) v_2$
Put $spol := u_2 \mathtt{rem}(p_2) v_2$
$- u_1 \mathtt{rem}(p_1) v_1$
Reduce $spol$ w.r.t. ALL
If not $spol = 0$ then
Add $spol$ to $SPOL$
LOOK FOR OVERLAPS
Remove a pair (p_1, p_2) of polynomials from $PAIRS$
Do the leading terms overlap?
NO
YES
$PAIRS$ is empty
$SPOL = \emptyset$?
NO
YES
OUTPUT
$G := ALL$

there are four possible ways in which an overlap can occur: $l_1 = u_2l_2v_2$, $u_1l_1v_1 = l_2$, $l_1v_1 = u_2l_2$ and $u_1l_1 = l_2v_2$. The *critical pair* (xi) resulting from an overlap is the pair of words resulting from applying each rule to the smallest word on which the overlap occurs. The critical pairs resulting from each of the four overlaps are: $(r_1, u_2r_2v_2)$, $(u_1r_1v_1, r_2)$, (r_1v_1, u_2r_2) and (u_1r_1, r_2v_2) respectively (see diagram).
In one pass the completion procedure finds all the critical pairs resulting from overlaps of rules of R. Both sides of each of the critical pairs are reduced as far as possible with respect to R to obtain a *reduced critical pair* (x) (c_1, c_2). The original pair is said to *resolve* (xi) if $c_1 = c_2$. The reduced pairs that have not resolved are orientated, so that $c_1 > c_2$, and added to R forming R_1. The procedure is then repeated for the rewrite system R_1, to obtain R_2 and so on. When all the critical pairs of a system R_n resolve (i.e. $R_{n+1} = R_n$) then R_n is a *complete rewrite system* (xii).

In terms of Gröbner basis theory applied to this special case we consider the *basis* (i) F which consists of a *set of two-term polynomials* (ii) of the form $l - r$ multiplied by ± 1 so that $l > r$. A *monomial* (iii) $m \in X^\dagger$ may be *reduced* (iv) with respect to F if it contains the *leading monomial* (v) l of a polynomial $l - r$ as a *submonomial* (vi) i.e. if $m = ulv$ for some $u, v \in X^*$. To reduce $m = ulv$ using the polynomial $l - r$ we replace l by the *remainder* (vii) r of the polynomial, and write $ulv \rightarrow_F urv$.
The Buchberger algorithm looks for *matches between polynomials* (viii). Given a pair of polynomials $l_1 - r_1$, $l_2 - r_2$ there are four possible ways in which an match can occur: $l_1 = u_2l_2v_2$, $u_1l_1v_1 = l_2$, $l_1v_1 = u_2l_2$ and $u_1l_1 = l_2v_2$. The *S-polynomial* (xi) resulting from a match is the difference between the pair of monomials resulting from applying each two-term polynomial to the smallest monomial on which the match occurs. The S-polynomials resulting from each of the four matches are: $r_1 - u_2r_2v_2$, $u_1r_1 - v_1, r_2$, $r_1v_1 - u_2r_2$ and $u_1r_1 - r_2v_2$ respectively (see diagram).
In one pass the completion procedure finds all the S-polynomials resulting from matches of polynomials of F. The S-polynomials are reduced as far as possible with respect to F to obtain a *reduced S-polynomial* (x) $c_1 - c_2$. Note that reduction can only replace one term with another so the reduced S-ploynomial will have two terms unless the two terms reduce to the same thing $c_1 = c_2$ in which case the original S-polynomial is said to *reduce to zero* (xi). The reduced S-polynomials that have not been reduced to zero are multiplied by ± 1, so that $c_1 > c_2$, and added to F forming F_1. The procedure is then repeated for the basis F_1, to obtain F_2 and so on. When all the S-polynomials of a basis F_n reduce to zero (i.e. $F_{n+1} = F_n$) then F_n is a *Gröbner basis* (xii).

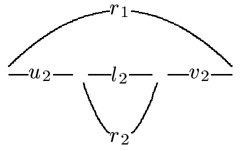
A critical pair in R will occur if and only if a corresponding S-polynomial occurs in F. Reduction of the pair by R is equivalent to reduction of the S-polynomial by F. Therefore at any stage any new rules correspond to the new two-term polynomials and $F_i := \{l - r : (l, r) \in R_i\}$. Therefore the completion procedures as applied to R and F correspond to each other at every step. □

The following pictures illustrate the four cases in which overlaps / matches and critical pairs / S-polynomials arise, showing their correspondence, as described in the proof above.

possible overlaps
of rules
$l_1 \to r_1$ and $l_2 \to r_2$

possible matches
of polynomials
$l_1 - r_1$ and $l_2 - r_2$

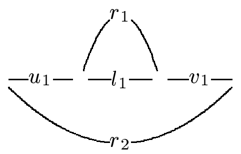
$l_1 = u_2l_2v_2$
$(r_1, u_2r_2v_2)$



$l_1 = u_2l_2v_2$
$u_2r_2v_2 - r_1$

$l_2 = u_1 l_1 v_1$

$(u_1 r_1 v_1, r_2)$

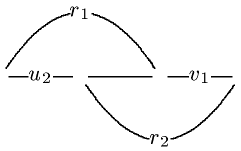


$l_2 = u_1 l_1 v_1$

$r_2 - u_1 r_1 v_1$

$l_1 v_1 = u_2 l_2$

$(r_1 v_1, u_2 r_2)$

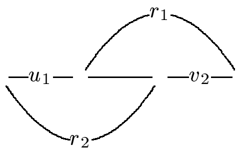


$l_1 v_1 = u_2 l_2$

$u_2 r_2 - r_1 v_1$

$u_1 l_1 = l_2 v_2$

$(u_1 r_1, r_2 v_2)$



$u_1 l_1 = l_2 v_2$

$r_2 v_2 - u_1 v_1$

Remark 3.2.3 The main conclusion to be drawn from this result is that there is no need for special Knuth-Bendix programs: the noncommutative Buchberger algorithm applied to a rewrite system (set of two-term polynomials with coefficients 1 and -1) is the Knuth-Bendix algorithm. All of the work in Chapter Two is in fact about the application of a special case of the Gröbner basis procedure; even Kan extensions are calculated using Gröbner bases.

Corollary 3.2.4 *Abelian semigroups have complete rewriting systems.*

Proof It is known that Buchberger's algorithm always terminates in the commutative case, and so it follows that presentations of abelian semigroups will have complete rewriting systems, which can be found by using the commutative Buchberger algorithm. □

3.3 One-sided Ideals

In this section we describe Gröbner basis theory for one-sided ideals in noncommutative polynomial algebras. The first result shows how to use the standard noncommutative Buchberger algorithm to compute a Gröbner basis for a one-sided ideal. Then we make explicit the correlation between the Gröbner basis theory for one-sided ideals and standard one-sided rewriting systems.

Let K be a field and let S be a semigroup. Let $F = \{f_1, \dots, f_n\}$ be a subset of polynomials (a basis for an ideal) in $K[S]$. We will assume that the f_i are all monic. Let $\langle F\rangle^r$ denote the right ideal generated in $K[S]$ by F i.e.

$$\langle F\rangle^r := \{f_1q_1 + \cdots + f_nq_n : q_1, \dots, q_n \in K[S]\}.$$

A **right congruence** on an algebra A is an equivalence relation $=^r$ such that for all $q \in A$

$$f =^r h \Rightarrow f + q =^r h + q \text{ and } fq =^r hq.$$

Lemma 3.3.1 *Let $K[S]$ be the free K-algebra on S. Let $F = \{f_1, \dots, f_n\}$ be a subset of $K[S]$. Then $\overset{*}{\leftrightarrow}{}^{r}_{F}$ defines a right congruence on $K[S]$ where*

$$f \rightarrow^{r}_{F} f - kf_iv$$

if $\mathtt{LM}(f_i)v$ *occurs in f with coefficient k for $f_i \in F$, $v \in S$ and $k \in K$. Furthermore*

$$f \overset{*}{\leftrightarrow}{}^{r}_{F} h \Leftrightarrow f - h \in \langle F \rangle^{r}.$$

Proof Suppose that $f \overset{*}{\leftrightarrow}{}^{r}_{F} h$ in n steps. The hypothesis is that $f =^{r}_{F} h$ and the proof is by induction on n. For the base step set $n = 0$. Then $f = h$ so $f - h = 0$ is in the ideal i.e. $f =^{r}_{F} h$. Assume the hypothesis is true for $n-1$ and suppose $f \overset{*}{\leftrightarrow}{}^{r}_{F} h$ in n steps. There exists f' such that $f \overset{*}{\leftrightarrow}{}^{r}_{F} f'$ in $n-1$ steps and $f' \leftrightarrow^{r}_{F} h$. By the induction hypothesis $f - f'$ is in the ideal Now either $f' \rightarrow h$ and $h = f' - k_if_iv_i$ or $h \rightarrow f'$ and $f' = h - k_if_iv_i$ for some $f_i \in F, k_i \in K$ and $u_i \in S$. So $h = f' - k_if_iv_i$ or $f' = h - k_if_iv_i$ which means that $f - h = f - f' \pm k_if_iv_i$ which is clearly in the ideal. So $f =^{r}_{F} h$.

Conversely, suppose that $f =^{r}_{F} h$. Then we can write $f - h = k_1f_1v_1 + \cdots + k_nf_nv_n$ for some $f_i \in F, k_i \in K$ and $u_i \in S$. The hypothesis is that $f \overset{*}{\leftrightarrow}{}^{r}_{F} h$ and the proof is by induction on n. For the base step put $n = 0$. Then $f = h$ so $f \overset{*}{\leftrightarrow}{}^{r}_{F} h$ by reflexivity. For the induction step assume the hypothesis holds for $n-1$ and consider $f - h = k_1f_1v_1 + \cdots + k_nf_nv_n$. By the induction hypothesis $f \overset{*}{\leftrightarrow}{}^{r}_{F} h$. There are three cases to consider.
In the first case $\mathtt{LT}(f_n)$ does not occur in h and so $h + k_nf_nv_n \rightarrow^{r}_{F} h$.
In the second case $\mathtt{LT}(f_n)$ does not occur in f and so $f - k_nf_nv_n \rightarrow^{r}_{F} f$ and since $f - k_nf_nv_n = h + k_1f_1v_1 + \cdots + k_{n-1}f_{n-1}v_{n-1}$ we have $f - k_nf_nv_n \overset{*}{\leftrightarrow}{}^{r}_{F} h$ by the induction hypothesis.
In the third case let $c_1 \neq 0$ be the coefficient of $\mathtt{LM}(f_n)$ in $h + k_nf_nv_n$ and let $c_2 \neq 0$ be the coefficient of $\mathtt{LM}(f_n)$ in h. Then

$$h + k_nf_nv_n \rightarrow^{r}_{f_n} h + k_nf_nv_n - c_1f_nv_n = h - (c_1 - k_n)f_nv_n,$$

eliminating the occurance of $\mathtt{LM}(f_n)$ in $h + k_nf_nv_n$. Now $c_2 = c_1 - k_n$ so $h \rightarrow^{r}_{f_n} h - (c_1 - k_n)f_nv_n$ so $h + k_nf_nv_n$ and h are joinable. □

We introduce a tagging notation which will allow the use of the two- sided Buchberger algorithm to compute a Gröbner basis which will allow us to solve the ideal membership problem for $\langle F \rangle^{r}$.

Definition 3.3.2 *Let $K[S]$ be the free K-algebra on a semigroup S. Let $\dashv$ be a symbol. Let* $\mathtt{tag} : K[S] \rightarrow K[\{\dashv\} \sqcup S]$ *be the morphism induced by* $\mathtt{tag}(m) := \dashv m$ *for all $m \in S$. So for $f = k_1m_1 + \cdots + k_nm_n$ where $k_1, \dots, k_n \in K$ and $m_1, \dots, m_n \in S$, $\dashv f := k_1 \dashv m_1 + \cdots + k_n \dashv m_n \in K[\dashv S]$. Therefore* $\mathtt{tag}$ *is well-defined. The inverse function* $\mathtt{tag}^{-1}$*, removing the tag, is similarly well- defined.*

We may refer to tagged and untagged polynomials. Let S be a semigroup given by a presentation $\langle X|R \rangle$. Let $\bar{F}$ be a set of polynomials $\{\bar{f_1}, \dots \bar{f_m}\}$, a basis for the one-sided ideal $\langle \bar{F} \rangle^{r}$ in $K[S]$. Define $H := \{l_i - r_i | (l_i, r_i) \in R\}$. Define a section σ of the factor morphism $\theta : K[X^{\dagger}] \rightarrow K[S]$, denoted by $\sigma(\bar{f_i}) := f_i$ and let $F := \sigma(\bar{F})$. Let $>$ be an admissible well-ordering on $X^{\dagger}$.

Definition 3.3.3 *Define the reduction relation $\rightarrow_{\dashv F \sqcup H}$ on $K[\dashv X^{\dagger}]$ by*

$$\dashv f \rightarrow_{\dashv F \sqcup H} \dashv f - k \dashv f_iv$$

whenever $\mathtt{LM}(f_i)v$ *occurs in f with coefficient k for $f \in K[X^{\dagger}]$, $v \in S$, $f_i \in F$ and by*

$$\dashv f \rightarrow_{\dashv F \sqcup H} \dashv f - k \dashv uh_iv$$

whenever $u\mathtt{LM}(h_i)v$ *occurs in f with coefficient k for $f \in K[X^{\dagger}]$, $u, v \in S$, $h_i \in H$.*

This corresponds to the function `ReducePoly` in the program. The reflexive, symmetric, transitive closure will be denoted $\overset{*}{\leftrightarrow}_{\dashv F \sqcup H}$.
Note how the reduction of f requires that we find a monomial of $\dashv f$ that is some multiple of a leading monomial from $\dashv F$ or H. This definition of reduction will allow the application of the standard Buchberger algorithm to $\dashv F \sqcup H$ to attempt to compute a Gröbner basis for the one-sided ideal $\langle \bar{F} \rangle^r$ in $K[S]$. First we require the following results.

Proposition 3.3.4 *The relation $\rightarrow_{\dashv F \sqcup H}$ is Noetherian on $K[\dashv X^\dagger]$. It is complete if and only if it is also locally confluent.*

Proof Any Noetherian reduction relation is complete if and only if it is locally confluent (see Chapter One). Suppose there exists an infinite sequence of reductions $p_1 \rightarrow_{\dashv F \sqcup H} p_2 \rightarrow_{\dashv F \sqcup H} \cdots$ of polynomials $p_1, p_2, \ldots \in K[\dashv X^\dagger]$. This implies the existence of an infinite sequence $m_1 > m_2 > \cdots$ of monomials $m_1, m_2, \ldots \in X^\dagger$ because the definition of reduction replaces one term with terms which are smaller with respect to $>$. The ordereing $>$ is Noetherian, therefore the sequence cannot exist, proving that $\rightarrow_{\dashv F \sqcup H}$ is Noetherian. □

Theorem 3.3.5 (Simulation of Right Reduction in a Monoid Ring)

$$\frac{K[S]}{=^r_{\bar{F}}} \cong \frac{K[\dashv X^\dagger]}{\overset{*}{\leftrightarrow}_{\dashv F \sqcup H}}$$

Proof Let θ represent the quotient morphism $X^\dagger \rightarrow S$. Extend θ to $\theta' : K[X^\dagger] \rightarrow K[S]$.
Define $\phi : K[\dashv X^\dagger] \rightarrow K[S]$ by $\phi(\dashv f) := \theta'(f)$. Define $\phi' : K[\dashv X^\dagger]/ \overset{*}{\leftrightarrow}^r_{\dashv F \sqcup H} \rightarrow K[S]/ =^r_{\bar{F}}$ by $\phi'[\dashv f]_{\dashv F \sqcup H} := [\phi(\dashv f)]^r_{\bar{F}}$. We require to prove that ϕ' is well-defined, i.e. that ϕ preserves the congruence classes of $\overset{*}{\leftrightarrow}^r_{\dashv F \sqcup H}$ on $K[\dashv X^\dagger]$. It is sufficient to prove that $\phi(\dashv f) =^r_{\bar{F}} \phi(\dashv f - k \dashv f_i v)$ and $\phi(\dashv f) =^r_{\bar{F}} \phi(\dashv f - k \dashv u h_i v$ whenever either $f_i v$ occurs in f with coefficient k for some $u, v \in X^\dagger$, $f_i \in F$, $h_i \in H$.

Now $\phi(\dashv f - k \dashv f_i v) = \theta'(f) - \theta'(k f_i v)$. Recall that $f_i = \sigma(\bar{f}_i)$ where σ is the extension of a section of θ. It follows that $k \bar{f}_i \theta(v) \in \langle \bar{F} \rangle$ therefore $\theta'(f) =^r_{\bar{F}} \theta'(f) - k \bar{f}_i \theta(v)$ as required. For the other case let $h_i \in H$, $f \in K[X^\dagger]$. Then $\phi(\dashv f - k \dashv u h_i v) = \theta'(f) - \theta'(k u h_i v)$. Recall the definitions of θ and H so $\theta'(k u h_i v) = 0$ and so $\theta'(f) =^r_{\bar{F}} \theta'(f) - \theta'(k u h_i v)$. Hence ϕ is well-defined.

We now prove that ϕ' is surjective. Let $\bar{f} \in K[S]$. Then the extension of σ uniquely defines $f \in K[X^\dagger]$ and $\theta'(f) = \bar{f}$. Thus we have $\dashv f \in K[\dashv X^\dagger]$ such that $\phi'[\dashv f]_{\dashv F \sqcup H} = [\bar{f}]^r_{\bar{F}}$.

Finally, we prove that ϕ' is injective. Let $f, h \in K[X^\dagger]$ such that $\phi'[\dashv f]_{\dashv F \sqcup H} = \phi'[\dashv h]_{\dashv F \sqcup H}$. Then immediately $[\theta'(f)]^r_{\bar{F}} = [\theta'(h)]^r_{\bar{F}}$. Therefore $\bar{f} - \bar{h}$ is a member of the right ideal generated by $\bar{F}$. It can be verified, using the definitions, that this implies that $f \overset{*}{\leftrightarrow}_{\dashv F \sqcup H} h$. Therefore ϕ' is a bijection. □

Recall that a pair of monomials is considered to have a match if there is some overlap between them.

Lemma 3.3.6 *Let $\rightarrow$ be a reduction relation on $K[\dashv X^\dagger]$. Let $\dashv p$ be a polynomial which reduces in two possible ways $\dashv p \rightarrow \dashv p_1$ and $\dashv p \rightarrow \dashv p_2$. If the S-polynomial $\dashv p_1 - \dashv p_2$ reduces to zero then there exists a polynomial $\dashv q$ such that $\dashv p_1 \overset{*}{\rightarrow} \dashv q$ and $\dashv p_2 \rightarrow \dashv q$.*

Proof Let $p, p_1, p_2 \in K[X^\dagger]$ such that $\dashv p \rightarrow \dashv p_1$ and $\dashv p \rightarrow \dashv p_2$ and $\dashv p_1 - \dashv p_2 \overset{*}{\rightarrow} 0$. then there exist $\zeta_1, \ldots, \zeta_t$ with $\zeta_i = k_i h_i v_i$ or $\zeta_i = k_i u_i f_i v_i$ for some $u_i, v_i \in X^*$, $k_i \in K$, $f_i \in F$, $h_i \in \sigma(H)$ such that

$\dashv p_1 - \dashv p_2 \to \dashv p_1 - \dashv p_2 - \dashv \zeta_1 \to \cdots \to \dashv p_1 - \dashv p_2 - \dashv \zeta_1 - \cdots - \dashv \zeta_t = 0$. Consider the first reduction. The leading monomial of ζ_1 occurs in p_1 with coefficient $k_{1,1}$ and in p_2 with coefficient $k_{2,1}$ for $k_{1,1}, k_{2,1} \in K$ such that $k_{1,1} + k_{2,1} = 1$. Therefore $\dashv p_1 \to \dashv p_1 - k_{1,1} \dashv \zeta_1$ and $\dashv p_2 \to \dashv p_2 + k_{2,1} \dashv \zeta_1$. Repeating this procedure for $\zeta_2, \dots, \zeta_t$ we obtain $\dashv p_1 \overset{*}{\to} \dashv p_1 - k_{1,1} \dashv \zeta_1 - \cdots - k_{1,t} \dashv \zeta_t$ and $\dashv p_2 \overset{*}{\to} \dashv p_2 + k_{2,1} \dashv \zeta_1 + \cdots + k_{2,t} \dashv \zeta_t$. Now $(\dashv p_1 - k_{1,1} \dashv \zeta_1 - \cdots - k_{1,t} \dashv \zeta_t) - (\dashv p_2 + k_{2,1} \dashv \zeta_1 + \cdots + k_{2,t} \dashv \zeta_t) = 0$. Therefore $\dashv p_1$ and $\dashv p_2$ are reduced to the same term. □

Lemma 3.3.7 *The reduction relation $\to$ generated by $\dashv F \sqcup H$ is confluent on $K[\dashv X^\dagger]$ if and only if all S-polynomials resulting from matches of $\dashv F \sqcup H$ reduce to zero by $\to$.*

Proof Let all S-polynomials resulting from matches of $\dashv F \sqcup H$ reduce to zero by $\to$. Let $\dashv p$ be a critical term of $(K[\dashv X^\dagger], \to)$. If the reductions apply to different terms of $\dashv p$ or to disjoint parts of the same term then it is clear that the S-polynomial will reduce to zero immediately (by applying the same two reductions again). If the reductions apply to the same term of $\dashv p$ and are not disjoint then there are three possibilities.

For the first case both rules come from $\dashv F$. So let $\dashv p \to \dashv p - k \dashv f_i v_i$ and $\dashv p \to \dashv p - k \dashv f_j v_j$ for some $f_i, f_j \in F$, $v_i, v_j \in X^*$ such that $l_i v_i = l_j v_j$ are monomials of p with coefficient k where $l_i := \mathtt{LM}(f_i)$ and $l_j := \mathtt{LM}(f_j)$. Then there is an overlap such that (without loss of generality) $\dashv l_i v = \dashv l_j$ for some $v \in X^*$. The S-polynomial resulting from this overlap is $\dashv r_j - \dashv r_i v$ where $r_i := \mathtt{rem}(f_i)$ and $r_j := \mathtt{rem}(f_j)$. Now $\dashv r_j - \dashv r_i v = \dashv f_i v - \dashv f_j \to 0$, therefore $(\dashv p - k \dashv f_i v_i) - (\dashv p - k \dashv f_j v_j) \to 0$. Therefore by Lemma 3.3.6 there exists $\dashv q \in K[\dashv X^\dagger]$ such that $\dashv p - k \dashv f_i v_i \overset{*}{\to} \dashv q$ and $\dashv p - k \dashv f_j v_j \overset{*}{\to} \dashv q$. So the critical pair $(\dashv p - k \dashv h_i v_i, \dashv p - k \dashv h_j v_j)$ resolves.

For the second case one rule is from $\dashv F$ and one is from H. So let $\dashv p \to \dashv p - k \dashv f_i v_i$ and $\dashv p \to \dashv p - k \dashv u_j h_j v_j$ for some $f_i \in F$, $h_j \in H$, $v_i, u_j, v_j \in X^*$ such that $l_i v_i = u_j l_j v_j$ are monomials of p with coefficient k where $l_i := \mathtt{LM}(f_i)$ and $l_j := \mathtt{LM}(h_j)$. Then there are two possibilities for the overlap.
For the first $\dashv l_i v = \dashv u l_j$ for some $u, v \in X^*$. The S-polynomial resulting from this overlap is $\dashv u r_j - \dashv r_i v$ where $r_i := \mathtt{rem}(f_i)$ and $r_j := \mathtt{rem}(h_j)$. Now $\dashv u r_j - \dashv r_i v = \dashv f_i v - \dashv u h_j \to 0$, therefore $(\dashv p - k \dashv f_i v_i) - (\dashv p - k \dashv u_j h_j v_j) \to 0$.
For the second $\dashv l_i = \dashv u l_j v$ for some $u, v \in X^*$. The S-polynomial resulting from this overlap is $\dashv u r_j v - \dashv r_i$ where $r_i := \mathtt{rem}(f_i)$ and $r_j := \mathtt{rem}(h_j)$. Now $\dashv u r_j v - \dashv r_i = \dashv f_i - \dashv u h_j v \to 0$, therefore $(\dashv p - k \dashv f_i v_i) - (\dashv p - k \dashv u_j h_j v_j) \to 0$.
In either case by Lemma 3.3.6 there exists $\dashv q \in K[\dashv X^\dagger]$ such that $\dashv p - k \dashv f_i v_i \overset{*}{\to} \dashv q$ and $\dashv p - k \dashv u_j h_j v_j \overset{*}{\to} \dashv q$. So the critical pair $(\dashv p - k \dashv f_i v_i, \dashv p - k \dashv u_j h_j v_j)$ resolves.

For the final case, both rules come from H. So let $\dashv p \to \dashv p - k \dashv u_i h_i v_i$ and $\dashv p \to \dashv p - k \dashv u_j h_j v_j$ for some $h_i, h_j \in H$, $u_i, v_i, u_j, v_j \in X^*$ such that $u_i l_i v_i = u_j l_j v_j$ are monomials of p with coefficient k where $l_i := \mathtt{LM}(h_i)$ and $l_j := \mathtt{LM}(h_j)$. Then there are two possibilities for the overlap.
For the first $l_i v = u l_j$ for some $u, v \in X^*$. The S-polynomial resulting from this overlap is $u r_j - r_i v$ where $r_i := \mathtt{rem}(h_i)$ and $r_j := \mathtt{rem}(f_j)$. Now $u r_j - r_i v = h_i v - u h_j \to 0$, therefore $(\dashv p - k \dashv u_i h_i v_i) - (\dashv p - k \dashv u_j h_j v_j) \to 0$.
For the second $l_i = u l_j v$ for some $u, v \in X^*$. The S-polynomial resulting from this overlap is $u r_j v - r_i$ where $r_i := \mathtt{rem}(h_i)$ and $r_j := \mathtt{rem}(h_j)$. Now $u r_j v - r_i = h_i - u h_j v \to 0$, therefore $(\dashv p - k \dashv u_i h_i v_i) - (\dashv p - k \dashv u_j h_j v_j) \to 0$.
In either case by Lemma 3.3.6 there exists $\dashv q \in K[\dashv X^\dagger]$ such that $\dashv p - k \dashv u_i f_i v_i \overset{*}{\to} \dashv q$ and $\dashv p - k \dashv u_j f_j v_j \overset{*}{\to} \dashv q$. So the critical pair $(\dashv p - k \dashv u_i f_i v_i, \dashv p - k \dashv u_j f_j v_j)$ resolves.

This proves that however the critical pair arises, it is a consequence of some match between polynomials and can be resolved. Therefore $\to$ is confluent.

The converse is easily checked. Suppose that $\to$ is confluent. Then any S-polynomial arising from a match between polynomials is the result of reducing one monomial in two different ways i.e. $\dashv p \to \dashv p_1$ and $\dashv p_2$ for some $p, p_1, p_2 \in K[X^\dagger]$. The S-polynomial is equal to $\dashv p_1 - \dashv p_2$. The relation $\to$ is locally confluent and so there exists $\dashv q \in K[\dashv X^\dagger]$ such that $\dashv p_1 \to \dashv q$ and $\dashv p_2 \to \dashv q$. Therefore $\dashv p_1 - \dashv p_2 \xrightarrow{*} \dashv q - \dashv q = 0$ as required. □

We have now proved the following theorem.

Theorem 3.3.8 *The Buchberger algorithm may be applied directly to a set containing tagged polynomials and non-tagged (two-term) polynomials to attempt to compute a Gröbner basis for a one-sided ideal in a free algebra on a finitely presented semigroup.*

This widens the scope of the Gröbner basis program `grobner.g` without modifying it. The program can now attempt to compute bases for one sided ideals.

3.3.1 Gröbner Bases for Coset Systems

A tagged Gröbner basis corresponds to a tagged complete rewrite system in that special case (two-term polynomials and no rewriting tags).

Lemma 3.3.9 *Let G be a group and K be a field. Let $F := \{f_1, \dots, f_n\} \subseteq K[G]$ where the polynomials in F each have only two terms, the larger of which has coefficient 1, the other having coefficient -1. Then the right ideal of F defines a subgroup of G.*

Proof Define H to be the set of elements m in G such that $m =^{r}{}_{F} id$.

$$H := \{m \mid m \in G \text{ and } m - id \in \langle F \rangle^r\}$$

We prove that H is a subgroup of G. Firstly, $H \subseteq G$, so composition is associative.
Let $m_1, m_2 \in H$, then $m_1 - id, m_2 - id \in \langle F \rangle^r$ by definition, and $m_1 m_2 - id = m_1(m_2 - id) + (m_1 - id)$. Therefore $m_1 m_2 \in H$, so H is closed under multiplication. Clearly $id - id = 0 \in \langle F \rangle^r$ because $0(f_i) \in \langle F \rangle^r$ for $f_i \in F$ since $0 \in K$, so $id \in H$. Finally, for any $m \in H$, we have $m - id \in \langle F \rangle^r$ so $m^{-1} - id = -m^{-1}(m - id) \in \langle F \rangle^r$, so $m^{-1} \in H$.
Therefore we have shown that $\langle F \rangle^r$ defines a subgroup of G. □

Corollary 3.3.10 *A complete right coset rewriting system for the finitely generated subgroup H of a finitely presented group G may be computed by finding a Gröbner basis for a particular right ideal over a particular algebra, when the Buchberger algorithm terminates.*

We will now apply the procedures that have been described to calculating some relations in finitely presented semigroups.

3.3.2 Example: Computing Green's Relations for Semigroups

Semigroups are often described using Green's relations, specifying their L-classes R-classes, D-classes and H-classes. Eggbox diagrams depict the partitions of a semigroup into these classes. We can determine the classes by using Gröbner bases applied directly to the presentation. The examples show that there is also the possibility of dealing with infinite semigroups having infinitely many H-classes, L-classes or

R-classes. First we recall some definitions [45].

A nonempty subset A of a semigroup S is a **right ideal** of S if $AS \subseteq A$. It is a **left ideal** of S if $SA \subseteq A$. If x is an element of S then the smallest right ideal of S containing x is $xS \cup \{x\}$, we denote this $\langle x\rangle^r$ as it is called the **right ideal generated by** x. Similarly the **left ideal generated by** x is $Sx \cup \{x\}$ and is denoted $\langle x\rangle^l$.

Green's Relations
Let S be a semigroup and let s and t be elements of S. We say that s and t are **L-related** if the left ideal generated by s in S is equal to the left ideal generated by t:

$$s \sim_L t \Leftrightarrow \langle s\rangle^l = \langle t\rangle^l.$$

Similarly they are **R-related** if the right ideals are the same:

$$s \sim_R t \Leftrightarrow \langle s\rangle^r = \langle t\rangle^r.$$

The L-relation is a right congruence on S and the R-relation is a left congruence on S. (The right action of S on itself is preserved by the mapping to the L-classes - so $[x^y]_{\sim_L} = [xy]_{\sim_L} = [x]^y{}_{\sim_L}$, similarly for the left action and R-classes.) The elements s and t are said to be **H-related** if they are *both* L-related *and* R-related, and are **D-related** if they are *either* L-related *or* R-related.

To determine whether s and t are R (or L)-related we can compute the appropriate Gröbner bases and compare them. First let K be (any) field. Let S have presentation $sgp\langle X|Rel\rangle$ Let F be a Gröbner basis for $K[S]$ – so $K[X^\dagger]/=_F$ is isomorphic to $K[S]$. We would add the polynomial $\dashv s$ to the Gröbner basis system for $K[S]$ and compute the Gröbner basis, and see whether this was equivalent to the basis obtained for $\dashv t$.

Example 3.3.11 (Symmetric Monoid)
The following example is for the finite semigroup $Sym(2)$ with monoid presentation

$$mon\langle e, s, id | e^2 = e, s^2 = id, sese = ese, eses = ese\rangle.$$

The Gröbner basis equivalent to the rewrite system is

$$H := \{e^2 - e,\ s^2 - id,\ eses - ese,\ sese - ese\}.$$

The elements are $\{id, e, s, es, se, ese, ses\}$. We calculate Gröbner bases for the right and left ideals for each of the elements. The results are displayed in the table below. In detail a Gröbner basis for $\langle ses\rangle^r$ in $K[S]$ is calculated in $K[\dashv X^*]$ by adding $\dashv ses$ to the set of polynomials H. A match $\dashv sess$ between $s^2 - id$ and $\dashv ses$. This results in the S-polynomial $\dashv se(id) - (0)s$ which simplifies to $\dashv se$. Another match $\dashv seses$ occurs between $eses - ese$ and $\dashv ses$. This results in the S-polynomial $\dashv s(ese) - (0)es$ which reduces to $\dashv ese$. Any further matches result in S-polynomials which reduce to zero. The polynomials we add to H to obtain a Gröbner basis are $\{\dashv se, \dashv ese\}$ (note that $\dashv ses$ is a multiple of $\dashv se$ so it is not required in the Gröbner basis). The table lists the polynomials which, together with H, will give the Gröbner bases for the right and left ideals generated by single elements.

element	right ideal	left ideal
id	$\dashv id$	$id\vdash$
e	$\dashv e$	$e\vdash$
s	$\dashv id$	$id\vdash$
es	$\dashv e$	$es\vdash, ese\vdash$
se	$\dashv se, \dashv ese$	$e\vdash$
ese	$\dashv ese$	$ese\vdash$
ses	$\dashv se, \dashv ese$	$es\vdash, ese\vdash$

Two elements whose right ideals are generated by the same Gröbner basis have the same right ideal (similarly left), and so it is immediately deducible that the R-classes are $\{id, s\}, \{e, es\}, \{se, ses\}$ and $\{ese\}$, the L-classes are $\{id, s\}, \{e, se\}, \{es, ses\}$ and $\{ese\}$, the H-classes are $\{id, s\}, \{e\}, \{se\}, \{es\}, \{ses\}$ and $\{ese\}$ and the D-classes are $\{id, s\}, \{e, es, se, ses\}$ and $\{ese\}$.

The eggbox diagram is as follows where L classes are columns, R-classes are rows, D classes are diagonal boxes and H classes are the small boxes:

id, s			
	e	es	
	se	ses	
			ese

This example could have been calculated by enumerating the elements of each of the fourteen ideals (which takes longer).

Example 3.3.12 (Bicyclic Monoid)
The next example is the Bicyclic monoid which is infinite and has monoid presentation

$$mon\langle p, q \mid pq = id\rangle.$$

This means that the equivalent Gröbner basis defined on the free monoid algebra $K[\{p, q\}^*]$ is $\{pq - id\}$. We begin the table as before:

element	right ideal	left ideal
id	$\dashv id.$	$id\vdash.$
p	$\dashv id.$	$p\vdash.$
q	$\dashv q.$	$q\vdash.$
p^2	$\dashv id.$	$p^2\vdash.$
qp	$\dashv q.$	$p\vdash.$
q^2	$\dashv q^2.$	$id\vdash.$
$\cdots$	$\cdots$	$\cdots$
q^np^m	$\dashv q^n.$	$p^m\vdash.$

It can be seen that there are infinitely many L-classes and infinitely many R-classes. Representatives for the L-classes are q^* because $q^np^m\vdash\rightarrow^l q^n\vdash$ – using the S-polynomial resulting from $p^n(q^np^m\vdash) \rightarrow^l p^n\vdash$ with $(p^nq^n)p^m\vdash\rightarrow^l p^m\vdash$. Similarly p^* is a set of representatives for the R-classes. All elements are D-related and none of them are H-related. So the eggbox diagram would be an infinitely large box of cells, with one element in each cell, this means that the monoid is *bisimple.*

Example 3.3.13 (Polycyclic Monoid)
Now consider the Polycyclic monoid P_n which has presentation

$$mon\langle x_1, \dots, x_n, y_1, \dots, y_n, o \mid ox_i = x_io = oy_i = y_io = o, x_iy_i = id, x_iy_j = o \text{ for } i, j = 1, \dots, n-1, i \neq j\rangle$$

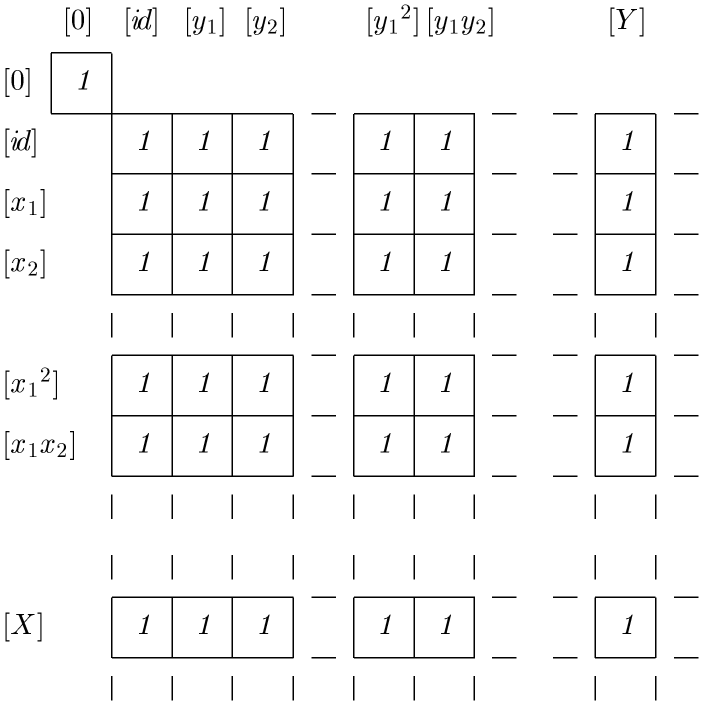
and therefore the Gröbner basis for $K[P_n]$, where K is a field, is

$$\{x_iy_i - id, x_iy_j - 0\text{for } i, j = 1, \dots, n-1,\ i \neq j\}.$$

As might be expected Green's relations for P_n are similar to those for the Bicyclic monoid. The L-classes are represented by sequences of y_i's and the R-classes are represented by sequences of x_i's.

To verify this let $X = x_{i_1} \cdots x_{i_k}$ be a general word in the x_i's, and let Y be $y_{j_1} \cdots y_{j_l}$ a general word in the y_j's. Then we can show that $YX \sim_L X$. To do this we consider the ideals $\langle YX \vdash \rangle$ and $\langle X \vdash \rangle$. To find a Gröbner basis for $\langle YX \vdash \rangle$ consider the match $x_{j_l} \cdots x_{j_1} y_{j_1} \cdots y_{j_l} x_{i_1} \cdots x_{i_k} \vdash$. This results in the S-polynomial $(id) x_{i_1} \cdots x_{i_k} \vdash - x_{j_l} \cdots x_{j_1}(0)$ which simplifies to $x_{i_1} \cdots x_{i_k} \vdash = X \vdash$. This is a Gröbner basis for $\langle YX \vdash \rangle$, and so $\langle YX \vdash \rangle = \langle X \vdash \rangle$. Similarly $\langle \dashv YX \rangle = \langle \dashv Y \rangle$ so $YX \sim_R X$ for any $Y = y_{j_1} \cdots y_{j_l}$, $X = x_{i_1} \cdots x_{i_k}$.

The eggbox diagram is drawn below. As before the L classes are the columns and the R-classes the rows, H-classes are the cells, and there is just one D-class other than the one containing the zero. This proves that the polycyclic monoids are bisimple. The diagram is more conventional than the previous one, as classes are listed but not individual elements, instead the number of elements in each cell is indicated.



	[0]	[id]	[y_1]	[y_2]	[${y_1}^2$]	[y_1y_2]	[Y]
[0]	1						
[id]		1	1	1	1	1	1
[x_1]		1	1	1	1	1	1
[x_2]		1	1	1	1	1	1
[${x_1}^2$]		1	1	1	1	1	1
[x_1x_2]		1	1	1	1	1	1
[X]		1	1	1	1	1	1

This illustrates the fact that Gröbner bases can be used to compute Green's relations for (infinite) semigroups which have finite complete presentations. In particular H-classes have groups called *Schützenberger groups* associated with them. It is known that H-classes in the same D-class have the same Schützenberger group [50].

Previous methods for calculating minimal ideals from presentations of semigroups have been variations on the classical Todd-Coxeter enumeration procedure [24]. The one-sided Gröbner basis methods have limitations in that a complete rewrite system with respect to the chosen order might not be found, but they do give the possibility of calculating the structure of infinite semigroups.

3.4 K-categories

A K-**category** is a category whose hom-sets (a hom-set is the set of all morphisms between a given pair of objects) are K-modules. A morphism of K-categories or K-**functor** F preserves the K-module structure of the hom-sets so $F(a + b) = F(a) + F(b)$, $F(ka) = kF(a)$ for all arrows a, b such that $a + b$ is defined and scalars k in K.

The **free K-category on a graph** Γ is the category whose objects are objects of Γ and whose arrows $\mathrm{Arr}P_K\Gamma$ are all polynomials of the form $p = k_1m_1 + k_2m_2 + \cdots + k_nm_n$ where $k_1, \ldots, k_n \in K$, $m_1, \ldots, m_n \in P\Gamma(A_1, A_2)$ for some $A_1, A_2 \in \mathrm{Ob}\Gamma$. The functions *src* and *tgt* are preserved.

The **relations of a K-category** could be of the form $l = r$ where each side has the same source and target. They can be written $l - r$ and so we assume that the relations are set of polynomials $R \subseteq \mathrm{Arr}P_K\Gamma$. If $R = \{r_1, \ldots, r_n\}$ is such a set of relations on $P_K\Gamma$ then the **congruence generated by** R is defined as follows:

$$f =_R h \text{ if and only if } f =_h +k_1p_1r_1q_1 + \cdots + k_np_nr_nq_n$$

for some $k_1, \ldots, k_n\ inK$, $p_1, \ldots, p_n, q_1, \ldots, q_n \in \mathrm{Arr}P_K\Gamma$ where $src(f) = src(h) = src(p_1) = \cdots = src(p_n)$ and $tgt(f) = tgt(h) = tgt(q_1) = \cdots = tgt(q_n)$ and $p_1r_1q_1, \ldots, p_nr_nq_n$ are defined in $\mathrm{Arr}P_K\Gamma$. The K-category $P_K\Gamma/=_R$ whose elements are the congruence classes of $\mathrm{Arr}P_K\Gamma$ with respect to R is the **factor K-category**.

Definition 3.4.1 *Let K be a field. A **K-category presentation** is a pair $cat_K\langle\Gamma|R\rangle$ where Γ is a graph and $R \subseteq \mathrm{Arr}P_K\Gamma \times \mathrm{Arr}P_K\Gamma$. The K-category it presents is the factor category $P_K\Gamma/=_R$.*

The easiest example is of the free K-category generated by a graph with one vertex and no arrows (except the identity). The arrows of the trivial K-category are simply the elements of K. If the graph now has a set of arrows X from the object to itself, then the arrows of the free K-category are the elements of the noncommutative algebra $K[X]$. It is possible to use Buchberger's algorithm to compute Gröbner bases which enable the specification of the morphisms of a general K-category presented in this way.

Let $>$ be an admissible well-ordering on $\mathrm{Arr}P\Gamma$. Define the **leading monomial** of a polynomial f to be the monomial occurring in f which is the greatest path in Γ with respect to $>$ and denote it $\mathtt{LM}(f)$. Define a **reduction relation $\rightarrow_R$ on** $\mathrm{Arr}P_K\Gamma$ by $f \rightarrow f - k_iu_ir_iv_i$ when $u_i(\mathtt{LM}(r_i))v_i$ occurs in f with coefficient $k_i \in K$ for $u_i, v_i \in \mathrm{Arr}P\Gamma$, $r_i \in R$. If the reduction relation generated by R is complete (i.e. Noetherian and locally confluent), then we say that R is a **Gröbner basis**.

Lemma 3.4.2

$$\frac{\mathrm{Arr}P_K\Gamma}{=_R} \cong \frac{\mathrm{Arr}P_K\Gamma}{\overset{*}{\leftrightarrow}_R}$$

Proof It is clear from the definitions that the equivalence relation $\overset{*}{\leftrightarrow}_R$ is contained in $=_R$. For the converse, suppose $f =_R h$. Then there exist $p_1, \ldots, p_n, q_1, \ldots, q_n \in P_K\Gamma$, such that $f = h + p_1r_1q_1 + \cdots + p_nr_nq_n$. By splitting p_i and q_i into their component terms for $i = 1, \ldots, n$ we obtain $f = h + k_1u_1r_1v_1 + \cdots + k_ju_jr_iv_j + \cdots + k_tu_tr_nv_t$ for some $k_1, \ldots, k_t \in K$, $u_1, \ldots, u_t, v_1, \ldots, v_t \in P\Gamma$. It follows immediately from this that $f \overset{*}{\leftrightarrow}_R h$. □

Proposition 3.4.3 *The relation $\rightarrow_R$ is Noetherian on* $\mathrm{Arr}P_K\Gamma$.

Proof Let $f_1 \rightarrow_R f_2 \rightarrow_R f_3 \rightarrow_R \cdots$ be an infinite reduction sequence. This implies the existence of an infinite sequence of terms $m_1, m_2, m_3, \ldots \in \mathrm{Arr}P\Gamma$ such that $m_1 > m_2 > m_3 > \cdots$. This cannot exist because $>$ is Noetherian on $\mathrm{Arr}P\Gamma$. □

Lemma 3.4.4 *If all S-polynomials resulting from matches of R reduce to zero by $\rightarrow_R$ then $\rightarrow_R$ is locally confluent on* $\mathrm{Arr}P_K\Gamma$.

Proof Let all S-polynomials resulting from matches of R reduce to zero by $\to_R$. We require to prove that $\to_R$ is locally confluent.
Let $f \in \mathrm{Arr}P_K\Gamma$ such that $f \to_R f - k_1u_1r_1v_1$ and $f \to_R f - k_2u_2r_2v_2$ i.e. $k_1u_1r_1v_1 - k_2u_2r_2v_2$ is an S-polynomial.
For the first case the polynomials do not overlap on their leading terms then the critical pair reduces immediately to $p - p_1 - p_2$.
For the second case the polynomials overlap on their leading terms l_1, l_2, here we can assume $k_1 = k_2$ and $u_1l_1v_1 = u_2l_2v_2$ for some $u_1, u_2, v_1, v_2 \in \mathrm{Arr}P\Gamma$. The S-polynomial is $u_2r_2v_2 - u_1r_1v_1$, and it reduces to zero by assumption. The S-polynomial is in fact equal to $u_1f_1v_1 - u_2f_2v_2$. Therefore $p - k_1u_1f_1v_1 - (p - k_2u_2f_2v_2) \overset{*}{\to}_R 0$ and by Lemma 3.3.6 this implies that there exists q such that $p - k_1u_1f_1v_1 \overset{*}{\to}_F q$ and $p - k_2u_2f_2v_2 \overset{*}{\to}_F q$. Hence $\to_R$ is locally confluent and therefore confluent.

For the converse suppose that all critical pairs of $\to_R$ resolve. Then it follows by by the usual argument that all S-polynomials of $\to_F$ reduce to zero by $\to_R$. □

Buchberger's algorithm calculates the S-polynomials of a system R and attempts to reduce them to zero by $\to_R$. If an S-polynomial cannot be reduced it is added to the system. The S-polynomials of the modified system $R'$are then computed – the process looping until a system is found whose S-polynomials can all be reduced to zero.

Theorem 3.4.5 *Buchberger's algorithm, applied to* $(R, >)$ *will return a Gröbner basis for* $=_R$ *on* $\mathrm{Arr}P_K\Gamma$.

Proof All that remains to be verified is that S-polynomials resulting from matches found in R can be added to R without altering $\overset{*}{\leftrightarrow}_R$. We assume all polynomials in R to be monic (possible since K is a field). Now S-polynomials result from two types of overlap.
For the first case let r_1, r_2 be polynomials in R such that $u\mathtt{LM}(r_1) = \mathtt{LM}(r_2)v$ for some $u, v \in \mathrm{Arr}P\Gamma$. Then the S-polynomial is $s := \mathtt{rem}(r_2)v - u\mathtt{rem}(r_1)$ where $\mathtt{rem}(r_i) := r_i - \mathtt{LM}(r_i)$ for $i = 1, 2$. Now $\mathtt{rem}(r_2)v - u\mathtt{rem}(r_1) = ur_1 - r_2v$ therefore $s = \mathtt{rem}(r_2)v - u\mathtt{rem}(r_1) =_R 0$, and hence the congruence generated by $R' := R \cup \{s\}$ coincides with $=_R$.
For the second case let r_1, r_2 be polynomials in R such that $u\mathtt{LM}(r_1)v = \mathtt{LM}(r_2)$ for some $u, v \in \mathrm{Arr}P\Gamma$. Then the S-polynomial is $s := \mathtt{rem}(r_2) - u\mathtt{rem}(r_1)v$. Now $\mathtt{rem}(r_2) - u\mathtt{rem}(r_1)v = ur_1v - r_2$ therefore $s = \mathtt{rem}(r_2) - u(r_1)v =_R 0$, and hence the congruence generated by $R' := R \cup \{s\}$ coincides with $=_R$.
□

Example 3.4.6 The free $\mathbb{Q}$-category generated by the graph below has arrows of the form $k\mathit{id}_{A_1}, k\mathit{id}_{A_2}$ and $k_1a_1 + k_2a_2$ for $k, k_1, k_2 \in \mathbb{Q}$.

$$\begin{tikzcd} A_1 \arrow[r, bend left, "a_1"] \arrow[r, bend right, "a_2"'] & A_2 \end{tikzcd}$$

If we factor this set of arrows by the relation $2a_1 = a_2$ then we have a well-defined $\mathbb{Q}$-category whose morphisms are completely represented by $\{k\mathit{id}_{A_1} \mid k \in \mathbb{Q}\} \cup \{k\mathit{id}_{A_2} \mid k \in \mathbb{Q}\} \cup \{ka_1 \mid k \in \mathbb{Q}\}$.

3.5 Kan Extensions

In the last chapter we showed that a number of combinatorial problems soluble by rewriting methods could be expressed in terms of the problem of computing a particular Kan extension over the category of sets. In this section, we investigate the Gröbner basis analogue to this by expressing the presentation of a non-commutative polynomial algebra as a problem of computing a Kan extension over framed modules

Mods (modules over a fixed field or ring).

As K is traditionally used to represent the field in Gröbner basis calculations, and to differentiate between Kan extensions over Sets, the notation (E, ε) will be used to denote the Kan extension.

Theorem 3.5.1 *Let K be a field. Let A be the trivial K-category generated by the graph with one object A and the identity arrow 1_A, and let B be the K-category with one object B, arrows generated by a set X and polynomial relations P. Let $M : \mathsf{A} \to \mathsf{Mods}$ be the K-functor that maps A to the K-module $K[1]$ and let $F : \mathsf{A} \to \mathsf{B}$ be the K-functor mapping A to B.*
Let $\theta : K[X^\dagger] \to K[X^\dagger]/\langle P \rangle$ be the homomorphism mapping polynomials f of the free algebra to ideals $\langle P \rangle + f$ in the quotient algebra. Let the dimension of the algebra be n and let $\{\theta(m_1), \ldots, \theta(m_n)\}$ be a monomial basis.
Then the Kan extension of M along F is the pair (E, ε), where $E : \mathsf{B} \to \mathsf{Mods}$ is the K-functor defined by $E(B) = K[\theta(m_1), \ldots, \theta(m_n)]$; $E(f)$ is defined by $E(f)(\theta(m_i)) = \theta(m_i)\theta(f)$ and $\varepsilon : M \to EF$ is given by $\varepsilon_A(1) = \theta(id)$.

Proof It is required to verify that E as defined above, is a K-functor, ε is a natural transformation of K-functors, and that for any other such pair (E', ε') there is a unique natural transformation $\alpha : E \to E'$.

First we verify that E is well-defined:
$E(f)(\theta(m_i)) = \theta(m_i)\theta(f) = \theta(m_i f) \in EB$ because $\{\theta(m_1), \ldots, \theta(m_n)\}$ is a basis for $\theta(K[X^*])$.
Also, E is a functor preserving the K multiplication:
$E(f_1 f_2)(\theta(m_i)) = \theta(m_i f_1 f_2) = E(f_2)\theta(m_i f_1) = E(f_2)(E(f_1)\theta(m_i)) = E(f_1) \circ E(f_2)(\theta(m_i))$,
$E(kf)(\theta(m_i)) = \theta(m_i)\theta(fk) = \theta(m_i f)k = k\theta(m_i f) = kE(f)(\theta(m_i))$ and
$E(f_1 + f_2)(\theta(m_i)) = \theta(m_i(f_1 + f_2)) = \theta(m_i f_1 + m_i f_2)) = \theta(m_i f_1) + \theta(m_i f_2)$
$\quad = E(f_1)(\theta(m_i)) + E(f_2)(\theta(m_i))$.

Now we prove that ε is a natural transformation: there is one generating arrow 1_A in A and, for all $k \in K$, we have $\varepsilon_A(M(1_A)(k)) = \varepsilon(k) = k\,id$ and $EF(1_A)(\varepsilon_A(k)) = E1_B(k\,id) = k\,id$.

The universal property follows from the fact that EB is essentially the K-algebra B as a K-module, but we verify the property for completeness. Let (E', ε') be a pair such that E' is a K-functor from $\mathsf{B} \to \mathsf{KMods}$ and ε' is a natural transformation of K-functors.
Any natural transformation of K-functors $\alpha : E \to E'$ such that $\varepsilon \circ \alpha = \varepsilon'$ must satisfy the commutative diagram:

$$\begin{tikzcd}
M(A) \arrow[r, "\varepsilon_A"] \arrow[rr, bend left, "{\varepsilon'}_A"] & EF(A) \arrow[r, "\alpha_{FA}"] \arrow[d, "E(m_i)"'] & E'F(A) \arrow[d, "E'm_i"] \\
 & EF(A) \arrow[r, "\alpha_{FA}"] & E'F(A)
\end{tikzcd}$$

which allows the unique definition $\alpha(m_i) = E'(m_i)(\varepsilon'(1_A))$ for $i = 1, \ldots, n$. Hence (E, ε) is universal. $\Box$

In Gröbner basis computations the set $\{m_1, \ldots, m_n\}$ is the set of irreducible monomials of the algebra with respect to the ideal $\langle P \rangle$, and so by using Gröbner bases to calculate this set we calculate the Kan extension.

3.6 Concluding Remarks

In relating Gröbner bases to rewriting systems we have come as far as expressing the presentation of a noncommutative polynomial algebra in the categorical terms of a Kan extension. It is not claimed that this result is particularly deep or difficult, but it illustrates the possibility of using Gröbner bases to compute different types of Kan extensions. The result proves that the Kan extension can be used to present a K-algebra, and so that there is a kind of Kan extension, (beyond the rewriting ones over sets) to which Gröbner basis methods of computation may be applied.

Expressing a presentation of a K-category as a Kan extension causes more problems. The reason for this is that the K-category B presented may have arrows from different sources leading into one target B. In this case the collection of irreducible monomial arrows with target B (which we might expect to be EB) cannot be a K-module (more like a union of K-modules), as addition across the hom-sets is not defined. Open questions remain, therefore, as to how to express the other algebra presentations in terms of Kan extensions, which would be likely to yield methods for using Gröbner bases to compute a greater range of Kan extensions.

Chapter 4

Reduction and Machines

In the first section automata are considered in the standard way, as acceptors, but applied to the Kan extensions of Chapter 2. We show how to construct automata which accept the unique normal forms of the elements of each set KB for $B \in \mathrm{Ob}\Delta$. Creating accepting automata for such structures is new, and we describe their construction from the complete rewriting systems as well as showing how to apply standard automata theory [41] to obtain a regular expression for the language which is the set of irreducible elements. Further, we extend the ideas to algebras. It appears that some work is being done in this line [60] (monomial acceptors) but it is still appropriate to include it here, to relate the concepts.

In the second section we move on to consider a more useful class of automata – those with output. These machines not restricted to accepting or rejecting strings, but can reduce them into the unique irreducible representative forms. The best known example of this is the use of the Cayley Graph to work out multiplication of group elements. The use of the Cayley Graph as a reduction machine is the first thing to be described. Rewriting systems for Kan extensions can be translated into reduction machines for Kan extensions. These machines are defined as Moore machines. The next consideration is of reduction machines for algebras, which are constructed from the Gröbner bases. I believe this to be a new idea. The construction and operation of the "Gröbner machines" is explained, using a small Hecke Algebra as an example.

The final section introduces a third type of machine: a Petri net. There are many different classes of Petri nets, and we show how to consider the "Gröbner machine" of the previous section as a Petri net. We also show how commutative Gröbner bases may be applied to successfully solve the standard problems posed for reversible Petri nets. This small section speculates on the relation between Petri nets and Gröbner bases and does not prove any results. It is hoped that it provides a starting point for further investigations into the relation between Petri nets and Gröbner bases.

4.1 Normal Forms Acceptors

4.1.1 Definitions and Notation

For a detailed introduction to automata theory refer to [28] or [41]. This section only outlines the essential ideas we use.

A (finite) **deterministic automaton** is a 5-tuple $\underline{A} = (S, \Sigma, s_0, \delta, Q)$ where S is a finite set of *states* (represented by circles), $s_0 \in S$ is the *initial state* (marked with an arrow), Σ is a finite *alphabet*, $\delta : S \times \Sigma \to S$ is the *transition*, $Q \subseteq S$ is the set of **terminal states** (represented by double circles). A deterministic automaton $\underline{A}$ is **complete** if δ is a function, and **incomplete** if it is only a partial function. If $\underline{A}$ is incomplete, then when $\delta(s, a)$ is undefined, the automaton is said to **crash**.

The **extended state transition** δ^* is the extension of δ to Σ^*. It is defined by $\delta^*(s, id) := s$, $\delta^*(s,a) := \delta(s,a)$, $\delta^*(s,aw) := \delta^*(\delta(s,a),w)$ where $s \in S$, $a \in \Sigma$ and w is a string in Σ^*. We are interested in the final state $\delta^*(s_0,w)$ of the machine after a string w has been completely read. If the machine crashes or ends up at a non-terminal state then the string is said to have been **rejected**. If it ends up at a terminal state then we say the string is **accepted**.

A **language** over a given alphabet Σ is a subset L of Σ^*. The set $L(\underline{A})$ of all acceptable strings is the **language accepted by the automaton** $\underline{A}$. A language L is a **recognisable** if it is accepted by some automaton $\underline{A}$. Two automata are **equivalent** if their languages are equal. The **complement** of a complete, deterministic automaton is found by making non-terminal states terminal and vice versa. If the language accepted by an automaton is L, then the language accepted by its complement is $\Sigma^* - L$.

Lemma 4.1.1 ([28]) *Let* $\underline{A} = (S, \Sigma, s_0, \delta, Q)$ *be an incomplete deterministic automaton. Then there exists a complete deterministic automaton* $\underline{A}^{CP}$ *such that* $L(\underline{A}) = L(\underline{A}^{CP})$.

Outline proof Define $\underline{A}^{CP} = (S \sqcup d, \Sigma, s_0, \delta_1, Q)$ where the transition $\delta_1 : S \times \Sigma \to S$ is defined by $\delta_1(s,a) := \delta(s,a)$ if $\delta(s,a)$ is defined, otherwise $\delta_1(s,a) := d$, and $\delta_1(d,a) := d$. □

Diagrammatically this means that automata may be completed by adding one further non-terminal (dump) state d and adding in all the missing arrows so that they point to this state.

A **non-deterministic automaton** is a 5-tuple $\underline{A} = (S, \Sigma, S_0, \delta, Q)$ where S is a finite set of states, $S_0 \subseteq S$ is a set of initial states, Σ is a finite alphabet, $Q \subseteq S$ is the set of terminal states and $\delta : S \times \Sigma \to \mathbb{P}(S)$ is the transition mapping where $\mathbb{P}(S)$ is the power set.

Lemma 4.1.2 ([28]) *Let* $\underline{A} = (S, \Sigma, S_0, \delta_1, Q)$ *be a non-deterministic automaton. Then there exists a deterministic automaton* $\underline{A}^d$ *such that* $L(\underline{A}^d) = L(\underline{A})$.

Outline proof Define $\underline{A}^d := (S^d, \Sigma, {S_0}^d, \delta^d, Q^d)$ where $S^d := \mathbb{P}(S)$ then ${S_0}^d = S_0 \in S^d$, $Q^d := \{U \in \mathbb{P}(S) | U \cap Q \neq \emptyset\}$. Define $\delta^d(U,a) := \bigcup_{u \in U} \delta(u,a)$ for $a \in \Sigma$. It can be verified that $L(\underline{A}^d) = L(\underline{A})$. □

In practice a non-deterministic automaton may be made deterministic by drawing a *transition tree* and then converting the tree into an automaton; for details of this see [28].

Let Σ be a set (alphabet). The following notation is standard when working with languages. The empty word will be denoted id. If $x \in \Sigma^*$ then we will write x for $\{x\}$. If $A, B \in \mathbb{P}\Sigma^*$ then $A + B := A \cup B$, $A - B := A \,/\, B$. Therefore, for example $(x+y)^* + z = \{x,y\}^* \cup \{z\}$.

A **regular expression** over Σ is a string of symbols formed by the rules

i) $a_1 \cdots a_n$ is regular for $a_1, \dots, a_n \in \Sigma$,

ii) $\emptyset$ is regular,

iii) id is regular,

iv) if x and y are regular then xy is regular,

v) if x and y are regular then $x + y$ is regular,

vi) if x is regular then x^* is regular.

A **right linear language equation** over Σ is an expression $X = AX + E$ where $A, X, E \subseteq \Sigma^*$.

Theorem 4.1.3 (Arden's Theorem [28]) *Let $A, X, E \subseteq \Sigma^*$ such that $X = AX + E$ where A and E are known and X is unknown. Then*

i) A^*E is a solution,

ii) if Y is any solution then $A^*E \in Y$,

iii) if $\mathit{id} \notin A$ then A^*E is the unique solution.

Theorem 4.1.4 ([28]) *A system of right linear language equations:*

$$\begin{array}{lclcccccc}
X_0 & = & A_{0,0}X_0 & + & \cdots & + & A_{0,n-1}X_{n-1} & + & E_0, \\
X_1 & = & A_{1,0}X_0 & + & \cdots & + & A_{1,n-1}X_{n-1} & + & E_1, \\
\cdots & & \cdots\cdots & & \cdots & & \cdots\cdots\cdots & & \cdots \\
X_{n-1} & = & A_{n-1,0}X_0 & + & \cdots & + & A_{n-1,n-1}X_{n-1} & + & E_{n-1}.
\end{array}$$

where $A_{i,j}, E_i \in (\Sigma^*)$ and $\mathit{id} \notin A_{i,j}$ for $i, j = 0, \dots, n-1$, has a unique solution.

Outline proof Begin with the last equation. By assumption $\mathit{id} \notin A_{n-1,n-1}$. So by Arden's theorem $X_{n-1} = A^*_{n-1,n-1}(A_{n-1,0}X_0 + \cdots + A_{n-1,n-2}X_{n-2} + E_{n-1})$. Substitute this value for X_{n-1} into the remaining $n-1$ equations and repeat the procedure. Eventually an equation in X_0 only will be obtained which can be solved explicitly. The back-substitution will give explicit values of $X_1, \dots, X_{n-1}$. □

Theorem 4.1.5 ([28]) *Let $\underline{A}$ be a (non)-deterministic automaton. Then $L(\underline{A})$ is regular.*

Outline proof (for the deterministic case)
Let $\underline{A} := (S, \Sigma, s_0, \delta, Q)$ where $S = \{s_0, \dots, s_{n-1}\}$. Define $X_i := \{z \in \Sigma^* : \delta(s_i, z) \in Q\}$ for $i = 0, \dots, n-1$. It is clear that $L(\underline{A}) = X_0$. Define $E_i := \emptyset$ if $s_i \notin Q$ and $E_i := \{\mathit{id}\}$ if $s_i \in Q$ for $i = 0, \dots, n-1$. Define $A_{i,j} := \{a \in \Sigma : \delta(s_i, a) = s_j\}$ for $i, j = 0, \dots, n-1$. Form the following system:

$$\begin{array}{lclcccccc}
X_0 & = & A_{0,0}X_0 & + & \cdots & + & A_{0,n-1}X_{n-1} & + & E_0, \\
X_1 & = & A_{1,0}X_0 & + & \cdots & + & A_{1,n-1}X_{n-1} & + & E_1, \\
\cdots & & \cdots\cdots & & \cdots & & \cdots\cdots\cdots & & \cdots \\
X_{n-1} & = & A_{n-1,0}X_0 & + & \cdots & + & A_{n-1,n-1}X_{n-1} & + & E_{n-1}.
\end{array}$$

This system of n right linear equations in n unknowns satisfies the conditions of the previous theorem and therefore has a unique solution. Moreover, the solution can easily be converted into regular expressions. □

So every non-deterministic automaton gives rise to a system of language equations from whose solutions a description of the language may be obtained.

Theorem 4.1.6 (Kleene's Theorem [28]) *A language L is regular if and only if it is recognisable.*

4.1.2 Acceptors for Kan Extensions

Throughout this section we will use the notation introduced in Chapter Two. Recall that a presentation of a Kan extension (K, ε) is a quintuple $\mathcal{P} := \mathit{kan}\langle \Gamma | \Delta | \mathit{RelB} | X | F \rangle$ where Γ and Δ are graphs, RelB is a set of relations on $\mathsf{P} := P\Delta$, while $X : \Gamma \to \mathsf{Sets}$ and $F : \Gamma \to \mathsf{P}$ are graph morphisms. Elements of the set

$$T := \bigsqcup_{B \in \mathrm{Ob}\Delta} \bigsqcup_{A \in \mathrm{Ob}\Gamma} XA \times \mathsf{P}(FA, B)$$

are written $t = x|b_1 \cdots b_n$ with $x \in XA$, and $b_1, \ldots, b_n \in \mathrm{Arr}\Delta$ are composable with $src(b_1) = FA$. The function $\tau : T \to \mathrm{Ob}\Delta$ is defined by $\tau(x|b_1 \cdots b_n) := tgt(b_n)$ and the action of P on T, written $t \cdot p$ for $t \in T$, $p \in \mathrm{Arr}\mathsf{P}$, is defined when $\tau(t) = src(p)$.

In Chapter Two we defined an initial rewriting system $R_{init} := (R_\varepsilon, R_K)$ on T, and gave a procedure for attempting to complete this system. We will be assuming that the procedure has terminated, returning a complete rewriting system $R = (R_T, R_P)$ on T. In this section automata will be used to find regular expressions for each of the sets KB for $B \in \mathrm{Ob}\Delta$.

Recall that $\sqcup XA$ is the union of the images under X of all the objects of Γ and $\sqcup KB$ is the union of the images under K of all the objects of Δ. In general the automaton for the irreducible terms which are accepted as members of $\sqcup KB$ is the complement of the machine which accepts any string containing undefined compositions of arrows of B, any string not containing a single x_i on the left-most end, and any string containing the left-hand side of a rule. This essentially uses a semigroup presentation of the Kan extension.

Lemma 4.1.7 *Let $\mathcal{P}$ present the Kan extension (K, ε). Then the set $\sqcup KB$ may be identified with the non-zero elements of the semigroup having the presentation with generating set*

$$U := (\sqcup XA) \sqcup \mathrm{Arr}\Delta \sqcup 0$$

and relations

$$\begin{array}{lll}
0u = u0 = 0 & \textit{for all} & u \in U, \\
ux = 0 & \textit{for all} & u \in U,\ x \in \sqcup XA, \\
xb = 0 & \textit{for all} & x \in XA,\ A \in \mathrm{Ob}\Gamma, b \in \mathrm{Arr}\Delta \quad \textit{such that} \quad src(b) \neq FA, \\
b_1 b_2 = 0 & \textit{for all} & b_1, b_2 \in \mathrm{Arr}\Delta \quad \textit{such that} \quad src(b_2) \neq tgt(b_1) \\
x(Fa) = (x \cdot a) & \textit{for all} & x \in XA,\ a \in \mathrm{Arr}\mathsf{A} \quad \textit{such that} \quad src(a) = A, \\
l = r & \textit{for all} & (l, r) \in RelB.
\end{array}$$

Proof The semigroup defined is the set of equivalence classes of T with respect to the second two relations (i.e. the Kan extension rules R_ε and R_K) with a zero adjoined and multiplication of any two classes of T defined to be zero. □

Lemma 4.1.8 *Let $\mathcal{P}$ be a presentation of a Kan extension (K, ε). Then T is a regular language over the alphabet $\Sigma := (\sqcup XA) \sqcup \mathrm{Arr}\Delta$.*

Proof To prove that T is regular over Σ we define an automaton with input alphabet Σ which recognises $T \subseteq \Sigma^*$. Define $\underline{A} := (S, \Sigma, s_0, \delta, Q)$ where $S := \mathrm{Ob}\Delta \sqcup s_0 \sqcup d$, $Q := \mathrm{Ob}\Delta$ and δ is defined as follows:

$$\delta(s_0, u) := \begin{cases} FA & \text{for } u \in XA, A \in \mathrm{Ob}\Gamma \\ d & \text{otherwise.} \end{cases}$$

$$\text{for } B \in \mathrm{Ob}\Delta, \quad \delta(B, u) := \begin{cases} tgt(u) & \text{for } u \in \mathrm{Arr}\Delta, src(u) = B \\ d & \text{otherwise.} \end{cases}$$

$$\delta(d, u) := \quad d \qquad \text{for all } u \in \Sigma.$$

It is clear from the definitions that the extended state transition δ^* is such that $\delta^*(s_o, t) \in \mathrm{Ob}\Delta$ if and only if $t \in T$. Hence $L(\underline{A}) = T$. □

Theorem 4.1.9 *Let $\mathcal{P}$ be a presentation of a Kan extension (K, ε). Let R be a finite rewriting system on T. Then the set of elements $\mathtt{IRR}(\to_R) \subseteq T$ which are irreducible with respect to $\to_R$ is a regular language over the alphabet $\Sigma := \sqcup XA \sqcup \mathrm{Arr}\Delta$.*

Proof We define an incomplete non-deterministic automaton $\underline{A}$ with input alphabet Σ, and language $\Sigma^* - \mathtt{IRR}(\rightarrow_R)$ i.e. that rejects only the irreducible elements of T and accepts all reducible and undefined elements. This is sufficient proof for the theorem, since a language recognised by an incomplete non-deterministic automaton $\underline{A}$ is recognisable and therefore regular. The complement of $\Sigma^* - \mathtt{IRR}(R)$ is $\mathtt{IRR}(R)$ and therefore if $\Sigma^* - \mathtt{IRR}(R)$ is regular then $\mathtt{IRR}(R)$ is regular.

Begin by defining $L(R_T)$ and $L(R_P)$ to be the sets of left hand sides of rules of R_T and R_P respectively. Then define $\mathtt{PL}(R_T)$ and $\mathtt{PL}(R_P)$ to be the sets of all prefixes of elements of $L(R_T)$ and $L(R_P)$ and define $\mathtt{PPL}(R_T)$ and $\mathtt{PPL}(R_P)$ to be the sets of all proper prefixes of elements of $L(R_T)$ and $L(R_P)$. The proper prefixes of a term $x|b_1 \cdots b_n$ are the terms $x|b_1, \dots, x|b_{n-1}$. Note that each x has its own state and we do not require that $x|id$ is a prefix. Similarly the proper prefixes of a path $b_1 \cdots b_n$ are the elements $b_1, \dots b_1 \cdots b_{n-1}$. The difference between proper prefixes and prefixes is that $x|b_1 \cdots b_n$ is considered to be a prefix of itself (but not a proper one), similarly for $b_1 \cdots b_n$. Note $\mathtt{PPL}(R_T) \cup L(R_T) = \mathtt{PL}(R_T)$, similarly for R_P.

Define $\underline{A} := (S, \Sigma, s_0, \delta, Q)$ where $S := s_0 \sqcup (\mathrm{Ob}\Delta \cup (\sqcup XA) \cup \mathtt{PPL}(R_T) \cup \mathtt{PPL}(R_P)) \sqcup D$, $Q := s_0 \sqcup D$. Let $x, b \in \Sigma$ so that $x \in \sqcup XA$ and $b \in \mathrm{Arr}\Delta$. Let $x_1 \in \sqcup XA$, $B \in \mathrm{Ob}\Delta$, $u \in \mathtt{PPL}(R_P)$ and $p \in \mathtt{PPL}(R_P)$. Define the transition $\delta : S \times \Sigma \to \mathbb{P}(S)$ by:

$$\begin{aligned}
\delta(s_0, x) &:= \begin{cases} \{x\} & \text{if } x \notin L(R_T), \\ \{D\} & \text{if } x \in L(R_T), \end{cases} \\
\delta(s_0, b) &:= \{D\}, \\
\delta(y, x) &:= \{D\}, \\
\delta(y, b) &:= \begin{cases} \{x_1|b, \mathit{tgt}(b)\} & \text{if } x_1|b \in \mathtt{PPL}(R_T), \\ \{\mathit{tgt}(b)\} & \text{if } \tau(y) = \mathit{src}(b), y|b \notin \mathtt{PL}(R_T), \\ \{D\} & \text{if } x_1|b \in L(R_T), \\ \{D\} & \text{if } \tau(y) \neq \mathit{src}(b), \end{cases} \\
\delta(B, x) &:= \{D\}, \\
\delta(B, b) &:= \begin{cases} \{b, \mathit{tgt}(b)\} & \text{if } \mathit{src}(b) = B, b \in \mathtt{PPL}(R_P), \\ \{\mathit{tgt}(b)\} & \text{if } \mathit{src}(b) = B, b \notin \mathtt{PL}(R_P), \\ \{D\} & \text{if } \mathit{src}(b) = B, b \in L(R_P), \\ \{D\} & \text{if } \mathit{src}(b) \neq B, \end{cases} \\
\delta(u, x) &:= \{D\}, \\
\delta(u, b) &:= \begin{cases} \{u \cdot b, \mathit{tgt}(b)\} & \text{if } u \cdot b \in \mathtt{PPL}(R_T), \\ \{\mathit{tgt}(b)\} & \text{if } \tau(u) = \mathit{src}(b), u \cdot b \notin \mathtt{PL}(R_T), \\ \{D\} & \text{if } u \cdot b \in L(R_T), \\ \{D\} & \text{if } \tau(u) \neq \mathit{src}(b), \end{cases} \\
\delta(p, x) &:= \{D\}, \\
\delta(p, b) &:= \begin{cases} \{pb, \mathit{tgt}(b)\} & \text{if } pb \in \mathtt{PPL}(R_P), \\ \{\mathit{tgt}(b)\} & \text{if } \mathit{tgt}(p) = \mathit{src}(b), pb \notin \mathtt{PL}(R_P), \\ \{D\} & \text{if } pb \in L(R_P), \\ \{D\} & \text{if } \mathit{tgt}(p) \neq \mathit{src}(b), \end{cases} \\
\delta(D, x) &:= \{D\}, \\
\delta(D, b) &:= \{D\}.
\end{aligned}$$

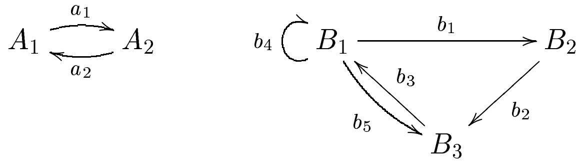
It follows from these definitions that the extended state transition function δ^* is such that $\delta^*(s_0, t) \cap Q \neq \emptyset$ if and only if t is in $\Sigma^* - T$ or if some part of t is the left-hand side of a rule of R (i.e. if t is reducible). Therefore $\Sigma^* - \mathtt{IRR}(R)$ is regular, hence $\mathtt{IRR}(R)$ is regular. □

Corollary 4.1.10 *Let R be a finite complete rewriting system for a Kan extension (K, ε). Then regular expressions for the sets KB of the extended action K can be calculated.*

Outline proof This follows from the preceding results. The automaton $\underline{A}$ of the theorem can be constructed using the specifications in the proof. By the results quoted in the introduction to this chapter a complete deterministic automaton that recognises the same language can be defined. The complement of this has a language that can be identified with $\sqcup KB$. Language equations for this automaton can be written down and Arden's theorem may be applied to find a solution, which gives the language of the automaton as a regular expression. $\square$

The following example illustrates the calculations outlined above.

Example 4.1.11 We construct simple automata which accept the terms which represent elements of some set KB for $B \in \mathsf{ObB}$ for the general example of a Kan extension 2.7. Recall that the graphs were



The relations are $RelB = \{b_1b_2b_3 = b_4\}$, X was defined by $XA_1 = \{x_1, x_2, x_3\}, XA_2 = \{y_1, y_2\}$ with $Xa_1 : XA_1 \to XA_2 : x_1 \mapsto y_1, x_2 \mapsto y_2, x_3 \mapsto y_1$, $Xa_2 : XA_1 \to XA_2 : y_1 \mapsto x_1, y_2 \mapsto x_2$, and F was defined by $FA_1 = B_1$, $FA_2 = B_2$, $Fa_1 = b_1$ and $Fa_2 = b_2b_3$.

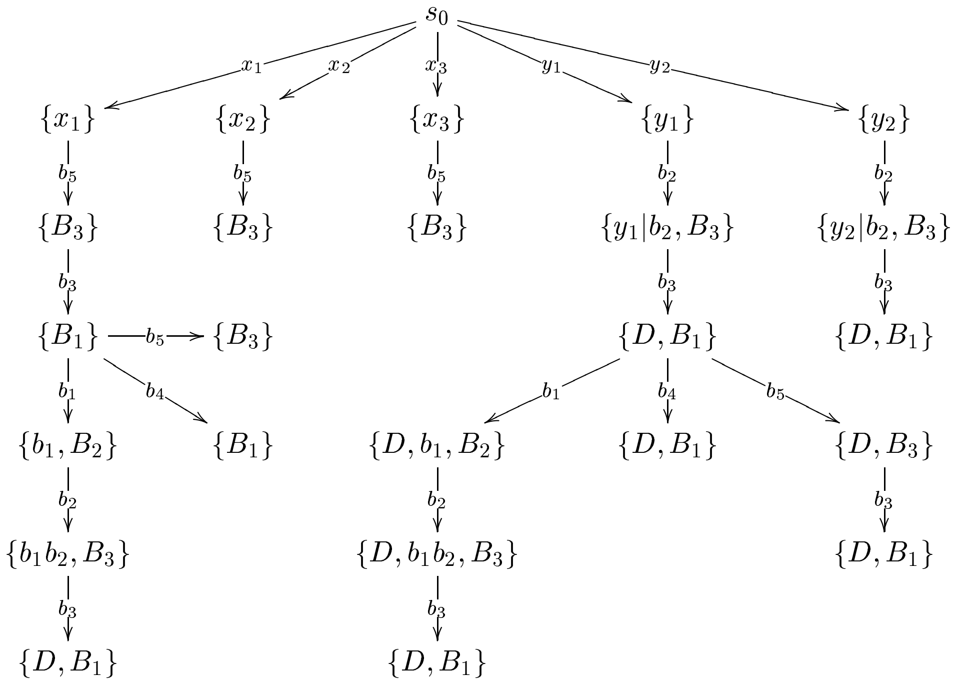
The completed rewriting system was:

$$\begin{array}{llll} x_1|b_1 \to y_1|id_{B_2}, & x_2|b_1 \to y_2|id_{B_2}, & x_3|b_1 \to y_1|id_{B_2}, & y_1|b_2b_3 \to x_1|id_{B_1}, \\ y_2|b_2b_3 \to x_2|id_{B_1}, & x_1|b_4 \to x_1|id_{B_1}, & x_2|b_4 \to x_2|id_{B_1}, & x_3|b_4 \to x_1|id_{B_1}, \\ b_1b_2b_3 \to b_4. & & & \end{array}$$

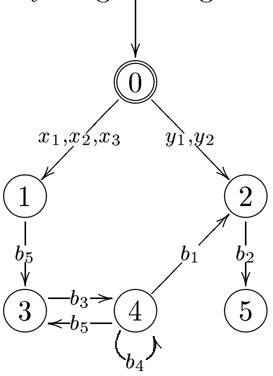
The proper prefix sets are $\mathtt{PPL}(R_T) := \{y_1|b_2, y_2|b_2\}$ and $\mathtt{PPL}(R_P) := \{b_1, b_1b_2\}$. The following table defines the incomplete non-deterministic automaton which rejects only the terms of T that are irreducible with respect to the completed relation $\to$. The alphabet over which the automaton is defined is $\Sigma := \{x_1, x_2, x_3, y_1, y_2, b_1, b_2, b_3, b_4, b_5\}$.

state/letter	x_1	x_2	x_3	y_1	y_2	b_1	b_2	b_3	b_4	b_5
s_0	x_1	x_2	x_3	y_1	y_2	D	D	D	D	D
x_1	D	D	D	D	D	D	D	D	D	B_3
x_2	D	D	D	D	D	D	D	D	D	B_3
x_3	D	D	D	D	D	D	D	D	D	B_3
y_1	D	D	D	D	D	D	$y_1\|b_2, B_3$	D	D	D
y_2	D	D	D	D	D	D	$y_2\|b_2, B_3$	D	D	D
$y_1\|b_2$	D	D	D	D	D	D	D	D	D	D
$y_2\|b_2$	D	D	D	D	D	D	D	D	D	D
B_1	D	D	D	D	D	b_1, B_2	D	D	B_1	B_3
B_2	D	D	D	D	D	D	B_3	D	D	D
B_3	D	D	D	D	D	D	D	B_1	D	D
b_1	D	D	D	D	D	D	b_1b_2, B_3	D	D	D
b_1b_2	D	D	D	D	D	D	D	D	D	D
D	D	D	D	D	D	D	D	D	D	D

By constructing the transition tree for this automaton, we will make it deterministic. The next picture is of the partial transition tree – the arrows to the node marked $\{D\}$ are omitted.



The tree is constructed with strict observation of the order on $\sqcup XA$ and ArrΔ, all arrows are drawn from $\{s_0\}$ and then arrows from each new state created, in turn. When a label e.g. $\{B_3\}$ occurs that branch of the tree is continued only if that state has not been defined previously. Eventually the stage is reached where no new states are defined, all the branches have ended. The tree is then converted into an automaton by 'gluing' all states of the same label. The initial state is $\{s_0\}$ and a state is terminal if its label contains a terminal state from the original automaton. The automaton can often be made smaller, for example, here all the terminal states may be glued together. One possibility is drawn below:



Here the state S_1 is labelled 1 and corresponds to the glueing together of $\{x_1\}$, $\{x_2\}$ and $\{x_3\}$ to form $\{x_1, x_2, x_3\}$ and the state S_2 is $\{y_1, y_2, b_1, B_2\}$. States S_3 and S_4 represent $\{B_3\}$ and $\{B_1\}$ respectively and state S_5 is $\{y_1|b_2, y_2|b_2, B_3, b_1b_2\}$. The complement of this automaton accepts all irreducible elements of $\sqcup KB$. When S_1 and S_4 are terminal the language accepted is KB_1. When S_2 is terminal the language accepted is KB_2. When S_3 and S_5 are terminal the language accepted is KB_3. The language equations from the automaton for KB_1 are:

$$
\begin{aligned}
X_0 &= (x_1 + x_2 + x_3)X_1 + (y_1 + y_2)X_2,\\
X_1 &= b_5X_3 + \mathit{id}_{B_1},\\
X_2 &= b_2X_5,\\
X_3 &= b_3X_4,\\
X_4 &= b_1X_2 + b_4X_4 + b_5X_3 + \mathit{id}_{B_1},\\
X_5 &= \emptyset.
\end{aligned}
$$

Putting $X_2 = \emptyset$ and eliminating X_1 and X_3 by substitution gives

$$
\begin{aligned}
X_0 &= (x_1 + x_2 + x_3)(b_5b_3X_4 + \mathit{id}_{B_1}),\\
X_4 &= (b_4 + b_5b_3)X_4 + \mathit{id}_{B_1}.
\end{aligned}
$$

Finally, applying Arden's Theorem to X_4 we obtain the regular expression

$$X_0 = (x_1 + x_2 + x_3)|(b_5b_3(b_4 + b_5b_3)^* + \mathit{id}_{B_1}).$$

The separator "|" may be added at this point. Similarly, we can obtain regular expressions for KB_2 and KB_3. For KB_2 we have

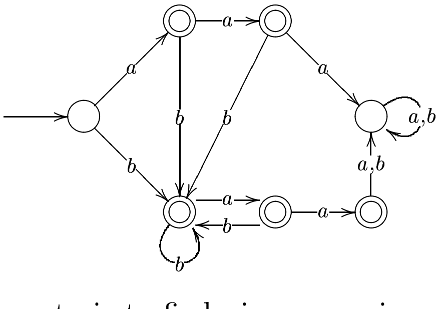
$$X_0 = (x_1 + x_2 + x_3)|b_5b_3(b_4 + b_5b_3)^*b_1 + (y_1 + y_2)|\mathit{id}_{B_2}.$$

For KB_3 the expression is

$$X_0 = (x_1 + x_2 + x_3)|(b_5b_3(b_4 + b_5b_3)^*(b_1b_2 + b_5) + b_5) + (y_1 + y_2)|b_2.$$

4.1.3 Accepting Automata for Algebras

We have discussed automata for rewriting systems which accept only irreducible words. The concept will now be generalised to Gröbner bases. The irreducibles of an algebra $K[S]/\langle P\rangle$ in which we are interested are the irreducible monomials; reducibility of a polynomial is determined by reducibility of the monomials it contains. Therefore the automaton we draw is over the alphabet X, the generators of S and the language it accepts is the set of irreducible monomials. The automaton below is for the infinite dimensional algebra $\mathbb{Q}[\{a,b\}^\dagger]$ factored by the ideal generated by the Gröbner basis $\{a^3 - b + 2,\ ba^2b - 2b^2 + 4a\}$.



The point of drawing acceptor automata is to find nice expressions for the sets of irreducibles. If an algebra is finite then the number of irreducible monomials it has is the *dimension* of the algebra. In the infinite example above we can at least find a regular expression for the set of irreducible monomials.
It is:

$$(a^2b + ab + b)(ab + b)^*(a^2 + a + \mathit{id}) + (a^2 + a)$$

Any element of the algebra is then uniquely expressible as a sum of K-multiples of these monomials.

It is possible to adapt the automaton so that it accepts polynomials by allowing $+$ and $-$ to be elements of the input alphabet, with transitions (from each state) labelled by $+$ and $-$ going to the initial state, and by adding k for $k \in K$ as a loop at the initial state. In this way it may be possible to define automatic algebras. One difficulty to such a definition is the fact that a multiplier/equality recogniser has to recognise that two polynomials are equal though the terms may be input in a different order ($b+a^2$ and a^2+b). There is not the option, as with the acceptor, of working only with monomials. The reason for this is that the normal form of a monomial w multiplied by a generator x (as if to define the multiplier automaton) may well not be a monomial. We mention these issues in passing, only here being concerned with the acceptors and with the reduction machines (next section).

4.2 Reduction Machines

4.2.1 Cayley Graphs

The Cayley graph Γ of a group G with generating set X, and quotient morphism $\theta : F(X) \to G$ is the graph with vertex set $\mathrm{Ob}\Gamma := G$ and edge set $\mathrm{Arr}\Gamma := G \times X$ with $src[g,x] = g$, $tgt[g,x] = g\theta(x)$. The Cayley graph is a representation of the whole multiplication table for the group. In this section we indicate how to use the Cayley graph of a group to help with rewriting procedures. The results are not surprising, but formalise certain procedures which may sometimes be useful.

Proposition 4.2.1 *Let G be the group given by the finite presentation $grp\langle X|Rel\rangle$. Let Γ be the Cayley graph of G. Let $\theta : F(X) \to G$ be the quotient map. Let $>$ be the length-lex order on X^* induced by a linear order on X. Then $>$ specifies a tree in the Cayley graph and a vertex labelling $V \subseteq X^*$ where for all $w_1 \in V$, $w_2 \in F(X)$ such that $\theta(w_1) = \theta(w_2)$ it is the case that $w_2 > w_1$ or $w_2 = w_1$.*

Proof Since G is finite the inverse of any generator can be represented by a positive power. So for any word $r \in F(X)$ there is a word r^+ obtained by replacing each x^{-1} with $x^{Order(x)-1}$, with $\theta(r) = \theta(r^+)$. Therefore we consider the presentation $mon\langle X|R\rangle$ where $R := \{(r^+, id) : r \in Rel\}$ of G. Let $T := \emptyset$, $V := \emptyset$. Start at vertex id and add this label to V. Go through the elements of X in order, adding the edge $[id, x]$ to T whenever it will not create a cycle in the graph. When an edge $[id, x]$ is added to T the target vertex label x should be added to V. Clearly, if $x_i \in V$ and $\theta(x_i) = \theta(x_j)$ for some x_j in X then $x_j > x_i$ and $x_j \notin V$ or else $x_j = x_i$.

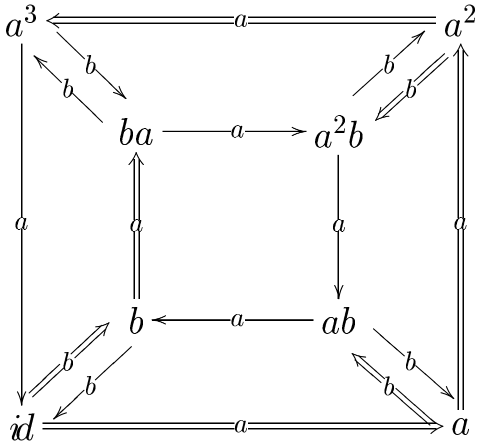
Now repeat the following step until all the vertices of the graph are represented in V; that is until $\theta(V) = G$. Choose the vertex with least label w of V in the graph and go through the elements of X in order adding edges $[w, x]$ to T whenever $\theta(wx) \notin \theta(V)$. This is the condition that to add that edge will not create a cycle. For each new edge $[w, x]$ added to T, add the vertex label wx to V.
It is immediate from the inductive construction that the set of vertex labels V is least in the sense that for any w in V, w is the least element of $F(X)$ with respect to $>$ with image $\theta(w)$. Furthermore, since Γ is connected and edges are chosen so as not to create cycles, T defines a spanning tree of Γ with edges $[\theta(w), x]$. □

Corollary 4.2.2 *The set of vertex labels V is a set of unique normal forms for G in $F(X)$ and the tree T defines a normal form function $N : F(X) \to V$.*

Proof It is immediate from the last result that V is a set of unique normal forms for R on X^*. The normal form function is defined by using the Cayley graph as a reduction machine operating on $F(X)$. Let $x_0^{\varepsilon_0} x_1^{\varepsilon_1} \cdots x_m^{\varepsilon_m}$ be an input word where $\varepsilon_i := \pm 1$ and $x_i \in X$. Start at the vertex with label id and follow the path $[id, x_0^{\varepsilon_0}][\theta(x_0^{\varepsilon_0}), x_1^{\varepsilon_1}] \cdots [\theta(x_0^{\varepsilon_0} \cdots x_{m-1}^{\varepsilon_{m-1}}), x_m^{\varepsilon_m}]$. The label of the target vertex $\theta(x_0^{\varepsilon_0} x_1^{\varepsilon_1} \cdots x_m^{\varepsilon_m})$ is the least element $w \in F(X)$ such that $\theta(w) = \theta(x_0^{\varepsilon_0} x_1^{\varepsilon_1} \cdots x_m^{\varepsilon_m})$. This defines a normal form function N.

□

Example 4.2.3 Consider the Cayley graph for the dihedral group D_8 which is presented by $grp\langle a,b|a^4,b^2,abab\rangle$. The Cayley graph is depicted below, with the vertices labelled according to the ordering induced by $a<b$.



Consider the word aba^3b. Beginning at id follow the path to a. Read b and go to vertex ab. Read a and so go to vertex b. When the final b is read, it takes us to the vertex with label a^2, hence $N(aba^3b)=a^2$.

4.2.2 Reduction Machines for Kan Extensions

We now generalise the reduction machine idea to Kan extensions. Formally, standard output automata are defined in two ways, as *Moore* machines or *Mealy* machines (see [41]). The reduction machines here are Moore machines.

A **Moore machine** is a six-tuple $\underline{M}:=(S,\Sigma,s_0,\delta,\lambda,\Theta)$ where S is the set of states with an initial state s_0, Σ is the input alphabet, Θ is the output alphabet, δ is the transition function from $S\times\Sigma\to S$ and $\lambda:S\to\Theta$ is a mapping which gives the output associated with each state. (All states are "terminal".) As before δ^* denotes the extended state transition function.

We continue with the assumption that $\mathcal{P}:=kan\langle\Gamma|\Delta|RelB|X|F\rangle$ is the finite presentation of the Kan extension (K,ε) and $R=(R_T,R_P)$ is a finite complete rewriting system on the P-set T given by $\mathcal{P}$. We will only work with finite machines, so for the rest of this chapter the Kan extensions will be assumed to be finite i.e. $\sqcup KB$ is finite.

Proposition 4.2.4 *Let $\mathcal{P}$ be a presentation of a finite Kan extension, with complete rewriting system R. Then there exists a Moore machine $\underline{M}=(S,\Sigma,s_0,\delta,\lambda,\Theta)$ such that $\lambda(\delta(w))$ is the irreducible form of w with respect to $\to_R$ on T.*

Proof Define a Moore machine M in the following way. Let $S:=(T/\overset{*}{\leftrightarrow}_R)\sqcup s_0\sqcup d$, $\Sigma:=XA\sqcup\mathrm{Arr}\Delta$, and $\Theta:=T\sqcup 0$. Let s_0 be the initial state. Define $\delta:S\times T\to S$ by $\delta(s_0,x):=[x|id_{FA}]$ and $\delta([t],x)=\delta(d,x):=d$ for all $x\in XA, A\in\mathrm{Ob}\Gamma$ and $t\in T$; and $\delta([t],b):=[t\cdot b]$ for all $t\in T,\ b\in\mathrm{Arr}\Delta$ such that $\tau(t)=src(b)$ and $\delta([t],b)=\delta(s,b)=\delta(d,b):=d$ otherwise. Then define $\lambda:S\to\Theta$ by $\lambda(s)=\lambda(d)=0$ and $\lambda([t]):=N(t)$. It is clear from these definitions that $\lambda(\delta(s,t))=N(t)$ for all $t\in T$.
□

Example 4.2.5 We conclude this subsection with an example of a reduction machine for a Kan extension. Let $\mathcal{P}$ be a Kan extension where Γ and Δ are as follows:

$$\begin{tikzcd} A_1 \arrow[r, bend left, "a_1"] \arrow[r, bend right, "a_2"'] & A_2 \end{tikzcd} \qquad \begin{tikzcd} B_1 \arrow[r, "b_1"] & B_2 \arrow[r, "b_2"] \arrow[rr, bend right, "b_4"'] & B_3 \arrow[r, "b_3"] \arrow[loop above, "b_5"] & B_4 \end{tikzcd}$$

The relations of B are $RelB := \{(b_2b_5b_3, b_4), (b_5^2, b_5)\}$. The functors F and X are defined by:- $FA_1 := B_1$, $FA_2 := B_4$, $Fa_1 := b_1b_2b_3$, $Fa_2 := b_1b_4$ and $XA_1 := \{x_1, x_2, x_3\}$, $XA_2 := \{y_1, y_2\}$, $Xa_1 : XA_1 \to XA_2 : x_1 \mapsto y_1, x_2 \mapsto y_1, x_3 \mapsto y_2$, $Xa_2 : XA_1 \to XA_2 : x_1 \mapsto y_1, x_2 \mapsto y_2, x_3 \mapsto y_2$. The initial rewriting system is in fact complete. It is

$$\{x_1|b_1b_2b_3 \to y_1|\mathit{id}_{B4},\ x_2|b_1b_2b_3 \to y_1|\mathit{id}_{B4},\ x_3|b_1b_2b_3 \to y_2|\mathit{id}_{B4},\ x_1|b_1b_4 \to y_1|\mathit{id}_{B4},$$
$$x_2|b_1b_4 \to y_2|\mathit{id}_{B4},\ x_3|b_1b_4 \to y_2|\mathit{id}_{B4},\ b_2b_5b_3 \to b_4,\ b_5^2 \to b_5\}.$$

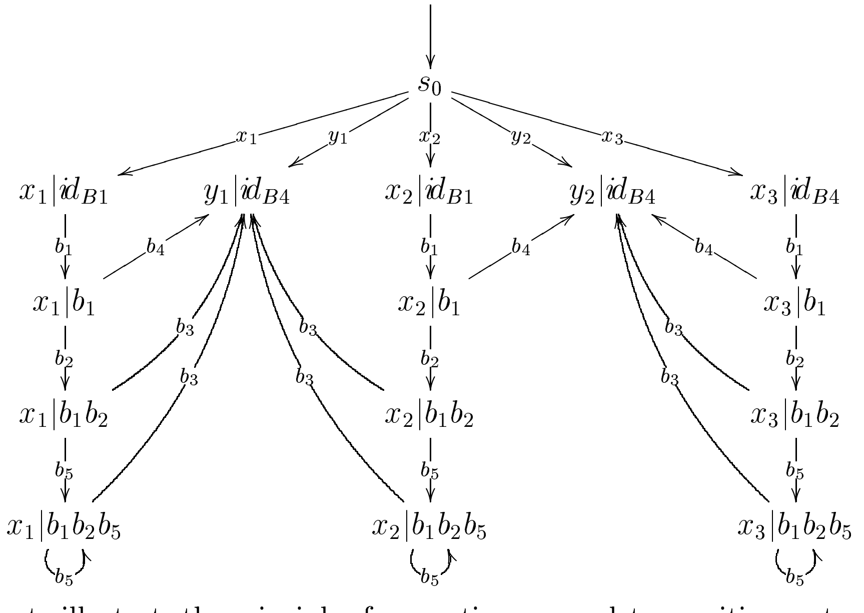
Following the directions in the proof above we construct the Moore machine. There are 14 states $[t] \in S$ and also the initial state s and the dump state d which rejects any terms that are not defined in T.

$$\lambda(S) := \{d, x_1|\mathit{id}_{B1}, x_2|\mathit{id}_{B1}, x_3|\mathit{id}_{B1}, y_1|\mathit{id}_{B4}, y_2|\mathit{id}_{B4},$$
$$x_1|b_1, x_2|b_1, x_3|b_1, x_1|b_1b_2, x_2|b_1b_2, x_3|b_1b_2, x_1|b_1b_2b_5, x_2|b_1b_2b_5, x_3|b_1b_2b_5\}.$$

The non-trivial part of the transition function is as follows:

$$\begin{array}{rll}
\delta(s,x_1) = [x_1|\mathit{id}_{B1}] & \delta(s,x_2) = [x_2|\mathit{id}_{B1}] & \delta(s,x_3) = [x_3|\mathit{id}_{B1}] \\
\delta(s,y_1) = [y_1|\mathit{id}_{B4}] & \delta(s,y_2) = [y_2|\mathit{id}_{B4}] & \delta([x_1|\mathit{id}_{B1}],b_1) = [x_1|b_1] \\
\delta([x_2|\mathit{id}_{B1}],b_1) = [x_2|b_1] & \delta([x_3|\mathit{id}_{B1}],b_1) = [x_1|b_1] & \delta([x_1|b_1],b_2) = [x_1|b_1b_2] \\
\delta([x_1|b_1],b_4) = [y_1|\mathit{id}_{B4}] & \delta([x_2|b_1],b_2) = [x_2|b_1b_2] & \delta([x_2|b_1],b_4) = [y_2|\mathit{id}_{B4}] \\
\delta([x_3|b_1],b_2) = [x_3|b_1b_2] & \delta([x_3|b_1],b_4) = [y_2|\mathit{id}_{B4}] & \delta([x_1|b_1b_2],b_3) = [y_1|b_1] \\
\delta([x_1|b_1b_2],b_5) = [y_1|b_2] & \delta([x_2|b_1b_2],b_3) = [y_2|b_1] & \delta([x_2|b_1b_2],b_5) = [y_2|b_2] \\
\delta([x_3|b_1b_2],b_3) = [y_1|b_1] & \delta([x_3|b_1b_2],b_5) = [y_2|b_2] &
\end{array}$$

The machine can be represented by a diagram – states have not been circled as the labels are too long, and the state d which rejects anything not defined is not drawn.



This example serves to illustrate the principle of converting a complete rewriting system R on T for which there are a finite number of irreducibles into a machine which accepts terms of T (which may be infinite) and gives as output their irreducible form i.e. representatives of elements of $\sqcup KB$.

4.2.3 Reduction Machines for Algebras

We have shown how to use general rewriting systems to construct automata. In a similar way Gröbner bases may be used to construct reduction machines for finite dimensional algebras. The concepts of reduction machines for the previous structures were new but based on standard automata for semigroups. The Gröbner reduction machines for algebras are different from basic output automata.

Let K be a field and let X be a set. Let $\to_R$ be a reduction relation on $K[X^\dagger]$. We define a reduction machine $\underline{M}$ to be a marked graph whose vertices V are labelled by monomials of X^* that are irreducible with respect to $\to_R$. (The monoid identity id represents the algebra identity 1.) Edges have the form (c, x) with $c \in K$, $x \in X$ and from every vertex m there will be at least one edge (c, x) for each $x \in X$. The targets of these edges are the monomials of the reduced form of mx with respect to $\to_R$.

A state of the machine can be represented by a vector in $K[X^\dagger]^n$, where n is the number of vertices. The value at each vertex represents the unprocessed input. When the Cayley graph machines were considered in this way, the state of a machine was essentially a function $V \to F(X)$. Thus it seems reasonable that the state of a Gröbner machine should be represented by a function $V \to K[X^\dagger]$. Essentially the state of a machine is the specification of a value $v \in K[X^\dagger]$ for each vertex m.

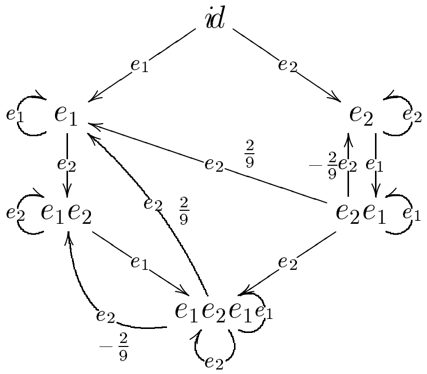
The machine acts by reading the first letter $x_1 \in X$ of a monomial $x_1 \cdots x_n$ of the value v at a vertex m and moves to a new state determined by all the edges leaving m that are labelled (c_i, x_1) and have target m_i. The value at m is decreased by $kx_1 \cdots x_m$ where k is the coefficient of $x_1 \cdots x_n$ in v and the value at each $m_i \in S$ is increased by $c_i x_2 \cdots x_n$. The vital difference between these machines and earlier ones is that monomials can reduce to polynomials, and so there may be more than one arrow with the same letter label coming from a vertex. This becomes clearer on examination of an example.

Example 4.2.6 The third Hecke algebra is $\mathbb{Q}[\{e_1, e_2\}^*]/\langle P \rangle$ where

$$P := \{e_1^2 - e_1, e_2^2 - e_2, e_2e_1e_2 - e_1e_2e_1 + 2/9\, e_2 - 2/9\, e_1\}.$$

In fact P is a Gröbner basis for this algebra. The algebra has dimension 6, the irreducible monomials being $\mathit{id}, e_1, e_2, e_1e_2, e_2e_1, e_1e_2e_1$. We draw a machine which acts to reduce polynomials in $\mathbb{Q}[\{e_1, e_2\}^*]$ The edges have two labels; a generator e_1 or e_2 and a coefficient from $\mathbb{Q}$, (1 where unmarked). For example $e_1e_2e_1e_2$ reduces to $e_1e_2e_1 - \frac{2}{9}e_1e_2 + \frac{2}{9}e_1$ so there are three arrows with letter label e_2 coming out of the vertex $e_1e_2e_1$.

The following diagram shows the "Gröbner machine" for the Hecke algebra defined above.



The machine operates to reduce monomials, for example: $e_1e_2e_1e_2e_1$. Start with the value $e_1e_2e_1e_2e_1$ at vertex id. Read e_1 and the new state of the machine is given by the value $e_2e_1e_2e_1$ at e_1 and 0 elsewhere.

Read e_2 and the state is now given by the value $e_1e_2e_1$ at e_1e_2 and 0 elsewhere. Read e_1 and the state of the machine is e_2e_1 at $e_1e_2e_1$ and 0 elsewhere. Read e_2 and the new state is given by e_1 at $e_1e_2e_1$, $-2/9e_1$ at e_1e_2 and $2/9e_1$ at e_1 with 0 elsewhere. At vertex $e_1e_2e_1$ read e_1 and the new state of the machine is 1 at $e_1e_2e_1$ and the values of the other vertices unchanged. At vertex e_1e_2 read $-2/9e_1$ and the new state of the machine is given by 7/9 at $e_1e_2e_1$ and $2/9e_1$ at e_1 and 0 elsewhere. To finish, read $2/9e_1$ at e_1, and the final state of the machine is given by the values of 7/9 at state $e_1e_2e_1$, 2/9 at e_1 and 0 elsewhere. The output polynomial is therefore $7/9e_1e_2e_1 + 2/9e_1$, this is the irreducible form of $e_1e_2e_1e_2e_1$.

The "Gröbner Machines" described are really no more than "pictures" of the Gröbner bases. We will formalise the ideas of reduction machines for algebras, for the general case, by using Petri nets.

4.3 Petri nets

This section introduces Petri nets and formalises the "Gröbner machines" devised in the previous section in terms of these well-defined structures.

4.3.1 Introduction to Petri nets

Petri nets are a graphical and mathematical modelling tool applicable to many systems. They may be used for specifying information processing systems that are concurrent, asynchronous, distributed, parallel, non-deterministic, and/or stochastic. Graphically, Petri nets are useful for illustrating and describing systems, and tokens can simulate the dynamic and concurrent activities. Mathematically, it is possible to set up models such as state equations and algebraic equations which govern the behaviour of systems. Petri nets are understood by practitioners and theoreticians and so provide a powerful link of communication between them. For example engineers can show mathematicians how to make practical and realistic models, and mathematicians may be able to produce theories to make the systems more methodical or efficient. A good introduction to the ideas of Petri nets is [58].

An integer-valued *Petri net* is a kind of directed graph together with an initial state (called an *initial marking* M_0). The underlying graph of a Petri net is a directed, weighted bipartite graph. The two kinds of vertices are *places* (represented by circles) and *transitions* (represented by rectangles). Edges go between places and transitions and are labelled with their weights. A *marking* assigns a non-negative integer to each place. If a place p is assigned k in a marking then we say p has k *tokens* (represented by black dots). In modelling, places represent conditions and transitions represent events. A transition has input and output places, which represent preconditions and postconditions (respectively) of the event.

A **Petri net** (without specific initial marking) is a 4-tuple $\underline{N} = (P, T, \mathcal{F}, w)$ where:
$P = \{p_1, \dots, p_m\}$ is a finite set – the places,
$T = \{t_1, \dots, t_n\}$ is a finite set – the transitions,
$\mathcal{F} \subseteq (P \times T) \cup (T \times P)$ is a set of edges – the flow relation,
$w : \mathcal{F} \to \mathbb{N}$ is a weight function,
and $P \cap T = \emptyset$, $P \cup T \neq \emptyset$.

The state of a Petri net is represented by a marking. A **marking** is a function $M : P \to \mathbb{N} + \{0\}$. Let $\underline{N}$ be a Petri net where each place is given a distinct label p_i. To every marking M we will associate a polynomial $pol(M) := \Sigma_P\, pM(p)$ that is the formal sum of terms where $M(p)$ is a non-negative integer and p is a place label.

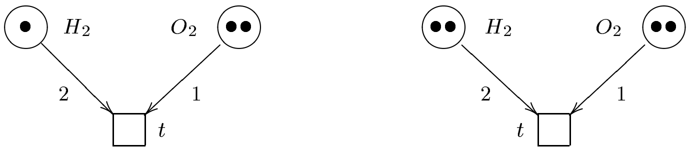
The behaviour of dynamic systems may be described in terms of system states and changes. A marking of a Petri net is changed according to the **firing rule**:

i) A transition t is **enabled** if each input place p of t is marked with at least $w(p,t)$ tokens where $w(p,t)$ is the weight of the edge from p to t.

ii) An enabled transition may or may not **fire** – depending on whether or not the relevant event occurs.

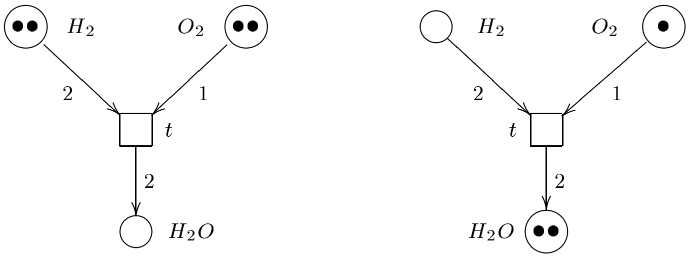
iii) Firing of an enabled transition t removes $w(p,t)$ tokens from each input place p of t and adds $w(t,q)$ tokens to each output place q of t where $w(t,q)$ is the weight of the edge from t to q.

Example 4.3.1 The markings of the nets below are given by the polynomials $H_2 + 2O_2$ and $2H_2 + 2O_2$ respectively. The transition t is enabled in the second case and not in the first:



Each transition t has an associated polynomial $pol(t) := \Sigma_P\, pw(p,t) - \Sigma_P\, pw(t,p)$, that is the sum of the weights of tokens that a firing of transition t takes from each input place minus the sum of weights of tokens that it adds to each output place. A firing/occurrence sequence is denoted by $M_0 \xrightarrow{t_1} M_1 \xrightarrow{t_2} \cdots \xrightarrow{t_n} M_n$ where the M_i are markings and the t_i are transitions (events) transforming M_{i-1} into M_i. For $i = 1, \ldots, n$ it follows from the definitions that $pol(M_i) = pol(M_{i-1}) - pol(t_i)$. Therefore the above firing sequence gives the information $pol(M_n) = pol(M_0) - pol(t_1) - pol(t_2) - \cdots - pol(t_n)$.

Example 4.3.2 The formula $2H_2 + O_2 = 2H_2O$ is represented by the transition in the diagrams below, the left diagram shows the initial marking and the right shows the marking after the transition has fired.



The polynomial for the transition is $2H_2 + O_2 - 2H_2O$ and the firing sequence would be denoted $2H_2 + 2O_2 \xrightarrow{t} O_2 + 2H_2O$.

One of the main problems in Petri net theory is *reachability* (see [32] for some examples). A marking M is said to be **reachable** from a marking M_0 in a net $\underline{N}$, if there is a sequence of firings that transforms M_0 to M.

Definition 4.3.3 *The* ***reachability problem*** *for a Petri net $\underline{N}$ is as follows:*

INPUT: *M_1, M_2, two markings of $\underline{M}$,*
QUESTION: *is M_2 reachable from M_1?*

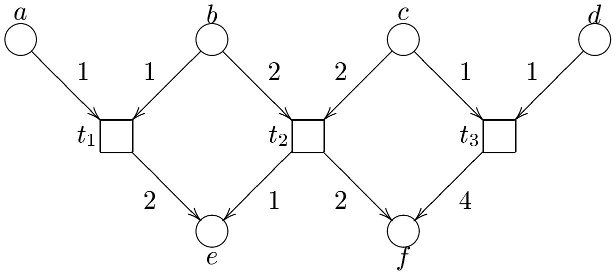
Often a Petri net comes with a specified **initial marking** M_0. Then the reachability refers to reachability from M_0 and the reachability problem refers to deciding whether a marking M is reachable from M_0. Note: For the type of Petri nets defined so far reachability is decidable [58] (in exponential time and space).

A Petri net $\underline{N}$ is called **reversible** if a marking M_2 is reachable from another marking M_1 implies that M_1 is reachable from M_2. A Petri net with initial marking may be called reversible if there is always a firing sequence of events that will transform the net from any reachable marking back to the initial marking.

Proposition 4.3.4 *Let $\underline{N}$ be a reversible Petri net. Define $F := \{pol(t) : t \in T\}$ and let $\langle F\rangle$ be the ideal generated by F in $\mathbb{Z}[P]$. Let M and M' be two markings of $\underline{N}$. Then M' is reachable from M only if $pol(M) - pol(M') \in \langle F\rangle$.*

Proof From the definitions above, if M' is reachable from M then there is a firing sequence $M = M_0 \xrightarrow{t_1} M_1 \xrightarrow{t_2} \cdots \xrightarrow{t_n} M_n = M'$ so that $pol(M') = pol(M) - pol(t_1) - \cdots - pol(t_n)$. This implies that $pol(M) - pol(M') = pol(t_1) + \cdots + pol(t_n) \in \langle F\rangle$. □

Example 4.3.5 Let $\underline{N}$ be the reversible Petri net given by the marked graph below:



The places are $P := \{a, b, c, d, e, f\}$ and the polynomials defined by the transitions are $t_1 := a + b - 2e$, $t_2 := 2b + 2c - e - 2f$ and $t_3 := c + d - 4f$. A Gröbner basis (using the order $f > e > d > c > b > a$) for the ideal generated in $\mathbb{Q}[P]$ is

$$F := \{d - 3c - 3b + a,\ e - \frac{1}{2}a - \frac{1}{2}b,\ f + \frac{1}{4}a - \frac{3}{4}b - c\}.$$

For any marking M the polynomial $pol(M)$ may be reduced, using the relation $\rightarrow_F$ defined by the Gröbner basis, to an irreducible form $irr(M) \in \mathbb{Q}^{\geqslant 0}[\{a, b, c\}^*]$. Here are three examples.

$$\begin{aligned}
pol(M_0) &= 2a + 2b + 3c + d \rightarrow_F 2a + 2b + 3c - (-3c - 3b + a) = a + 5b + 6c \\
pol(M_1) &= 4e + 2c + 4f \rightarrow_F 4(\frac{1}{2}a + \frac{1}{2}b) + 2c + 4(-\frac{1}{4}a + \frac{3}{4}b + c) = a + 5b + 6c \\
pol(M_2) &= a + d + 3e + 5f \rightarrow_F a + (3c + 3b - a) + 3(\frac{1}{2}a + \frac{1}{2}b) + 5(-\frac{1}{4}a + \frac{3}{4}b + c) = \frac{1}{4}a + \frac{33}{4}b + 8c
\end{aligned}$$

So M_2 is not reachable from M_0 because the corresponding polynomials do not reduce to the same form. It is here the case that M_1 is reachable from M_0 but this result does not necessarily follow from the reduced polynomials for these markings being the same.

Remark 4.3.6 We can draw a rational-valued Petri net that is equivalent to the original net $\underline{N}$ but whose transition polynomials are the Gröbner basis and whose markings are a function $P \rightarrow \mathbb{Q}^{\geqslant 0}$. This is constructed by drawing a state for each letter and a transition for each polynomial. The arcs into a transition come from the letters with positive coefficient and are weighted with that coefficient. Similarly the arcs leaving a transition correspond to the negative terms in the polynomial.

4.3.2 Gröbner Machines as Petri-Nets

The Gröbner machine for reducing polynomials which was described at the end of Section 4.2 can be expressed quite nicely as a Petri net.

Theorem 4.3.7 *Let K be a field, let X be a set and let $F \subseteq K[X^\dagger]$ be a Gröbner basis for the ideal $\langle F\rangle$. Then there is a Petri net $\underline{N}$ which can be marked with a polynomial $f \in K[X^\dagger]$ so that any resulting sequence of firings can be extended to a finite sequence of firings that terminates with a unique non-live state. All states reachable from the initial marking may be identified with polynomials that are equivalent under $=_F$ to f.*

Proof We will define a type of Petri net and firing rule from the Gröbner basis. Let $\underline{N} := (P, T, \mathcal{F}, w)$. The set of places P is the set of monomials m of $K[X^\dagger]$ which are irreducible with respect to $\to_F$, together with an 'initial' place labelled id. The set of transitions T is identified with $P \times X$.

The flow relation $\mathcal{F}$ is described as follows. The transition (m, x) has a single input edge from m with weight x. If $mx \in P$ then (m, x) has a single output edge to mx with weight 1. If $mx \notin P$ then mx is the leading monomial of some $f = mx - \Sigma_{i=1}^n k_i m_i$ in F. In this case there is an output edge from (m, x) to each non-leading term in f, the edge to m_i having weight k_i.

The Petri net just defined differs from the standard type in that the weight function returns elements of K or elements of X rather than just integers. So $w : \mathcal{F} \to K[X^\dagger]$. Similarly a marking is a function $M : P \to K[X^\dagger]$ and is identified with the polynomial $pol(M) := \Sigma_P \, mM(m)$

Let M_1 be a marking, with $M_1(m) \in K[X^\dagger]$ for each $m \in P$. Let (m, x) be an enabled transition, so that $M_1(m)$ contains a term kxv for some $k \in K$, $v \in X^*$. If mx is irreducible, then when (m, x) fires, the term kxv is removed from m while mx gains a term kv, so the resulting marking M_2 is such that

$$pol(M_2) := \Sigma_P m M_2(m) = \Sigma_P m M_1(m) - m(kxv) + mx(kv) = pol(M_1).$$

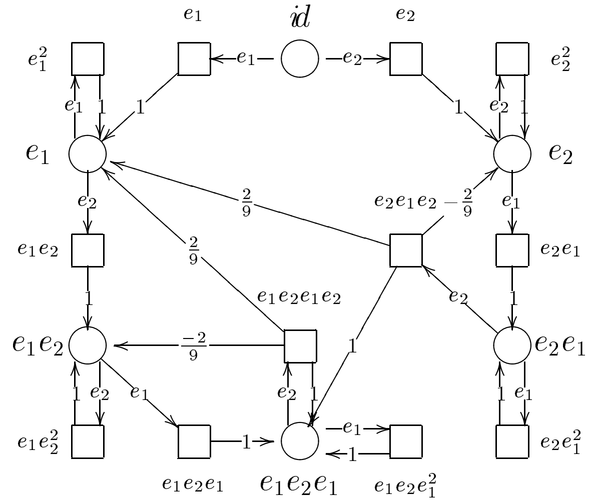
Alternatively, when $f = mx - \Sigma_{i=1}^n k_i m_i \in F$ and (m, x) fires, M_2 is such that

$$pol(M_2) = pol(M_1) - m(kxv) + \Sigma_{i=1}^n m_i(kk_i v) = pol(M_1) - kfv,$$

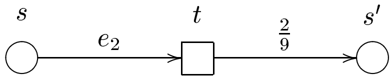
and so $pol(M_1) \to_F pol(M_2)$.

Thus a firing represents a single step reduction by $\to_F$. The relation is complete, since F is a Gröbner basis, and therefore there exists a unique non-live marking (irreducible polynomial) which may be reached within a finite firing sequence (sequence of reductions). □

Example 4.3.8 The picture for the third Hecke Algebra Petri net (whose Grobner machine was Example 4.2.6) is as follows (with each transition label (m, x) written mx):



The states of the Petri net are labelled by the irreducible monomials. To reduce a polynomial p take the initial marking M_0 to be such that $M_0(id) = p$ and $M_0(m) = 0$ for all other $m \in P$. A transition is enabled if the input states to it hold terms which are right multiples of the weight on their input arcs. Firing of a transition transforms the input and all output states simultaneously. For example, if in the situation illustrated here the state s holds tokens to a value of e_2v for some string v then the transition t is enabled (to the value of v).



If transition t then fires, the output state s' receives tokens to the value of $\frac{2}{9}v$, which is added to the token value it already holds. The marking remaining on the net when all enabled transitions have fired and the net is no-longer live (this happens due to the Noetherian property of the Gröbner basis), represents the irreducible form of the polynomial given by the initial marking. This polynomial is extracted from the Petri net by adding the token multiples of the states, i.e. if there are 9 tokens at state e_1 and $\frac{5}{3}$ tokens at state e_1e_2 then the polynomial is $9e_1 + \frac{5}{3}e_1e_2$.

Remark 4.3.9 The nature of Petri nets is to allow for concurrent operations, and this ties in well with the different ways in which a polynomial may be reduced by a set of other polynomials. A Petri net can be used to model reduction by a set of non-commutative polynomials. It is only in those sets which are Gröbner bases, however, that the non-live state eventually reached is entirely determined by the initial marking.

4.4 Remarks

The main theme of Chapter Four was the relation between rewrite systems / Gröbner bases and various types of machine.

Automata can be useful for determining whether or not a structure is finite (has a finite number of elements). The automaton is drawn directly from the complete rewriting system, the equations for it (see [28]) can be solved (Arden's theorem) to obtain a regular expression for the language (i.e. the set of normal forms of the elements) which will be infinite if the free monoid (Kleene star) of some sub-expression occurs. Beyond acceptance or rejection of words, these automata have no output. It is more helpful to consider the type of machines ("Cayley machines") which take any word as input and output its reduced form. We introduced such Cayley machines (or "Gröbner machines") for algebras. Input is a polynomial and the unique irreducible form of that algebra element is the output. These machines can be seen as types of automata with output or – as illustrated for the polynomial ring case – as Petri nets.

The main result of the second section was the definition of reduction machines for finite Kan extensions. The final section of this chapter on machines introduced Petri nets. It is of interest to model Gröbner bases with Petri nets, because it would be extremely useful to find some equivalences between them, so that Petri nets could be analysed using Gröbner bases. With this aim in mind we showed how the "Gröbner machine" for an algebra is a type of Petri net. An example of an application of commutative Gröbner bases to the reachability problem in reversible Petri nets is also given. There is much scope for further work in this area.

Chapter 5

Identities Among Relations

There is a large number of papers on computing resolutions of groups, in the usual sense of homological algebra. Many of these computations are for particular classes of groups (e.g. p-groups, nilpotent groups) and some of these compute only resolutions mod p. In general, they do not compute modules of identities among relations because they are not specific to a presentation.
This problem can be put more generally as that of extending a *partial resolution* of a group. That is, we are given an exact sequence of free $\mathbb{Z}G$-modules $C_n \to C_{n-1} \to \cdots \to C_1$, and we are asked to extend it by further stages. For the identities among relations for a presentation $\mathcal{P} = grp\langle X|R\rangle$, the initial case is $n = 2$ with the boundary given by the Whitehead-Fox derivative

$$\partial_2 = (\partial r/\partial x) : (\mathbb{Z}G)^R \to (\mathbb{Z}G)^X.$$

The problem is to extend this by one or two more stages – the boundaries of the free generators of C_3 then give generators for the module of identities. If also we find C_4 and the boundary to C_3, then we have a module presentation of the module of identities.

This problem is usually expressed as 'choose generators for the kernel of ∂_2'. However, it is not clear how this can be done algorithmically. The main result of Brown/Razak [17] relates this problem to the construction of a partial contracting homotopy for a partial free crossed resolution of the universal covering groupoid of the group G. This contracting homotopy is related to choices of what are often called 0- and 1-combings of the Cayley graph.

The main results of this chapter show how to define an "extra information rewriting system" or EIRS and how to use this to construct the homotopy h_1. The EIRS records the steps that have been taken in rewriting. The 'record' is a sequence of elements of the free crossed module of the presentation. This shows that the normal form function of a complete rewriting system for a group presentation determines (up to some choices) a set of free generators for the part C_3 of a resolution, together with the boundary to C_2. In fact the generators of C_3 are in one to one correspondence with the elements of $G \times R$, but the boundary depends on the choice of complete EIRS. This method of computing h_1 means that the computation of a set of generators for the module of identities among relations is completely algorithmic. This work was done with the help of Chris Wensley. The computer program `idrels.g` implements the procedure.

The next problem is that of reducing the generating set of the $|R| \times |G|$ identities computed. When the group is small (e.g. S_3) this can be done by trial and error. In fact S_3 is a Coxeter group, and for these it has already been proven [68, 67] that the standard presentation yields a minimum of 4 generators for the module of identities. The methods of these papers do not, however, produce relations among these module generators.

The example of S_3 is used to demonstrate how reduced sets of generators at one level determine the identities at the next level, and the way in which the reducible elements are expressed in terms of the irreducibles allows the calculation of these new identities. The example is a good illustration because it is small enough to be done by hand, whilst illustrating that the crossed resolution for even a small group given by a familiar presentation may be quite complex.

The final part of the chapter identifies why the problem of reducing the set of generators is difficult, and expresses it in terms of a Gröbner basis problem (the submodule problem).

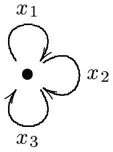
The crossed complex construction of [17], together with an enhanced rewriting procedure and noncommutative Gröbner basis theory over rings are brought together to indicate an algorithmic method for constructing a free crossed resolution of a group. This is an area that will require much further development.

5.1 Background

There are strong geometrical and algebraic reasons for studying the *module of identities among relations* [15, 63]. The following exposition gives some of the topological background.

We assume the usual notion of a presentation $\mathcal{P} := grp\langle X|R\rangle$ of a group G, where X is a set generating G and $R \subseteq F(X)$ is called the set of relators. To allow for repeated relators we can also consider presentations of the form $grp\langle X, \mathcal{R}, w\rangle$ where $w : \mathcal{R} \to F(X)$ is a function such that $w(\mathcal{R}) = R$.

From $\mathcal{P}$ we form the *cell-complex* $K = K(\mathcal{P})$ *of the presentation.* This is a 2-dimensional complex. Its 1-skeleton K^1 is $\bigvee_{x \in X} S^1_x$, a wedge of directed circles - one for each generator $x \in X$:



This topological space has fundamental group $\pi_1(K^1, *)$ isomorphic to the free group $F(X)$ on the set X. Now K is formed as

$$K = K^1 \cup_{\{f_r\}} \{e^2_r\},$$

by attaching to K^1 a 2-cell by a map $f_r : S^1_r \to K^1$ chosen in the homotopy class $w(r) \in F(X) = \pi_1(K^1)$ for each $r \in \mathcal{R}$. The homotopy type of K is independent of the choice of f_r in its homotopy class.
In the next section we shall define the free crossed module $(\delta_2 : C(w) \to F(X))$ on a function $w : \mathcal{R} \to F(X)$. Whitehead [77, 78, 79] proved that $(\pi_2(K^2, K^1, *) \to \pi_1(K^1, *))$ is the free crossed module on $w : \mathcal{R} \to \pi_1(K^1, *) = F(X)$, and so is isomorphic to $(C(w) \to F(X))$. In particular $ker\delta_2 \cong \pi_2(K, *)$, the second homotopy group of the geometrical model of the presentation, and so this homotopy group is also called the module of identities among relations for the group presentation.

Example 5.1.1 The torus $T = S^1 \times S^1$ has a cell structure $(S^1 \vee S^1) \cup_{f_r} \{e^2_r\}$ and its fundamental group is presented by $\mathcal{P} := grp\langle a, b \mid aba^{-1}b^{-1}\rangle$. In this case $\pi_2(T) = 0$, since $\pi_2(S^1) = 0$, but it is not so obvious that $\ker \delta_2 = 0$.

More background to these topological ideas may be found in [11]. There have been many papers written on $\pi_2(K^2, *) = ker(C(R) \to F(X))$ (some examples are [4, 12, 14, 77, 78, 79, 36, 37]). The methods often use a geometrical notion of "pictures" [6, 63, 64, 65, 66, 67] to work with identities among relations. Although the computation of $\pi_2(K^2, *)$ is reduced to an algebraic problem on crossed modules, this has not previously helped the computation. We shall follow the paper [17] in developing algorithmic methods for this computation. For this, we need the language of free crossed modules.

Let $\mathcal{P} := grp\langle X|R\rangle$ be a group presentation. An **identity among relations** is a specified product of conjugates of relations

$$\iota \;=\; ({r_1}^{\varepsilon_1})^{u_1}({r_2}^{\varepsilon_2})^{u_2}\cdots({r_n}^{\varepsilon_n})^{u_n}$$

where $r_i \in R, \varepsilon_i = \pm 1, u_i \in F(X)$ such that ι equals the identity in $F(X)$.

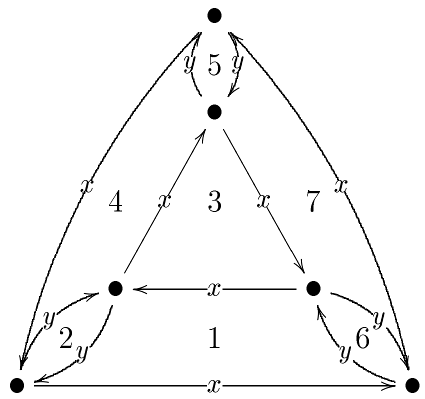
Example 5.1.2 Let $grp\langle X|R\rangle$ be a group presentation. Then for any elements $r, s \in R$ we have the identities

$$\begin{array}{rcl} r^{-1}s^{-1}rs^r & = & id, \\ rs^{-1}r^{-1}s^{r^{-1}} & = & id. \end{array}$$

When a group has a Cayley graph which forms a simply connected region comprised of cells whose boundaries correspond to relators, an identity ι may be obtained by the following procedure:

- Order the cells as $\gamma_1, \ldots, \gamma_m$ in such a way that for all $i = 1, \ldots, m$ the first i cells form a simply connected sub-region Λ_i.
- Choose to transverse each cell in an anti-clockwise direction.
- Form a product of of conjugates of relators $v_1 \cdots v_m$ where v_i is determined as cell γ_i is added to Λ_{i-1}. To add γ_i, start from the vertex id and move clockwise around the boundary of Λ_{i-1} until a suitable start vertex on the boundary of γ_i is reached. A start vertex is such that the word formed by the anti-clockwise boundary of γ_i starting at that vertex is either the relator r_i or the inverse r_i^{-1} of the relator label corresponding γ_i. Let u_i be the word given by the path from id to the start vertex. Then the required term is $v_i := (r_i^{\varepsilon_i})^{u_i^{-1}}$.
- Finally set $\iota := v_1 \cdots v_m r_b^{\varepsilon_b}$ where $r_b^{\varepsilon_b}$ is the relator associated to the boundary.

Example 5.1.3 In the case of a specific group presentation, $S_3 = grp\langle x, y \,|\, x^3, y^2, xyxy\rangle$, label the relators in S_3 as r, s, t respectively, and order the cells of the Cayley graph as shown below:



Here cells 1,4,7 (traversed in an anti-clockwise direction) correspond to t; cells 2,5,6 correspond to s^{-1} while cell 3 and the outer boundary (considered as the boundary of the "outside cell") correspond to r^{-1}. We obtain

$$\iota \;:=\; t(s^{-1})(r^{-1})^{y^{-1}}t^{y^{-1}}(s^{-1})^{x}(s^{-1})^{x^{-1}y^{-1}x}t^{y^{-1}x}r^{-1}.$$

We can verify algebraically that ι is an identity:

$$\begin{aligned}
&\mapsto (xyxy)(y^{-2})(x^{-3})^{y^{-1}}(xyxy)^{y^{-1}}(y^{-2})^{x}(y^{-2})^{x^{-1}y^{-1}x}(xyxy)^{y^{-1}x}x^{-3} \\
&= (xyxy)(y^{-2})(yx^{-3}y^{-1})(yxyxyy^{-1})(x^{-1}y^{-2}x)(x^{-1}yxy^{-2}x^{-1}y^{-1}x)(x^{-1}yxyxyy^{-1}x)(x^{-3}) \\
&= id.
\end{aligned}$$

5.2 The Module of Identities Among Relations

To discuss relations among generators of G we use free groups. To discuss identities among the relations of G we need free crossed modules. The precise idea of a consequence of the relations, and in particular of an identity is similar to that of specifying a relator as an element of the free group, but takes the action of F into account.

Peiffer and Reidemeister were the first to detail the construction in [61, 69] in 1949. Reidemeister sets up the necessary group action by associating each element of a first group with an automorphism of a second group, defining a homomorphism between the two groups, requiring that it fulfills CM1. He looks at the class of Peiffer relations of the kernel of this homomorphism, and factors the first group by the congruence generated by the Peiffer relations. The construction is the same as that detailed below, but he does not mention the terms "group action" or "crossed module". Given that "crossed module" had only been defined by Whitehead in 1946, this is not so surprising. It was not until 1982 that perhaps the first paper [15] to recognise and name the structures that Reidemeister defined was published.

Formally, given a group F, a **pre-crossed F-module** is a pair (C, δ) where $\delta : C \to F$ is a group morphism with an action of F on C denoted c^u $(u \in F)$ so that:

$$\text{CM1)} \quad \delta(c^u) \;=\; u^{-1}(\delta c)u \quad \text{for all } \; c \in C, u \in F.$$

A **crossed F-module** is a pre-crossed F-module that also satisfies the **Peiffer relation**:

$$\text{CM2)} \quad c^{-1}c_1c = c_1^{\delta c} \qquad \text{for all } \; c, c_1 \in C.$$

When (δ, C, F) is a crossed module it is also common to refer to it as the crossed $F(X)$-module (δ, C). For more information on crossed modules see [18, 19, 20, 49].

The following exposition is a combination of ideas in [15, 30, 69]. It details the construction of the module of identities among relations. The construction is not exactly the same as that in the references, since it is in terms of rewriting systems on a free monoid rather than normal subgroups of a free group.

Let $\mathcal{P} := grp\langle X, \mathcal{R}, w\rangle$ be a presentation of a group G where $\mathcal{R}$ is a set of labels for the relators identified by the (not necessarily injective function) $w : \mathcal{R} \to F(X)$ and $R := w(\mathcal{R})$.

A crossed $F(X)$-module (C, δ) is **free** on the function $w : \mathcal{R} \to F(X)$ if, given any other crossed $F(X)$-module (D, γ) with a map $\beta : \mathcal{R} \to D$, there exists a unique morphism of crossed $F(X)$-modules $\phi : C \to D$ which satisfies $\alpha \circ \phi = \beta$.

Define $Y := \mathcal{R} \times F(X)$, and write elements of Y in the form (ρ, u), where $\rho \in \mathcal{R}, u \in F(X)$.
Put $Y^+ := \{y^+ : y \in Y\}$ and $Y^- := \{y^- : y \in Y\}$. Elements of the free monoid $(Y^+ \sqcup Y^-)^*$ are called **Y-sequences** and have the form

$$(\rho_1, u_1)^{\varepsilon_1} \cdots (\rho_n, u_n)^{\varepsilon_n}.$$

Define an action of $F(X)$ on Y by

$$(\rho, u)^x := (\rho, ux) \text{ for } x \in F(X).$$

This induces an action of $F(X)$ on $(Y^+ \sqcup Y^-)^*$. Define a monoid morphism $\delta : (Y^+ \sqcup Y^-)^* \to F(X)$ to be that induced by

$$\delta(\ (\rho, u)^\varepsilon) = u^{-1}(w\rho)^\varepsilon u \text{ where } \varepsilon = \pm.$$

Define

$$\begin{array}{rcl} R_P & := & \{(y^- z^+ y^+, z^{+\delta y^+}) : y, z \in Y\} \\ & \cup & \{(y^+ z^- y^-, z^{-\delta y^-}) : y, z \in Y\} \\ & \cup & \{(y^- y^+, id) : y \in Y\} \\ & \cup & \{(y^+ y^-, id) : y \in Y\} \end{array}$$

and define $\to_{R_P}$ to be the reduction relation generated by R_P on $(Y^+ \sqcup Y^-)^*$. For $a, b \in (Y^+ \sqcup Y^-)^*$ if $a \overset{*}{\leftrightarrow}_{R_P} b$ then a and b are said to be **Peiffer Equivalent**.

Definition 5.2.1 *The **Peiffer Problem** is as follows:*

INPUT:	$a, b \in (Y^+ \sqcup Y^-)^*$	*two elements of the free monoid,*
QUESTION:	$a \overset{*}{\leftrightarrow}_{R_P} b$*?*	*are they Peiffer Equivalent?*

The motivation for solving this Peiffer Problem comes from the fact that we wish to construct a particular free crossed module, whose kernel will be the module of identities among relations. Define

$$C(R) := \frac{(Y^+ \sqcup Y^-)^*}{\overset{*}{\leftrightarrow}_{R_P}}.$$

Lemma 5.2.2 *$C(R)$ is a group.*

Proof Let $a, b \in (Y^+ \sqcup Y^-)^*$. The congruence $\overset{*}{\leftrightarrow}_{R_P}$ preserves the composition of Y-sequences so we define $[a]_{R_P}[b]_{R_P} := [ab]_{R_P}$. The identity is $[id]_{R_P}$, and if $a = y_1^{\varepsilon_1} \cdots y_n^{\varepsilon_n}$ for $y_1, \ldots, y_n \in Y$, $\varepsilon_1, \ldots, \varepsilon_n = \pm$ then $[a]_{R_P}^{-1} := [y_n^{-\varepsilon_n} \cdots y_1^{-\varepsilon_1}]_{R_P}$ is the inverse. □

Lemma 5.2.3 *There is an action of $F(X)$ on $C(R)$ defined by*

$$[a]^x := [a^x] \text{ for } x \in F(X).$$

Proof Let $y, z \in Y$, $x \in F(X)$ then $y = (\rho, u)$ and $z = (\sigma, v)$ for some $u, v \in F(X), \rho, \sigma \in R$.

$$\begin{array}{rcll} (y^- z^+ y^+)^x & = & (\rho, ux)^- (\sigma, vx)^+ (\rho, ux)^+ & \\ & = & y_1^- z_1^+ y_1^+ & \text{where } y_1 = (\rho, ux), z_1 = (\sigma, vx) \in Y \\ & \overset{*}{\leftrightarrow}_P & {z_1}^{+\delta y_1^+} & \\ & = & (\sigma, vx)^{+\delta(\rho, ux)^+} & \text{by definition of } y_1, z_1 \\ & = & (\sigma, vx(x^{-1}\delta(\rho, u)^+ x)^+ & \text{by definition of the action on } (Y^+ \sqcup Y^-)^* \\ & = & (\sigma, v\delta(\rho, u)^+ x)^+ & \\ & = & ((\sigma, v)^{+\delta(\rho, u)^+})^x & \\ & = & (z^{+\delta y^+})^x & \text{by definition of } y, z \end{array}$$

Similarly $(y^+ z^- y^-)^x \overset{*}{\leftrightarrow}_{R_P} (z^{-\delta y^-})^x$, and it is also clear that $(y^+ y^-)^x \overset{*}{\leftrightarrow}_{R_P} (id)^x = id$ and $(y^- y^+)^x \overset{*}{\leftrightarrow}_{R_P} (id)^x = id$. Therefore the action of $F(X)$ on $C(R)$ is well-defined by $[a]^x := [a^x]$. □

Lemma 5.2.4 *There is a group homomorphism* $\delta_2 : C(R) \to F(X)$ *defined by*

$$\delta_2[a]_{R_P} := \delta(a) \textit{ for } a \in (Y^+ \sqcup Y^-)^*.$$

Proof Let $a, b \in (Y^+ \sqcup Y^-)^*$. We require to prove that if $a \overset{*}{\leftrightarrow}_{R_P} b$ then $\delta(a) = \delta(b)$. It is therefore sufficient to prove, for all $y, z \in Y$, that $\delta(y^- z^+ y^+) = \delta(z^{+\delta y^+})$, $\delta(y^+ z^- y^-) = \delta(z^{-\delta y^-})$ and $\delta(y^+ y^-) = \delta(y^- y^+) = id_{F(X)}$. Let $y = (\rho, u), z = (\sigma, v) \in Y$. Then

$$\begin{aligned}
\delta(y^- z^+ y^+) &= \delta(\rho, u)^- \delta(\sigma, v)^+ \delta(\rho, u)^+, \\
&= u^{-1} w(\rho)^{-1} u v^{-1} w(\sigma) v u^{-1} w(\rho) u, \\
&= \delta(\sigma, v u^{-1} w(\rho) u)^+, \\
&= \delta(\sigma, v \delta(\rho, u)^+)^+, \\
&= \delta((\sigma, v)^{+\delta(\rho, u)^+}), \\
&= \delta(z^{+\delta y^+}),
\end{aligned}$$

and

$$\begin{aligned}
\delta(y^+ y^-) &= \delta(\rho, u)^+ \delta(\rho, u)^-, \\
&= u^{-1} w(\rho) u u^{-1} w(\rho)^{-1} u, \\
&= id_{F(X)}.
\end{aligned}$$

The other two cases can be proved in the same way, therefore δ_2 is well-defined. □

Theorem 5.2.5 $(C(R), \delta_2)$ *is the free crossed* $F(X)$*-module on* $w : \mathcal{R} \to F(X)$.

Proof First we verify the crossed module axioms.
CM1: Let $a = (\rho_1, u_1)^{\varepsilon_1} \cdots (\rho_n, u_n)^{\varepsilon_n}$ for $(\rho_1, u_1), \ldots, (\rho_n, u_n) \in Y$, $\varepsilon_1, \ldots, \varepsilon_n = \pm$ and let $x \in F(X)$. Then

$$\begin{aligned}
\delta_2([a]^x_{R_P}) &= \delta([(\rho_1, u_1)^{\varepsilon_1}]^x) \cdots \delta([(\rho_n, u_n)^{\varepsilon_n}]^x) \\
&= x^{-1} u_1^{-1} w(\rho_1)^{\varepsilon_1(1)} u_1 x \cdots x^{-1} u_n^{-1} w(\rho_n)^{\varepsilon_n(1)} u_n x, \\
&= x^{-1} (u_1^{-1} w(\rho_1)^{\varepsilon_1(1)} u_1 \cdots u_n^{-1} w(\rho_n)^{\varepsilon_n(1)} u_n) x, \\
&= x^{-1} \delta((\rho_1, u_1)^{\varepsilon_1} \cdots (\rho_n, u_n)^{\varepsilon_n}) x, \\
&= x^{-1} \delta_2[(\rho_1, u_1)^{\varepsilon_1} \cdots (\rho_n, u_n)^{\varepsilon_n}]_{R_P} x, \\
&= x^{-1} \delta_2[a]_{R_P} x.
\end{aligned}$$

CM2: Let $y, z \in Y$. We first use the basic rules of R_P to verify that $y^+ z^+ y^- \overset{*}{\leftrightarrow}_{R_P} z^{+\delta y^-}$ and $y^- z^- y^+ \overset{*}{\leftrightarrow}_{R_P} z^{-\delta y^+}$.

$$\begin{aligned}
z^{+\delta y^+ y^-} &\overset{*}{\leftrightarrow}_{R_P} (y^+ y^-)^- z^+ (y^+ y^-), \\
&= \quad y^+ y^- z^+ y^+ y^- \\
&\to_{R_P} y^+ z^{+\delta y} y^-.
\end{aligned}$$

Therefore

$$y^+ z^{+\delta y^+} y^- \overset{*}{\leftrightarrow}_{R_P} (z^{+\delta y^+})^{\delta y_-}.$$

So for all $z_1 \in Y$

$$y^+ z_1^+ y^- \overset{*}{\leftrightarrow}_{R_P} z_1^{+\delta y^-}.$$

The other case may be proved in the same way but using the basic rule $y^+ z^- y^- \rightarrow_{R_P} z^{-\delta y^-}$. Therefore the Peiffer relation $y^{-\varepsilon} z^\eta y^\varepsilon \overset{*}{\leftrightarrow}_{R_P} z^{\eta \delta y^\varepsilon}$ holds for all $y^\varepsilon, z^\eta \in (Y^+ \sqcup Y^-)^*$.

Let $a = y_1^{\varepsilon_1} \cdots y_n^{\varepsilon_n}$, $b = z_1^{\eta_1} \cdots z_m^{\eta_m}$. We prove that $[a]_{R_P}^{-1} [b]_{R_P} [a]_{R_P} = [b^{\delta(a)}]_{R_P}$. First note that $[a]_{R_P}^{-1} = [y_n^{-\varepsilon_n} \cdots y_1^{-\varepsilon_1}]_{R_P}$. Now

$$\begin{aligned} y_n^{-\varepsilon_n} \cdots y_1^{-\varepsilon_1} z_1^{\eta_1} \cdots z_m^{\eta_m} y_1^{\varepsilon_1} \cdots y_n^{\varepsilon_n} = &\quad y_n^{-\varepsilon_n} \cdots y_2^{-\varepsilon_2} (y_1^{-\varepsilon_1} z_1^{\eta_1} y_1^{\varepsilon_1}) \cdots (y_1^{-\varepsilon_1} z_m^{\eta_m} y_1^{\varepsilon_1}) y_2^{\varepsilon_2} \cdots y_n^{\varepsilon_n}, \\ &\overset{*}{\leftrightarrow}_{R_P} y_n^{-\varepsilon_n} \cdots y_2^{-\varepsilon_2} z_1^{\eta_1 \delta y_1^{\varepsilon_1}} \cdots z_m^{\eta_m \delta y_1^{\varepsilon_1}} y_2^{\varepsilon_2} \cdots y_n^{\varepsilon_n}. \end{aligned}$$

Repeating the procedure we obtain

$$\begin{aligned} &\overset{*}{\leftrightarrow}_{R_P} z_1^{\eta_1 \delta y_1^{\varepsilon_1} \cdots \delta y_n^{\varepsilon_n}} \cdots z_m^{\eta_m \delta y_1^{\varepsilon_1} \cdots \delta y_n^{\varepsilon_n}}, \\ &= \quad (z_1^{\eta_1} \cdots z_m^{\eta_m})^{\delta(y_1^{\varepsilon_1} \cdots y_n^{\varepsilon_n})}. \end{aligned}$$

Therefore we have verified CM2:-

$$[y_1^{\varepsilon_1} \cdots y_n^{\varepsilon_n}]_{R_P}^{-1} [z_1^{\eta_1} \cdots z_m^{\eta_m}]_{R_P} [y_1^{\varepsilon_1} \cdots y_n^{\varepsilon_n}]_{R_P} = [(z_1^{\eta_1} \cdots z_m^{\eta_m})^{\delta(y_1^{\varepsilon_1} \cdots y_n^{\varepsilon_n})}]_{R_P}.$$

Finally we show that $(C(R), \delta_2)$ is free on $w : \mathcal{R} \rightarrow F(X)$. Recall that $F(X)$ acts on Y by $(\rho, u)^x = (\rho, ux)$. Define $\alpha : \mathcal{R} \rightarrow C(R)$ by $\alpha(\rho) := [(\rho, id)]_{R_P}$. Then let (D, γ) be any other crossed $F(X)$-module with a map $\beta : \mathcal{R} \rightarrow D$. We can define a unique morphism of crossed modules $\phi : C(R) \rightarrow D$ which satisfies $\alpha \circ \phi = \beta$ by putting $\phi([(\rho, u)]_{R_P}) := \beta(\rho)$.

Therefore we have proved that $(C(R), \delta_2)$, as defined on $(Y^+ \sqcup Y^-)^*$ using R_P, is the free crossed $F(X)$-module generated by $w : \mathcal{R} \rightarrow F(X)$. □

Remark 5.2.6 The usual method of construction of $C(R)$ does not use rewriting systems but factors the free precrossed module $(F(Y), \delta')$ by the congruence $=_P$ generated by the set of all Peiffer relations P on $F(Y)$. Detail of this construction are found in [15]. It may be verified that the natural map $\theta : (Y^+ \sqcup Y^-)^* \rightarrow F(Y)$ induces an isomorphism

$$\theta' : \frac{(Y^+ \sqcup Y^-)^*}{\overset{*}{\leftrightarrow}_{R_P}} \longrightarrow \frac{F(Y)}{=_P}.$$

The motivation for this section is to give an exposition of the construction of $C(R)$. Since this thesis is concerned with rewriting, we've presented the exposition in terms of rewriting. It is simply an alternative exposition of standard work that is necessary background for what is to follow.

The Peiffer Problem that we have identified is that of determining whether two Y-sequences represent the same element of $C(R)$. If $a \in (Y^+ \sqcup Y^-)^*$ and $\delta_2(a) = id$ then $[a]_{R_P} \in ker\delta_2$, the module of identities among relations, and a is called an **identity Y-sequence**. There is a special property which will allow us to convert the Peiffer Problem for identity sequences into a Gröbner basis problem, and this will be discussed in Section 6. In general there is no procedure for solving the Peiffer Problem. As a result the example here is a simple one, included to demonstrate the rewriting procedure.

Example 5.2.7 The result of the following example is proved in [18].

The multiplicative cyclic group C_n of order n has a presentation $grp\langle x \mid x^n \rangle$. Let r represent the relator x^n, then $Y := \{(r, x^i) : i \in \mathbb{Z}\}$. with $\delta : (Y^+ \sqcup Y^-)^* \to F(X)$ defined by $\delta(r) = x^n$ so

$$\begin{aligned} \delta_2(r, x^i)^+ &= x^{-i}\delta(r)x^i = x^{-i}(x^n)x^i = x^n. \\ \delta_2(r, x^i)^- &= x^{-i}\delta(r)^{-1}x^i = x^{-i}(x^n)x^i = x^{-n}. \end{aligned}$$

The action of $F(X)$ on $(Y^+ \sqcup Y^-)^*$ is given by

$$(r, x^i)^x = (r, x^{i+1}).$$

The elements of $Y^+ \sqcup Y^-$ can be denoted a_i, A_i $i \in \mathbb{Z}$ where $a_i := (r, x^i)^+$, $A_i := (r, x_i)^-$. We consider the rewriting system R_P on $(Y^+ \sqcup Y^-)^*$ given by:

$$\{(A_i a_j a_i, a_j^{\delta a_i}) : i, j \in \mathbb{Z}\} \cup \{(a_i A_j A_i, A_j^{\delta A_i}) : i, j \in \mathbb{Z}\} \cup \{(a_i A_i, \mathit{id}) : i \in \mathbb{Z}\} \cup \{(A_i a_i, \mathit{id}) : i \in \mathbb{Z}\}$$

The rewriting system is clearly infinite. Put $i = j$ in the above rules and we obtain $A_i a_i a_i \leftrightarrow_{R_P} a_{i+n}$ and $a_i A_i A_i \leftrightarrow_{R_P} A_{i-n}$. So $a_{i+n} \to_{R_P} a_i$ and $A_i \to_{R_P} A_{i-n}$ for all $i \in \mathbb{Z}$. It follows immediately from these rules that $\{a_0, \ldots, a_{n-1}, A_0, \ldots, A_{n-1}\}$ is a complete set of generators for $C(R)$ as a monoid. The now finite set of relations is $\{(a_i A_i, \mathit{id}), (A_i a_i, \mathit{id}), (A_i a_j a_i, a_j), (a_i A_j A_i, A_j)\}$ Therefore $C(R)$ for C_n is the free abelian group on n generators $a_0, a_1, \ldots, a_{n-1}$. Further, we find that $a_i^x = a_{i+1}$ for $i = 0, \ldots, n-1$ and $a_{n-1}^x = a_0$. Thus the $C(R)$, which is a C_n-module is isomorphic to $\mathbb{Z}[\mathsf{C}_n]$, the free C_n-module on one generator.

Remark 5.2.8 The Peiffer Problem (of deciding when two sequences are Peiffer equivalent) does not arise only in crossed modules. When a 2-category is constructed, by factoring a sesquicategory (see [74, 76]) by the interchange law, the pairs arising from that interchange law are relations among the two cells involving the whiskering action of the category morphisms. Tim Porter identified this in [62] calling them Peiffer pairs. Thus the Peiffer Problem is not restricted to the construction of crossed modules.

5.3 Free Crossed Resolutions of Groups

The following exposition was constructed with Ronnie Brown.

The notion of resolution of $\mathbb{Z}G$-modules for G a group is a standard part of homological algebra and the cohomology of groups [27, 10]. It has been shown in [18, 16, 17] that there are computational advantages in considering free *crossed* resolutions of groups. This will be confirmed by bringing these calculations into the context of rewriting procedures. For this we need to give some basic definitions in the form we require.

An important aspect of the calculation in [17] is the use of the Cayley graph, being seen here as data for a free crossed resolution of the universal covering groupoid $\widetilde{G}$ of the group G. This groupoid corresponds to the action of G on itself by right multiplication. That is, the objects of $\widetilde{G}$ are the elements of G and an arrow of $\widetilde{G}$ is a pair $(g_1, g_2) : g_1 \to g_1 g_2$, with the obvious composition. We have the covering morphism of groupoids $p_0 : \widetilde{G} \to G : (g_1, g_2) \mapsto g_2$.

If X is a set of generators of the group G, we have a standard morphism $\theta : F(X) \to G$. We also have a standard morphism $\widetilde{\theta} : F(\widetilde{X}) \to \widetilde{G}$. Here

i) $\widetilde{X}$ is the Cayley graph of (X, G) with arrows $[g, x] : g \to g\theta(x)$ for $x \in X, g \in G$.

ii) $F(\widetilde{X})$ is the groupoid with objects again the elements of G and arrows pairs $[g,u] : g \to g(\theta u)$ for $g \in G$, $u \in F(X)$, with composition defined by $[g,u][g(\theta u),v] := [g,uv]$. In fact $F(\widetilde{X})$ is the free groupoid on the graph $\widetilde{X}$, so that a morphism f from $F(\widetilde{X})$ to a groupoid is determined by the graph morphism $f|_{\widetilde{X}}$.

Then $\widetilde{\theta} : F(\widetilde{X}) \to \widetilde{G}$ is given on arrows by $\widetilde{\theta}[g,u] := [g,\theta(u)]$. There is also the covering morphism $p_1 : F(\widetilde{X}) \to F(X)$ given by $p_1[g,u] := u$. This gives the commutative diagram of morphisms of groupoids

$$\begin{tikzcd}
F(\widetilde{X}) \arrow[r, "\widetilde{\theta}"] \arrow[d, "p_1"'] & \widetilde{G} \arrow[d, "p_0"] \\
F(X) \arrow[r, "\theta"'] & G
\end{tikzcd} \tag{5.1}$$

In fact this diagram is a pullback in the category of groupoids. Also, p_1 maps $F(\widetilde{X})(1,1)$ isomorphically to $ker\theta$, and $F(\widetilde{X})$ is the free groupoid on the graph $\widetilde{X}$.
Now let $\mathcal{P} = grp\langle X|R\rangle$ be a presentation of G. As explained in the previous section, this gives rise to a free crossed $F(X)$-module $\delta_2 : C(R) \to F(X)$, whose kernel is $\pi_2(\mathcal{P})$, the $\mathbb{Z}G$-module of identities among relations. The aim is to compute a presentation for this module in terms of information on the Cayley graph. For this we extend diagram 5.1 in the first instance to

$$\begin{tikzcd}
C(\widetilde{R}) \arrow[r, "\tilde{\delta}_2"] \arrow[d, "p_2"'] & F(\widetilde{X}) \arrow[r, "\widetilde{\theta}"] \arrow[d, "p_1"'] & \widetilde{G} \arrow[d, "p_0"] \\
C(R) \arrow[r, "\delta_2"] & F(X) \arrow[r, "\theta"] & G
\end{tikzcd} \tag{5.2}$$

Here $\tilde{\delta}_2 : C(\widetilde{R}) \to F(\widetilde{X})$ is a free crossed module of groupoids. For details, we refer the reader to [17]. All the reader needs to know for now is that

i) $C(\widetilde{R})$ is a disjoint union of groups $C(\widetilde{R})(g)$ for $g \in G$ and $\tilde{\delta}_2$ maps $C(\widetilde{R})(g)$ to $F(\widetilde{X})(g,g)$.

ii) for each $g \in G$, p_2 maps the group $C(\widetilde{R})(g)$ isomorphically to $C(R)$, so that elements of $C(\widetilde{R})(g)$ are specified by pairs $[g,c]$ where $c \in C(R)$.

iii) $F(\widetilde{X})$ operates on $C(\widetilde{R})$ by $[g,c]^{[g,u]} := [g\theta(u), c^u]$ for $g \in G$, $c \in C(R)$, $u \in F(X)$.

iv) The morphisms $\tilde{\delta}_2$, p_2 are given by $\tilde{\delta}_2[g,c] := [g,\delta_2 c]$ and $p_2[g,c] := c$.

A proof that $\tilde{\delta}_2 : C(\widetilde{R}) \to F(\widetilde{X})$ is the free crossed $F(\widetilde{X})$-module on $\widetilde{R} := G \times R$ is given in [17]. This implies that morphisms and homotopies on $C(\widetilde{R})$ can be defined by their values on the elements $[g,r]$ for $g \in G$, $r \in R$.

The key feature of this construction is that $\widetilde{G}$ is a contractible groupoid, i.e. it is connected and has trivial vertex groups. We are going to construct a partial contracting homotopy of $\tilde{\delta}_2 : C(\widetilde{R}) \to F(\widetilde{X})$. This is a key part of the procedure of constructing generators (and then relations) for $\pi_2(\mathcal{P})$. The philosophy as stated in [17] is to construct a "home" for a contracting homotopy – this will be explained later. The point is that this leads to a "tautological" proof that the generators constructed do in fact generate $\pi_2(\mathcal{P})$.

Such a partial contracting homotopy consists of functions

$$h_0 : G \to F(\widetilde{X}) \quad \text{and} \quad h_1 : F(\widetilde{X}) \to C(\widetilde{R})$$

with the properties that

i) $h_0(g) : g \mapsto id$ in $F(\widetilde{X})$, $g \in G$.

ii) h_1 is a morphism (from a groupoid to a group).

iii) $\tilde{\delta}_2 h_1[g,u] = (h_0 g)^{-1}[g,u]h_0(g(\theta u))$ for all $[g,u] \in F(\widetilde{X})$.

We always assume that $h_0(id) = id \in F(\widetilde{X})(id)$

Remark 5.3.1 h_0 and h_1 are related to what are commonly called 0- and 1-combings of the Cayley graph [39]. We hope to pursue this elsewhere.

The choice of h_0 is equivalent to choosing a section σ of $\theta : F(X) \to G$, i.e. a representative word for each element of G, by $h_0(g) = [g, \sigma(g)^{-1}]$, for $g \in G$. What h_1 does is provide for each word $u \in F(X)$ a representation

$$u = \delta_2(\texttt{proc}_R(u))N_R(u)$$

where $\texttt{proc}(u) = p_2 h_1[id, u] \in C(R)$ – the procedure through which the normal form $N_R(u) := (\sigma\theta(u))^{-1}$ is reached. To verify this consider (iii), assuming $h_0(id) = id$, we have

$$\tilde{\delta}_2 h_1[id, u] = [id, u]h_0(\theta u).$$

Then

$$\begin{aligned} \delta_2(proc(u)) &= \delta_2 p_2 h_1[id, u] \\ &= p_1 \tilde{\delta}_2 h_1[id, u] \\ &= p_1([id, u]h_0(\theta u)) \\ &= u p_1 h_0(\theta u) \end{aligned}$$

Thus $\texttt{proc}(u)$ shows how to write $u(N_R(u))^{-1} \in \delta_2 C(R)$ as a consequence of the relators R. Conversely, a rewriting procedure to be given later will allow us to determine h_1 given h_0 and a complete rewriting system for $\mathcal{P} = grp\langle X|R\rangle$.

We can now state

Proposition 5.3.2 *Given h_0, h_1 as above, the module $\pi_2(\mathcal{P})$ is generated by the (separation) elements*

$$sep(g,r) := p_2(h_1\tilde{\delta}_2[g,r])^{-1} r^{\sigma(g)^{-1}} \tag{5.3}$$

for all $g \in G$, $r \in R$.

Outline proof The fact that the elements $sep(g,r)$ of 5.3 are identities among relations is easily checked, as follows:

$$\begin{aligned} \delta_2(p_2(h_1\tilde{\delta}_2[g,r])^{-1} r^{\sigma(g)^{-1}}) &= \delta_2(p_2(h_1[id, \delta_2(r^g)])^{-1} r^{\sigma(g)^{-1}}) \\ &= \delta_2(p_2([id, c])^{-1} r^{\sigma(g)^{-1}}) \text{ where } c \text{ satisfies } \delta_2(c) = \delta_2(r^{\sigma(g)^{-1}}), \\ &= \delta_2(c)^{-1}\delta_2(r^{\sigma(g)^{-1}}) \\ &= id. \end{aligned}$$

The important point is that these elements $sep(g,r)$ generate the *module* of identities. The proof of this can be made tautologous by taking the construction one step further, i.e.

$$\begin{tikzcd} \widetilde{C}_3 \arrow[r, "\tilde{\delta}_3"] \arrow[d, "p_3"'] & C(\widetilde{R}) \arrow[r, "\tilde{\delta}_2"] \arrow[d, "p_2"] & F(\widetilde{X}) \arrow[r, "\tilde{\theta}"] \arrow[d, "p_1"] & \widetilde{G} \arrow[d, "p_0"] \\ C_3 \arrow[r, "\delta_3"] & C(R) \arrow[r, "\delta_2"] & F(X) \arrow[r, "\theta"] & G \end{tikzcd}$$

Here C_3 is the free $\mathbb{Z}G$-module on $(g,r) \in \bar{R}$ where $\bar{R} := G \times R$ – we use round brackets to distinguish elements of $\bar{R}$ from those of $\widetilde{R}$. The morphism δ_3 is defined by

$$\delta_3(g,r) := p_2((h_1\tilde{\delta}_2[g,r])^{-1})r^{\sigma(g)^{-1}}.$$

The definition is verified by checking that $\delta_2\delta_3(g,r) = id$ i.e.

$$\begin{aligned}\delta_2\delta_3(g,r) &= \delta_2 p_2((h_1\tilde{\delta}_2[g,r])^{-1}r^{\sigma(g)^{-1}})\\ &= \delta_2(c^{-1}r^{\sigma(g)^{-1}}) \text{ where } c \text{ satisfies } \delta_2(c) = \delta_2(r^{\sigma(g)^{-1}})\\ &= id.\end{aligned}$$

(Mapping a free $\mathbb{Z}G$-module into a free crossed G-module, is acceptable because the image lies in $ker\delta_2$ which is a $\mathbb{Z}G$-module.) In fact we define $\widetilde{C}_3$, h_2 and $\tilde{\delta}_3$ as follows

$$\begin{aligned}\widetilde{C}_3(g) &:= \{g\} \times C_3,\\ h_2[g,r] &:= (id,(g,r)),\\ \tilde{\delta}_3(g_2,[g_1,r]) &:= (g_2,\delta_3(g_1,r)).\end{aligned}$$

We now check directly that

$$\begin{aligned}\tilde{\delta}_3 h_2[g,r] &= [id,\delta_3(g,r)],\\ &= [id, p_2((h_1\tilde{\delta}_2[g,r])^{-1})r^{\sigma(g)^{-1}}],\end{aligned}$$

so that

$$= (h_1(\delta_1[g,r]))^{-1}r^{\sigma(g)^{-1}}.$$

In the partial resolution of $\widetilde{G}$ we have, for any $c \in C(\widetilde{R})$,

$$\tilde{\delta}h_2(c) = (h_1(\tilde{\delta}_2 c))^{-1}c^{h_0 id},$$

since this holds for all $c = [g,r] \in \widetilde{R}$. So

$$\tilde{\delta}_2(c) = 0 \text{ implies that } c = \tilde{\delta}_3((h_2c)^{(h_0 id)^{-1}}).$$

Hence $ker\tilde{\delta}_2 \subseteq im\tilde{\delta}_3$, so $ker\tilde{\delta}_2 = im\tilde{\delta}_3$. Therefore $ker\delta_2 = im\delta_3$. □

To summarise: the problem of constructing a crossed resolution of a group given a particular presentation has been reduced to the problem of constructing a contracting homotopy and a covering crossed complex that begins with a groupoid defined on the Cayley graph.

5.4 Completion Procedure and Contracting Homotopies

In this section we define what we call an "extra information completion procedure". The implementation may be found in *kb2.g*. Input to the procedure is a set of relators for a group. If the procedure terminates then the output is a set of "extra information" rules. These rules will not only reduce any word in the free group to a unique irreducible but will express the actual reduction in terms of the original relators.

Definition 5.4.1 *An **extra information rewriting system** for a group presentation $grp\langle X|R\rangle$ is a set of triples $R2 := \{(l_1, c_1, r_1), \ldots, (l_n, c_n, r_n)\}$, where $R1 := \{(l_1, r_1), \ldots, (l_n, r_n)\}$ is a rewriting system on $F(X)$ and $c_1, \ldots, c_n \in C(R)$, such that $l_i = \delta_2(c_i)r_i$ for $i = 1, \ldots, n$. We say $R2$ is **complete** if $R1$ is complete.*

Lemma 5.4.2 *Let $R2$ be a complete EIRS for $grp\langle X|R\rangle$. Then for any $w \in F(X)$ there exists (c, z), $c \in C(R)$, $z \in F(X)$ such that z is irreducible with respect to $\rightarrow_{R1}$, and $w = (\delta_2 c)z$.*

Proof If w is irreducible then we take $z = w$ and $c = id_{C(R)}$. Otherwise there is a sequence of reductions

$$\begin{aligned} w &= u_1 l_1 v_1 \\ u_1 r_1 v_1 &= u_2 l_2 v_2 \\ \cdots & \quad \cdots \\ u_n r_n v_n &= z \end{aligned}$$

where $n \geq 1$, and for $i = 1, \ldots, n$, $u_i, v_i \in F(X)$ and there exists $c_i \in C(R)$ such that $(l_i, c_i, r_i) \in R2$. Then since $l_i = (\delta_2 c_i)\, r_i$ for $i = 1, \ldots, n$

$$\begin{aligned} w &= u_1\,(\delta_2 c_1)\,r_1 v_1 \\ u_1 r_1 v_1 &= u_2\,(\delta_2 c_2)\,r_2 v_2 \\ \cdots & \quad \cdots \\ u_n r_n v_n &= z. \end{aligned}$$

Hence $w = ((\delta_2 c_1)^{u_1^{-1}} \cdots (\delta_2 c_n)^{u_n^{-1}})z$. □

This defines the function `ReduceWord2`, which accepts as input $(w, R2)$ and returns as output (c, z). We will write $w \rightarrow_{R2} (c, z)$.

Lemma 5.4.3 *Let $grp\langle X|R\rangle$ be a finite group presentation which is completable with respect to an ordering $>$. Then there exists a procedure* `KB2` *which will return the complete EIRS for the group.*

Proof Define $R2 := \{(\delta\rho, (\rho, id), id) : \rho \in R\}$. It is clear that this defines an EIRS since $\delta\rho = \delta_2(\rho, id)id$. If $R1$ is complete then $R2$ is complete. If $R1$ is not complete then there is an overlap between a pair of rules $(l_1, r_1), (l_2, r_2)$ of $R1$ where $(l_1, c_1, r_1), (l_2, c_2, r_2) \in R2$. There are two cases to consider.
For the first case suppose $u_1 l_1 v_1 = l_2$ for some $u_1, v_1 \in F(X)$. Then the critical pair resulting from the overlap is $(u_1 r_1 v_1, r_2)$. Reduce each side of the pair using `ReduceWord2`, so $u_1 r_1 v_1 \rightarrow_{R2} (d_1, z_1)$ and $r_2 \rightarrow_{R2} (d_2, z_2)$. Then if $z_1 > z_2$ add the extra information rule $(z_1, d_1^{-1} c_1{}^{-u_1^{-1}} c_2 d_2, z_2)$ or if $z_2 > z_1$ add $(z_2, d_2^{-1} c_2^{-1} c_1^{u_1^{-1}} d_1, z_1)$.
For the second case suppose $u_1 l_1 = l_2 v_2$ for some $u_1, v_2 \in F(X)$. Then the critical pair resulting from the overlap is $(u_1 r_1, r_2 v_2)$. Reduce each side of the pair by $R2$ as before, so that $u_1 r_1 \rightarrow_{R2} (d_1, z_1)$ and $r_2 v_2 \rightarrow_{R2} (d_2, z_2)$. Then if $z_1 > z_2$ add the extra information rule $(z_1, d_1^{-1} c_1^{-u_1^{-1}} c_2 d_2, z_2)$ or if $z_2 > z_1$ add $(z_2, d_2^{-1} c_2^{-1} c_1^{u_1^{-1}} d_1, z_1)$.
It can be seen immediately from the above that the effect on $R1$ is a standard completion of the rewriting system, and that the triples (l, c, r) added to $R2$ satisfy the requirement $l = \delta_2(c)r$, so that when the completion procedure terminates $R2$ will be a complete extra information rewriting system. □

This defines the procedure `KB2`.

Example 5.4.4 $Q8$ is presented by $grp\langle a, b \mid a^4, b^4, abab^{-1}, a^2b^2\rangle$. Let r, s, t and u denote the relators i.e. $\delta(r) = a^4, \delta(s) = b^3, \ldots$. We begin with the EIRS

$$R2 := \{(a^4, r, id),\ (b^4, s, id),\ (aba, t, b),\ (a^2b^2, u, id)\}.$$

As explained before, all the extra information rules are triples (l, c, r) such that $l = (\delta_2 c)r$ and we write $l \rightarrow_{R2} (c, r)$, thinking of the (c) part as the record of the procedure by which r is obtained from l using the original group relators. For example $aba \rightarrow_{R2} (t, b)$ – we have to work with a monoid presentation and choose to make use of the fact that $Q8$ is finite, rather than introduce generators for the inverses, which is what the computer program does. We look for overlaps between the left hand sides of the rules. The first overlap we examine is between the first and third rules:

$$\begin{tikzcd}
 & a^4ba \arrow[dl, "a^4\to id"'] \arrow[dr, "aba\to b"] & \\
ba & & a^3b \arrow[ll, dashed]
\end{tikzcd}$$

Without the extra information the critical pair is (a^3b, ba) and the new rule is $a^3b \to ba$. For the EIRS rule we need c so that $a^3b = \delta_c(c)ba$ where c is a product of conjugates of relators. The new EIRS rule as defined in the proof (second case) is $(a^3b, t^{-a^{-3}}r, ba)$. This is checked by:

$$a^4ba = (a^4)ba \rightarrow_{R2} (r, id)ba = (r, ba) \ \mathit{and}\ a^4ba = a^3(aba) \rightarrow_{R2} a^3(t, b) = (t^{a^{-3}}, a^3b).$$

Therefore $\delta_2(r)ba = \delta_2(t^{a^{-3}})a^3b$, so $a^3b = \delta_2(t^{a^{-3}})^{-1}(r)ba = \delta_2(t^{-a^{-3}}r)ba$. so $c = t^{-a^{-3}}r$. If we continue this "extra information completion" for $Q8$ we end up with the EIRS

$$\begin{aligned}
b^2 &\rightarrow_{R2} (r^{-1}u^{a^{-2}}, a^2),\\
aba &\rightarrow_{R2} (t, b),\\
ba^2 &\rightarrow_{R2} (t^{-1}t^{-a^{-1}}r^{b^{-1}a^{-2}}, a^2b),\\
bab &\rightarrow_{R2} (r^{-b^{-1}a^{-1}b^{-1}}t^{b^{-1}}r^{-1}u^{-a^{-2}}r, a),\\
a^4 &\rightarrow_{R2} (r, id),\\
a^3b &\rightarrow_{R2} (t^{-a^{-3}}r, ba).
\end{aligned}$$

So, for example, a^5ba^3 reduces to a^2b and $a^5ba^3 = \delta_2(rt^{-a^{-1}}r^{b^{-1}a^{-2}})a^2b$.

The "extra information" Knuth-Bendix procedure `KB2` results in a rewriting system with information on where the rules came from. This extra information is in no way unique.

Let $grp\langle X|R\rangle$ be a presentation of a group G. Let $\widetilde{X}$ denote the Cayley graph. Edges of the graph are recorded as pairs $[g, x]$, where g is the group element identified with the source vertex, and x is a group generator identified with the edge label.

Lemma 5.4.5 (Complete Rewriting Systems Determine h_0)
Let G be a finite group, finitely presented by $grp\langle X|R\rangle$, with quotient morphism $\theta : F(X) \to G$. Then a complete rewriting system $R1$ for the presentation determines $h_0 : G \to F(\widetilde{X})$.

Proof Let N be the normal form function defined by $\rightarrow_{R1}$ on $F(X)$. Define $h_0(g) := [id, N(g)]^{-1}$. Then $h_0(g) : g \to id$ in $F(\widetilde{X})$ as required. □

Theorem 5.4.6 (Complete EIRS's Determine h_1)
Let G be a finite group, finitely presented by $grp\langle X|R\rangle$, with quotient morphism $\theta : F(X) \to G$. Then a complete EIRS $R2$ for the presentation determines $h_1 : F(\widetilde{X}) \to C(\widetilde{R})(id)$.

Proof Recall that $\widetilde{X}$ is the Cayley graph of G. Let $[g, x] \in \widetilde{X}$. Define

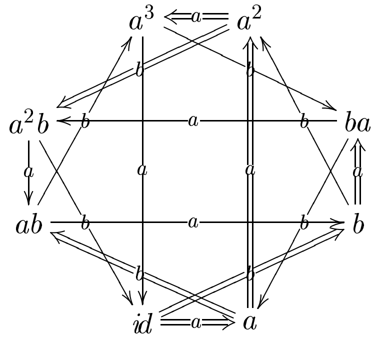
$$h_1[g,x] := [id, \texttt{ReduceWord2}(N(g)xN(g\theta x)^{-1}, R2)[1]].$$

Then clearly $h_1[g,x] \in C(\widetilde{R})(id)$ and $\widetilde{\delta}_2 h_1[g,x] = [id, N(g)][g,x][id, N(g\theta x)]^{-1} = h_0(g)^{-1}[g,x]h_0(g\theta x)$. Extending this definition of h_1 on $\widetilde{X}$ therefore gives the morphism h_1 of the groupoid $F(\widetilde{X})$ to the group $C(\widetilde{R})(id)$ satisfying the required conditions. □

Corollary 5.4.7 *There exists an algorithm for defining h_0, h_1 for any finite completable group presentation $grp\langle X|R\rangle$.*

Proof Calculate $R2$, using `KB2`. Let N be the normal form function defined by $\to_{R1}$ (recall $R1$ is part of $R2$). Put $h_0(g) := [id, N(g)]^{-1}$. Put $h_1[g,x] := [id, \texttt{ReduceWord2}(N(g)xN(g\theta x)^{-1}, R2)[1]]$. □

Example 5.4.8 Below is the Cayley graph for $Q8$. The double edges indicate the tree defined by the length-lex ordering.



A typical relator cycle is $[id,b][b,a][ba,a][a^2b,a][ab,a][id,b]^{-1}$ this is equivalent to ba^4b^{-1} or $r^{b^{-1}}$, the cycles represent conjugates of relators in the graph.

The extra information in our rewriting system may be used to express the cycle created by adding an edge α to the tree as such a product, or in fact to express its retraction as a product of conjugates of relators. For example, add the edge $[a^3, a]$ and the cycle $[id,a][a,a][a^2,a][a^3,a]$ is created. The retraction is a^4. We know that $\delta_2(r) = a^4$, so $[id, r]$ is the cycle as a product of relator cycles.

That was an easy example. If we add the edge $[a^3, b]$ then the retraction is $a^3ba^{-1}b^{-1}$ or $a^3b(a^3)(a^2b)$, (since the rewriting system is defined on the monoid presentation we replace inverse elements by their normal forms). It is more difficult to see how this word may be written as a product of conjugates of relators. In fact we just reduce it using the extra information rules :

$$\begin{aligned}
a^3ba^5b &\to_{R2} a^3ba(r,id)b \\
&\to_{R2} a^3bab(r^b,id) \\
&\to_{R2} a^3((r^{-b^{-1}a^{-1}b^{-1}}t^{b^{-1}}r^{-1}u^{-a^{-2}}r,a)(r^b,id) \\
&\to_{R2} a^4(r^{-b^{-1}a^{-1}b^{-1}}t^{b^{-1}}r^{-1}u^{-a^{-2}}r)^a,id)(r^b,id) \\
&\to_{R2} (r(r^{-b^{-1}a^{-1}b^{-1}}t^{b^{-1}}r^{-1}u^{-a^{-2}}r)^a r^b,id)
\end{aligned}$$

The order in which the rules are applied does not matter for our purposes – it does affect the answer but we only wish to find a representation of the word as a product of conjugates of relators, which representation it is is not important – though smaller ones are preferable for efficiency reasons. The list below gives the cycles created by adding in non-tree edges as products of relator cycles.

$$\begin{array}{llll}
\underline{[g,x]} & \mapsto \underline{h_1[g,x]} & & \\
[b,b] & \mapsto bb(a^2)^{-1} & \to b^2a^{-2} & \to su^{-1}, \\
[ab,a] & \mapsto ab(b)^{-1} & \to aba^2b & \to t, \\
[ab,b] & \mapsto ab^2(a^3)^{-1} & \to ab^2a & \to u^ar^{-1}, \\
[ba,a] & \mapsto ba^2(a^2b)^{-1} & \to ba^4ba^6 & \to su^{-1}u^{ba^{-2}}s^{-a^{-2}}, \\
[ba,b] & \mapsto bab(a)^{-1} & \to baba^3 & \to t^{-a^{-1}b^{-1}}u^{b^{-1}}, \\
[a^3,a] & \mapsto a^4(id)^{-1} & \to a^4 & \to r, \\
[a^3,b] & \mapsto a^3b(ba)^{-1} & \to a^3ba^5b & \to rt^{-a}, \\
[a^2b,a] & \mapsto a^2ba(ab)^{-1} & \to a^2ba^3ba^3 & \to t^{a^{-1}}, \\
[a^2b,b] & \mapsto a^2b^2(id)^{-1} & \to a^2b^2 & \to u.
\end{array}$$

This example gives 32 generators for the module of identities. In fact this can be reduced to 7 but the reduction requires methods not dealt with in this thesis.

5.5 Algorithm for Computing a Set of Generators for π_2

Section 5.3 described how the problem of specifying a free crossed resolution of a group reduced to the problem of defining a contracting homotopy of a covering crossed complex.

The computation of a complete rewriting system for the group is used to define the first part of the contracting homotopy h_1 on the edges of the Cayley graph. The formulae from the definition of the covering crossed complex are used to find a complete set of generators for the kernel of δ_2 (the identities among relations). The pre-images of these elements generate C_3 as a $\mathbb{Z}G$-module. By reducing this set of generators and writing each of the reducible generators in terms of the irreducible ones we define h_2 on the generators of C_2. This is made clear in the example, and is the part which corresponds to the Gröbner basis computation, though we do it by inspection.

Now the crossed complex formulae with h_2 are used to find a complete set of generators for the kernel of δ_3 (the identities among identities). Again, we reduce the set of identities, so that their pre-images freely generate C_4 as a $\mathbb{Z}G$-module. The process of reduction of the identities defines the next contracting homotopy h_3, and again we use the formulae to find a complete set of generators for $ker\delta_4$, and reduction to a set whose pre-image freely generates C_5 as a $\mathbb{Z}G$-module.

This procedure may in theory be repeated as much as is wished, in order to compute the resolution of the group up to any level. The limitations are ones of practicality: in our example the reduction of the set of identities is done by inspection (involving a lot of trial and error) this takes time (weeks). A Gröbner basis procedure (over the group ring) would provide a computerisable method for defining h_n, and this would mean that the computation of the resolution was limited only by the computer's capacity. The correspondence between the homotopy definition and the Gröbner basis computation (for reduction) is explained more fully in the next section.

5.5.1 Specification of the Program

A collection of GAP3 functions has been written to perform these calculations and will be rewritten in GAP4 and submitted as a share package. The function `IdRel1` accepts as input a free group and a list of relators. It goes through a number of calculations, including an "extra information" Knuth-Bendix completion procedure and returns a complete set of generators for the module of identities among relations. The structure of the program `idrel.g` is outlined below.

Preliminary functions necessary are:

`ReduceWord`$(word, R1)$: reduces a word with respect to a rewriting system $R1$, in the standard way.

`ReduceWord2`$(word, R2)$: applies an EIRS $R2$ to a word and reduces it as far as possible within that system. Output is a pair $[c, w]$ where $word = \delta_2(c)w$, where c is a Y-sequence.

`InverseYsequence`(a): Y-sequences are represented by lists $a = [s_1, u_1], \dots, [s_n, u_n]$ where $u_i \in F$ and s_i is a relator or an inverse of a relator. This function inverts such a sequence to $[s_n, u_n], \dots, [s_1, u_1]$. This is used to invert products of conjugates of relators which are represented as Y-sequences.

`KB2`$(R2)$: is an implementation of the "extra information" Knuth-Bendix procedure described in Section 4. The input rules are in the form of lists of length three where the middle entry represents the product of conjugates of relators $(r_1, u_1)^{\varepsilon_1} \cdots (r_n, u_n)^{\varepsilon_n}$ as a Y-sequence $[[r_1^{\varepsilon_1}, u_1], \dots, [r_n^{\varepsilon_n}, u_n]]$. The output rules will have the same form. If $[l, c, r]$ is a rule in such a system then $l \to r$ and $l = \delta_2(c)r$.

Given a presentation $grp\langle X|relts\rangle$, define $F := F(X)$. The main function is:

`IdRel1`$(F, relts)$. First G is defined to be the quotient of the free group F by the relators $relts$. Let $\theta : F \to G$ be the quotient morphism. It is necessary to keep track of whether an element is in G or F. The next step is to construct the initial EIRS from the relators. The program uses the monoid presentation of the group to enable it to accept relators containing inverses without changing them. The resulting EIRS is then completed using `KB2` to obtain $K2$. The analogous ordinary system is $K1$. The Cayley graph is represented by a list of edges, which are pairs $[g, x]$ where g is an irreducible in F and x is a generator. The so-called alpha-edges are the edges not in the spanning tree given by the length-lex order. The map h_1 is defined on these alpha-edges by $h_1[g, x] := [id, \texttt{ReduceWord2}(N(g)x, K2)]$ and we apply p_2 immediately, so recording only the second part of this pair. To obtain the identities among relations all relator cycles in the Cayley graph must be considered. These are recorded as pairs $[g, r]$ where g is a vertex and r is a relator. The boundary $\tilde{\delta}_2$ of the cycle is basically found by splitting up the relator r to obtain a list of edges. Non-alpha edges are removed since h_1 maps any edge of the tree to id. The remaining edges of each cycle are identified with their images under p_2h_1. The identities are calculated by manipulating the information held so as to obtain a representation of $p_2(h_1\tilde{\delta}_2[g, r])^{-1}r^{\sigma(g)^{-1}}$ for each $[g, r]$ pair.

The output is in the form of a record `id1` (say) with the following fields:

`id1.free` the free group F;

`id1.rels` the relators $relts$;

`id1.elF` the normal forms of the group elements;

`id1.K` the (ordinary) completed rewriting system;

`id1.idents` the generating set of identities among relations;

`id1.isIdsRecord` true – a check that the identities generated all have the image id.

A small example is printed here – others are on disk in files `idreleg1.g` to `idreleg3.g`. If `IdRelPrintLevel` is set to be greater than 1 (up to 3) information on the progression through the program is printed to the screen.

```
gap> Read("idrel.g");
gap> IdRelPrintLevel:=1;;
gap> F:=FreeGroup("a","b");;
gap> a:=F.1;;b:=F.2;;
gap> R:=[a^3,b^2,a*b*a*b];;
gap> id1:=IdRel1(F,R);;
gap> id1.idents;
[ [ [ r1-1, IdWord ], [ r1^-1, IdWord ] ],
  [ [ r1^-1, IdWord ], [ r1, a^-1 ] ],
  [ [ r3^-1, IdWord ], [ r2, a^-1*b^-1*a^-1 ],
    [ r2^-1, IdWord ], [ r1^-1, b^-1 ], [ r3, a^-2*b^-1 ], [ r1, b^-1 ] ],
  [ [ r1^-1, IdWord ], [ r1, a^-2 ] ],
  [ [ r2^-1, IdWord ], [ r1^-1, b^-1 ], [ r3, a^-2*b^-1 ],
      [ r3^-1, IdWord ], [ r2, a^-1*b^-1*a^-1 ], [ r1, b^-1*a^-1 ] ],
  [ [ r3^-1, IdWord ], [ r2, a^-1*b^-1*a^-1 ],
      [ r2^-1, IdWord ], [ r1^-1, b^-1 ], [ r3, a^-2*b^-1 ],
      [ r1, a^-1*b^-1 ] ], [ [ r2^-1, IdWord ], [ r2, IdWord ] ],
  [ [ r2^-1, a^-1 ], [ r2, a^-1 ] ],
  [ [ r2^-1, IdWord ], [ r2, b^-1 ] ],
  [ [ r3^-1, a^-2 ], [ r1, IdWord ], [ r2^-1, a^-1*b^-1 ],
  [ r1^-1, IdWord ], [ r3, a^-2 ], [ r2, a^-2 ] ],
  [ [ r2^-1, a^-1 ], [ r2, b^-1*a^-1 ] ],
  [ [ r2^-1, a^-1*b^-1 ], [ r1^-1, IdWord ], [ r3, a^-2 ],
      [ r3^-1, a^-2 ], [ r1, IdWord ], [ r2, a^-1*b^-1 ] ],
  [ [ r2^-1, IdWord ], [ r3^-1, IdWord ], [ r2, a^-1*b^-1*a^-1 ], [ r3, IdWord ] ],
  [ [ r2^-1, a^-1 ], [ r2^-1, IdWord ], [ r1^-1, b^-1 ], [ r3, a^-2*b^-1 ],
      [ r2^-1, a^-1*b^-1 ], [ r1^-1, IdWord ], [ r3, a^-2 ], [ r3, a^-1 ] ],
  [ [ r1^-1, IdWord ], [ r3^-1, a^-2 ], [ r1, IdWord ], [ r3, b^-1 ] ],
  [ [ r3^-1, a^-2 ], [ r1, IdWord ], [ r1^-1, IdWord ], [ r3, a^-2 ] ],
  [ [ r2^-1, IdWord ], [ r3^-1, IdWord ], [ r2, a^-1*b^-1*a^-1 ], [ r3, b^-1*a^-1 ] ],
  [ [ r2^-1, a^-1*b^-1 ], [ r1^-1, IdWord ], [ r3, a^-2 ], [ r2^-1, a^-1 ],
    [ r2^-1, IdWord ], [ r1^-1, b^-1 ], [ r3, a^-2*b^-1 ], [ r3, a^-1*b^-1 ] ] ]
gap>
```

The program returns a set of 18 generators for $ker\delta_2$, these are the images under δ_3 of a set of generators for $\widetilde{C}_3$. For the output of higher stages to be useful implementation of some Gröbner basis procedures will be necessary. This is discussed in Section 6.

Example 5.5.1 We now present the results obtained for S_3 followed by some of the details of the calculations which can be done by hand in this case, beginning with the presentation

$$G := grp\langle x, y \mid x^3, y^2, (xy)^2\rangle.$$

The description of the partial free crossed resolution is as follows. Let $X = \{x, y\}$ and define $\mathcal{R}$ to be the set of relator labels $\{r, s, t\}$ whose images under w are

$$\{x^3, y^2, (xy)^2\}.$$

C_2 is the free crossed $F(X)$-module on $w : \mathcal{R} \to F(X)$.
C_3 is the free $\mathbb{Z}G$-module generated by four elements $\{\iota_1, \dots, \iota_4\}$ whose images under δ_3 generate $ker\delta_2$ and are

$$\{r^{-1}r^{x^{-1}},\ s^{-1}s^{y^{-1}},\ t^{-1}t^{y^{-1}x^{-1}},\ \ ts^{-xy}r^{-y}s^{-1}t^{x}s^{-x}r^{-1}t^{x^{-1}}\}.$$

C_4 is the free $\mathbb{Z}G$-module generated by five elements $\{\eta_1, \dots, \eta_5\}$ whose images under δ_4 generate $ker\delta_3$ and are

$$\{\iota_1(id+x+x^2),\ \iota_2(id+y),\ \iota_3(x+y),\ \iota_4(x^2-id)-\iota_2(yx+x^2)-\iota_1(xy-id),\ \iota_4(y-1)-\iota_3(x-yx+id)+\iota_2\}.$$

C_5 is the free $\mathbb{Z}G$-module generated by six elements $\{\mu_1,\dots,\mu_6\}$ whose images under δ_5 generate $ker\delta_4$ and are

$$\{\eta_1(x-id),\ \eta_2(y-id),\ \eta_3(x^2-y),\ \eta_4(id+x+x^2)+\eta_2(id+x+x^2)-\eta_1(id-y),\\ \eta_5(id+yx)+\eta_4(x+y)+\eta_3+\eta_2(x^2),\ \eta_5(id+y)+\eta_3(id-x+y)-\eta_2\}.$$

C_6 is the free $\mathbb{Z}G$-module generated by seven elements $\{\nu_1,\dots,\nu_7\}$ whose images under δ_6 generate $ker\delta_5$ and are

$$\{\mu_1(id+x+x^2),\ \mu_2(id+y),\ \mu_3(x+y),\ \mu_4(x^2-id)-\mu_1(x^2+y),\\ \mu_6(id+x+x^2)-\mu_5(id+y+xy)+\mu_4(id+y)-\mu_3(y)-\mu_2(x^2),\\ \mu_5(x^2-y)+\mu_2(x)-\mu_3,\ \mu_6(yx-x)-\mu_3(id+x+y)\}.$$

This defines the resolution of the group (C_0) up to the sixth level C_6. If identities among relations ι_i are equivalent to first order syzygies then the ν_i are like the fourth order syzygies.

The calculations proceeded as follows:
First of all we computed an "extra information" complete rewriting system for the group (GAP output):

```
gap> R:=[x^3,y^2,x*y*x*y];
[ x^3, y^2, x*y*x*y ]
gap> R2:=List( R, r -> [ r, [ [ r, IdWord ] ], IdWord ] );
[ [ x^3, [ [ x^3, IdWord ] ], IdWord ], [ y^2, [ [ y^2, IdWord ] ],
IdWord ],
  [ x*y*x*y, [ [ x*y*x*y, IdWord ] ], IdWord ] ]
gap> KB2(R2);
[ [ y^2, [ [ y^2, IdWord ] ], IdWord ],
  [ x^3, [ [ x^3, IdWord ] ], IdWord ],
  [ x^2*y, [ [ y^-1*x^-1*y^-1*x^-1, x^-2 ],
             [ y^2, x^-1*y^-1*x^-3 ],  [ x^3, IdWord ] ],  y*x ],
  [ x*y*x, [ [ y^-2, x^-1*y^-1*x^-1 ], [ x*y*x*y, IdWord ] ], y ],
  [ y*x^2, [ [ y^-1*x^-1*y^-1*x^-1, x^-2*y^-1 ],
             [ x^3, y^-1 ], [ y^2, IdWord ] ], x*y ],
  [ y*x*y, [ [ x^-3, IdWord ], [ x*y*x*y, x^-2 ] ], x^2 ] ]
```

The six rules may be translated as follows:

$$\begin{array}{ll} y^2 \rightarrow_{R2} (s, id) & x^3 \rightarrow_{R2} (r, id) \\ x^2y \rightarrow_{R2} (t^{-x^{-2}} s^{x^{-1}y^{-1}x^{-3}} r, yx) & xyx \rightarrow_{R2} (s^{-x^{-1}y^{-1}x^{-1}} t, y) \\ yxy \rightarrow_{R2} (r^{-1} t^{x^{-2}}, x^2) & yx^2 \rightarrow_{R2} (t^{-x^{-2}y^{-1}} r^{y^{-1}} s, xy) \end{array}$$

The word on the left hand side reduces to the word at the right hand end, and is equal to the boundary of the entry in brackets multiplied by that reduced word. $N(g)$ denotes the normal form (unique reduced word) in $F(X)$ representing the element g and θ is the quotient map : $F(X) \rightarrow G$. The homotopy h_1 is defined on the edges $[g,x]$ of the Cayley graph $(G \times X)$ by finding products of conjugates of the relators (R) whose images under δ_2 are $N(g)xN(g\theta(x))^{-1}$. (For small groups like this one it is possible to do this quite efficiently by inspection.) In general one defines h_1 algorithmically by using the "extra information" rewriting system introduced in the previous section. The definition of h_1 in this example is as follows: (I have chosen to use a more efficient definition than that suggested by the computer program because it simplifies the manual calculations to follow. The only loss by using the computer generated definition is that of space. With groups even a little larger or more complex there is no option but to use the computer generated definition.)

edge $[g,x]$ in $\widetilde{C}_1$	$h_1[g,x]$ in $\widetilde{C}_2$	$p_2h_1[g,x]$ in C_2
$[id,x]$	1	1
$[id,x]$	1	1
$[x,x]$	1	1
$[x,y]$	1	1
$[y,x]$	1	1
$[y,y]$	$[id,s]$	s
$[x^2,x]$	$[id,r]$	r
$[x^2,y]$	$[id,rs^xt^{-x}]$	rs^xt^{-x}
$[xy,x]$	$[id,ts^{-1}]$	ts^{-1}
$[xy,y]$	$[id,ts^{xy}t^{-1}]$	$ts^{xy}t^{-1}$
$[yx,x]$	$[id,sr^yt^{-1}]$	sr^yt^{-1}
$[yx,x]$	$[id,st^ys^{-1}r^{-1}]$	$st^ys^{-1}r^{-1}$

Table 1: Defining h_1

The formulae for the crossed complex give us a complete set of generators for $ker\delta_2$.

$[g,r]$ in $\widetilde{C}_2$	$\tilde{\delta}_2[g,r]$ in $\widetilde{C}_1$	$p_2((h_1\tilde{\delta}_2[g,r])^{-1}[g,r]^{[g,g^{-1}]})$ in C_2	$p_3h_2[g,r]$ in C_3
$[id,r]$	$[1,x][x,x][x^2,x]$	1	0
$[x,r]$	$[x,x][x^2,x][1,x]$	$r^{-1}r^{x^{-1}}$	ι_1
$[y,r]$	$[y,x][yx,x][xy,x]$	1	0
$[x^2,r]$	$[x^2,x][1,x][x,x]$	$r^{-1}r^{x^{-2}}$	$\iota_1(1+x^2)$
$[xy,r]$	$[xy,x][y,x][yx,x]$	$r^{-x^{-1}y^{-1}x^{-1}}r^{y^{-1}x^{-1}}$	$-\iota_1(xy)$
$[yx,r]$	$[yx,x][xy,x][y,x]$	$r^{-y^{-1}}r^{x^{-1}y^{-1}}$	$\iota_1(y)$
$[id,s]$	$[1,y][y,y]$	1	0
$[x,s]$	$[x,y][xy,y]$	$s^{-y^{-1}x^{-1}}s^{x^{-1}}$	$-\iota_2(x^2)$
$[y,s]$	$[y,y][1,y]$	$s^{-1}s^{y^{-1}}$	ι_2
$[x^2,s]$	$[x^2,y][yx,y]$	$t^{y^{-1}x^{-3}}t^{-x^{-2}}$	$-\iota_3(x)$
$[xy,s]$	$[xy,y][x,y]$	1	0
$[yx,s]$	$[yx,y][x^2,y]$	$t^xs^{-x}t^{-y^{-1}}s^{-x^{-1}y^{-1}}$	$\iota_3(y)-\iota_2(yx)$
$[id,t]$	$[1,x][x,y][xy,x][y,y]$	1	0
$[x,t]$	$[x,x][x^2,y][yx,x][xy,y]$	$ts^{-xy}r^{-y}s^{-1}t^xs^{-x}r^{-1}t^{x^{-1}}$	ι_4
$[y,t]$	$[y,x][yx,y][x^2,x][1,y]$	1	0
$[x^2,t]$	$[x^2,x][1,y][y,x][yx,y]$	$t^{y^{-1}x^{-3}}t^{-x^{-2}}$	$-\iota_3(x)$
$[xy,t]$	$[xy,x][y,y][1,x][x,y]$	$t^{-1}t^{y^{-1}x^{-1}}$	ι_3
$[yx,t]$	$[yx,x][xy,y][x,x][x^2,y]$	$t^xs^{-x}r^{-1}ts^{-xy}r^{-y}s^{-1}t^{x^{-1}y^{-1}}$	$\iota_4(1)-\iota_3(yx)$

Table 2: Calculating $ker\delta_2$ and defining h_2

The last column shows how the other identities found may be expressed (in C_3) in terms of the four generating ones. The main result so far is that the module of identities among relations for this group presentation is generated by four elements. This result can be obtained by other methods. However, we now use the results of that last column to calculate a set of generators for the module of identities among identities. This last column defines h_2 on the free generators of $\widetilde{C}_2$ (listed in the second column of the table) so that it annihilates the action of $\widetilde{C}_1$ as required.

The elements $p_3(-h_2\tilde{\delta}_3[g,\iota]+[g,\iota]^{h_0(g)})$ for $[g,\iota]\in\widetilde{C}_3$ are a generating set of identities among the identities. The table below gives the identity resulting from each generator $[g,\iota]$ of $\widetilde{C}_3$. These were obtained by first calculating the images under $\tilde{\delta}_3$. This effectively gives us the boundary of the generator.

For example, $\tilde{\delta}_3[id, \iota_1]$ is $[id, r]^{-1}[x, r]^{[x,x^{-1}]}$, This is because $\delta_3(\iota_1) = r^{-1}r^{x^{-1}}$, and $\tilde{\delta}_n(g, \gamma) := [g, \delta_n(\gamma)]$ and we then write $[g, \delta_n(\gamma)]$ as a product of the generators of C_{n-1} as a C_1-module as h_2 will be defined on these generators. Similarly, $\tilde{\delta}_3[x^2, \iota_4]$ is $[x^2, t][y, s]^{-[y,xy]}[yx, r]^{-[yx,y]}[x^2, s]^{-1}[x, t]^{[x,x]}[x, s]^{-[x,x]}[x^2, r]^{-1}[id, t]^{[id,x^{-1}]}$. (Recall that the action is defined as $[g, \gamma]^{[g,y]} = [g\theta y, \gamma^y]$.)

When we have turned the $[g, \iota]$ into such a product of $\widetilde{C}_2$ generators, we can calculate $h_2(\tilde{\delta}_3[g, \iota])$ using the last table. Note that a property of h_2 is that it must annihilate the action of $\widetilde{C}_1$, it is also a morphism, in that it preserves the multiplication of the elements of $\widetilde{C}_2$. Therefore $h_2\tilde{\delta}_3[id, \iota_1]$ is $h_2[id, r]^{-1} = h_2[x, r]$ and $h_2\tilde{\delta}_3[x^2, \iota_4]$ is $h_2[x^2, t] - h_2[y, s] - h_2[yx, r] - h_2[x^2, s] + h_2[x, t] - h_2[x, s] - h_2[x^2, r] + h_2[id, t]$. We can read these values off the previous table, as we have defined h_2 on all the elements $[g, r]$. So $h_2\tilde{\delta}_3[id, \iota_1]$ is $[id, -0 + \iota_1] = [id, \iota_1]$
and $h_2\tilde{\delta}_3[x^2, \iota_4]$ is $[id, \iota_4 - \iota_2 - \iota_1(y) - (-\iota_3(x)) + \iota_4 - (-\iota_2(x^2)) - \iota_1(1 + x^2) + 0]$.

To obtain the identities we negate the above $h_2\tilde{\delta}_3[g, \iota]$'s and add $[g, \iota]^{h_0(g)}$ which is effectively $[id, \iota(g)]$. We finally project this sum down to C_3: $p_2h_2\tilde{\delta}_3[id, \iota_1]$ is $-\iota_1 + \iota_1 = 0$ and $p_2h_2\tilde{\delta}_3[x^2, \iota_4]$ is $\iota_4(x - 1) - \iota_2(x^2 - id) + \iota_1(id + x^2 + y)$.

The following table gives the identities resulting from all the generators.

$[g, \iota]$ *in* $\bar{C}_3$	$p_3(-h_2\tilde{\delta}_3[g, \iota] + [g, \iota]^{h_0(g)})$ in C_3	$p_4h_3[g, \iota]$ in C_4
$[id, \iota_1]$	0	0
$[x, \iota_1]$	0	0
$[y, \iota_1]$	0	0
$[x^2, \iota_1]$	$\iota_1(id + x + x^2)$	η_1
$[xy, \iota_1]$	0	0
$[yx, \iota_1]$	$\iota_1(y + xy + yx)$	$\eta_1(y)$
$[id, \iota_2]$	0	0
$[x, \iota_2]$	0	0
$[y, \iota_2]$	$\iota_2(id + y)$	η_2
$[x^2, \iota_2]$	$\iota_2(x + yx) - \iota_3(x + y)$	$\eta_2(x) - \eta_3$
$[xy, \iota_2]$	$\iota_2(x^2 + xy)$	$\eta_2(x^2)$
$[yx, \iota_2]$	$\iota_3(x + y)$	η_3
$[id, \iota_3]$	0	0
$[x, \iota_3]$	$\iota_3(x^2 + yx)$	$\eta_3(x)$
$[y, \iota_3]$	$\iota_3(x + y)$	η_3
$[x^2, \iota_3]$	0	0
$[xy, \iota_3]$	$\iota_3(id + xy)$	$\eta_3(y)$
$[yx, \iota_3]$	0	0
$[id, \iota_4]$	0	0
$[x, \iota_4]$	$\iota_4(x^2 - id) + \iota_3(x + y) - \iota_2(yx + x^2) - \iota_1(xy - 1)$	$\eta_4 + \eta_3$
$[y, \iota_4]$	$\iota_4(y - 1) - \iota_3(x - yx + id) + \iota_2$	η_5
$[x^2, \iota_4]$	$\iota_4(x - 1) - \iota_2(x^2 - id) + \iota_1(id + x^2 + y)$	$-\eta_4(x) - \eta_2(x^2) - \eta_1$
$[xy, \iota_4]$	$\iota_4(xy - 1) - \iota_3(id - yx - y) - \iota_2(yx) - \iota_1(xy - id)$	$\eta_5(x^2) + \eta_4 + \eta_3(x)$
$[yx, \iota_4]$	$\iota_4(yx - id) - \iota_3(id - y - yx) - \iota_2(x^2 + yx - id) + \iota_1(x^2 + id + y)$	$-\eta_5(yx) - \eta_4(x) - \eta_2(x^2) + \eta_1$

Table 3: Calculating $ker\,\delta_3$ and defining h_3

The images of the η_i generate the kernel as a $\mathbb{Z}G$-module, the η_i themselves provide a set of generators for $\bar{C}_4$. We use the formula $p_4(-h_3\tilde{\delta}_4[g, \eta] + [g, \eta]^{h_0(g)})$ to calculate a generating set of 30 elements for

$ker\delta_4$, which we can reduce to six. The last table defines h_3 ("in $\widetilde{C}_4$" column) on the generators of C_3 ($[g,\iota]$ column).

$[g,\eta]$ in $\bar{C}_4$	$p_4(-h_3\tilde{\delta}_4[g,\eta]+[g,\eta]^{h_0(g)})$ in C_4	$p_5h_4[g,\eta]$ in C_5
$[id,\eta_1]$	$-\eta_1+\eta_1$	0
$[x,\eta_1]$	$-\eta_1+\eta_1(x^2)$	$-\mu_1(x^2)$
$[y,\eta_1]$	$-\eta_1(y)+\eta_1(y)$	0
$[x^2,\eta_1]$	$-\eta_1+\eta_1(x)$	μ_1
$[xy,\eta_1]$	$-\eta_1(y)+\eta_1(xy)$	$\mu_1(y)$
$[yx,\eta_1]$	$-\eta_1(y)+\eta_1(yx)$	$-\mu_1(yx)$
$[id,\eta_2]$	$-\eta_2+\eta_2$	0
$[x,\eta_2]$	$-\eta_2(x^2)+\eta_2(x^2)$	0
$[y,\eta_2]$	$-\eta_2+\eta_2(y)$	μ_2
$[x^2,\eta_2]$	$-\eta_2(x)+\eta_2(x)$	0
$[xy,\eta_2]$	$-\eta_2(x^2)+\eta_2(xy)$	$\mu_2(x^2)$
$[yx,\eta_2]$	$-\eta_2(x)+\eta_2(yx)$	$\mu_2(x)$
$[id,\eta_3]$	$-\eta_3+\eta_3$	0
$[x,\eta_3]$	$-\eta_3(y)+\eta_3(x^2)$	μ_3
$[y,\eta_3]$	$-\eta_3(y)+\eta_3(y)$	0
$[x^2,\eta_3]$	$-\eta_3(x)+\eta_3(x)$	0
$[xy,\eta_3]$	$-\eta_3(x)+\eta_3(xy)$	$\mu_3(yx)$
$[yx,\eta_3]$	$-\eta_3+\eta_3(yx)$	$\mu_3(y)$
$[id,\eta_4]$	$-\eta_4+\eta_4$	0
$[x,\eta_4]$	$\eta_4(x+id)+\eta_2(x^2+x+id)-\eta_1(id-y)+\eta_4(x^2)$	μ_4
$[y,\eta_4]$	$\eta_5(id+yx)+\eta_4(x)+\eta_3+\eta_2(x^2)+\eta_4(y)$	μ_5
$[x^2,\eta_4]$	$-\eta_4(x)+\eta_4(x)$	0
$[xy,\eta_4]$	$\eta_5(x^2+id)+\eta_4+\eta_3(x-id)+\eta_2(x+id)+\eta_4(xy)$	$-\mu_6+\mu_5(y)-\mu_2(x)$
$[yx,\eta_4]$	$-\eta_5(x^2+yx)-\eta_4(x+id)-\eta_3(x)+\eta_1(y-id)+\eta_4(yx)$	$-\mu_6(x^2+x)+\mu_5(xy)$ $-\mu_4+\mu_3(y)-\mu_2$
$[id,\eta_5]$	$-\eta_5+\eta_5$	0
$[x,\eta_5]$	$-\eta_5(x^2)+\eta_5(x^2)$	0
$[y,\eta_5]$	$\eta_5+\eta_3(id-x+y)-\eta_2+\eta_5(y)$	μ_6
$[x^2,\eta_5]$	$\eta_5(yx)+\eta_3(x-y+id)-\eta_2(x)+\eta_5(x)$	$\mu_6(x)+\mu_3(id+x)$
$[xy,\eta_5]$	$\eta_5(x^2)+\eta_3(y-id+x)-\eta_2(x^2)+\eta_5(xy)$	$\mu_6(x)+\mu_3(x^2-id)$
$[yx,\eta_5]$	$-\eta_5(yx)+\eta_5(yx)$	0

Table 4: Calculating $ker\delta_4$ and defining h_4

So now we have six generators for $\widetilde{C}_5$: $\{\mu_1,\dots,\mu_6\}$ and their images $\{\eta_1(x-id),\ \eta_2(y-id),\ \eta_3(x^2-y),\ \eta_4(id+x+x^2)+\eta_2(id+x+x^2)-\eta_1(id-y),\ \eta_5(id+yx)+\eta_4(x+y)+\eta_3+\eta_2(x^2),\ \eta_5(id+y)+\eta_3(id-x+y)-\eta_2\}$ generate the module of identities among the identities among identities ($ker\delta_4$). The last column defines h_4.

$[g,\mu]$	$p_5(-h_4\tilde{\delta}_5[g,\mu]+[g,\mu]^{h_0(g)})$	in C_6
$[id,\mu_1]$	0	0
$[x,\mu_1]$	0	0
$[y,\mu_1]$	0	0
$[x^2,\mu_1]$	$\mu_1(id+x+x^2)$	ν_1
$[xy,\mu_1]$	$\mu_1(y+xy+yx)$	$\nu_1(y)$
$[yx,\mu_1]$	0	0
$[id,\mu_2]$	0	0
$[x,\mu_2]$	0	0
$[y,\mu_2]$	$\mu_2(id+y)$	ν_2
$[x^2,\mu_2]$	0	0
$[xy,\mu_2]$	$\mu_2(x^2+xy)$	$\nu_2(x^2)$
$[yx,\mu_2]$	$\mu_2(x+yx)$	$\nu_2(x)$
$[id,\mu_3]$	0	0
$[x,\mu_3]$	$\mu_3(xy+x^2)$	$\nu_3(x)$
$[y,\mu_3]$	0	0
$[x^2,\mu_3]$	$\mu_3(x+y)$	ν_3
$[xy,\mu_3]$	$\mu_3(id+xy)$	$\nu_3(y)$
$[yx,\mu_3]$	0	0
$[id,\mu_4]$	0	0
$[x,\mu_4]$	$\mu_4(x^2-id)-\mu_1(x^2+y)$	ν_4
$[y,\mu_4]$	$\mu_6(id+x+x^2)-\mu_5(1+y+xy)+\mu_4(1+y)-\mu_3(y)-\mu_2(x^2)$	ν_5
$[x^2,\mu_4]$	$\mu_4(x-id)+\mu_1(yx+id)$	$-\nu_4(x)$
$[xy,\mu_4]$	$\mu_6(id+x+x^2)-\mu_5(1+y+xy)+\mu_4(1+xy)-\mu_3(y)-\mu_2(x^2)+\mu_1(y+x^2)$	$\nu_5-\nu_4(xy)$
$[yx,\mu_4]$	$\mu_6(id+x+x^2)-\mu_5(1+y+xy)+\mu_4(1+yx)-\mu_3(y)-\mu_2(x^2)-\mu_1(yx+id)$	$\nu_5+\nu_4(y)$
$[id,\mu_5]$	0	0
$[x,\mu_5]$	$\mu_5(x^2-y)+\mu_2(x)-\mu_3$	ν_6
$[y,\mu_5]$	0	0
$[x^2,\mu_5]$	$-\mu_6(x^2+x+id)+\mu_5(id+y+x)-\mu_4(id+y)-\mu_3(x+x^2)+\mu_2(id+x^2)$	$\nu_6(x^2)-\nu_5+\nu_3$
$[xy,\mu_5]$	$-\mu_6(x^2+x+id)+\mu_5(id+y+yx)-\mu_4(id+y)-\mu_3(x+x^2+yx)+\mu_2(x^2)$	$-\nu_5-\nu_3(x+id)$
$[yx,\mu_5]$	$\mu_5(yx-id)-\mu_3(y)-\mu_2(x^2)$	$\nu_6(y)-\nu_2(x^2)$
$[id,\mu_6]$	0	0
$[x,\mu_6]$	$-\mu_3(x^2+yx)$	$-\nu_3(x^2)$
$[y,\mu_6]$	$\mu_6(y-id)+\mu_3(yx)$	ν_7
$[x^2,\mu_6]$	$-\mu_3(x+y)$	$-\nu_3$
$[xy,\mu_6]$	$\mu_6(xy-x^2)+\mu_3(y-x^2+yx)+\mu_2(x^2)$	$\nu_7(x^2)+\nu_3(x)$
$[yx,\mu_6]$	$\mu_6(yx-x)-\mu_3(x+y+id)+\mu_2(x)$	$\nu_7(x)-\nu_3(y+id)$

Table 5: Calculating $ker\delta_5$ and defining h_5

We could calculate the identities for the next level, using the last table as a definition for h_5, computing a set of 42 generators for $ker\delta_6$ (using $p_6(-h_5\tilde{\delta}_6[g,\nu]+[g,\nu]^{h_0(g)})$) and reducing them as before. It does not get more complicated: for $n \geq 3$ C_n is a $\mathbb{Z}G$-module and the expression $p_n(-h_{n-1}\tilde{\delta}_n[g,\gamma]+[g,\gamma]^{h_0(g)})$, where γ is a generator of C_n, gives a set of generators for C_{n+1} as a $\mathbb{Z}G$-module (which may be reduced over the $\mathbb{Z}G$-module). It is in principle possible to continue this exercise further, but it is not of value to do so here. The obvious conjecture it that C_n will be the free $\mathbb{Z}G$-module generated by $n+1$ elements.

Notice that every time we are choosing a set of independent generators for the $\mathbb{Z}G$-submodule; the set is not unique, and we do not have an algorithm for determining which generator is expressible in terms of the others or how to express it in this way. The method used is no more than inspection and trial and error. The purpose of including this example is that it best shows what may be achieved using the covering groupoids and homotopies methods, the complexity of even a very small example, and thus illustrates the necessity for a computer algorithm to extract such information as was summarised at the beginning of this example. The next section shows that these problems can be expressed in terms of noncommutative Gröbner bases over group rings. New work is being developed [52] on algorithms for such problems, and so expressing the problem of devising an algorithm for obtaining *reduced* sets of identities and higher identities is a step forward, and until such Gröbner basis algorithms become available we cannot expect to be able to have algorithms for reducing the sets of generating identities.

5.6 The Submodule Problem

The previous sections have shown that a variation of the noncommutative Buchberger algorithm (Knuth-Bendix algorithm) may be applied to a group presentation to obtain the contracting homotopy h_1, and a set of generators for the module of identities among relations for the group presentation. This much has been implemented in the program `idrel.g` for GAP. The remaining problem is that of reducing the set of generators with respect to the action of $\mathbb{Z}G$ on the module.

We discussed earlier the Peiffer Problem which occurs at the first level (identities among relations: $ker\delta_2 \subseteq C(R)$). This problem is difficult because we need to test for equality in the free crossed $F(X)$-module, in other words, to test for Peiffer equivalence of two sequences (recall that the Peiffer rules imply that $[s,v][r,u] = [r,u][s, v\delta(r)^u] = [r, u\delta(s)^v]$). In this case we essentially wish to be able to reduce the set of generating identities to a set $\{\iota_1, \dots, \iota_k\}$ that is in some sense minimal over $\mathbb{Z}G$ i.e. no ι_j can be written as a sum of $\mathbb{Z}G$-multiples of the other identities. To summarise – there are great difficulties in reducing the set of generators of the module of identities among relations. Furthermore, unless we can express each of the original generators in terms of those in the reduced set it is not practical to define h_2 on such a large set.

We will now use a property which converts the Peiffer Problem into a Gröbner basis problem. This property is fully explained in [15]. First, recall that the crossed module is defined by taking the Peiffer equivalence classes of the free group $F(R \times F(X))$. This is the same as looking at the free monoid $(Y^+ \sqcup Y^-)^*$ factored by the relations needed for the group as well as by the Peiffer relations. Elements of $(Y^+ \sqcup Y^-)^*$ are called **Y-sequences**.

An **identity Y-sequence** is one whose image under δ_2 is the identity in $F(X)$.

The identity property uses a result on the abelianisation of $C(R)$ to describe a useful way of determining whether an identity Y-sequence (i.e. one identified with an element of the kernel of δ_2, which is abelian) is Peiffer equivalent to the empty sequence.

An identity Y-sequence $a = (r_1, u_1)^{\varepsilon_1}, \dots, (r_k, u_k)^{\varepsilon_1}$ has the **Primary Identity Property** if the indexing numbers $1, \dots, k$ of the sequence y can be paired (i,j) so that $r_i = r_j$, $\theta(u_j) = \theta(u_j)$ and $\varepsilon_i = -\varepsilon_j$.

Lemma 5.6.1 ([15]) *Let $a \in (Y^+ \sqcup Y^-)^*$. Then a has the Primary Identity Property if and only if it is Peiffer equivalent to the empty sequence.*

Let X be a set and let K be a ring. Recall that the **free right K-module** $K[X]$ on X has as elements all formal sums $x_1k_1 + \cdots + x_nk_n$ where $x_1, \dots, x_n \in X$ and $k_1, \dots, k_n \in K$. Right multiplication by elements of K and addition of elements of $K[X]$ are defined, with a zero and inverses, and $(x_1 + x_2)k = x_1k + x_2k$. Let $P := \{p_1, \dots, p_n\} \subseteq K[X]$. Recall that the **sub $\mathbb{Z}G$-module generated by P** is

$$\langle P \rangle := \{p_1\zeta_1 + \cdots + p_n\zeta_n : \zeta_1, \dots, \zeta_n \in K\}$$

Let $grp\langle X|R\rangle$ be a presentation of a group G. The group ring $\mathbb{Z}G$ is the free right $\mathbb{Z}$-module on G together with a composition, making it an algebra over the ring $\mathbb{Z}$. The free right $\mathbb{Z}G$-module $\mathbb{Z}G[R]$ on the set R has elements of the form $r_1\zeta_1 + \cdots + r_n\zeta_n$ where $r_1, \ldots, r_n \in R$ and $\zeta_1, \ldots, \zeta_n \in \mathbb{Z}G$.

Lemma 5.6.2 *Let $grp\langle X|R\rangle$ be a presentation of a group G, with quotient morphism $\theta : F(X) \to G$. Let $\iota = (r_1, u_1)^{\varepsilon_1} \cdots (r_n, u_n)^{\varepsilon_n}$ be an identity Y- sequence and let λ denote the empty sequence. Define $\alpha : (Y^+ \sqcup Y^-)^* \to \mathbb{Z}G[R]$ by $\alpha((r,u)^{\varepsilon}) := r(\theta u \varepsilon)$ with $\alpha(\lambda) = 0$. Then $\iota \overset{*}{\leftrightarrow}_{R_P} \lambda$ if and only if $\alpha(\iota) = 0$.*

Proof We verify that α preserves the G-action: $\alpha(((r,u)^{\varepsilon})^v) = \alpha((r,uv)^{\varepsilon}) = r(\theta(uv)\varepsilon) = (\alpha(r,u)^{\varepsilon})^{\theta v}$. The result now follows immediately from the definition of α, the Primary Identity Property and the previous lemma. □

Corollary 5.6.3 *Let ι_1, ι_2 be identity Y-sequences. Then $\iota_1 \overset{*}{\leftrightarrow}_{R_P} \iota_2$ if and only if $\langle \iota_1 \rangle = \langle \iota_2 \rangle$ in $\mathbb{Z}G[R]$.*

Definition 5.6.4 *Let $K[X]$ be a right K-module and let $a, b \in K[X]$. The Submodule Problem is*

INPUT	$a, b \in K[X]$	*(two elements of the right K-module,)*
QUESTION	$\langle a \rangle = \langle b \rangle$*?*	*(do they generate the same submodule?)*

So we have shown that the Peiffer Problem for identity Y-sequences simplifies to the Submodule Problem. If the Submodule Problem can be solved then it is possible to reduce the set of generators of $ker\delta_2$ to a set of generating identities $\{\iota_1, \ldots \iota_t\}$ such that no subset of this will generate the same sub $\mathbb{Z}G$-module. This is in some sense a minimal set of generators for $ker\delta_2$ (see later note).

At the next levels, $ker\delta_n$ for $n \geq 3$, the problem is simpler in that we are now working entirely in $\mathbb{Z}G$-modules, and do not encounter the Peiffer Problem. The only problem we now encounter is the Submodule Problem.

In the $ker\delta_3$ case (Table 3) we have a set of 24 generators as elements of C_2, which here is the free $\mathbb{Z}G$-module on $\{\iota_1, \ldots, \iota_4\}$. Some of these generators are zero, others are of the form $\iota_1(id{+}x{+}x^2)$ and $\iota_2(x{+}yx) - \iota_3(x{+}y)$.

The problem may be phrased in the terms of a Gröbner basis problem. This is a reasonable approach, because methods for dealing with commutative Gröbner bases over rings exist [1] (essentially for Principal Ideal Domains) and methods for noncommutative Gröbner bases over rings (specifically group and monoid rings) are being developed [52]. Let $P := \{p_1, \ldots, p_n\}$ be a set of polynomials with coefficients in $\mathbb{Z}G$ and monomials from a set M i.e. $p_1, \ldots, p_n$ are elements of the $\mathbb{Z}G$-module $\mathbb{Z}G(M)$. The task is to find a set $Q := \{q_1, \ldots, q_m\}$ that generates the same sub $\mathbb{Z}G$-module, but is such that no q_i is a sum of $\mathbb{Z}G$-multiples of the other q_j.

Bases for modules are not in general unique or of the same rank. So it is possible that there are two such sets Q and Q' and that these are of different sizes. We are concerned not with finding the generating set with smallest cardinality but with finding a set which contains no subset which would generate the same submodule.

If Q is a Gröbner basis for P then by definition $\langle P \rangle = \langle Q \rangle$. If Q is a reduced Gröbner basis then it is such that no element q_i of Q is a sum of $\mathbb{Z}G$-multiples of the other elements q_j of Q. This puts the problem of finding a reduced set of sub-module generators in terms of a Gröbner basis problem.

5.7 Concluding Remarks

The purpose of this chapter was to make algorithmic the methods given in [17]. In fact we have computerised the initial part of the construction, using rewriting theory and the Knuth-Bendix completion procedure to algorithmically define the first contracting homotopies h_0 and h_1. The program *idrels.g* will compute, from a group presentation, a complete generating set for the module of identities among relations.

Unfortunately we cannot yet produce an algorithm for the minimalisation of this set of generators. Two major barriers to a reduction procedure have been identified. Firstly, the Peiffer Problem, a particularly difficult word problem encountered in crossed modules and 2-categories as a result of the Peiffer rules or interchange law. This has been reduced, using a property defined in [15] to the Submodule Problem, which is also encountered at higher levels, and indicates that methods for noncommutative Gröbner bases over group rings are required. Methods for solving this problem are progressing, thanks to collaboration with Birgit Reinert (Kaiserslautern). A program for reducing the first generating set of identities exists. This work will continue with the aim of extending the program so that it will compute minimal generating sets for the $\mathbb{Z}G$-modules C_n for any given n.

Investigation of whether the completion of a monoid presentation yields something useful for the construction of a resolution of the monoid would also be an interesting area of work. We do not know whether the covering groupoids methods of [17] might generalise to a covering categories of monoids method for calculating something corresponding to identities among relations for monoids. This looks like the beginnings of a noncommutative syzygy theory, and would definitely be worth investigating.

File 1: knuth.g

The first program is an implementation of the standard Knuth-Bendix procedure which may be applied to string rewriting. A rewrite system R is input in the form of a list `R` of pairs of words. The important subroutines are:

- `OnePass(word,R)`: reduces `word` (if possible) by applying one rule from `R`. This procedure involves searching to see if the left side of a rule in `R` is a subword of `word` and then replacing that part of `word` with the right side of the rule.
- `ReduceWord(word,R)` reduces `word` as far as possible with respect to `R` by the repeated application of the previous function. (Note that the reduced form can only be guaranteed to be unique if `R` is complete.)
- `CriticalPairs(R)`: overlaps between the left hand sides of the rules in `R` are found, and the resulting critical pairs are found and reduced with respect to `R`.
- `OnePassKB(R)`: this function computes the critical pairs of a rewrite system `R` and then resolves these critical pairs by adding then to `R`.
- `SystemReduce(R)`: is an efficiency measure rather than theoretically essential. It normalises an ordinary rewrite system by reducing the rules (both sides of each rule are reduced by the other rules and the rules implied by other rules within the system are hence removed).

The main function of the program is `KB`.

- `KB(R)`: attempts to complete the rewrite system (with respect to the length-lex order). If it achieves the completion it returns the complete (reduced) rewrite system as a list of ordered pairs.

When the rewriting system is for a monoid there are further functions which will enumerate the elements of the monoid.

- `NextWords(F,Words)`: creates new words of length $n+1$ by composing single generators from (the free group) `F` with irreducible words of length n.
- `Enumerate(F,R)`: uses the previous function and `reduce(word,R)` to build up blocks of words of the same length (on the irreducibles one unit shorter) and then to reduce these words as far as possible. When a whole block of new words is reducible, there are no more irreducible words to be found.

File 2: kan.g

The main function of the program is called `Kan`. The input, functions and output are fully described in Chapter Two.

- `InitialRules(KAN)`: The first sub-routine constructs the initial rewrite system of mixed one-sided and two-sided rules. All the rules of the form $(x\iota Fa, Xa(x))$ for $a \in \mathsf{A}$ are added to the relations of the category B. This establishes an initial rewriting system for the group.
- `Kan(KAN)`: This completes the rewriting system with respect to length-lex (where possible) by calling `knuth.g`. It then enumerates the elements of the sets which make up the Kan extension. The action of B on the resulting elements can easily be computed.

File 3: ncpoly.g

This file provides definitions and some operations for polynomials with rational coefficients and non-commutative monomials in a semigroup.

• `PolyFromTerms`($[[\mathtt{k}_1,\mathtt{m}_1],..,[\mathtt{k}_n,\mathtt{m}_n]]$): creates a (noncommutative) polynomial from a list of terms. A polynomial is stored as a record but printed nicely as a polynomial `k1 m1` $+ \cdots +$ `kn mn`. There are a number of operations:

• `IsNonCommPoly(poly)`: tests whether a record is a polynomial.
• `LengthPoly(poly)`: returns the number of terms.
• `LeadTerm(poly)`: extracts the leading term (which consists of the monomial of greatest size with respect to the length-lex order and its coefficient).
• `LeadCoeff(poly)`: returns the coefficient of the leading term.
• `LeadMonom(poly)`: returns the monomial part of the leading term.
• `MakeMonic(poly)`: divides a a polynomial by its leading coefficient to return a monic polynomial.
• `NeatenPoly(poly)`: adds like terms (non-destructive).
• $\mathtt{poly}_1 = \mathtt{poly}_2$: equality between polynomials is well defined.
• `AreEquivPolys`($\mathtt{poly}_1,\mathtt{poly}_2$): polynomials are equivalent if one is a multiple of the other.
• `AddPoly`($\mathtt{poly}_1,\mathtt{poly}_2$) : returns the 'neatened' sum of two 'neat' polynomials.
• `SubtractPoly`($\mathtt{poly}_1,\mathtt{poly}_2$) : returns the 'neatened' difference of two 'neat' polynomials.

To summarise: a polynomial record `poly` has the following fields: `poly.IsNonComPoly` is true; `poly.terms` is a list of terms [`c`,`m`] where `c` is a rational and `m` is a word; `poly.isNeat` is either true or false; `poly.operations` will be `NonCommPolyOps`; `poly.lead` is a term [`c`,`m`]; `poly.leadmon` is `poly.lead`[2]; `poly.isMonic` is either true or false.

All these functions are required for the noncommutative Gröbner basis program.

File 4: grobner.g

This is a program for computing the noncommutative Gröbner basis of a set of polynomials. It consists of a number of functions:

• `ReducePoly(poly,POL)`: reduces a polynomial *poly* by subtracting multiples of polynomials in `POL`. The reduced form can only be guaranteed to be unique with a Gröbner basis.
• `OrderSystem(POL)`: orders a set of polynomials with respect to their leading monomials.
• `PolySystemReduce(POL)`: Removes polynomials which are sums of multiples of other polynomials in the system.
• `SPolys(ALL,NEW)`: compares two lists of polynomials for matches (if the lists are equal then this is the standard procedure and finds all matches in the system) and calculates the resulting S-polynomials.
• `GB(POL)`: returns (where possible) a Gröbner basis for a system of noncommutative polynomials over the rationals (with respect to the length-lex order).

File 5: idrel.g

This program accepts as input a free group and a list of relators. It goes through a number of calculations, including an "extra information" Knuth-Bendix completion procedure and returns a complete set of generators for the module of identities among relations. The input, functions and output are fully described in Chapter Five, with examples.

Bibliography

[1] W. W. Adams and P. Loustaunau : An Introduction to Gröbner Bases, *Graduate Studies in Mathematics, publishers: American Math. Soc.* (1994).

[2] B. Amrhein and O. Gloor : The Fractal Walk, *in Gröbner Bases and Applications, B. Buchberger and F. Winkler (eds), Proc. London Math. Soc. vol.251 p305-322* (1998).

[3] F. Baader and T. Nipkow : Term Rewriting and All That, *Cambridge University Press* (1998).

[4] Y. G. Baik and S. J. Pride : Generators of the Second Homotopy Module of Presentations arising from Group Constructions, *University of Glasgow Preprint 92-49* (1992).

[5] G. Bergman : The Diamond Lemma for Ring Theory, *Advances of Mathematics, vol.29, p178-218* (1978).

[6] W. A. Bogley and S. J. Pride : Calculating Generators of Π_2, *in Group Theory and Low-Dimensional Homotopy Theory, C. Hog-Angeloni, W. Metzler, A. Sieradski (eds), Cambridge University Press* (1993).

[7] R. Book and F. Otto : String-Rewriting Systems, *Springer-Verlag, New York* (1993).

[8] M. A. Borges and M. Borges : Gröbner Bases Property for an Elimination Ideal in the Noncommutative Case, *in Gröbner Bases and Applications, B. Buchberger and F. Winkler, (eds) Proc. London Math. Soc. vol.251* (1998).

[9] BooBarkee : Gröbner Bases: The Ancient Secret Mystic Power of Algu Compubraicus,

[10] K. Brown : Cohomology of Groups: *Graduate Texts in Mathematics (87) Springer-Verlag, New York* (1982).

[11] R. Brown : Elements of Modern Topology, *McGraw Hill (Maidenhead)* (1968).

[12] R. Brown : On the Second Relative Homotopy Group of an Adjunction Space: An Exposition on a Theorem of J. C. H. Whitehead, *Journal of London Math. Soc. (2) 22, p146-152* (1980).

[13] R. Brown : Peiffer Equivalences for Pre-crossed Modules over Groupoids, *School of Mathematics, Bangor University* (1996).

[14] R. Brown and P. J. Higgins : On the Connection Between the Second Relative Homotopy Groups of some Related Spaces, *Proc. London Math. Soc. (3) vol.36 p193-212* (1978).

[15] R. Brown and J. Huebschuman : Identities Among Relations, *in Low-Dimensional Topology, Brown and Thickstun (eds) Proc. London Math. Soc. vol.48 p153-202* (1982).

[16] R. Brown and T. Porter : On the Schreier Theory of Nonabelian Extensions: Generalisations and Computations, *School of Mathematics, Bangor University* (1995).

[17] R. Brown and A. R. Salleh : On the Computation of Identities Among Relations and of Free Crossed Resolutions of Groups, *School of Mathematics, Bangor University* (1997).

[18] R. Brown and C. D. Wensley : On Finite Induced Crossed Modules and the Homotopy 2-Type of Mapping Cones, *Theory and Applications of Categories, vol.1 p54-71* (1995).

[19] R. Brown and C. D. Wensley : Computing Crossed Modules Induced by an Inclusion of a Normal Subgroup, with Applications to Homotopy 2-Types, *Theory and Applications of Categories, vol.2 p3-16* (1996).

[20] R. Brown and C. D. Wensley : On the Computation of Induced Crossed Modules, *University of Wales Bangor Preprint 97.07* (1997).

[21] M. R. Bush, M. Leeming and R. F. C. Walters : Computing Left Kan Extensions, *Journal of Symbolic Computation, vol.11 p11-20* (1997).

[22] B. Buchberger : Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal, *Dissertation Math. Inst. Universität Innsbruck* (1965).
An Algorithmic Criterion for the Solvability of a System of Algebraic Equations, *translation by M. Abramson and R. Lumbert in Gröbner Bases and Applications, B. Buchberger and F. Winkler, (eds) Proc. London Math. Soc. vol.251* (1998).

[23] B. Buchberger and F. Winkler : Gröbner Bases and Applications, *"33 Years of Gröbner Bases" RISC-Linz 2-4 Feb 1998, Proc. London Math. Soc. vol.251* (1998).

[24] C. M. Campbell, N. Ruskŭc, E. F. Robertson and R. M. Thomas : Rewriting a Semigroup Presentation, *International Journal of Algebra and Computation, vol.5 no.1 p81-103* (1995).

[25] S. Carmody and R. F. C. Walters : The Todd-Coxeter Procedure and Left Kan Extensions, *Research Reports of the School of Mathematics and Statistics, The University of Sydney 90-19* (1990).

[26] S. Carmody and R. F. C. Walters : Computing Quotients of Actions on a Free Category, *Research Reports of the School of Mathematics and Statistics, The University of Sydney 90-20* (1990).

[27] H. P. Cartan and S. Eilenberg : Homological Algebra, *Princeton Princeton University Press* (1956).

[28] D. E. Cohen : Introduction to Computer Theory, *Revised Edition, New York : Wiley* (1991).

[29] D. A. Cox, J. B. Little and D. O'Shea : Ideals, Varieties and Algorithms, *Undergraduate Texts in Mathematics, Springer-Verlag* (1992).

[30] R. Cremanns : Finiteness Conditions for Rewriting Systems, *PhD Thesis Universität Gesamthochschule Kassel* (1995).

[31] M. Dehn: Papers on group theory and topology, (translated and introduced by John Stillwell) *Springer-Verlag, New York* (1987).

[32] J. Desel and W. Reisig : The Synthesis Problem of Petri Nets, *Acta informatica 33, p297-315 Springer-Verlag* (1996).

[33] 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, *Boston : Jones and Bartlett Publishers* (1992).

[34] M. Fleming, R. Gunther and R. Rosebrugh : User Guide for the Categories Database and Manual, *anonymous ftp://sun1.mta.ca/pub/papers/rosebrugh/catdsalg.dvi,tex and /catuser.dvi,tex* (1996).

[35] E. L. Green : Noncommutative Gröbner bases. A Computational and Theoretical Tool, *lectures, New Mexico State University, Las Cruces, January* (1997).

[36] K. W. Gruenberg : Resolutions by Relations, *Journal London Math. Soc. vol.35 p481-494* (1960).

[37] K. W. Gruenberg : Cohomological Topics in Group Theory: *Springer Lecture Notes, vol.143* (1970).

[38] G. Hermann : Die Frage der endlich vielen Schritte in der Theorie der Polynomideale, *Math. Ann, vol.95 p736-788* (1926).

[39] S. M. Hermiller : Tame Combings, Almost Convexity and Rewriting Systems for Groups, *University of Melbourne and New Mexico State University* (1997).

[40] P. J. Higgins : Presentations of Groupoids, with Applications to Groups, *Pro. Camb. Phil. Soc. vol.60 p7-20* (1964).

[41] J. E. Hopcroft and J. D. Ullman : Introduction to Automata Theory, Languages and Computation, *Addison-Wesley Publishing Company* (1979).

[42] D. F. Holt : Knuth-Bendix in Monoids, and Automatic Groups, *Mathematics Institute, University of Warwick* (1996).

[43] D. F. Holt : Rewriting Techniques in Finitely Presented Groups and Monoids, *Lectures: New Mexico State University, Las Cruces, Jan 3-7* (1997).

[44] D. F. Holt and D. F. Hurt : Computing Automatic Coset Systems and Subgroup Presentations, *Journal of Symbolic Computation* (1996).

[45] J. M. Howie: Automata and Languages, *Oxford University Press* (1991).

[46] D. Johnson : Presentations of Groups, *Cambridge University Press* (1990).

[47] K. H. Kim and F. W. Roush : Applied Abstract Algebra, *Ellis Horwood Ltd, Halstead Press* (1983).

[48] D. Knuth and P. Bendix : Simple Word Problems in Universal Algebras, in J. Leech (ed) Computational Problems in Abstract Algebra, *Pergamon Press, New York* (1970).

[49] R. Lavendhomme and R. Lucas : On Modules and Crossed Modules, *Journal of Algebra, vol.179, p936-963* (1996).

[50] S. A. Linton, G. Peiffer, E. F. Robertson and N. Ruskŭc : Groups and Actions in Transformation Semigroups, *Mathematische Zeitschrift* (to appear).

[51] S. Mac Lane : Categories for the Working Mathematician, *Springer-Verlag* (1971).

[52] K. Madlener and B. Reinert : Gröbner Bases in Non-Commutatice Reduction Rings, *in Gröbner Bases and Applications, B. Buchberger and F. Winkler (eds) Proc. London Math. Soc. 251 p408-420* (1998).

[53] B. Mitchell : Rings with Several Objects, *Academic Press vol.8 no.1* (1972).

[54] H. M. Möller : On the Construction of Gröbner bases using Syzygies, *in Computational Aspects of Comouter Algebra, L. Robbiano (ed) Academic Press, San Diego p211-225* (1989).

[55] F. Mora : Gröbner bases for Noncommutative Polynomial Rings, in J. Calmet (ed) *AAECC-3, Lect. Notes of Computer Science 229 p353-362* (1986).

[56] T. Mora : Gröbner Bases and the Word Problem, *Preprint, University of Genova* (1987).

[57] T. Mora : An Introduction to Commutative and Noncommutative Gröbner Bases, *Theoretical Computer Science vol.134 p131-173* (1994).

[58] T. Murata : Petri-nets: Properties, Analysis and Applications, *Proceedings of the IEEE, vol.77 no.4 April* (1989).

[59] J. Neubuser : An Elementary Introduction to Coset Table Methods in Computational Group Theory, *London Math. Soc. Lecure Notes Series, vol.71, p1-45* (1981).

[60] P. Nordbeck : On Some Basic Applications of Gröbner Basis Methods In Noncommutative Polynomial Rings. *in Gröbner Bases and Applications, B. Buchberger and F. Winkler (eds) Proc. London Math. Soc. 251 p408-420* (1998).

[61] R. Peiffer : Uber Identitaten Zwischen Relationen, *Math. Annalen. vol.121 p67-99* (1949).

[62] T. Porter : Internal Categories and Crossed Modules, *Proc. Inter. Conf. of Category Theory, Gummersbach, 1981, in Springer Lecture Notes in Mathematics, vol.962* (1982).

[63] S. J. Pride : Identities Among Relations of Group Presentations, *in Group Theory from a Geometric Viewpoint, editors: E. Ghys, A. Haefliger and A. Verjovsky, World Scientific, p687-717* (1990).

[64] S. J. Pride : The (Co)homology of Groups given by Presentations in which Each Defining Relator Involves At Most Two Types of Generators, *Journal of the Australian Math. Soc. series A, vol.52, p205-218* (1992).

[65] S. J. Pride : Low-Dimensional Homotopy Theory for Monoids, *International Journal of Algebra and Computation* (1993).

[66] S. J. Pride : Geometric Methods in Combinatorial Semigroup Theory, *Proc. International Conference on Groups, Semigroups and Formal Languages, York, Kluwer Publishers* (1993).

[67] S. J. Pride and R. Stohr : Relation Modules of Groups with Presentations in which Each Relator Involves Exactly Two Types of Generators, *Journal of the London Math. Soc. series 2, vol.38 p99-111* (1988).

[68] S. J. Pride and R. Stohr : The (Co)homology of Aspherical Coxeter Groups, *Journal of the London Math. Soc. series 2, vol.42 p49-63* (1990).

[69] K. Reidemeister : Uber Identitaten von Relationen, *Abhandlungen aus dem Mathematischen Seminar der Universität Hamberg vol.16 p114-118* (1949).

[70] B. Reinert : On Gröbner Bases in Monoid and Group Rings *PhD Thesis, Universität Kaiserslautern* (1995).

[71] I. D. Redfern : Automatic Coset Systems, *PhD Thesis, University of Warwick* (1993).

[72] E. F. Robertson, N. Ruskŭc and J. Wiegold : Generators and Relations of Direct Products of Semigroups, *Transactions of the American Math. Soc.* (to appear).

[73] C. C. Sims : Computation with Finitely Presented Groups, *Cambridge University Press* (1994).

[74] J. G. Stell : Modelling Term Rewriting Systems by Sesqui-Categories, *University of Keele, Dept. of Computer Science, Technical Report TR94-02* (1994).

[75] K. Stokkermans : A Categorical Framework and Calculus for Critical Pair Completion, *Phd Thesis, Royal Institute for Symbolic Computation, Johannes Kepler University, Linz* (1995).

[76] R. Street : Categorical Structures, *in Handbook of Algebra, M. Hazewinkel (ed), vol.1, p530-577* (1992).

[77] J. H. C. Whitehead : On Adding Relations to Homotopy Groups, *Ann. of Math. vol.42 p409-428* (1941).

[78] J. H. C. Whitehead : Note on a Previous Paper Entitled 'On Adding Relations to Homotopy Groups', *Ann. of Math. vol.47 p806-810* (1946).

[79] J. H. C. Whitehead : Combinatorial Homotopy II, *Bull. American Math. Soc. vol.55 p453-496* (1949).

[80] G. Zacharius : Generalised Gröbner Bases in Commutative Polynomial Rings, *Batchelor's Thesis, MIT* (1978).