J. Symbolic Computation (2000) **29**, 5–31
doi: 10.1006/jsco.1999.0294
Available online at http://www.idealibrary.com on IDEAL

Using Rewriting Systems to Compute Left Kan Extensions and Induced Actions of Categories

RONALD BROWN[†‡] AND ANNE HEYWORTH[§¶]

School of Mathematics, University of Wales, Bangor, Gwynedd LL57 1UT, UK

The aim is to apply string-rewriting methods to compute left Kan extensions, or, equivalently, induced actions of monoids, categories, groups or groupoids. This allows rewriting methods to be applied to a greater range of situations and examples than before. The data for the rewriting is called a Kan extension presentation. The paper has its origins in earlier work by Carmody and Walters who gave an algorithm for computing left Kan extensions based on extending the Todd–Coxeter procedure, an algorithm only applicable when the induced action is finite. The current work, in contrast, gives information even when the induced action is infinite.

1. Introduction

This paper extends the usual string-rewriting procedures for words w in a free monoid to 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 $x|ulv \to x|urv$ given by a relation (l, r). The second derives from a given action of certain words on elements, so allowing rewriting $x|F(a)v \to x \cdot a|v$ (a kind of tensor product rule). Further, the elements x and $x \cdot a$ are allowed to belong to different sets.

The natural setting for this rewriting is a *presentation* of the form $kan\langle\Gamma|\Delta|RelB|X|F\rangle$ where:

- Γ, Δ are (directed) graphs;
- $X : \Gamma \to \mathsf{Sets}$ 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 the free category $P\Delta$.

The main result defines rewriting procedures on the $P\Delta$-set

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

[†]Research partially supported by INTAS Project 94-436 ext "Algebraic K-theory, groups and categories".
[‡]E-mail: `r.brown@bangor.ac.uk`
[§]Supported 1995-8 by an EPSRC Earmarked Research Studentship, "Identities among relations for monoids and categories", and 1998–9 by a University of Wales, Bangor, Research Assistantship.
[¶]E-mail: `map130@bangor.ac.uk`

0747–7171/00/010005 + 27 $35.00/0

When such rewriting procedures complete, the associated normal form gives in effect a computation of what we call the *Kan extension defined by the presentation.*

So the power of rewriting theory may now be brought to bear on a much wider range of combinatorial enumeration problems. Traditionally, string-rewriting is used for solving the word problem for monoids. It has also been used for coset enumeration problems (Redfern, 1993; Holt and Hurt, 1999). It may now also be used in the specification of

- equivalence classes and equivariant equivalence classes,
- arrows of a category or groupoid,
- right congruence classes given by a relation on a monoid,
- orbits of an action of a group or monoid,
- conjugacy classes of a group,
- coequalizers, pushouts and colimits of sets,
- induced permutation representations of a group or monoid

and many others (see Section 8).

In this paper we are concerned with the description of the theory and the implementation in GAP of the procedure with respect to one ordering. It is hoped to consider implementation of efficiency strategies and other orderings on another occasion. The advantages of our abstraction should then become even clearer, since one efficient implementation will be able to apply to a variety of situations, including some not yet apparent.

The papers by Walters *et al.* (Bush *et al.*, 1997; Carmody and Walters, 1990, 1991) on generalized Todd–Coxeter procedures for left Kan extensions, together with the implementation (Fleming *et al.*, 1996) by Rosebrugh were very influential on the current work. Our work generalizes Knuth–Bendix procedures and so we work with a set of rules rather than tables which would build up a catalogue of elements. Our techniques can give results when the induced action is infinite, and this is the main advantage of rewriting here. Further work is needed to make a detailed comparison of efficiency between the procedures at the group level of specialization. At this general level where the procedures have yet to be implemented in the same environment it is not possible to make a fair comparison of the implementations. However, in the special cases of groups and monoids, it can be remarked that sometimes the Knuth–Bendix procedures are thought more efficient and sometimes it is the Todd–Coxeter which are more efficient.

2. Kan Extensions of Actions

The general definitions of Kan extensions may be found in Mac Lane (1971). The case which we consider is the left Kan extension where the codomain is the category of sets. To avoid repeating the phrase "left Kan extension over Sets" endlessly, we simply refer to the "Kan extension". The categorical definition of this case follows.

Let A be a category. A *category action* X of A is a contravariant 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 contravariant functor. The action of identity arrows is trivial, so if id is an identity arrow at A then $x \cdot id = x$ for all $x \in XA$.

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 covariant functor. Then an *extension of the action X along F* is a pair (K, ε) where $K : \mathsf{B} \to \mathsf{Sets}$ is a contravariant 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 (K, ε) of the action 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 $\varepsilon' = \alpha \circ \varepsilon$. Thus K may thought of as the universal extension of the action of A to an action of B.

An alternative and possible more concrete description of the Kan extension is to form the disjoint union $\overline{X}$ of the sets $X(A)$ for all A in $\mathrm{Ob}\mathsf{A}$. There is then a partial action of the category on the set $\overline{X}$. This will be described in detail in Section 4. The functor $F : \mathsf{A} \to \mathsf{B}$ then determines what is often called an *induced partial action* of the category B on a set $F_*(\overline{X})$, which is a disjoint union of sets $F_*(\overline{X})_B$ for all B in $\mathrm{Ob}\mathsf{B}$. This gives a functor F_* on actions of A which is left adjoint to the "composite functor" F^* from actions of B to actions of A. In this way we see how Kan extensions, or induced actions, are related to the problems and examples listed in the introduction. This will be pursued in detail in Section 8.

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 a definition of a *presentation* of a Kan extension. This definition turns out to be a special case of the definition of Carmody and Walters who required a more general notion to capture the content of the Todd–Coxeter procedure.

First, we set out our notation for free categories. Let Δ be a directed graph, that is, Δ consists of two functions $src, tgt : \mathrm{Arr}\Delta \to \mathrm{Ob}\Delta$. Any small category P has an underlying graph $U\mathsf{P}$. The *free category* $P\Delta$ on Δ consists of the objects of Δ with an identity arrow at each object and non-identity arrows $p : B \to B'$ given by the sequences $(d_1, d_2, \ldots, d_n)$ of arrows of Δ which are composable, i.e. $tgt(d_i) = src(d_{i+1}), i = 1, \ldots, n-1$, and such that $src(d_1) = B, tgt(d_n) = B'$. As usual, such a word is written $d_1 \ldots d_n : B \to B'$, and composition is by juxtaposition. Of course, the free functor P is left adjoint to the forgetful functor U.

A *graph of relations* Rel for the free category $P\Delta$ has objects those of Δ and arrows $B \to B'$ sets of pairs (l, r) such that $l, r : B \to B'$ in Δ. Then the quotient category $P\Delta/Rel$ is defined.

A *presentation* $cat\langle\Delta|Rel\rangle$ for a category B consists of a graph Δ of generators of B and a graph of relations for $P\Delta$ such that the natural morphism of categories $P\Delta \to \mathsf{B}$ induces an isomorphism of categories $(P\Delta)/Rel \to \mathsf{B}$. (For an introduction to category presentations see Mitchell (1972).)

Next, we define "Kan extension data".

DEFINITION 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 3.2. A *Kan extension presentation* is a quintuple $\mathcal{P} := kan\langle\Gamma|\Delta|RelB|X|F\rangle$ where:

(1) Γ and Δ are graphs,
(2) $cat\langle\Delta|RelB\rangle$ is a category presentation,
(3) $X : \Gamma \to U\mathsf{Sets}$ is a graph morphism, and
(4) $F : \Gamma \to UP\Delta$ is a graph morphism.

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

(1) Γ is a generating graph for A and $X : \Gamma \to \mathsf{Sets}$ is the restriction of $X' : \mathsf{A} \to \mathsf{Sets}$,
(2) $cat\langle\Delta|RelB\rangle$ is a category presentation of B, and
(3) $F : \Gamma \to P\Delta$ 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 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. In other words, a given Kan extension presentation always defines a Kan extension data where A is the free category $P\Gamma$ and (X', F') are induced by X, F. 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.

4. P-sets

In this section we establish the concepts and notation used to apply rewriting procedures to presentations of Kan extensions of actions. Our terminology is modelled on the standard in rewriting theory.

DEFINITION 4.1. Let P be a category. 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 satisfies the following properties for all $t \in T, p, q \in \mathrm{Arr}\mathsf{P}$:

(1) if $\tau(t) = src(p)$, then $t \cdot p$ is defined and $\tau(t \cdot p) = tgt(p)$;
(2) $t \cdot id_{\tau(t)} = t$; and
(3) $(t \cdot p) \cdot q = t \cdot (pq)$ if the left-hand side is defined.

Note: the "stronger" notion of discrete fibration could be used here, but the above definition is more consistent with rewriting.

DEFINITION 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 $\tau(t_1) = \tau(t_2)$. The 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 $t_1 \cdot p \to t_2 \cdot p$ for all $p \in \mathrm{Arr}\mathsf{P}$ such that $src(p) = \tau(t_1)$.

For the rest of this paper 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

$$T := \bigsqcup_{B\in\mathrm{Ob}\Delta} \bigsqcup_{A\in\mathrm{Ob}\Gamma} XA \times \mathsf{P}(FA, B). \tag{1}$$

The elements of the set T will be referred to as *terms*, and a pair $(x,p) \in XA \times \mathsf{P}(FA,B)$ will be written $x|p$. The function $\tau : T \to \mathrm{Ob}\mathsf{P}$ is defined by

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

Then T becomes a P-set by the action

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

A *rewrite system* for a Kan presentation $\mathcal{P}$ is a pair $R := (R_T, R_P)$ such that:

(1) R_T is a reduction relation on the P-set T; and
(2) R_P is a set of relations on P—pairs of paths in P sharing the same source and the same target.

The *initial rewrite system* that results from the presentation is the pair $R_{\mathrm{init}} := (R_\varepsilon, R_K)$ defined as follows.

$$\begin{aligned} R_\varepsilon &:= \{(x|Fa, x\cdot a|id_{FA_2}) | x \in XA_1, a \in \Gamma(A_1,A_2), A_1, A_2 \in \mathrm{Ob}\Gamma\}.\\ R_K &:= RelB. \end{aligned}$$

The first type of rule we call the "ε-rules" $R_\varepsilon \subseteq T \times T$. They are to ensure that the action is an extension by F of the action of $P\Gamma$—this is the requirement for $\varepsilon : X \to KF$ to be a natural transformation.

The second type we call the "K-rules" $R_K \subseteq \mathrm{Arr}\mathsf{P} \times \mathrm{Arr}\mathsf{P}$. They are to ensure that the action preserves the relations and so gives a functor on the quotient $\mathsf{B} = (P\Delta)/RelB$.

Remark 4.3. If the Kan extension presentation is finite, then R_{init} is finite. The number of initial rules is by definition $(\Sigma_{a\in\mathrm{Arr}\Gamma}|X(src(a))|) + |RelB|$.

Definition 4.4. The *reduction relation* $\to_R$ *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:

(1) There exist $(s,u) \in R_T, q \in \mathrm{Arr}\mathsf{P}$ such that $t_1 = s\cdot q$ and $t_2 = u\cdot q$.
(2) 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 relation $\xrightarrow{*}_R$ is defined to be the reflexive, transitive closure of $\to_R$ on T, and $\xleftrightarrow{*}_R$ is the reflexive, symmetric, transitive closure of $\to_R$.

Remark 4.5. 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—these might be called "tagged rewrite rules", the "tag" being the part x to the left of the separator of $x|p$, but in a more general sense than previous uses since the tags are being rewritten.

LEMMA 4.6. *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.*

The proof is straightforward.

The equivalence class of $t \in T$ under $\overset{*}{\leftrightarrow}_R$ will be denoted $[t]$. A suggestive notation for the class $[x|p]$ would also be $x \otimes p$.

We apply the standard terminology of reduction relations to the reduction relation $\to_R$ on T. In particular, we have a notion of $\to_R$ being complete. A rewrite system $R := (R_T, R_P)$ will be called *complete* when $\to_R$ is complete. In this case $\overset{*}{\leftrightarrow}_R$ admits a normal form function.

We expect that 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 $\bigsqcup_B KB$;
- a function $\overline{\tau} : \bigsqcup_B KB \to \mathrm{Ob}\mathsf{B}$;
- a partial function (action) $\bigsqcup_B KB \times \mathrm{Arr}\mathsf{P} \to \bigsqcup_B KB$; and
- and a function $\varepsilon : \bigsqcup_A XA \to \bigsqcup_B KB$.

Here, $\bigsqcup_B KB$ and $\bigsqcup_A XA$ are the disjoint unions of the sets KB, XA over $\mathrm{Ob}\mathsf{B}$, $\mathrm{Ob}\mathsf{A}$, respectively; if $z \in KB$, then $\overline{\tau}(z) = B$ and if further $src(p) = B$ for $p \in \mathrm{Arr}\mathsf{P}$, then $z \cdot p$ is defined.

THEOREM 4.7. *Let $\mathcal{P} = kan\langle \Gamma|\Delta|RelB|X|F\rangle$ be a Kan extension presentation, and let* P*, T, $R = (R_\varepsilon, R_K)$ be defined as above. Then the Kan extension (K, ε) presented by $\mathcal{P}$ may be given by the following data:*

(1) the set $\bigsqcup_B KB = T/\overset{*}{\leftrightarrow}_R$,
(2) the function $\overline{\tau} : \bigsqcup_B KB \to \mathrm{Ob}\mathsf{B}$ induced by $\tau : T \to \mathrm{Ob}\mathsf{P}$,
(3) the action of B *on $\bigsqcup_B KB$ induced by the action of* P *on T, and*
(4) the natural transformation ε determined by $x \mapsto [x|id_{FA}]$ for $x \in XA$, $A \in \mathrm{Ob}\mathsf{A}$.

PROOF. We give the proof in some detail since this is helpful for the implementations described in the next section.

Claim $\overset{*}{\leftrightarrow}$ preserves the function τ.

PROOF. 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 $t_1 = s_1 \cdot q$ and $t_2 = s_2 \cdot q$ for some $q \in \mathrm{Arr}\mathsf{P}$. Clearly $\tau(t_1) = \tau(t_2)$. For the other case suppose that there exist $(l, r) \in R_K$ such that $t_1 = s \cdot (lq)$ and $t_2 = s \cdot (rq)$ for some $s \in T$, $q \in \mathrm{Arr}\mathsf{P}$. Again, it is clear that $\tau(t_1) = \tau(t_2)$. Hence $\overline{\tau} : T/\overset{*}{\leftrightarrow}_R \to \mathrm{Ob}\mathsf{P}$ is well defined by $\overline{\tau}[t] = \tau(t)$. □

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 $(\overset{*}{\leftrightarrow}_{RelB}) \subseteq (\overset{*}{\leftrightarrow})$. Therefore P acts on $T/\overset{*}{\leftrightarrow}$. This action preserves the relations of B and so defines an action of B on $T/\overset{*}{\leftrightarrow}$. Furthermore, $\overline{\tau}([t] \cdot p) = \overline{\tau}[t \cdot p] = tgt(p)$ 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

$$KB := \{[x|p] : \overline{\tau}[x|p] = B\}. \tag{2}$$

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

$$Kb : KB_1 \to KB_2 : [t] \mapsto [t \cdot p] \qquad \text{for } [t] \in KB_1 \text{ where } p \in [b]. \tag{3}$$

It can be verified that this definition of the action is a functor $K : \mathsf{B} \to \mathsf{Sets}$. Then define

$$\varepsilon : X \to KF : x \mapsto [x|id_{FA}] \qquad \text{for } x \in XA, A \in \mathrm{Ob}\mathsf{A}. \tag{4}$$

It is straightforward to verify that this is a natural transformation. 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 $\alpha : K \to K'$, defined by

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

is a natural transformation which satisfies $\varepsilon \circ \alpha = \varepsilon'$ and is clearly the only such. □

5. Rewriting Procedures for Kan Extensions

In this section we will explain the completion process for the initial rewrite system. To this end we give a convenient notation for the implementation of the data structure for a *finite* presentation $\mathcal{P}$ of a Kan extension.

5.1. structure of input data

Kan presentation data is input in the form of various lists. Note that in so doing, we are in each case choosing an arbitrary order on the elements of the list. The notation we use is chosen to reflect our later GAP implementation:

`ObA`	This is a list of objects of Γ.
`ArrA`	This is a list of arrows of Γ.
`ObB`	This is a list of objects of Δ.
`ArrB`	This is a list of arrows of $\mathrm{Arr}\Delta$.
`RelB`	This is a finite list of the pairs of paths which are the relations defining B.
`FObA`	This is a list of the images of the objects of Γ under the functor F—objects of Δ.
`FArrA`	This is a list of the images of the arrows of Γ under the functor F—arrows of P.
`XObA`	This is a list of the images of the objects of Γ under the functor X—sets of distinct elements.
`XArrA`	This is a list of the images of the arrows of Γ under the functor X—functions between pairs of sets.

All the above lists are finite since in this paper we are dealing only with finite Kan presentation data. In Section 7 we show by example how to input this data.

5.2. lists

Elements of T are called *terms* and are represented by lists of generators, where the generators may be thought of as labels. The first entry in the list must be a label for an element of XA for some $A \in \mathrm{Ob}\Gamma$. The subsequent entries will be labels for composable arrows of Δ, with the source of the first being FA. Formally, an element $t \in T$ is represented by a list

$$\mathtt{List}(x|p) = \begin{cases} [x, b_1, \ldots, b_n] & \text{if } p = b_1 \ldots b_n, n \geqslant 1, \\ [x, 1_{FA}] & \text{if } p = 1_{FA}. \end{cases}$$

This also allows us to use list notation, so that if $t = x|b_1 \ldots b_n$, then $t[1] = x, t[i+1] = b_i, 1 \leqslant i \leqslant n$. Also, $\mathtt{Length}(t)$ means the number of elements in the list corresponding to t and $\mathtt{Position}(\mathtt{ObA}, \mathtt{A})$ returns the position of the element A in the list `ObA`. If $t = [x|p]$ we also write $t[2..]$ for p.

5.3. initial rules procedure

Algorithm 5.1. (Initial Rules) Given the data for a Kan presentation in the form of a record with the fields named as above, the initial rewrite system $R_{\text{init}} := (R_\varepsilon, R_K)$ is determined.

(1) (Input:) `ObA, ArrA, ObB, ArrB, RelB, FObA, FArrA, XObA, XArrA`.
(2) (Procedure:) Set $R_\varepsilon := \emptyset$, then for each arrow $a \in \mathtt{ArrA}$, set $\mathtt{i} := \mathtt{Position}(\mathtt{ArrA}, \mathtt{a})$; $\mathtt{XA} := \mathtt{XObA}[\mathtt{Position}(\mathtt{ObA}, \mathtt{a}[1])]$; $\mathtt{Xa} := \mathtt{XArrA}[\mathtt{i}]$; and set $\mathtt{Fa} := \mathtt{FArrA}[\mathtt{i}]$. Then for each element `x` in `XA` , set $\mathtt{j} := \mathtt{Position}(\mathtt{XA}, \mathtt{x})$ and add the rule $[\mathtt{x} * \mathtt{Fa}, \mathtt{Xa}[\mathtt{j}]]$ to R_ε. Set $R_K := \mathtt{RelB}$.
(3) (Output:) $R_{\text{init}} := (R_\varepsilon, R_K)$.

5.4. ORDERINGS

To work with a rewrite system R on T we will require certain concepts of order on T. We give properties of orderings $>_X$ on $\bigsqcup_A XA$ and $>_P$ on $\mathrm{Arr}\mathsf{P}$ to enable us to construct an ordering $>_T$ on T with the properties needed for the rewriting procedures.

DEFINITION 5.2. A binary operation $>$ on a set S is called a *strict partial ordering* if it is irreflexive, antisymmetric and transitive. It is called a *total ordering* if also for all $x, y \in S$ either $x > y$ or $y > x$ or else $x = y$. An ordering $>$ is *well founded* on S if there is no infinite sequence $x_1 > x_2 > \cdots$ of elements of S. An ordering $>$ is a *well ordering* if it is well founded and a total ordering.

DEFINITION 5.3. Let $>_P$ be a strict partial ordering on $\mathrm{Arr}\mathsf{P}$. It is called a *total path ordering* if it induces a total order on $\mathsf{P}(B, B')$ for all objects $B, B' \in \mathsf{P}$. It is called a *well ordering* if it is well founded and a total path ordering. 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 $upv, uqv \in \mathrm{Arr}\mathsf{P}$. An admissible well ordering is called a *monomial ordering.*

LEMMA 5.4. *Let $>_X$ be a well ordering on the finite set $\bigsqcup_A XA$ and let $>_P$ be an admissible well ordering on P. For $t_1, t_2 \in T$ define*

$$\begin{aligned} t_1 >_T t_2 \ \textit{if}\ & t_1[2..] >_P t_2[2..] \ \textit{or} \\ & t_1[2..] = t_2[2..] \ \textit{and}\ t_1[1] >_X t_2[1]. \end{aligned}$$

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 $>_P$. Since $>_X$ and $>_P$ are both well founded there are no such sequences. $\Box$

The last result shows that there is scope for choosing different orderings on T. The actual choice is even wider than this, and is related to efficiency, see Holt and Hurt (1999)—there may even be completion with respect to one order and not another. We do not discuss these matters here.

In this paper we work only with a "length-lexicographical ordering" defined in the following way.

DEFINITION 5.5. (IMPLEMENTED ORDERING) Let $>_X$ be any linear order on (the finite set) $\bigsqcup_A XA$. Let $>_\Delta$ be a linear ordering on (the finite set) $\mathrm{Arr}\Delta$. This induces an admissible ordering $>_P$ on Arr$\mathsf{P}$ where

$$\begin{aligned} p >_P q \Leftrightarrow\ & \mathtt{Length}(p) > \mathtt{Length}(q) \\ \text{or}\ & \mathtt{Length}(p) = \mathtt{Length}(q) \text{ and there exists } k > 0 \text{ such that} \\ & p[i] = q[i] \text{ for all } i < k \text{ and } p[k] >_\Delta q[k]. \end{aligned}$$

The ordering $>_T$ is then defined as follows:

$$\begin{aligned} t_1 >_T t_2 \text{ if }\ & \mathtt{Length}(t_1) > \mathtt{Length}(t_2) \\ \text{or}\ & \mathtt{Length}(t_1) = \mathtt{Length}(t_2) \text{ and } t_1[1] >_X t_2[1] \\ \text{or}\ & \mathtt{Length}(t_1) = \mathtt{Length}(t_2) \text{ and there exists } k \in [1..\mathtt{Length}(t_1)] \\ & \text{such that } t_1[i] = t_2[i] \text{ for all } i < k, \text{ and } t_1[k] >_\Delta t_2[k]. \end{aligned}$$

PROPOSITION 5.6. *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 $>_X$, $>_\Delta$). 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 $t_1 >_T t_2 >_T t_3 > \cdots$ is an infinite sequence. Then for each $i > 0$ either $\mathtt{Length}(t_i) > \mathtt{Length}(t_{i+1})$ or if $\mathtt{Length}(t_i) = \mathtt{Length}(t_{i+1})$ and $t_i[1] >_X t_{i+1}[1]$, or if $\mathtt{Length}(t_i) = \mathtt{Length}(t_{i+1})$ and there exists $k \in [1..\mathtt{Length}(t_i)]$ such that $t_i[j] = t_{i+1}[j]$ for all $j < k$ and $t_i[k] >_\Delta t_{i+1}[k]$. 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 $\bigsqcup_A XA$ or else of type $p_1 >_\Delta p_2 >_\Delta p_3 >_\Delta \cdots$ of arrows of Δ, none of which is possible as $>$, $>_X$, and $>_\Delta$ are well founded on $\mathbb{N}$, $\bigsqcup_A XA$ and ArrΔ, respectively. Hence $>_T$ is well founded. □

PROPOSITION 5.7. *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 5.8. The proposition can also be proved for the earlier definition of $>_T$ induced from $>_X$ and $>_P$.

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 5.9. *Let R be a rewrite system on T. Orientate the rules of R so that for all (l, r) in R, if $l, r \in$ Arr*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, by Definition 4.2. For the first case let $t_1 = s_1 \cdot p$, $t_2 = s_2 \cdot p$ for some $s_1, s_2 \in T$, $p \in$ 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 5.7 $s \cdot p_1 >_T s \cdot p_2$. Hence $t_1 >_T t_2$ by admissibility of $>_T$ on T. Therefore, in either case $t_1 >_T t_2$ so $\rightarrow_R$ is compatible with $>_T$. □

It is a standard result that if a reduction relation is compatible with an admissible well ordering, then it is Noetherian. The next algorithm describes the function `Reduce`.

Algorithm 5.10. (Reduce) Given a term $t \in T$ and a rewrite system $R = (R_P, R_P)$ a term $t_n \in [t]$, which is irreducible with respect to $\rightarrow_R$, is determined.

Recall that all terms are represented by lists. Sucessively replace sublists l_i of t which occur as the left-hand side of rules (l_i, r_i) of R with the right-hand side r_i of the rule. Stop when no left-hand side can be found as a sublist and return the modified list t'.

5.6. critical pairs

We can now discuss what properties of R will make $\rightarrow_R$ a complete (Noetherian and confluent) reduction relation. By standard abuse of notation the rewrite system R will be called complete when $\rightarrow_R$ is complete. The following result is called Newman's lemma (Baader and Nipkow, 1998).

Lemma 5.11. *A Noetherian reduction relation on a set is confluent if it is locally confluent.*

Hence, if R is compatible with an admissible well ordering on T and $\rightarrow_R$ is locally confluent, then $\rightarrow_R$ is complete. By orienting the pairs of R with respect to the chosen ordering $>_T$ on T, R is made to be Noetherian. The problem remaining 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 our situation. In particular, the overlaps involve either just R_T, or just R_P or an interaction between R_T and R_P.

Definition 5.12. 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 $crit1 \neq crit2$. A pair $(crit1, crit2)$ of distinct terms 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 (common) term res such that both $crit1$ and $crit2$ reduce to a res, i.e. $crit1 \xrightarrow{*}_R res$, $crit2 \xrightarrow{*}_R res$.

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.

If $t = x|b_1 \cdots b_n$, then a **part** of t is either a term $x|b_1 \cdots b_i$ for some $1 \leqslant i \leqslant n$ or a word $b_i b_{i+1} \cdots b_j$ for some $1 \leqslant i \leqslant j \leqslant n$.

Definition 5.13. Let $(rule1, rule2)$ be a pair of rules of the rewrite system $R = (R_T, R_P)$ where $R_T \subseteq T \times T$ and $R_P \subseteq \mathrm{Arr}\mathsf{P} \times \mathrm{Arr}\mathsf{P}$. If $rule1$ and $rule2$ may both be applied to the same term *crit* in such a way there is a part of the term *crit* that is affected by both the rules then we say that an *overlap* occurs.

There are five types of overlap for this kind of rewrite system, as shown in the following table:

#	rule1	in	rule2	in	overlap	critical pair
(i)	(s_1, u_1)	R_T	(s_2, u_2)	R_T	$s_2 = s_1 \cdot q$ for some $q \in \mathrm{Arr}\mathsf{P}$	$(u_1 \cdot q, u_2)$
(ii)	(l_1, r_1)	R_P	(l_2, r_2)	R_P	$l_1 = pl_2q$ for some $p, q \in \mathrm{Arr}\mathsf{P}$	(r_1, pr_2q)
(iii)					$l_1q = pl_2$ for some $p, q \in \mathrm{Arr}\mathsf{P}$	(r_1q, pr_2)
(iv)	(s_1, u_1)	R_T	(l_1, r_1)	R_P	$s_1 \cdot q = s \cdot l_1$ for some $s \in T, q \in \mathrm{Arr}\mathsf{P}$	$(u_1 \cdot q, s \cdot r_1)$
(v)					$s_1 = s \cdot (l_1q)$ for some $s \in T, q \in \mathrm{Arr}\mathsf{P}$	$(u_1, s \cdot r_1q)$

Overlap table

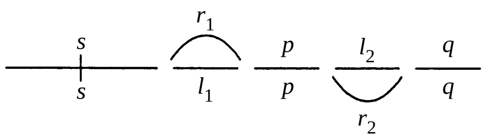
A pair of rules may overlap in more than one way, giving more than one critical pair. For example the rules $(x|a^2ba, y|ba)$ and (a^2, b) overlap with critical term $x|a^2ba$ and critical pair $(y|ba, x|b^2a)$ and also with critical term $x|a^2ba^2$ and critical pair $(y|ba^2, x|a^2b^2)$.

LEMMA 5.14. *Let R be a finite rewrite system on the P-set T. Consider applications of rules rule1 and rule2 affecting part c of term $t \in T$, resulting in a critical pair (c_1, c_2) from c and (t_1, t_2) from t. If there is no overlap, then (t_1, t_2) resolves immediately. Otherwise (t_1, t_2) resolves providing (c_1, c_2) does.*

PROOF. Let (t_1, t_2) be a critical pair. Then there exists a critical term t and two rules *rule*1, *rule*2 such that t reduces to t_1 with respect to *rule*1 and to t_2 with respect to *rule*2.

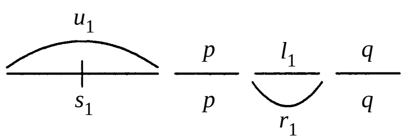
We first give the two non-overlap cases.

Suppose $rule1 := (l_1, r_1)$, $rule2 := (l_2, r_2) \in R_P$. Then there exist $s \in T$, $p, q \in \mathrm{Arr}\mathsf{P}$ such that $t = s \cdot l_1pl_2q$ as shown:



The pair (t_1, t_2) immediately resolves to $s \cdot r_1pr_2q$ by applying *rule*2 to t_1 and *rule*1 to t_2.

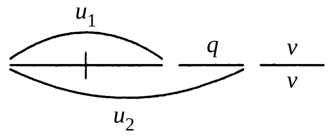
Suppose that $rule1 := (s_1, u_1) \in R_T$ and $rule2 := (l_1, r_1) \in R_P$ and the rules do not overlap. Then there exist $p, q \in \mathrm{Arr}\mathsf{P}$ such that $t = s_1 \cdot pl_1q$ and then $t_1 = u_1 \cdot pl_1q$ and $t_2 = s_1 \cdot pr_1q$ as shown:



The pair (t_1, t_2) immediately resolves to $u_1 \cdot pr_1q$ by applying *rule*2 to t_1 and *rule*1 to t_2.

We now give the overlap cases in the order given in the table.

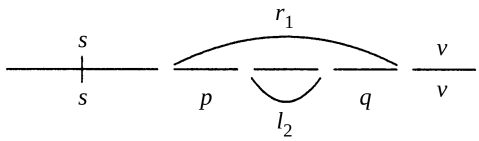
(i) Suppose $rule1 := (s_1, u_1), rule2 := (s_2, u_2) \in R_T$. Then there exist $v, q \in \mathrm{Arr}\mathsf{P}$ such that $c = s_1 \cdot q = s_2$, $t = c \cdot v$ and then $t_1 = u_1 \cdot qv$ and $t_2 = u_2 \cdot v$ as shown:



The critical pair here is $(u_1 \cdot q, u_2)$ and if this resolves to r, then (t_1, t_2) resolves to $r \cdot v$.

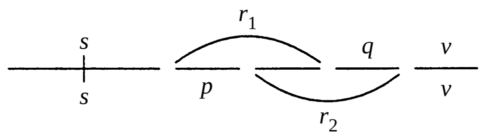
Suppose $rule1 := (l_1, r_1)$, $rule2 := (l_2, r_2) \in R_P$. There are two possible overlap cases.

(ii) In the first case there exist $s \in T$, $p, q, v \in \mathrm{Arr}\mathsf{P}$ such that $c = l_1 = pl_2q$ and $t = s \cdot cv$ and then $t_1 = s \cdot r_1 v$ and $t_2 = s \cdot pr_2qv$.



The critical pair here is (r_1, pr_2q) and if this resolves to r, then (t_1, t_2) resolves to $s \cdot rv$.

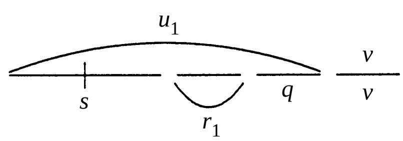
(iii) In the second case there exist $s \in T$, $p, q, v \in \mathrm{Arr}\mathsf{P}$ such that $c = l_1 q = pl_2$ and $t = s \cdot cv$ and then $t_1 = s \cdot r_1 qv$ and $t_2 = s \cdot pr_2 v$.



The critical pair is $(r_1 q, pr_2)$ and if this resolves to r, then (t_1, t_2) resolves to $s \cdot rv$.

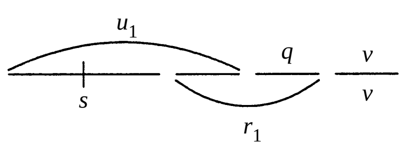
Suppose finally that $rule1 := (s_1, u_1) \in R_T$ and $rule2 := (l_1, r_1) \in R_P$. Then there are two possible overlap cases.

(iv) In the first case there exist $s \in T$, $q, v \in \mathrm{Arr}\mathsf{P}$ such that $c = s_1 = s \cdot l_1 q$ and $t = c \cdot v$ and then $t_1 = u_1 v$ and $t_2 = sr_1qv$.



The critical pair is $(u_1, s \cdot r_1 q)$ and if this resolves to r, then (t_1, t_2) resolves to $r \cdot v$.

(v) In the second case there exist $s \in T$, $q, v \in \mathrm{Arr}\mathsf{P}$ such that $c = s_1 \cdot q = s \cdot l_1$ and $t = c \cdot v$ and then $t_1 = u_1 \cdot qv$ and $t_2 = s \cdot r_1 v$.



The critical pair is $(s_1 \cdot q, s \cdot r_1)$ and if this resolves to 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 5.15. *If all the overlaps between rules of a rewrite system R on T resolve, then all the critical pairs for the reduction relation $\to_R$ resolve, and so $\to_R$ is confluent.*

Proof. This is immediate from Lemma 5.14. □

Lemma 5.16. *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 $\mathtt{List}(t)$ is the representation of a term $t \in T$ as a list. The first type of list overlap occurs when $\mathtt{List}(l_2)$ is a sublist of $\mathtt{List}(l_1)$ (or vice versa). This happens in cases (i), (ii) and (v). The second type of list overlap occurs when the end of $\mathtt{List}(l_1)$ matches the beginning of $\mathtt{List}(l_2)$ (or vice versa). This happens in cases (iii) and (iv). □

The program for finding overlaps and the resulting critical pairs is outlined in the algorithm below.

Algorithm 5.17. (Critical Pairs) Given a rewrite system R all critical pairs are determined.

Recall that terms are represented by lists. Take pairs of rules (l_1, r_1) and $(l_2, r_2)$from R. Test (a) whether $\mathtt{List}(l_2)$ is a sublist of $\mathtt{List}(l_1)$. If it is then find u and v such that $u \cdot l_2 v = l_1$. Add the critical pair $(u \cdot r_2 v, r_1)$ to a list $CRIT$. Now test (b) whether for $i = 1, 2 \ldots$ the sublist of length i at the right of $\mathtt{List}(l_1)$ is equal to the sublist of length i on the left of $\mathtt{List}(l_2)$. For each i where this occurs, set u to be the part of $\mathtt{List}(l_1)$ not in the overlap, and v to be the part of $\mathtt{List}(l_2)$ not in the overlap. Add the critical pair $(r_1 \cdot v, u \cdot r_2)$ to $CRIT$. Repeat the procedure until all (ordered) pairs of rules have been examined for overlaps. Then $CRIT$ is an exhaustive list of critical pairs from R.

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.

5.7. completion procedure

We have shown: (i) how to find overlaps between rules of R; (ii) how to test whether the resulting critical pairs resolve; and (iii) that if all the critical pairs resolve, then this implies $\to_R$ is confluent. We now show that critical pairs which do not resolve may be added to R without affecting the equivalence relation R defines on T.

Lemma 5.18. *Any critical pair (t_1, t_2) of a rewrite system R may be added to the rewrite system without changing the equivalence relation $\overset{*}{\leftrightarrow}_R$.*

Proof. By definition, (t_1, t_2) 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 $\overset{*}{\leftrightarrow}_R$. □

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 R^C. The benefit of such a system is that $\rightarrow_{R^C}$ then acts as a normal form function for $\overset{*}{\leftrightarrow}_{R^C}$ on T.

THEOREM 5.19. *Let $\mathcal{P} = \langle \Gamma | \Delta | RelB | X | F \rangle$ be a finite presentation of a Kan extension (K, ε). Let $P := P\Delta$, $T := \bigsqcup_{\mathrm{Ob}\Delta} \bigsqcup_{\mathrm{Ob}\Gamma} XA \times \mathsf{P}(FA, B)$, and let R 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 the ordering $>_T$ and such that the equivalence relations $\overset{*}{\leftrightarrow}_R$, $\overset{*}{\leftrightarrow}_{R^C}$ 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. □

ALGORITHM 5.20. (COMPLETION) Given the presentation of a Kan extension and the ordering $>_T$, a complete rewrite system with respect to $>_T$ is determined—if the algorithm terminates.

(1) (Input:) A rewrite system R on T and an ordering $>_T$ on T.
(2) (Initialize:) Set $NewRules := R$ and $OldRules := \emptyset$.
(3) (Loop:) While $NewRules \neq OldRules$, set $OldRules := NewRules$. Use the algorithm Critical Pairs to determine all the critical pairs of $NewRules$. Remove each critical pair in turn from the list, and reduce both sides of the pair with respect to $NewRules$ using the algorithm Reduce. If the left entry is greater than the right (with respect to $>_T$), then add the reduced critical pair to $NewRules$. If the right entry is greater than the left, then add the reversed, reduced critical pair to $NewRules$. Repeat this loop until all critical pairs resolve and no rules are added.
(4) (Output:) A complete rewrite system $NewRules$ on T.

Supposing that the completion procedure outlined above terminates, we will now briefly discuss how to interpret the complete rewrite system on T.

6. Interpreting the Output

6.1. FINITE ENUMERATION OF THE KAN EXTENSION

When every set KB is finite we may catalogue the elements of all of the sets $\bigsqcup_B KB$ in stages.

The first stage catalogues the elements $x|id_{FA}$ where $x \in XA$ for some $A \in \mathrm{Ob}\Gamma$. These elements are considered to have length one. 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 two. The next stage 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 $\bigsqcup_B KB$. The subsets KB of $\bigsqcup_B KB$ are determined by the function $\overline{\tau}$, i.e. if $x|b_1 \cdots b_n$ is a normal form in $\bigsqcup_B KB$ 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*.

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 of Heyworth (1998).

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 $(X', G' \circ F')$

Lemma 6.1. *Let $kan\langle \Gamma | \Delta | RelB | X | F \rangle$ be a presentation for a Kan extension (K, ε). Let $cat\langle \Lambda | RelC \rangle$ present a category C and let $G' : \mathsf{B} \to \mathsf{C}$ be a functor. Then the Kan extension presented by $kan\langle \Gamma | \Lambda | RelC | X | G \circ F | \rangle$ is equal to the Kan extension presented by $kan\langle \Delta | \Lambda | RelC | K | G \rangle$.*

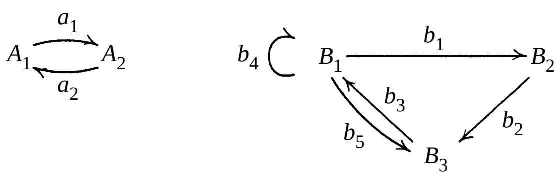
Proof. Let $kan\langle \Gamma | \Delta | RelB | X | F \rangle$ present the Kan extension data (X', F') for the Kan extension (K, ε). Let C be a category finitely presented by $cat\langle \Lambda | RelC \rangle$ and let $G' : \mathsf{B} \to \mathsf{C}$. Then $kan\langle \Delta | \Lambda | RelC | K | G \rangle$ presents the Kan extension data (K', G') for the Kan extension (L, η).

We require to prove that $(L, \eta \circ \varepsilon)$ is the Kan extension presented by $kan\langle \Gamma | \Lambda | RelC | X | G \circ F \rangle$ having data $(X', G' \circ F')$. It is clear that $(L, \eta \circ \varepsilon)$ defines an extension of the action X along $G \circ F$ because L defines an action of C and $\eta \circ \varepsilon : X \to L \circ G \circ F$ 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 $(M \circ G, \nu)$, it is an extension of X along F. Therefore there exists a unique natural transformation $\alpha : X \to M \circ G \circ F$ such that $\alpha \circ \varepsilon = \nu$ 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 $\beta \circ \eta = \alpha$ by universality of (L, η). Therefore β is the unique natural transformation such that $\beta \circ \eta \circ \varepsilon = \nu$, which proves the universality of the extension $(L, \eta \circ \varepsilon)$. □

7. Example of a GAP Session on the Rewriting Procedure

Here we give an example to show the use of the implementation. Let A and B be the categories generated by the graphs below, where B has the relation $b_1b_2b_3 = b_4$.



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 read in the program and set up the variables:

```
gap> RequirePackage("kan");
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 (choice of names is unimportant):

```
gap> OBJa:=[1,2];;
gap> ARRa:=[[1,2],[2,1]];;
gap> OBJb:=[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 (the field names are important):

```
gap> KAN:=rec( ObA:=OBJa, ArrA:=ARRa,  ObB:=OBJb, ArrB:=ARRb,
               RelB:=RELb,
FObA:=fOBa, FArrA:=fARRa, XObA:=xOBa,
               XArrA:=xARRa );;
```

To calculate the initial rules do:

```
gap> 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:

$$(x_1|Fa_1, x_1.a_1|id_{FA_2}),\ (x_2|Fa_1, x_2.a_1|id_{FA_2}),\ (x_3|Fa_1, x_3.a_1|id_{FA_2}),$$
$$(y_1|Fa_2, y_1.a_1|id_{FA_1}),\ (y_2|Fa_2, y_2.a_1|id_{FA_1}),\ \text{i.e. } x_1|b_1 \to y_1|id_{B_2},$$
$$x_2|b_1 \to y_2|id_{B_2},\ x_3|b_1 \to y_1|id_{B_2},\ y_1|b_2b_3 \to x_1|id_{B_1},\ y_2|b_2b_3 \to x_2|id_{B_1}$$

and one initial K-rule: $b_1b_2b_3 \to b_4$.

To attempt to complete the Kan extension presentation do:

```
gap> KB( InitialRules(KAN) );
```

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. (To change this the user should type `EnumerationLimit:= 5000`—or whatever, after reading in the program.) In fact the above 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:

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

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.

The general method of obtaining regular expressions for these computations will be given in a separate paper (see Chapter 4 of Heyworth, 1998).

8. Special Cases of the Kan Rewriting Procedure

Mac Lane wrote in Section 10.7 of Mac Lane (1971) (entitled "All Concepts are Kan Extensions") that "the notion of Kan extensions subsumes all the other fundamental concepts of category theory". We now illustrate his statement by showing how some familiar problems can be expressed in the terms of a left Kan extension over the category of sets and will see how our computational methods apply to these problems. Most of these examples are also familiar from Carmody and Walters (1990) and Fleming *et al.* (1996). 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.

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 0 and no arrows, so A is the singleton category. Let $X0$ be a one point set on which A acts trivially. 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 $K0$ 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 maps the unique element 0 of $X0$ to the element $K0$ representing the monoid identity. This ensures that the identity of B acts trivially and helps to define the normal form function which is $w \mapsto \varepsilon_0(1)\cdot(w) := Kw(\varepsilon_0(1))$.

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.

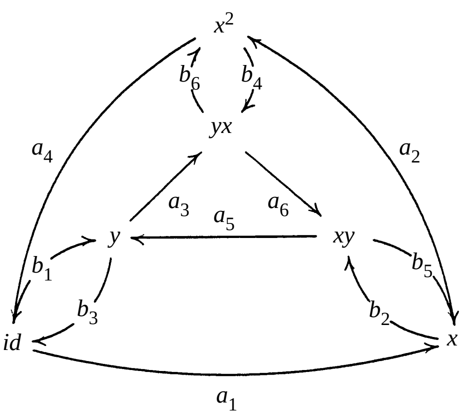
8.2. GROUPOIDS AND CATEGORIES

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 ObΛ. 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 maps the unique element of a set XA to the identity arrow for the object FA of Δ. This makes sure that the identities of B act trivially and helps to define the normal form function which is $w \mapsto \varepsilon_A \cdot (w) := Kw(\varepsilon_A)$.

EXAMPLE 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 $G \times X$:

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

To make calculations clearer, we relabel them $\{a_1, a_2, a_3, \ldots, a_6, b_1, b_2, \ldots, b_6\}$. The groupoid has 18 relators (the boundaries of irreducible cycles of the graph) $G\times R$, 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 $\{A_1, A_2, \ldots, A_6, B_1, B_2, \ldots, B_6\}$ 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 rewrite 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 rewrite
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 obtain 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 cannot build any longer paths; the covering groupoid has 30 morphisms and six identity arrows and is the tree groupoid with six objects.

Example 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

$$\begin{tikzcd} \bullet A \arrow[r, "a"] & \bullet B \arrow[r, "c"] \arrow[loop above, "b"] & \bullet C \arrow[ll, bend left, "d"] \end{tikzcd}$$

with the relations $b^2c = c,\ ab^2 = a$. This rewrite 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 of Heyworth, 1998), and from this we can specify the language which is the set of normal forms. It is in fact

$$a(cd(acd)^*ab + bcd(acd)^*ab) + b^*b + cd(acd)^*ab + d(acd)^*ab$$

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

8.3. COSET SYSTEMS AND CONGRUENCES

Let B be a group considered as a category with one object 0, and let $F : \mathsf{A} \to \mathsf{B}$ be the inclusion of the subgroup A. Let X map the object of A to a one point set. The set $K0$ represents the (right) cosets of A in B, with the right action of any group element b of B taking the representative of the coset $\mathsf{A}g$ to the representative of the coset $\mathsf{A}gb$. The natural transformation ε picks out the representative for the subgroup A.

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 $X0$ 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 $K0$ is a set of representatives for the (right) cosets, Kb defines the (right) action of the group on the cosets $Hg \mapsto Hgb$ and ε_0 picks out the subgroup from the cosets by mapping the single element of $X0$ to the representative for H in $K0$.

The left cosets can be similarly represented, defining the right action K by a left action on the cosets.

Alternatively, let B be a monoid considered as a category with one object 0 and let A be generated by arrows which map under F to a set of generators for a right congruence. Then the set $K0$ represents the congruence classes, the action of any arrow b of B (monoid elements) taking the representative (in $K0$) 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.)

In the monoid 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 8.3. Let B be 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 be 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 for B:

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

Note: on completion of this rewrite 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 the well-known fact that this 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 rewrite 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 rewrite system for the group.)

This time the subgroup has index 2, and the coset representatives are id and c.

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 $K0 := \Omega/\overset{*}{\leftrightarrow}_{Rel}$ 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.

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, then A is G thought of as a category. Let $X0 := \Omega$. 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 $K0$ is a set of representatives for the distinct orbits of the action of the group G on Ω. The action of B on $K0$ is trivial. The natural transformation ε maps each element of the set $X0$ to its orbit representative in $K0$.

We present a short example to demonstrate the procedure in this case.

Example 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 8.5. Consider the quarternion group, presented by

$$\langle a, b \mid a^4, b^4, abab^{-1}, a^2b^2 \rangle,$$

and $\Omega = \{id, a, b, a^2, ab, ba, a^3, a^2b\}$ (we can enumerate the elements using the variation of the Kan extensions 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 $\{id,\ a,\ b,\ a^2,\ ab\}$, and these are representatives of the five conjugacy classes.

8.6. COLIMITS OF DIAGRAMS OF Sets

Let $X : \mathsf{A} \to \mathsf{Sets}$ be any functor on the small category A and let $F : \mathsf{A} \to \mathsf{B}$ be the null functor to the trivial category. Then the Kan extension corresponds to the colimit of (the diagram) $X : \mathsf{A} \to \mathsf{Sets}$; $K0$ is the colimit object, and ε defines the colimit functions from each set XA to $K0$. Examples of this are: (i) when A has two objects A_1 and A_2, and two non-identity arrows $a_1, a_2 : A_1 \to A_2$; the colimit is then the *coequalizer* of the functions Xa_1 and Xa_2 in Sets; (ii) when A has three objects A_1, A_2 and A_3 and two arrows $a_1 : A_1 \to A_2$ and $a_2 : A_1 \to A_3$; the colimit is then the *pushout* of the functions Xa_1 and Xa_2 in 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 $K0$ is the colimit object, with a trivial action of B, and ε defines the colimit functions from each set XA to $K0$.

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 (*coequalizer* 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 the power of rewriting but to show another situation where presentations of Kan extensions can be used to express a problem in rewriting terms.

EXAMPLE 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 coequalizer. We have a number of ε-rules $y_1|id_0 \to x_1|id_0, y_2|id_0 \to x_2|id_0, y_3|id_0 \to x_3|id_0, y_1|id_0 \to x_1|id_0$, $y_2|id_0 \to x_1|id_0, y_3|id_0 \to x_3|id_0$. There is just one overlap, between $(y_2|id_0 \to x_1|id_0)$ and $(y_2|id_0 \to$

$x_2|id_0$): to resolve the critical pair we add the rule $x_2|id_0 \to x_1|id_0$, and the system is complete:

$$\{y_1|id_0 \to x_1 id_0|,\ y_2|id_0 \to x_1|id_0,\ y_3|id_0 \to x_3|id_0,\ x_2|id_0 \to x_1|id_0\}.$$

The elements of the set $K0$ are easily enumerated:

$$x_1|id_0,\ x_2|id_0 \to x_1|id_0,\ x_3|id_0,\ y_1|id_0 \to x_1|id_0,\ y_2|id_0 \to x_1|id_0,$$
$$y_3|id_0 \to x_3|id_0,\ y_4|id_0.$$

So the colimit object is the set

$$K0 = \{x_1|id_0, x_3|id_0, y_4|id_0\},$$

and the coequalizer function to it from XA_2 is given by $y_i \mapsto y_i|id_0$ for $i = 1, \ldots, 4$ followed by reduction defined by $\to$ to an element of $K0$.

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 $\overline{X}$. 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 $F_*(\overline{X})$. It can be constructed simply as the set $\overline{X} \times \mathsf{B}$ factored by the equivalence relation generated by $(xa, b) \sim (x, F(a)b)$ for all $x \in \overline{X}, a \in \mathsf{A}, b \in \mathsf{B}$. 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 $F_*(\overline{X})$ in these cases, as follows.

Suppose first that F is an epimorphism with kernel N. Then we can take as a representative of $F_*(\overline{X})$ 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 $\overline{X} \times T$ 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 factorization of F may not be a convenient way of determining the induced representation.

In the case A, B are monoids, so that X is a transformation representation of A on the set $\overline{X}$, 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. This yields a quotient of the free product of the monoids $\{x\} \times \mathsf{B}$, $x \in \overline{X}$ by the equivalence relation generated by $(x, F(a)b) \sim (x \cdot a, b)$, $a \in \mathsf{A}, b \in \mathsf{B}$.

Acknowledgements

We would like to acknowledge the help given by Larry Lambe in computational and mathematical advice since the early 1990s. He further suggested in 1995 that data structures of free categories implemented by Brown and Dreckmann could be relevant to work of Carmody and Walters on computations of Kan extensions. In visits in 1996 and 1997 under an EPSRC Visiting Fellowship he gave further crucial direction to the work, including suggestions on the connections with Gröbner bases which are developed elsewhere.

References

Baader, F., Nipkow, T. (1998). *Term Rewriting and All That*, New York, Cambridge University Press.

Book, R. V., Otto, F. (1993). *String-Rewriting Systems*, New York, Springer.

Bush, M. R., Leeming, M., Walters, R. F. C. (1997). Computing left Kan extensions. *J. Symb. Comput.*, **11**, 11–20.

Carmody, S., Walters, R. F. C. (1990). The Todd–Coxeter procedure and left Kan extensions. In Research Reports of the School of Mathematics andStatistics, The University of Sydney 90–19, with M. Leeming (1995). *J. Symb. Comput.*, **19**,459–488.

Carmody, S., Walters, R. F. C. (1991). Computing quotients of actions on a free category. In *Category Theory, Proceedings of the International Conference, Como, Italy, 22–28 July 1990,* LNM **1488**, Carboni, A., Pedicchio, M. C., Rosolini, G. eds., Springer.

Epstein, D. B. A., Cannon, J. W. *et al.* (1992). *Word Processing in Groups*, Boston, Jones and Bartlett Publishers.

Fleming, M., Gunther, R., Rosebrugh, R. User Guide for the Categories Database and Manual, *anonymous ftp://*`sun1.mta.ca/pub/papers/rosebrugh/catdsalg.dvi,tex` *and* /`catuser.dvi,tex` (1996).

Heyworth, A. (1998). Applications of rewriting systems and Gröbner bases to computing Kan extensions and identities among relations, Ph.D. Thesis, Bangor, `http://xxx.soton.ac.uk/abs/math.CT/9812097`

Holt, D. F. (1996). *Knuth–Bendix in Monoids and Automatic Groups*, Mathematics Institute, University of Warwick.

Holt, D. F., Hurt, D. F. (1999). Computing automatic coset systems and subgroup presentations. *J. Symb. Comput.*, **27**, 1–19.

Hopcroft, J., Ullman, J. (1979). *Introduction to Automata Theory, Languages and Computation*, Reading, MA, Addison-Wesley.

Mac Lane, S. (1971). *Categories for the Working Mathematician*, New York, Springer-Verlag.

Mitchell, B. (1972). Rings with several objects. *Adv. Math.*, **8**, 1–161.

Mora, T. (1987). *Gröbner Bases and the Word Problem*, University of Genova (manuscript).

Redfern, I. D. (1993). Automatic coset systems, Ph.D. Thesis, University of Warwick.

Originally Received 15 February 1999
Accepted 20 September 1999