Revised: 93jul22 14:00

On the Adequacy of Graph Rewriting for Simulating Term Rewriting

J.R. KENNAWAY [1], J.W. KLOP [2,3], M.R. SLEEP [4] & F.J. DE VRIES [5]

(1,4) *School of Information Systems, University of East Anglia, Norwich NR4 7TJ, United Kingdom*

(2,5) *Department of Software Technology, CWI P.O. Box 4079, 1009 AB Amsterdam, The Netherlands*

(3) *Department of Mathematics and Computer Science, Vrije Universiteit, de Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands*

(1) jrk@sys.uea.ac.uk, (2) jwk@cwi.nl, (4) mrs@sys.uea.ac.uk, (5) ferjan@cwi.nl

ABSTRACT Several authors have investigated the correspondence between graph rewriting and term rewriting. Almost invariably they have considered only acyclic graphs. Yet cyclic graphs naturally arise from certain optimisations in implementing functional languages. They correspond to infinite terms, and their reductions correspond to transfinite term reduction sequences, which have recently received detailed attention. We formalise the close correspondence between finitary cyclic graph rewriting and a restricted form of infinitary term rewriting, called rational term rewriting. This subsumes the known relation between finitary acyclic graph rewriting and finitary term rewriting.

Surprisingly, the correspondence breaks down for general infinitary rewriting. We present an example showing that infinitary term rewriting is strictly more powerful than infinitary graph rewriting.

The study also clarifies the technical difficulties resulting from the combination of collapsing rewrite rules and cyclic graphs.

AMS Subject Classification (1991): 68Q42

CR Subject Classification (1991): F1.1, F4.1, F4.2.

Keywords and Phrases: Orthogonal term rewriting, graph rewriting, functional programming.

Note: All authors were partially sponsored by SEMAGRAPH, ESPRIT Basic Research Action 3074. The first author was also partially supported by a SERC Advanced Fellowship, and by SERC grant no. GR/F 91582.

Contents

1. Introduction

Graph rewriting is a well-known technique for implementing term rewrite systems. The advantage it has over 'naive' term rewriting is that where a term rewrite rule such as:

$$\text{Times}(\text{ Succ}(\text{ x }), \text{ y }) \rightarrow \text{Add}(\text{ y}, \text{Times}(\text{ x}, \text{ y }))$$

would require making two copies of the second argument to Times, the graph rewrite implementation can instead make two references to a single copy. This saves time and space, especially when the argument is not itself in normal form, as can be the case when the language has call-by-need semantics.

Graph rewriting is frequently used to implement functional languages [Pey85]. Besides the advantages illustrated above, graph rewriting allows a subtle optimisation in the computation of fixed points. Consider the following rule for constructing an infinite list of copies of its argument:

$$\text{Repeat}(\text{ n }) \rightarrow \text{Cons}(\text{ n}, \text{Repeat}(\text{ n }))$$

Each time the rule is used, two copies of a pointer to the argument n are made. But in addition, the reference on the right-hand side of the rule to the subterm Repeat(n) can be replaced by a reference to the root of the right-hand side, giving a graph rewrite rule with a cyclic right hand side which we can write as Repeat(n) → x:Cons(n, x). The syntax x:Cons(n, x) here denotes a graph with two nodes, one of which is labelled with the symbol Cons and has two out-arcs, the second of which points to itself. A single application of this rule constructs the entire infinite list at once, as a finite cyclic structure. Figure 1 shows the effects of these two different representations of the rule.

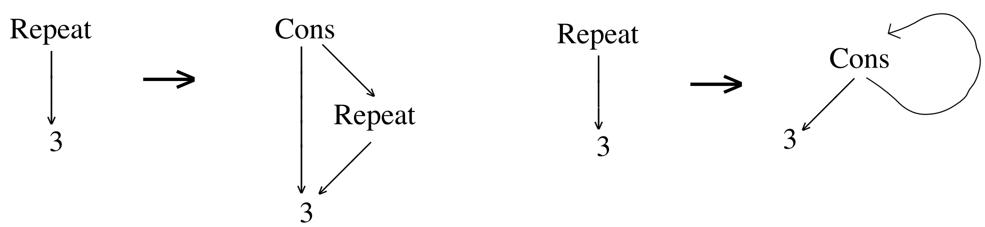

Figure 1

The correctness of the graph rewriting implementation of term rewriting is a piece of well-known folklore. For acyclic graphs the formal relationship has been studied in [Sta80, Bar87, Far89, Far90]. Only the last two of these consider cyclic graphs at all.

In an obvious and intuitive sense, acyclic graphs can be "unravelled" to trees — the syntax trees of terms. Cyclic graphs can be similarly unravelled, but give rise to infinite trees, which we can regard as infinite terms. A single reduction in a cyclic graph can correspond to the reduction of infinitely many redexes in the corresponding infinite term. A finite sequence of graph reductions may correspond to a term reduction sequence of length greater than ω. A precise account of the relationship between graph rewriting, including cyclic graphs, and term rewriting must therefore consider infinitary term rewriting. In a related paper [Ken93b] the authors have set out the foundations of infinitary term rewriting for orthogonal term rewrite systems.

In this paper we define finitary and infinitary term and term graph rewriting, and a notion of one rewrite system implementing another. We show that for orthogonal systems of rewrite rules, finitary graph rewriting implements in this sense a restricted version of infinitary term rewriting. This subsumes and makes more precise the result of [Bar87] that for orthogonal systems, finitary acyclic graph rewriting implements finitary term rewriting. We show by means of a counterexample that, surprisingly, infinitary graph rewriting does not implement infinitary term rewriting.

Our present definition of an implementation by one system of another (called here an *adequate mapping* from one to the other) adds to the abundance of concepts of simulation, in term rewriting (e.g. [Bar87, ODo85]), complexity theory (for an overview see [vEB90]) or programming languages [Mit91].

2. Finitary and infinitary term rewriting

2.1. Basic definitions

General introductions to finitary term rewriting may be found in [Der90] and [Klo92]. Here we shall define the basic concepts of infinitary term rewriting, which can be seen as including finitary rewriting as a special case.

For the formalisation of infinitary rewriting, we will require certain mathematical concepts: metric spaces, continuity, and ordinal numbers. No advanced knowledge is required, only a familiarity with the basic definitions. [Smy92] and [Phi92] give clear expositions of the material.

A *signature* Σ consists of a countably infinite set Var_Σ of variables (x,y,z,...) and a non-empty set of function symbols of various finite arities ≥ 0. Constants are function symbols with arity 0. Variables are also deemed to have arity 0

A(n infinitary) *term over* Σ is a finite or infinite ordered tree. Each node of the tree is labelled by a member of Σ and has a tuple of descendants. The size of the tuple is equal to the arity of the label of the node. $\mathrm{Ter}^\infty(\Sigma)$ is the set of infinitary terms over Σ. $\mathrm{Ter}(\Sigma)$ is the subset of $\mathrm{Ter}^\infty(\Sigma)$ consisting of the finite trees.

An *occurrence* is a finite sequence of positive integers. We write an occurrence as e.g. $1{\cdot}3{\cdot}2{\cdot}2$. The set O(t) of occurrences of t is defined by induction as follows: the empty occurrence ε is a member of O(t) for all t. If t is a variable or a function symbol of arity 0, this is the only member of O(t). An occurrence $i{\cdot}u$, where i is an integer and u is an occurrence, belongs to O(t) if t has the form $F(t_1,\ldots,t_n)$, $1 \leq i \leq n$, and $u \in O(t_i)$. Note that O(t) is in 1–1 correspondence with the nodes of t. For example, O(Cons(3, Cons(2, Nil))) = { ε, 1, 2, $2{\cdot}1$, $2{\cdot}2$ }. If u is a prefix of v, we write $u \leq v$. The concatenation of u and v is written $u{\cdot}v$.

If $u \in O(t)$ then the subterm $t|u$ at occurrence u is defined as follows: $t|\varepsilon = t$ and $F(t_1,\ldots,t_n)|i{\cdot}u = t_i|u$. The *depth* of a subterm of t at occurrence u is the length of u.

For example, let t = Cons(4, Cons(12, AddList(Plus(8, 3), Nil))). Then $t|2{\cdot}2{\cdot}1$ = Plus(8, 3).

There is a useful metric on the set $\mathrm{Ter}^\infty(\Sigma)$. Define d(t,t') to be 0 if t=t', otherwise 2^{-n}, where n is the length of the shortest occurrence u common to t and t' such that the nodes of t and t' corresponding to that occurrence bear different symbols.

A *substitution* is a map σ:$Var_\Sigma \to Ter^\infty(\Sigma)$. σ is extended to a function on the whole of Ter(Σ) by defining $\sigma(F(t_1,...,t_n)) = F(\sigma(t_1),...,\sigma(t_n))$, and to $Ter^\infty(\Sigma)$ by requiring that it be continuous with respect to the metric. Intuitively, this just means that $\sigma(t)$ is obtained from t by replacing every occurrence of every variable x in t by $\sigma(x)$. There may be infinitely many such occurrences. $\sigma(t)$ is called an *instance* of t. We may write a substitution defined on a finite set of variables $x_1,\ldots,x_n$ as $[x_1:=t_1,\ldots,x_n:=t_n]$.

A *rewrite rule* is a pair (t_l,t_r) of terms in $Ter^\infty(\Sigma)$, written as $t_l \to t_r$, such that t_l is finite and not a variable, and every variable occurring in t_r occurs also in t_l. t_l and t_r are called the *left-* and *right-hand sides* of the rule. It is *finitary* if t_r is also finite. A *redex* (reducible expression) of a term t consists of an occurrence u of t, a rule $t_l \to t_r$, and a substitution σ, such that $t|u = \sigma(t_l)$. The result of *reducing* this redex is the term resulting from replacing the subterm of t at u by $\sigma(t_r)$. If t reduces to t' by reduction of some redex, we write $t \to t'$. Concatenating reduction steps we get either a *finite reduction seequence* $t_0 \to t_1 \to ... \to t_n$, which we also denote by $t_0 \to_n t_n$, or an infinite reduction $t_0 \to t_1 \to t_2 \to$... A *normal form* is a term containing no redexes.

We require the left-hand side of a rule to be finite on both philosophical grounds (the question of whether a term is a redex by a given rule should be decidable in a finite time) and technical grounds (infinite left-hand sides cause some of the properties of infinitary rewriting to fail — see [Ken93b]). Infinite right-hand sides do not cause problems. We will in fact require infinite right-hand sides in order to model graph rewrite rules with cyclic right-hand sides. Unbounded left-hand sides are also unproblematic — that is, while we require each left-hand side to be finite, we do not require that in a system having infinitely many rules, there should be an upper bound on the size of the left-hand sides.

A (*finitary*)*term rewrite system* (or *TRS*) over a signature Σ is a pair (T,**R**), where **R** is a set of rewrite rules in T×T, and T is a set of terms over Σ which is closed under reduction by **R** and the subterm relation.

We do not require a TRS to contain every term which can be constructed from the signature. This allows us to uniformly treat such things as typed or sorted systems, where there are constraints on which terms are "legal". The closure conditions on T are for convenience. Note that T need not be closed under substitutions, since, for example, in a typed system where addition cannot be applied to boolean values, the legality of terms Add(x,y) and True must not imply that Add(True,y) is also legal.

A *transfinite reduction sequence* of a TRS (T,**R**) consists of a function f whose domain is an ordinal α, such that f maps each $\beta<\alpha$ to a reduction step $f_\beta \to f_{\beta+1}$. f is *Cauchy continuous* if the sequence of terms $\{ f_\beta \mid \beta<\alpha \}$ is a continuous function from α (with the usual topology on ordinals) to T (with the metric topology defined above). For each $\beta<\alpha$, let d_β be the depth of the redex reduced in the step from f_β to $f_{\beta+1}$. The sequence is *strongly continuous* if for every limit ordinal $\lambda<\alpha$, the sequence $\{ d_\beta \mid \beta<\lambda \}$ tends to infinity. It is *Cauchy convergent* if it is Cauchy continuous and converges topologically to a limit, denoted by f_α. It is *strongly convergent* if in addition the sequence $\{ d_\beta \mid \beta < \alpha \}$ tends to infinity.

We consider strongly convergent reduction sequences to be the appropriate notion of transfinite reduction sequence. Cauchy convergence alone is insufficient to allow the definition of the fundamental notions of residuals and projection, as shown in the next section.

We write a$_{\square}\!\!\rightarrow^{\alpha}\square$b (resp. a$_{\square}\!\!\rightarrow^{\leq\alpha}\square$b) to denote a strongly converging reduction of length α (resp. at most α) starting from a and converging to b, and a $\rightarrow^{\infty}$ b for a strongly converging reduction of any finite or infinite length. a $\rightarrow^{*}$ b denotes a reduction of finite length (including zero).

Consider the following rule systems and reduction sequences.

(i) Rule A(x,y) → A(y,x), sequence A(B,C) → A(C,B) → A(B,C) → A(C,B) →...

(ii) Rule A(x,y) → A(y,x), sequence A(D,D) → A(D,D) → A(D,D) → A(D,D) →...

(iii) Rule C → S(C), sequence C → S(C) → S(S(C)) →... S(S(S(…))).

Example (i) is a diverging reduction sequence. Example (ii) is Cauchy convergent with limit A(D,D). Example (iii) is strongly convergent with limit S^{ω} (i.e. S(S(S(…)))).

An *infinitary term rewriting system* over a signature Σ consists of a triple (T,R,S), where T is a subset of $Ter^{\infty}(\Sigma)$, R is a set of infinitary rewrite rules, and S is a set of strongly convergent term rewrite sequences, subject to the following conditions:

(i) T is closed under finite reduction sequences and the subterm relation.

(ii) Every term appearing in S is in T, and every reduction step in any member of S is a reduction by a rule in R.

(iii) S contains every finite reduction sequence of members of T.

(iv) S is closed under finite sequential composition.

(v) Every subsequence of a member of S is in S.

As for the finitary case, we do not require that T contain every possible term. In addition, we do not require that S contain every possible infinite reduction sequence over T, since we will later want to consider infinitary TRSs where S is restricted to the so-called rational sequences.

When we speak simply of a TRS, we shall mean an infinitary TRS. This includes the finitary TRSs, which are the special case where T and R contain only finite terms, and S is empty.

2.2. Basic properties of finitary and infinitary orthogonal systems

[Der90] and [Klo92] give tutorial accounts of the theory of orthogonal term rewriting for finitary orthogonal term rewrite systems. In [Ken93b] we have generalised this theory to the infinitary setting. We state here the main results: the Compressing Lemma, the Strip Lemma, Complete Developments, and the Church-Rosser property. The last of these is also treated in [Ken93a]. It is important to note that, in contrast to the finitary setting, the existence of complete developments and the Church-Rosser property do not hold in general, but are subject to various restrictions.

2.2.1 DEFINITION. Let R be a term rewriting system.

(i) R is *left-linear* if no variable occurs more than once in the left-hand side of a rewrite rule of R.

(ii) R is *non-overlapping* if for any two (not necessarily distinct) rules of R, with left-hand sides s and t, any occurrence u in t, and any substitutions σ and τ it holds that if $\sigma(t|u) = \tau(s)$ then either $t|u$ is a variable or t and s are left-hand sides of the same rewrite rule and u is the empty occurrence ε, the position of the root. If for some pair of rules this condition fails to hold, the rules are said to *conflict*.

(iii) R is *orthogonal* if R is both left-linear and non-overlapping. □

2.2.2 EXAMPLE. The rules $F(G(x)) \to H$ and $G(K(x)) \to K$ conflict with each other. The rule $F(F(x)) \to G$ conflicts with itself (take u = 1). □

Throughout the paper, all TRSs will be assumed to be orthogonal.

2.2.3 PROPOSITION. (Compressing Lemma [Ken93b]) In an orthogonal TRS, if $t \to^{\infty} t'$ then $t \to^{\leq\omega} t'$. □

This result is important because it allows us to be unconcerned about the fact that from sequences of length ω, we can obtain sequences of greater lengths through composition and projection. Such sequences may be given computational meaning by this result, independently of the motivation of cyclic graph rewriting.

The Compressing Lemma is one reason for preferring strong convergence to Cauchy convergence, as it is false for the latter form of reduction. A counterexample is given by the rules $A(x) \to A(C(x))$, $C(x) \to D(x)$, and the sequence of length ω+1: $A(B) \to A(C(B)) \to A(C(C(B))) \to \ldots A(C^{\omega}) \to A(D(C^{\omega}))$. There is no reduction of length ≤ω from A(B) to $A(D(C^{\omega}))$.

2.2.4 DEFINITION. Let $t \to t'$ by reduction of a redex r at occurrence u in t. Let $u' \in O(t)$. Then there is a set of occurrences in t', denoted by u'/r, which in an intuitive sense "descends from" u' in t, and which may be formally defined thus. Partition O(t) into three parts: those elements u' which u is not a prefix of; those elements of the form $u \cdot v$, where v is an occurrence of a function symbol in the left-hand side of the rule of r, and those elements of the form $u \cdot v \cdot w$, where v is an occurrence of a variable x on the left hand side of r. In the first case, $u'/r = \{u'\}$. In the second, $(u \cdot v)/r = \emptyset$. In the third, $(u \cdot v \cdot w)/r = \{ u \cdot v' \cdot w \mid v' \text{ is an occurrence of } x \text{ on the right hand side of } r \}$.

The members of u'/r are called the *residuals* of u by r. For $U \subseteq O(t)$, U/r is the union of u'/r over all $u' \in U$. □

2.2.5 DEFINITION. Let $s: t_0 \to^{\alpha} t_{\alpha}$ be a strongly convergent reduction sequence and $U \subseteq O(t)$. Then U/s is defined by induction on the steps of s. Let $U_0 = U$. If U_β is defined for some $\beta<\alpha$, then $U_{\beta+1} = U_\beta/(t_\beta \to t_{\beta+1})$ as given by definition 2.2.4. If $\lambda \leq \alpha$ is a limit ordinal, then U_λ contains every occurrence which is a member of U_β for all large enough $\beta<\lambda$. Equivalently, $U_\lambda = \bigcup_{\gamma<\lambda} \square \bigcap_{\gamma<\beta<\lambda} U_\beta$.

For $u \in O(t)$, u/s is defined to be {u}/s. □

The strong convergence of s implies that U_λ could equivalently be defined as containing those occurrences which are in U_β for arbitrarily large $\beta<\lambda$, rather than only those which are in U_β for all $\beta<\lambda$. In other words, U_λ is also equal to $\bigcap_{\gamma<\lambda} \square \bigcup_{\gamma<\beta<\lambda} U_\beta$. This is because strong convergence implies that whatever the length of an occurrence u is, from some point onwards in the sequence s up to stage λ, every reduction is performed at a place in the term which is deeper than u. If this state of affairs is reached at stage γ, then u is either a member of every U_β when $\gamma<\beta<\lambda$, or is a member of no such U_β.

2.2.6 PROPOSITION. In an orthogonal TRS, let t → t' by reduction of a redex r at occurrence u in t. Let s:t $\to^\infty$ t" be a strongly convergent reduction sequence. Then t" has a redex by the same rule as r at every occurrence in u/s.

PROOF. This is trivial when s is the empty sequence. If it is true for a sequence s, then it is true for a sequence consisting of s followed by a single step, from the theory of orthogonal finitary reduction [Der90, Klo92]. Suppose the length of s is a limit ordinal λ, and the theorem holds for all shorter sequences. Let u'∈u/s. The left hand side of the redex r has a finite maximum depth n. Suppose u' has length n'. From some point $\alpha<\lambda$ in the sequence onwards, every reduction is performed at depth n+n' or greater. At that point, there is by the inductive hypothesis a residual of r at occurrence u'. Since every later step is at depth at least n+n', that residual must be present in each later term in the sequence before the limit, and hence is present in the limit also. □

2.2.7 DEFINITION. The redexes given by proposition 2.2.6 are called the *residuals* of r by s, and the set of residuals is denoted by r/s. □

Proposition 2.2.6 provides another reason for using strongly convergent sequences. Consider example (ii) from section 2.1. If there is in addition a rule whose left-hand side is D, then the term A(D,D) contains two redexes. Each of these redexes in the initial term has exactly one residual in the terms at each finite stage of the sequence; however, the two redexes change places with each step. The limit term also contains two such redexes; which of the two initial redexes should either of these be deemed to be a residual of?

2.2.8 DEFINITION. A *development* of a pairwise non-conflicting set of redexes R of a term t is a reduction sequence in which after each initial segment s, the next step, if any, is the reduction of a residual of a member of R by s. The development is *complete* if it strongly converges to a limit which does not contain any residual of any member of R. □

We now come to a point on which finitary and infinitary term rewriting differ.

2.2.9 DEFINITION. A *collapsing rule* is a left-linear rule whose right-hand side is a variable. Its *collapsing occurrence* is the (unique) occurrence in its left-hand side of the variable which is its right-hand side. A *collapsing redex* is a redex of a collapsing rule. In a term t, a *collapsing tower* is an infinite set of redexes at occurrences of the form u_1, $u_1 \cdot u_2$, $u_1 \cdot u_2 \cdot u_3, \ldots$, such that the redex at $u_1 \cdot \ldots \cdot u_i$ is a collapsing redex with collapsing occurrence u_{i+1}. □

Given a collapsing rule such as I(x) → x, an example of a collapsing tower is the infinite term I(I(I(…))). Given an additional collapsing rule J(K,x) → x, there are more complicated collapsing towers such as I(J(K,I(J(K,I(J(K,…)))))).

In finitary orthogonal rewriting, every set of redexes has a complete development. For infinitary rewriting this is not so. Consider, for example, the collapsing tower introduced above, I(I(I(…))). The set of all redexes of this term has no complete development, since any attempt to reduce every redex results in a sequence which is only Cauchy convergent but not strongly convergent. For a more complicated collapsing tower such as the I(J(K,I(J(K,I(J(K,…)))))) introduced above, some attempts to reduce every redex result in a sequence which is not even Cauchy convergent.

2.2.10 PROPOSITION. (Complete developments [Ken93b]) In an orthogonal system, let R be a set of redexes in a term t. If R contains no collapsing tower, then R has a complete development. (In particular, every finite set of redexes has a complete development.) Every complete development of R, if any, ends at the same term. □

Note that collapsing towers do not arise in finitary term rewriting. In that setting, complete developments always exist and are finite.

2.2.11 COROLLARY. In an orthogonal TRS, if R and R' are sets of redexes of t, R is a subset of R', and R' contains no collapsing tower, then every complete development s of R can be extended to a complete development of R', by appending to it a complete development of the set R'/s. This set depends only on R and not on the choice of s. □

2.2.12 DEFINITION. We write R'/R for the set of redexes constructed in the previous corollary. Note that R'/R always exists when R' is a finite set. □

2.2.13 PROPOSITION. (Parallel Moves Lemma.) In an orthogonal TRS, let R and R' be sets of redexes of the same term t, whose union does not include a collapsing tower. Then R/R' and R'/R, considered as complete developments, have the same final term.

PROOF. By definition, R·(R'/R) and R'·(R/R') are both complete developments of R∪R', and hence by Proposition 2.2.10 have the same final term. □

2.2.14 PROPOSITION. (Strip Lemma [Ken93b]) In an orthogonal TRS, let r be a redex of the initial term of a reduction sequence s: $t \to^{\infty} t'$. Then r/s has a complete development, and there is a reduction sequence, denoted by s/r, such that r(s/r) and a complete development of r/s have the same final term. □

The proof of the Strip Lemma not only proves the existence of r/s and s/r, but constructs particular sequences. More generally, we can consider a set of redexes R instead of a single redex R, and obtain a set of redexes R/s and a sequence s/R, although in that case the construction cannot always be performed — R/s may not have a strongly convergent complete development, and s/R may not exist. The construction is pictured in Figure 2. Each of the squares in this diagram has sides of the form R, R', R/R', and R'/R. Such a diagram is called a *projection diagram*. Note that we use the notation R/s to denote not only a set of redexes, but also a strongly convergent complete development of that set. In general it will not matter which complete development is chosen.

$$\begin{tikzcd} t_0 \arrow[r, "s_1"] \arrow[d, "R"'] & t_1 \arrow[d, "R_1=R/s_1"'] & \cdots & t_\beta \arrow[r, "s_\beta"] \arrow[d, "R_\beta"'] & t_{\beta+1} \arrow[d, "R_{\beta+1}=R_\beta/s_\beta"'] & \cdots & t_\alpha \arrow[d, "R_\alpha=R/s"] \\ \bullet \arrow[r, "s_1/R"'] & \bullet & \cdots & \bullet \arrow[r, "s_\beta/R_\beta"'] & \bullet & \cdots & \bullet \end{tikzcd}$$

Figure 2

In the finitary case, the Parallel Moves Lemma and the Strip Lemma immediately yield two different proofs of the Church-Rosser property of orthogonal term rewrite systems. (The former is the method of proof in e.g. [Hue91].) This is the property that given any two sequences $s_0{:}t\to^* t_0$ and $s_1{:}t\to^* t_1$, there exist a term t_2 and two sequences $s_2{:}t_0\to t_2$ and $s_3{:}t_1\to t_2$. Both proofs give more information than this, by constructing a particular pair of sequences s_2 and s_3, denoted by s_1/s_0 and s_0/s_1 and called the *projection* of s_1 over s_0 and of s_0 over s_1 respectively.

Neither proof extends immediately to the infinitary case. The Strip lemma implies the transfinite Church-Rosser property (where s_0, s_1, s_2, and s_3 may be of any transfinite length) only in the case where at least one of the two sequences is finitely long, and the Parallel Moves Lemma implies the Church-Rosser property only when both sequences are concatenations of finitely many complete developments, and not even for all such sequences. If we apply the Compressing Lemma to s_0 and s_1 to transform them into sequences s'_0 and s'_1 of length at most ω, we can then apply the Strip Lemma to s'_0 and finite initial segments of s'_1, and vice versa. This results in a construction of sequences s'_0/s'_1 and s'_1/s'_0, but in general they might not be strongly convergent or converge to the same limit. For later reference, we shall call this construction of s'_0/s'_1 and s'_1/s'_0 the Standard Construction.

The fact that the Standard Construction does not immediately give a proof of the transfinite Church-Rosser property for orthogonal systems is not surprising, since in general the property does not hold. A counterexample is given by the rules $C \to A(B(C))$, $A(x) \to x$, $B(x) \to x$. C can be strongly convergently reduced in ω steps to either A^ω or B^ω. These have no common reduct (whether by strongly or Cauchy convergent reduction). However, certain restricted forms of the transfinite Church-Rosser property do hold for orthogonal systems.

One way of forcing the Church-Rosser property to hold is to restrict the form of rewrite rules. The above counterexample suggests that there is a problem with collapsing rules. It is possible to obtain the transfinite Church-Rosser property by suitably restricting the possible forms of such rules.

2.2.15 DEFINITION. An orthogonal set of term graph rewrite rules is *almost non-collapsing* if it contains at most one collapsing rule, and that collapsing rule contains at most one variable on its left-hand side. □

Examples of the single collapsing rule that may be contained in an almost non-collapsing system are $I(x) \to x$ or $A(B(C,x),D(E)) \to x$. The rule $K(x,y) \to x$ is not allowed. In any orthogonal system containing this rule, violations of the Church-Rosser property result which are similar to that given before Definition 2.2.17. For example, the graph $x{:}K(K(x,A),B)$ reduces both to $x{:}K(x,A)$ and $x{:}K(x,B)$.

2.2.16 THEOREM. An orthogonal set of rewrite rules is transfinitely Church-Rosser if it is almost non-collapsing.

PROOF. This is proved in [Ken93b].

For the sake of self-containedness, we shall briefly and very informally indicate the method of proof. As remarked above, we can apply the Standard Construction to the two sequences s_0 and s_1 of length ω, obtaining s_0/s_1 and s_1/s_0, after which it only remains to show that these sequences (or, as we shall see, sequences obtained from them by omitting certain steps) strongly converge to the same term.

We begin with *depth-preserving* systems, that is, systems where the depth of every occurrence of a variable in the right hand side of each rule is at least as great as its depth in the left hand side. For these, is it easy to see that the set of rows and columns of the projection diagram are uniformly strongly convergent, and hence that s_0/s_1 and s_1/s_0 are strongly convergent, and converge to the same term.

To extend this to the non-collapsing systems, we transform the given system into a depth-preserving one by padding out the right hand sides with a unary function symbol ε. New rules must also be added in which both left and right hand sides are padded out in this way. From the Church-Rosser property for depth-preserving systems the property for non-collapsing systems follows by simply erasing all occurrences of ε in the sequences that were constructed in the transformed system. This is possible since the absence of collapsing rules prevents any subterm of the form ε^ω (which has no counterpart in the original system) from arising.

When there are collapsing rules, the transformation to a depth-preserving system can still be applied, but the Church-Rosser property for the transformed system in general no longer carries back to the original system. This is because the sequences s_0/s_1 and s_1/s_0 which are constructed in the depth-preserving system may now contain subterms of the form ε^ω. These arise from collapsing towers in the original system. It is possible for occurrences of ε^ω in the transformed system in the common final terms of s_0/s_1 and s_1/s_0 to correspond to distinct collapsing towers of the original system which have no common reduct, resulting in a counterexample to the Church-Rosser property. However, for almost non-collapsing systems, all collapsing towers are identical and reduce only to themselves. By omitting reductions of redexes in such towers we obtain sequences of the required form in the original system. □

The converse holds provided the system contains enough terms. If a system is almost non-collapsing, then one can find two distinct collapsing redexes t and t' such that both t and t' reduce in a single step to x. This can be done if the system is not almost non-collapsing. Then construct the infinite term t[x:=t'[x:=t[x:=t'[...]]]]. This reduced both to t[x:=t[x:=t[...]]] and to t'[x:=t'[x:=t'[...]]], which have no common reduct.

Another form of the transfinite Church-Rosser property is obtained by instead weakening the requirement that the constructed sequences end with the same term.

2.2.17 DEFINITION. A *congruence* on a TRS is an equivalence relation ~ such that

(i) If t ~ t', then for any term t" and variable x, [x:=t]t" ~ [x:=t']t".

(ii) For any substitution σ, if t ~ t' then σt ~ σt'.

(iii) If $t \twoheadrightarrow^{\infty} t'$ then $t \sim t'$.

Given a congruence $\sim$, a TRS is CR/$\sim$ if whenever $t_0 \twoheadrightarrow^{\infty} t_1$ and $t_0 \twoheadrightarrow^{\infty} t_2$, there exist sequences $t'_1 \twoheadrightarrow^{\infty} t_3$ and $t'_2 \twoheadrightarrow^{\infty} t_4$ such that $t_1 \sim t'_1$, $t_2 \sim t'_2$, and $t_3 \sim t_4$. See Figure 3. □

$$
\begin{tikzcd}
t_0 \arrow[r] \arrow[d] & t_1 \sim t'_1 \arrow[d] \\
t_2 \sim t'_2 \arrow[r] & t_3 \sim t_4
\end{tikzcd}
$$

Figure 3

2.2.18 DEFINITION. A *hyper-collapsing reduction* is a reduction sequence containing infinitely many collapsing reductions performed at the root. A *hyper-collapsing term* is a term from which there is a hyper-collapsing reduction. $\sim_{hc}$ is the smallest conguence which identifies all hyper-collapsing terms with each other. □

2.2.19 THEOREM. [Ken93b] Every orthogonal infinitary TRS is CR/$\sim_{hc}$. Given coinitial reduction sequences s and t, sequences of the form required by the CR/$\sim_{hc}$ property may be constructed by applying the Standard Construction, and then omitting from the constructed reduction sequences all reductions which take place in hyper-collapsing subterms. □

3. Graph rewriting

Graph rewriting is a common method of implementing term rewrite languages [Pey85]. It relies on the basic insight, that when a variable occurs many times on the right-hand side of a rule, one need only copy pointers to the corresponding parts of the term being evaluated, instead of making copies of the whole subterm. The reader familiar with graph rewriting may skip this section. Note however that we allow cyclic graphs; as we will see, these correspond to certain infinite terms.

3.1 DEFINITION. A *graph* g over a signature $\Sigma = (\mathcal{F},\mathcal{V})$ is a quadruple (*nodes*(g), *lab*(g), *succ*(g), *roots*(g)), where *nodes*(g) is a finite or countable set of nodes, *lab*(g) is a function from a subset of *nodes*(g) to $\mathcal{F}$, *succ*(g) is a function from the same subset to tuples of nodes of g, and *roots*(g) is a tuple of (not necessarily distinct) nodes of g. Furthermore, every node of g must be *accessible* (defined below) from at least one root. Nodes of g outside the common domain of *lab*(g) and *succ*(g) are called *empty*. □

3.2 DEFINITION. A *path* in a graph g is a finite sequence a,i,b,j,… of alternating nodes and integers, beginning and ending with a node of g, such that for each m,i,n in the sequence, where m and n are nodes, n is the i'th successor of m. An *occurrence* of g is a sequence of integers obtained by omitting all the nodes from some path which starts from a root of g. For an occurrence u, $g|u$ is the node of g which the corresponding path of g ends at. The length of the path is the number of integers in it. If the path starts from a node m and ends at a node n, it is said to be a path from m to n. If there is a path from m to n, then n is said to be *accessible from* m. When this is so, the *distance* of n from m is the length of a shortest path from m to n. If n is a node of g, then $g|n$ is the one-rooted graph consisting of all nodes of g accessible from n, rooted at n. □

In order to textually represent a particular graph, we use the notations of [Bar87]. We write $n:F(n_1,\ldots,n_k)$ to indicate that *lab*(g)(n) = F and *succ*(g)(n) = $(n_1,\ldots,n_k)$. A finite graph can then be presented as a list of such *node definitions*. For example, x:F(y,z), z:G(y,w,w), w:H(w) represents the graph shown in Figure 4.

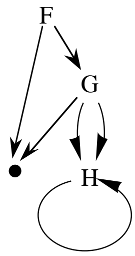

Figure 4

In such pictures, we can omit the names x, y, z, etc., as their only function in the textual representation is to identify the nodes. In particular, x, y, z, etc. do not represent variables: variables are represented by empty nodes. Different empty nodes need only be distinguished by the fact that they are different nodes; we do not need any separate alphabet of variable names. Multiple references to the same variable in a term are represented in a graph by multiple references to the same empty node.

The tabular description demonstrated above may conveniently be condensed, by nesting the definitions and omitting unnecessary identifiers; for example, another way of writing the same graph is F(y,z:G(y,w,w:H(w))).

In general a graph may have more than one root. We will only use graphs with one root (which represent terms) and graphs with two roots (which represent term rewrite rules).

3.3 DEFINITION. A *graph homomorphism* from a graph g to a graph h is a function f from the nodes of g to the nodes of h, such that for all nodes n in the domain of *lab*(g), *lab*(h)(f(n)) = *lab*(g)(n), and *succ*(h)(f(n)) = f(*succ*(g)(n)), where f is extended in the obvious way to tuples. f is *strict* if for every empty node n of g, f(n) is also empty. □

Note that a graph homomorphism is not required to map the roots of its domain to the roots of its codomain.

On graphs one can define many general graph rewrite mechanisms. We are concerned with one particular form: term graph rewriting.

3.4 DEFINITION. A *term graph* is a graph with one root. □

3.5 DEFINITION. A *term graph rewrite rule* is a graph with two, not necessarily distinct, roots (called the *left and right* roots), in which every empty node is accessible from the left root. The *left* (resp. *right*) *hand side* of a term graph rewrite rule r is the subgraph consisting of all nodes and edges accessible from the left (resp. right) root: notation *left*(r) (resp. *right*(r)). □

3.6 DEFINITION. A *redex* of a term graph rewrite rule r in a term graph g is a homomorphism from the left-hand side of r to g. The redex is *rooted at* the node to which the homomorphism maps the left root of the rule. The *depth* of a redex is the distance from the root of g to the node to which the redex maps the left root of r. A node of g is *pattern-matched* by the redex if it is the image of a nonempty node of r. □

The result of *reducing* a redex of the rule r in a graph g is the graph obtained by the following construction.

3.7 CONSTRUCTION. (i) (*Build.*) Construct a graph h by adding to g a copy of all nodes and edges of r not in *left*(r). Where such an edge has one endpoint in *left*(r), the copy of that edge in h is connected to the image of that endpoint by the homomorphism.

(ii) (*Redirect.*) Let n_l be the node of h corresponding to the left root of r, and n_r the node corresponding to the right root of r. (These are not necessarily distinct.) In h, replace every edge whose target is n_l by an edge with the same source and target n_r, obtaining a graph k. The root of k is the root of h, unless the root of h is n_l, otherwise the root of k is n_r.

(iii) (*Collect garbage.*) Remove all nodes which are not accessible from the root of k. The resulting graph is the result of the rewrite. □

Step (i) adds to g all the nodes which the rewrite must create.

Step (ii), which replaces all references to n_l by references to n_r, is customarily implemented in a different manner. When n_r would be a new node, it is not created; instead, n_l is instead overwritten with the function symbol and out-arcs which n_r would have. When n_r is an existing node, n_l is replaced by an “indirection” node, which contains a single out-arc, pointing to n_r. This indirection node is thereafter imagined to be transparent to pattern-matching: whenever the machine reads the contents of n_l, it gets the contents of n_r instead (or if that too is an indirection node, the contents of the node it points to, and so on). We will comment on this in more detail in the concluding remarks.

Step (iii) is known as “garbage collection”. In implementations of functional languages it is normally not performed as a part of each reduction step. Instead, the inaccessible nodes are allowed to accumulate until memory runs out, and are then all destroyed together to recover space. If is not difficult to see that this gives the same result as performing a garbage collection as part of each rewrite.

Figure 5 illustrates the stages of a rewrite, using the rule AddList(acc, Cons(h, t)) → Cons(newacc, AddList(newacc, t)), newacc:Plus(acc, h) and the initial graph Cons(x:8, AddList(x, Cons(3, Nil))).

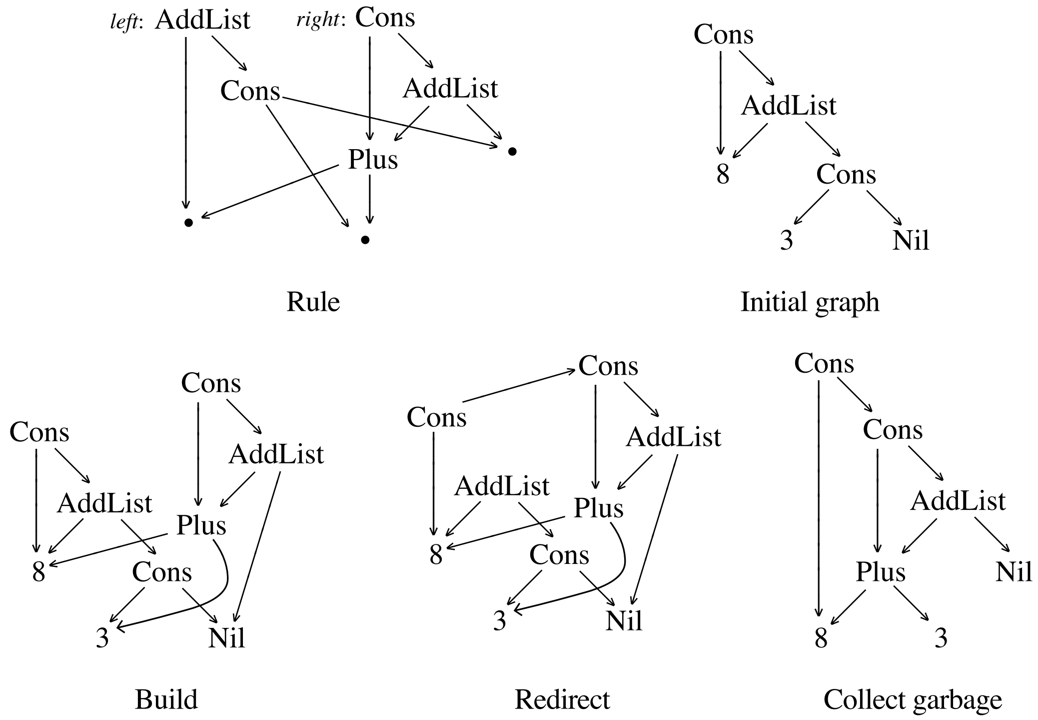

Figure 5

3.8 DEFINITION. A (finitary) Term Graph Rewrite System (GRS for short) over a signature Σ is a pair (G,**R**) where G is a set of finite term graphs over Σ, **R** is a set of term graph rewrite rules over Σ whose left and right hand sides are in G, and G is closed under reduction and inverse strict homomorphisms. □

The inverse strict homomorphism condition corresponds to the subterm condition for TRSs. It also implies that G is "closed under unsharing". That is, where there are two or more references in a graph g to a node n, the graph resulting from g by replacing n by a number of copies must also be in G. We do not require that G be "closed under sharing". This is because closure under sharing would always allow empty nodes of a graph in G to be merged to give another graph in G, but if the set of valid graphs G derives from some type discipline, this might be an invalid transformation, since distinct empty nodes might have incompatible types.

Having defined term graph rewriting and the notion of depth on term graphs, the concepts of normal form, orthogonality, collapsing rule, and residual carry over to term graphs. There are sufficient differences that we make explicit definitions.

Notice that the left-linearity part of the definition of orthogonality means in the case of graphs that the subgraph of a rule accessible from its left root is a tree. In addition, besides residuals of occurrences and redexes, we have the notion of a residual of a node.

3.9 DEFINITION. A *normal form* of a GRS is a graph containing no redexes.

A GRS is *orthogonal* if (i) for each rule, the subgraph accessible from the left root is a tree, and (ii) for any graph g, and any two distinct redexes r and r' of g, the root of r is not pattern-matched by r'. Rules satisfying condition (i) are called *left-linear*. Pairs of redexes violating condition (ii) are said to *conflict* with each other.

The concrete definition of graph rewriting implies that when g → g', many of the nodes of g' are nodes of g. Each such node of g' is said to be the *residual* of the corresponding node of g. Let r be the redex of g which is reduced in g → g', and let n be a node of g. The set of residuals of n by r (a set with either 0 or 1 members) is denoted by n/r. When this set is nonempty, we may also write n/r to denote its unique element. An occurrence u' of g' is a *residual* of an occurrence u of g if that relation holds between the nodes they respectively specify. We write u/r for the set of such residuals. (It may have many elements, but all will represent the same node of g'.) A redex r' of g' is a *residual* of a redex r of g if that relation holds between their respective roots, and the redex reduced by g → g' does not conflict with r.

Give a set N of nodes of g, the set of residuals of all members of N by r is denoted N/r. Similarly, for a reduction sequence s from g to g', we may define n/s, N/s, and u/s.

Developments are defined as in the term case: a development of a set of redexes in a graph g is a reduction sequence starting from g in which each step reduces a residual of a member of the original set. The development is *complete* if its final term graph contains no such residuals. □

As we deal only with finitary graph rewriting, the issue of convergence does not arise in the definition of development. Furthermore, since (unlike the term case) reduction of one redex can never cause multiple copies to be made of another, every redex has at most one residual by any other redex. This implies that the length of a development is always finite, bounded by the number of redexes in the given set. In particular, every set of pairwise disjoint redexes has a complete development.

Non-left-linearity has a rather different meaning for graphs than for terms. When the rule If(x,y,y) → y is understood as a term rewrite rule, a term of the form If(t,t',t") is a redex if and only if t' and t" are syntactically identical. Understood as a graph rewrite rule, a node of the form w:If(x,y,z) is the root of a redex of this rule if and only if y and z are the same node. If they are different nodes, then w is not a redex-root, even if the subgraphs rooted at y and z are isomorphic. Non-left-linearity causes technical problems in the theory of term rewriting — for example, it is responsible for the failure of the Church-Rosser property for systems such as the union of combinatory logic with rules similar to the above example. To a great extent, non-left-linear rules, as defined in graph rewriting, cause no such problems. Accordingly, when studying graph rewriting for its own sake one might omit the left-linearity condition from the definition of orthogonality. We retain it here in order that the definition match more closely the definition of orthogonality for term rewriting.

Throughout the paper, all GRSs will be assumed to be orthogonal.

Cyclic graphs can arise through optimisations of term rewriting. An example is the rule for the Y combinator. The term rewrite rule for Y is Apply(Y, f) → Apply(f, Apply(Y, f)). This can also be read as a graph rule. However, the presence of the subexpression Apply(Y, f) on the right-hand side of the rule, isomorphic to the left-hand side, suggests the optimised version Apply(Y, f) → x:Apply(f, x). These two versions of the rule are illustrated in Figure 6.

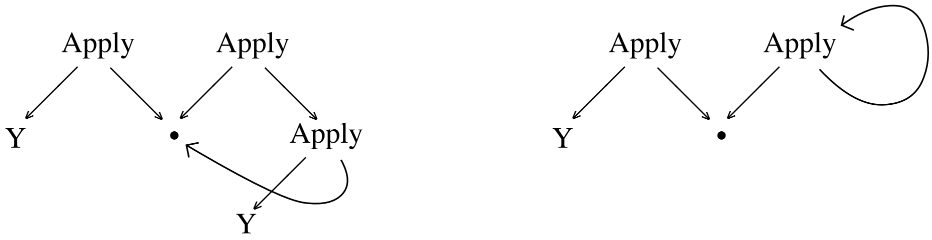

Figure 6

Cyclic graphs are practically useful, but they introduce a technical complication in certain cases, analogous to the collapsing towers we studied in the previous section. Consider the rule I(x) → x and the graph y:I(y), shown in Figure 7.

Figure 7

It is clear that the graph is a redex of the rule. It reduces to itself. Circular I, as we call it, is one instance of a class of redexes having the same behaviour, the circular redexes.

3.10 DEFINITION. A redex of a rule r in a graph g is *circular* if the roots of r are distinct and the homomorphism from *left*(r) to g maps both roots of r to the same node. (This can only happen if the right root of r is accessible from the left root.) □

3.11 PROPOSITION. *Every circular redex reduces to itself.*

PROOF. This can be verified by following through the construction 3.7. □

3.12 DEFINITION. A term graph rule is a *collapsing* rule if its right root is an empty node. □

An example of a collapsing rule is x:Head(Cons(y,z)) → y. An example of a non-collapsing rule which admits circular redexes is x:F(y:F(z)) → y. Note that this rule conflicts with itself: it has two overlapping redexes in the graph F(F(F(G))). A circular redex of this rule is x:F(x).

3.13 PROPOSITION. In an orthogonal term graph rewrite system, every circular redex is a redex of a collapsing rule. If the set of graphs of the system is closed under sharing, then every collapsing rule has a circular redex.

PROOF. A circular redex maps the left root and the right root to the same node, the right root being accessible from the left root. This implies that the graph constructed by replacing the subgraph accessible from the right root by a copy of the left hand side contains two redexes of the rule, one at its root and one at the root of the new copy of the left-hand side. If the right root were not an empty node, these redexes would conflict.

Conversely, suppose the GRS contains a collapsing rule R. Consider the graph obtained from the left hand side of R by identifying its left and right roots. This is obtained from R by sharing and is therefore a graph of the GRS. □

Collapsing rules and circular redexes introduce problems for cyclic graph rewriting similar to those which arise in infinitary term rewriting. The Church-Rosser property fails, as demonstrated by an example similar to the one near the end of section 2.2. Consider the rules A(x) → x and B(x) → x, and the graph y:A(B(y)). The graph contains two redexes, one by each rule. Reducing the outer redex yields z:B(z), reducing the inner redex yields z:A(z). These two graphs have no common reduct, as is the case for the infinitary terms A^{ω} and B^{ω}.

It is possible to formulate theorems for graph rewriting similar to those of section 2.2. However, we will not actually need such results for the purposes of the present paper. The acyclic case (in which circular redexes cannot arise) has been treated by [Sta80] and [Bar87]. We will only note here a point concerning complete developments in graph rewriting. For reasons corresponding to the phenomenon of collapsing towers in infinitary term graph rewriting, not all complete developments of the same set of redexes need end with the same graph, when cyclic graphs are present. The example of the previous paragraph illustrates this (a complete development of both redexes may yield either z:A(z) or z:B(z)). However, the present paper does not require an analysis of which sets of redexes may behave in this way.

4. Unravelling

Unravelling transforms acyclic term graphs to finite terms and cyclic graphs to infinite terms. Both graphs and computations can be unravelled. In this section we will prove that for any term graph rewrite system, if g reduces to g', then the same relation holds between their respective unravellings in the corresponding infinitary term rewrite system. This is so even for non-orthogonal systems.

4.1 DEFINITION. The *unravelling* U(g) of a graph g is a tuple of terms in 1-1 correspondence with the tuple of roots of g. The nodes of the term corresponding to the root node n are the paths of g which start from n. Given a node a,i,b,j,…y of one of the terms of U(g), if y is a nonempty node of g, then this node of the term is labelled with the function symbol *lab*(g)(y), and its successors are all paths of the form a,i,b,j,…y,n,z, where z is the n'th successor of y in g. If y is empty, then the node of the term is labelled with a variable symbol. A different variable symbol is used for each different empty node of g. Where g contains multiple references to the same empty node, there will be multiple occurrences in U(g) of the corresponding variable. When g is a term graph or a term graph rewrite rule, U(g) will be respectively a term or a term rewrite rule.

Given a node n of a term graph g, U(g,n) is the set of occurrences of U(g) corresponding to all the paths from the root of g to n.

Given a redex r rooted at a node n of a term graph g, U(r) is the set of redexes, by the same rule as r, at each of the occurrences in U(g,n) (it is easy to see that every such occurrence is such a redex). □

A cyclic graph has an infinite unravelling. For example, the unravelling of the graph shown in Figure 4 is the term $F(y,G(y,H^{\omega},H^{\omega}))$, where by H^{ω} we denote the infinite term H(H(H(…))).

4.2 DEFINITION. An *unravelling* of a GRS (G,**R**) is a TRS of the form (U(G),U(**R**),S) whose terms and rules are the unravellings of the term graphs and rules of (G,**R**), and whose transfinite reduction sequences include all those which are unravellings of finite reduction sequences of (G,**R**). □

Later results will imply that the closure conditions for U(G) to be the set of terms of a TRS are satisfied, and that the notion of the unravelling of a sequence is well-defined (Theorem 4.7).

The following proposition is immediate.

4.3 PROPOSITION. A graph g in a GRS (G,**R**) is a normal form iff its unravelling U(g) is a normal form in (U(G),U(**R**),S). □

4.4 THEOREM. In a left-linear GRS, let g $\to$ g' by reduction of a redex r. Then U(g) $\to^\infty$ U(g') in the corresponding TRS. If the redex reduced in g is circular, then the sequence is empty. Otherwise it is a complete development of U(r). Moreover, the depth of every redex reduced in the term sequence is at least equal to the depth of the redex reduced in g. If the GRS is acyclic, then the sequence U(g) $\to^\infty$ U(g') is finite.

PROOF. Let l$\to$r be the rule that was applied to reduce g to g', and n the node at which it was applied. Let t = U(g) and t' = U(g'). We distinguish two cases.

Firstly, if the redex in g is a circular redex, then g' = g, and therefore the empty sequence from U(g) to itself satisfies the requirements of the theorem.

Otherwise, there is a redex of U(l$\to$r) at every occurrence in U(g,n). This set of redexes contains no collapsing tower. Proposition 2.2.10 implies that there is a complete development of these redexes. Let t' be the limit of such a complete development. We demonstrate that U(g') = t' by giving an explicit description of both g' and t'.

An occurrence of g can be schematically depicted as a concatenation of segments thus:

The thin sections are parts which end at but do not pass through the node n. Each thick section v_i begins at n, and v_i is an occurrence of an empty node x_i in the left hand side of the rule. Thus each thick section lies within the part of the graph matched by the redex. The final segment may be thin or thick, and need not end, respectively, at n or at the image of an empty node.

The occurrences of g' have the same description, except that thick segments are drawn from the right hand side of the rule instead of the left.

Each occurrence $u_1v_1u_2v_2\ldots$ of g which does not end with a “short” thick segment gives rise to occurrences of the form $u_1v'_1u_2v'_2\ldots$ of g', where v_i and v'_i are occurrences of x_i in the left and right-hand sides of the rule respectively. Such corresponding occurrences of g and g' have the same function symbol. For an occurrence $u_1v_1u_2v_2\ldots v_n$ of g where v_k is an occurrence of a nonempty node in the left-hand side of the rule, g' will have occurrences $u_1v'_1u_2v'_2\ldots v'_k$, where $v'_1\ldots v'_{k-1}$ are as before, and v'_k is an occurrence of any nonempty node in the right-hand side of the rule. g' has the same function symbol at this occurrence as that node of the rule does.

An occurrence of t can be schematically depicted as a concatenation of segments thus:

An initial segment of this occurrence is a member of U(g,n) if and only if it ends with the whole of a thin segment u_i (except possibly if i=k). A thick segment v_i is as before an occurrence of a variable in the left-hand side of the term rewrite rule. As with occurrences of g, the last segment of an occurrence of t represented in the above form may be short, i.e. if thin, not reach to a member of U(g,n), and if thick, not reach to a variable in the left-hand side of the rule.

The occurrences of t' are of a similar form, where the thick segments are replaced by occurrences of variables in the right-hand side of the rule.

In the same way as for g and g', an occurrence $u_1v_1u_2v_2\ldots$ of t gives rise to occurrences $u_1v'_1u_2v'_2\ldots$ of t' by replacing each occurrence v_i of a variable x_i of the left-hand side of the rule by an occurrence v'_i of x_i of the right-hand side, with the same exception in the case of a short final segment.

It is immediate that t' = U(g').

Since the depth of n is the length of the shortest path from the root of g to n, this is a lower bound on the lengths of the occurrences in U(g,n).

If the GRS is acyclic, then g and g' are acyclic, t is finite, and the finite set of redexes chosen in t is pairwise disjoint. That is, none of their occurrences is a prefix of any other. Therefore the set has a finitely long complete development with exactly as many steps as there are members of the set. □

Our description of t' corresponds to the notion of reducing all the redexes in U(r) simultaneously. If the system is not orthogonal, there may be ways of reducing them in a particular order so as not to yield the same result. However, if they are reduced in outermost-first order, this will not happen. An example is the rule $F(F(x)) \to G(x)$ and the graph y:F(y). This reduces to y:G(y) by a single graph rewrite. The unravelled term F^ω contains infinitely many redexes. If the second-outermost redex is reduced first, one can eventually obtain the normal form $F(G^\omega)$; but if one instead first reduces the outermost redex one obtains G^ω, as described by the theorem. The reader may find it instructive to work out which of the redexes in F^ω are actually reduced in the simultaneous reduction described in the proof of the theorem, and which vanish without being reduced, and to confirm that this coincides with the effect of outermost-first reduction.

The construction of Theorem 4.4 yields the following immediate corollaries.

4.5 COROLLARY. Let r be a redex and n a node of g, and let reduction of r give the graph g'. Then U(n/r) = U(n)/U(r). If n is the root of a redex r', then U(r'/r) = U(r')/U(r). □

4.6 COROLLARY. Let $g \to^* g'$ in a left-linear GRS. Then $U(g) \to^\infty U(g')$ in any unravelling of the GRS. □

4.7 THEOREM. In a left-linear GRS, let $g \to^* g'$ by a complete development of a set of non-conflicting redexes R. Then $U(g) \to^\infty U(g')$ by a complete development of some subset of U(R).

PROOF. In the complete development of R, suppose there is some step which reduces a circular redex. Since such a redex reduces to itself, we can omit that step without changing the final result. We thus obtain a reduction of g to g' which is a complete development of some subset R' of R, and which does not at any point reduce a circular redex. Note that R' is in general not simply the set of non-circular members of R, since a non-circular redex can become a circular redex through reduction of other redexes.

We can apply Theorem 4.4 to each step of the complete development of R', obtaining a reduction of U(g) to U(g'). Corollary 4.5 implies that it is a complete development of U(R'). □

5. Adequate mappings between rewrite systems

A formulation of the relationship between graph rewriting and term rewriting must take account of the fact that while, as we have seen in Theorem 4.4, every graph rewrite sequence corresponds to a term rewrite sequence, the converse does not hold even for the most well-behaved of rewrite systems. For example, consider any graph containing two references to a redex r, such as F(x:G,x), and the rule G → H. The graph unravels to the term F(G,G), which can be reduced in one step to F(G,H). This is not the unravelling of any graph to which F(x,x:G) can reduce. However, the term can be further reduced to F(H,H). F(x:G,x) can be reduced in one step to F(x:H,x), which unravels to F(H,H). In general, we find that for systems satisfying certain conditions such as orthogonality, every term reduction sequence can be extended to a sequence which does correspond to some graph reduction sequence.

In the abstract, we define the following notion of an *adequate mapping* between rewrite systems.

5.1 DEFINITION. Let (A,R_1,S_1) and (B,R_2,S_2) be finitary or infinitary term or term graph reduction systems. A mapping U from A to B is *adequate* if:

(i) U is surjective.

(ii) $a \in A$ is a normal form iff U(a) is a normal form.

(iii) If $a \rightarrow^{\infty} a'$ in A, then $U(a) \rightarrow^{\infty} U(a')$ in B.

(iv) For $a \in A$ and $b \in B$, if $U(a) \rightarrow^{\infty} b$ then there is a $a' \in A$ such that $a \rightarrow^{\infty} a'$ and $b \rightarrow^{\infty} U(a')$. See Figure 8. □

$$\begin{tikzcd} U(a) \arrow[r] & b \arrow[r, dashed] & U(a') \\ a \arrow[rr, dashed] & & a' \end{tikzcd}$$

Figure 8

We refer to the four conditions of definition 5.1 respectively as surjectivity, preservation of normal forms, preservation of reduction, and cofinality. Surjectivity of U ensures that every object of B has a representation in A. The normal form condition ensures that an object of A is a possible final result of a computation if the object of B which it represents also is, and vice versa. Preservation of reduction ensures that every computation possible in A represents some computation in B. Cofinality is the notion we described informally at the beginning of the section, ensuring that for every computation in B, there is a computation in A which can be mapped, not necessarily to that computation of B, but to some extension of it.

Condition (iii) and one direction of condition (ii) can be read as expressing a soundness condition; the remaining conditions express a notion of completeness.

6. Adequacy of finite graph rewriting for rational term rewriting

Real machines can only handle finite, though possibly cyclic, graphs. It is clear that not all infinite terms are the unravellings of finite graphs, and so the surjectivity condition cannot hold for a finitary graph rewrite system and an infinitary term rewrite system allowing arbitrary infinite terms. Therefore we seek to formulate a restricted version of infinitary term rewriting for which the unravelling of finite cyclic graph rewrite systems will yield an adequate mapping. Such a restricted version we shall call rational term rewriting.

6.1 DEFINITION. A *rational term* is a term containing only finitely many non-isomorphic subterms. □

The following equivalent characterisation is well-known.

6.2 THEOREM. A term is rational iff it is the unravelling of a finite graph.

PROOF. Let t be a rational term. Define a graph whose nodes correspond with isomorphism classes of subterms of t. Given a node n, let t' be a member of the isomorphism class corresponding to n. Attach to n the function symbol at the root of t. The successors of n are the nodes corresponding to the isomorphism classes of the successors of the root of t'. The root of the graph is the node corresponding to the isomorphism class of t itself. It is obvious that the resulting graph unravels to t. □

6.3 DEFINITION. (i)□A *rational set* of nodes of a rational term is a set of nodes such that, if each of the nodes in the set is marked (e.g. with a tick mark), the resulting term is still rational, taking the marks into account when testing isomorphism.

(ii) A *rational set of redexes* of a rational term is a set of redexes whose roots are a rational set of nodes. □

6.4 THEOREM. A set of nodes of a rational term t is rational iff there is a graph g unravelling to t, and a set of nodes of g which map by the unravelling to the given set of nodes of t.

PROOF. Similar to Theorem 6.2. □

6.5 DEFINITION. The *rational term reduction sequences* are defined by the following axioms:

(i) A complete development, of length not more than ω, of a rational set of redexes, is rational. (In particular, finite reduction sequences are rational.)

(ii) A concatenation of finitely many rational reduction sequences is rational.

(iii) There are no other rational reduction sequences.

A *rational* TRS is an infinitary TRS whose infinitely long sequences are all rational. □

An immediate consequence of the definition is that any segment of a rational reduction sequence is rational. (For a final segment of a complete development of length ω, this follows from the fact that the set of residuals of a rational set of redexes over a finite reduction sequence is a rational set; for other types of segment it follows by induction on the definition of a rational reduction sequence.)

We shall demonstrate that finitary graph rewriting is adequate for rational term rewriting, in a sense to be made precise. First, we note that adequacy in general fails for non-Church-Rosser systems. Let t_0 be reducible to both t_1 and t_2, and suppose that there is no term to which both t_1 and t_2 are reducible. Now consider a graph $g = F(x,x{:}t_0)$, where F is some function symbol of arity 2. The unravelling of g is the term $F(t_0,t_0)$, which can be reduced to $F(t_1,t_2)$. If there is no rule for reducing a term of the form F(–,–), the graph g may be reduced to $F(x,x{:}t_1)$ or $F(x,x{:}t_2)$, but cannot be reduced to any graph g' such that $F(t_1,t_2)$ reduces to U(g'). Thus the cofinality condition is not satisfied.

We have already seen that when cyclic graphs are present, the Church-Rosser property may fail, even for orthogonal systems. If we have the rules $A(x) \to x$ and $B(x) \to x$, then the graph F(x,x:A(B(x))) behaves exactly as in the above description. It can be reduced only to F(x,x:A(x)) or to F(x,x:B(x)). Its unravelling can be reduced in infinitely many steps to $F(A^\omega,B^\omega)$, which cannot be reduced to $F(A^\omega,A^\omega)$ or to $F(B^\omega,B^\omega)$.

Adequacy can therefore be expected to hold only for orthogonal systems of rewrite rules which are Church-Rosser for transfinite term reduction, that is, the almost non-collapsing systems.

We briefly remark that the almost non-collapsing condition is not quite strong enough to imply the finite Church-Rosser property for finitary orthogonal graph rewriting. An example is given by the rule $F(A,A,x) \to x$. This has two non-isomorphic circular redexes a:F(A,A,a) and a:F(b:A,b,a). The graph a:F(A,A, F(b:A,b,a)) can be reduced to either of these, and they have no common reduct. Notice that the two circular redexes unravel to the same rational term, and so this failure of the Church-Rosser property does not constitute an obstacle to the adequacy relation between this system and its unravelling. The sufficient condition for a finitary orthogonal graph rewrite system to be CR is that it be almost non-collapsing, and that if it has a collapsing rule, its left hand side must not contain two distinct subgraphs (not necessarily non-isomorphic) with the same unravelling. The underlying reason is the same as for the case of term rewriting: the existence of two non-isomorphic collapsing redexes allows the construction of a graph containing one of each, each of which collapses to the root of the other, as in the above example.

6.6 DEFINITION. Let (G,R) be a graph rewrite system. Its *rational unravelling* is the unravelling whose transfinite reduction sequences are the transfinite rational term reduction sequences over U(G). Its *infinitary unravelling* is the unravelling whose set of terms is the closure of U(G) under arbitrary transfinite reductions (denoted $\overline{U}(G)$, and whose set of sequences is the set of all transfinite reduction sequences over those terms. If (G,R) is acyclic, its *finitary unravelling* is the finitary TRS (U(G),U(R)). □

To justify this definition, we must show that the sets of terms and transfinite reduction sequences have the closure properties required by the definition of a TRS.

6.7 THEOREM. Let (G,R) be a graph rewrite system. Both U(G) and $\overline{U}(G)$ are closed under finite reduction by U(R) and the subterm relation.

PROOF. We consider first U(G). Let $t \in U(G)$, and let $t \to t'$ by reduction of a redex r by a rule in U(R). By hypothesis, $t=U(g)$ for some $g \in G$. Therefore there is a strict homomorphism from t to g, Since U(G) is closed under inverse strict homomorphisms, t itself must be in G. Graph reduction of r will yield some graph g', and by Theorem 4.4, U(g') is the result of a complete development of U(r) by term reduction. But U(r) is just r, therefore U(g') = t'. Therefore T is closed under finite reduction.

If t is a subterm of $t' \in T$, then there is a strict graph homomorphism from t to t'. Therefore $t \in G$, and $U(t) = t \in T$.

For $\overline{U}(G)$, we first note that the Compressing Lemma implies that $\overline{U}(G)$ is the set of terms which are limits of sequences of length ω, all of whose finite stages are in U(G). Hence $\overline{U}(G)$ is closed under finite reduction. For the subterm condition, consider any $t \in \overline{U}(G)$ given as the limit of a sequence of length ω whose finite stages are in U(G). Then a sequence converging to the subterm of t at any given position u can be obtained by considering a final segment of the given sequence in which the depth of each reduction is at least the length of u, taking the subterm at u of each term from that point on, and omitting all reduction steps outside that subterm. Hence the subterm of t at u is in $\overline{U}(G)$. □

6.8 DEFINITION. Given a finitary acyclic GRS (G,R), its finitary unravelling is the finitary TRS (U(G),U(R)). Given a finitary GRS (G,R), its rational unravelling is the infinitary TRS (U(G),U(R),S), where S is the set of rational reduction sequences over U(G). Its infinitary unravelling is the TRS ($\overline{U}$(G),U(R),S), and S is the set of all transfinite reduction sequences over $\overline{U}$(G). □

6.9 PROPOSITION. The set of rational reduction sequences over U(G) is closed under the constructions required by the Parallel Moves Lemma, the Strip Lemma, and the finite and transfinite Church-Rosser theorems.

PROOF. We first prove that for rational sets of redexes R and R' of the same term t, if R contains no collapsing tower, then R'/R is rational. Without loss of generality we may assume that R and R' have no members in common, since R'/R = (R'–R)/R. Since R and R' are rational, there are graphs g and g' which both unravel to t, and set of redexes R_g of g and $R_{g'}$ of g' such that $U(R_g) = R$ and $U(R_{g'})=R'$. Label each node n of t by the pair $(n_g,n_{g'})$ of nodes of g and g' for which $n \in U(n_g)$ and $n \in U(n_{g'})$. It is clear that if two nodes of t receive the same label, then the labelled subtrees rooted at those nodes must be isomorphic. Since there are only finitely many non-isomorphic unleballed subtrees of t, and only finitely many different possible labels, there can be only finitely many non-isomorphic labelled subtrees of t. (The number is bounded by the number of non-isomorphic unlabelled subtrees, the number of nodes of g, and the number of nods of g'.) It follows that there is a graph g" and two sets of redexes $R_{g''}$ and $R''_{g''}$ of g", such that U(g") = t, $U(R_{g''})=R$, and $U(R'_{g''})=R'$. It is immediate that $U(R'_{g''}/R_{g''}) = R'/R$, proving that R'/R is rational.

This immediately proves the claim concerning the Parallel Moves Lemma.

For the Strip Lemma, if the given sequence s is rational, then it is a concatenation of finitely many complete developments, and therefore the sequence s/r is constructed by finitely many applications of the Parallel Moves Lemma. Similarly, when given rational sequences s_0 and s_1, the Standard Construction underlying the Church-Rosser theorems for s_0 and s_1 is constructed by finitely many applications of the Parallel Moves Lemma. For the version relating to almost non-collapsing systems, the elision of all reductions performed in collapsing towers preserves rationality, and for the version modulo $\sim_{hc}$, the same is the case for the elision of steps performed within hyper-collapsing subterms. □

6.10 THEOREM. Under any of the following conditions, the unravelling mapping from an orthogonal GRS to a TRS is adequate.

(i) The GRS is finitary and acyclic, and the TRS is its finitary unravelling.

(ii) The GRS is finitary, the TRS is its rational unravelling, and the rule-system is almost non-collapsing.

(iii) The GRS is finitary, the TRS is its rational unravelling, and hyper-collapsing graphs and terms are identified.

PROOF. Surjectivity and preservation of normal forms are immediate in all three cases. Preservation of reduction is Corollary 4.6. For cofinality, we consider a diagram of the form of Figure 9, in which each arrow is a reduction sequence of length ≤ ω.

$$
\begin{tikzcd}
t_0 = t'_0 = U(g_0) \arrow[r, "R_0"] & t_1 = t'_1 \arrow[r, "R_1"] \arrow[d] & t_2 \arrow[r, "R_2"] \arrow[d] & t_3 \arrow[r, "R_3"] \arrow[dd] & \ldots \arrow[r, "R_{n-1}"] & t_n \arrow[dddd] \\
& U(g_1) \arrow[r, "R''_1"'] & t'_2 \arrow[d] & & & \\
& & U(g_2) \arrow[r, "R''_2"'] & t'_3 \arrow[d] & & \\
& & & U(g_3) \arrow[r, "R''_3"'] & \ldots & \\
& & & & \ldots \arrow[r, "R''_{n-1}"'] & t'_n \arrow[d] \\
& & & & & U(g_n) \\
g_0 \arrow[r, "R'_0"] & g_1 \arrow[r, "R'_1"] & g_2 \arrow[r, "R'_2"] & g_3 \arrow[r, "R'_3"] & \ldots \arrow[r, "R'_{n-1}"] & g_n
\end{tikzcd}
$$

Figure 9

The given term rewrite sequence starts from t_0 and forms the top side of the diagram. It is divided into segments, each of which is a complete development of a nonempty set of redexes R_i. (This is always possible, by definition in the rational cases, and in the finitary case because each single step is the complete development of a single redex.) g_0 is taken to be t_0, considered as a graph.

The rest of the diagram is constructed by induction. t'_0 is defined to be t_0. When the diagram has been constructed up to and including t_i, t'_i , g_i, and the sequences joining them, it is extended to the right as follows.

(1) In case (i), $U(g_i) \rightarrow t'_{i+1}$ and $t_{i+1} \rightarrow t'_{i+1}$ are the projections of R_i and $t_i \rightarrow t'_i \rightarrow U(g_i)$ over each other, which by the finitary Church-Rosser theorem for orthogonal TRSs ([Der90, Klo92]; see also the remark following Proposition 2.2.14) must end at the same term t'_{i+1}.

In case (ii), they result from these projections by omitting all steps which belong to collapsing towers. The construction which proves Proposition 2.2.16 implies that they end at the same term t'_{i+1}.

In case (iii), they result from these projections by omitting all steps which are contained in hyper-collapsing subterms. In these cases, $U(g_i) \rightarrow t'_{i+1}$ is the complete development of a subset R''_i of $R_i/(t_i \rightarrow t'_i \rightarrow U(g_i))$. Theorem 2.2.19 implies that they end at the same term t'_{i+1}.

(2) R'_i is the set of redexes r of g_i such that U(r) contains at least one element of R''_i. Briefly, $R'_i = U^{-1}(R''_i)$. $g_i \rightarrow g_{i+1}$ is a complete development of R'_i.

(3) Since every redex of $U(g_i)$ is in the unravelling of some redex of g_i, $U(R'_i)$ contains R''_i. In cases (ii) and (iii), the construction of R''_i implies that complete development of R'_i does not require the reduction of any circular redexes. Therefore by Corollary 2.2.11 and Theorem 4.7 there is an extension of $U(g_i) \rightarrow t'_{i+1}$ to a complete development of $U(R'_i)$, ending with $U(g_{i+1})$. $t'_{i+1} \rightarrow U(g_{i+1})$ is taken to be such an extension. □

7. A counterexample to adequacy for infinite rewriting

The adequacy theorem for finitary and rational rewriting fails for general infinitary graph and term rewriting, even for orthogonal systems. This is a further reason for restricting the theorem to rational term rewriting.

Infinitary graph rewriting may be formalised by analogy with infinitary term rewriting. The notion should be sufficiently intuitive that the counterexample can be understood without further formalisation. Consider the following symbols, graph rewrite rule, and initial (infinite) graph:

Symbols: The natural numbers

Rules: For each natural number **n**: **n**(x,y) → **n+1**(x,y)

Graph: $g_0 = a_0{:}0(a_0,a_1),\ a_1{:}1(a_1,a_2),\ \ldots$

For simplicity, we allow ourselves the use of infinitely many function symbols and rules. We later show how the counterexample can be expressed with only finitely many function symbols and rules.

g_0 is an infinite chain of nodes, each labelled with a different integer, pointing to itself with its first argument, and pointing to the next node with its second argument. $t_0 = U(g_0)$ is an infinite binary tree, with root labelled 0, and where each node labelled **n** has left and right descendants labelled **n** and **n+1** respectively. g_0 and t_0 are illustrated in Figure 10.

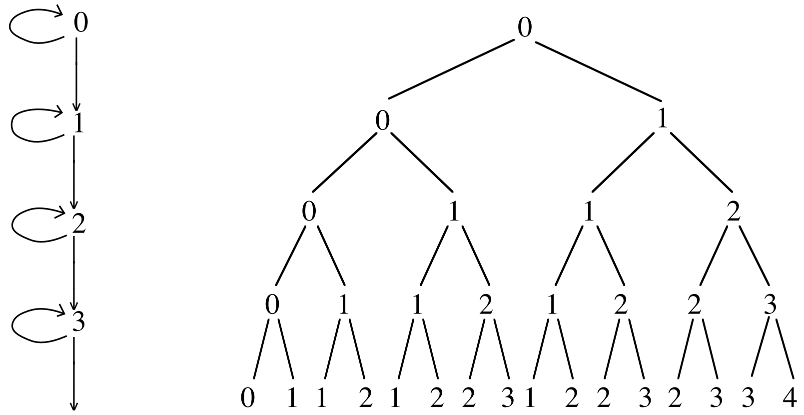

Figure 10

Every node of the term is a redex. In each row of the tree, the rightmost node has the largest label. Each node to its left in the same row can, by being reduced a finite number of times, come to have the same node label as it does. Applying such reductions to each row in the tree yields a strongly convergent reduction of the term, of length ω, converging to the term t_ω shown in Figure 11.

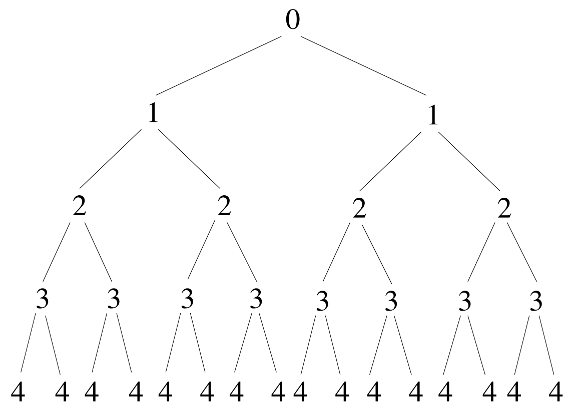

Figure 11

In this term, there are for every **n** only finitely many nodes of the form **n**(–,–). However, every graph which g_0 can be reduced to, by finite or infinite reduction, contains cycles which unravel to give an infinite number of such nodes, for some **n**. Therefore there is no graph g to which g_0 can be reduced such that $U(g) = t_\omega$. We thus see that infinitary orthogonal graph rewriting is not adequate for infinitary orthogonal term rewriting, as the cofinality condition fails.

The example used an infinite set of function symbols and rules, but it is easy to encode them into a finite set. Consider the single rule $F(x,y) \rightarrow G(F(x,y))$. Define $t_i = G^i(F(x,y))$. Then there is a reduction sequence $t_0 \rightarrow t_1 \rightarrow t_2 \rightarrow \ldots$ In the previous counterexample, replace each term of the form **n**(t,t') by $[x{:=}t,y{:=}t']t_n$. Each of the rules of the example is then an instance of the one rule. A similar counterexample results. Such a construction can be performed for any system containing an infinite term reduction sequence $t_0 \rightarrow t_1 \rightarrow t_2 \rightarrow \ldots$ (converging or otherwise) such that each term t_i contains at least one occurrence of both the variables x and y, and t_i cannot reduce to t_j when i>j.

8. Concluding remarks

8.1. Technical notes.

Theorem 4.4 underlies the intuitive notion of graph rewriting as an implementation technique. It guarantees that every computation which is possible with graph rewriting can be performed by (possibly transfinite) term rewriting. However, one requires more than that to justify the technique. One must also show that graph rewriting can always compute the normal forms which term rewriting can compute. This holds only under much more stringent conditions. The precise formulation is surprisingly complex. The main condition we impose is that the rewrite rules are orthogonal. Further conditions are also required, concerning the presence of cyclic graphs or infinite terms, the permitted forms of collapsing rules, or an axiom equating all hyper-collapsing terms.

The counterexamples demonstrated that non-left-linearity and failure of the Church-Rosser property usually cause failure of adequacy. These do not exhaust the "easy" counterexamples. A system which is non-orthogonal, but left-linear and Church-Rosser can also fail to be adequately implemented by graph rewriting, for reasons very similar to the failure in the other counterexamples. An example of this is the set of rules A → B, B → A, F(B,A) → C and the graph F(x:A,x). The term rewrite system is non-orthogonal but left-linear and Church-Rosser. The term F(A,A) reduces to the normal form C, but the graph F(x:A,x) reduces only to F(x:B,x), violating cofinality. The overlap of the rules allows the term system to compare the redex A with a reduct of A, something which is not possible in an orthogonal system. While there are non-orthogonal systems for which graph rewriting happens to be adequate, we do not foresee any significant extension of the adequacy theorem to a useful wider class of system.

In our definition of graph rewriting, we remarked that the redirection step differs from what real implementations usually do, but is equivalent. For non-collapsing rules, real implementations usually overwrite the root node of a redex with the root node of the right hand side, instead of making a new copy of the latter node and redirecting to it the in-arcs of the redex root. For a collapsing rule, the redex root is overwritten with a special "indirection" symbol and an arc pointing to the node to which the redex is collapsed. During pattern-matching of subsequent redexes, the indirection symbol is considered to be transparent; when an indirection node is required to be pattern-matched, its target is pattern-matched instead. This is more efficient than the redirection method, but more complicated to reason about. One may easily persuade oneself that the method is equivalent to redirection, with the exception of circular redexes. The redirection method reduces a circular redex to itself, while the indirection method reduces it to an indirection node whose target is itself. An attempt to pattern-match such a node will very likely be signalled as a program error. One may alternatively analyse the indirection method as equivalent to replacing the right hand side x of every collapsing rule by I(x), where I is a new symbol, and adding a rule I(x) → x, a transformation which makes the whole system hyper-collapsing.

8.2. Historical notes.

Graph rewriting as a computational mechanism for functional programming began with Wadsworth, who in his thesis [Wad71] proposed the idea of sharing in the setting of the lambda calculus, and used it to implement lazy evaluation. Term graph rewriting was first studied by Staples [Sta80], who demonstrated a correspondence with term rewriting, for finitary orthogonal rewrite systems and acyclic graphs. [Bar87] extended this by establishing results about normalising strategies. Raoult [Rao84], refined by Kennaway [Ken87], studied the relationship between graph rewriting and term rewriting using a category-theoretic description of term graph rewriting. Hoffmann and Plump [Hof88] gave an equivalent description in a different category of graphs.

All of the above treatments of the subject considered only acyclic graphs and finitary rewriting. Farmer and Watro [Far89] were the first to consider cyclic graphs and the resulting transfinite reduction sequences. They proved the soundness of term graph rewriting in this setting, but did not consider any notion of completeness. In so doing, they developed a part of the theory of transfinite term rewriting. [Far90] proves a notion of correctness for finitary cyclic graph rewriting relative to finitary term rewriting. Ariola and Arvind [Ari92] present a distinctive approach to graph rewriting which focusses on proving the correctness of practical optimisations.

References

[Abr92] S. ABRAMSKY, D. GABBAY, and T. MAIBAUM, *Handbook of Logic in Computer Science*, two vols. (Oxford University Press, 1992).

[Ari92] Z.M. ARIOLA and ARVIND, *Graph Rewriting Systems*, to appear as report, MIT (1992).

[Bar87] H.P. BARENDREGT, M.C.J.D. VAN EEKELEN, J.R.W. GLAUERT, J.R KENNAWAY, M.J. PLASMEIJER and M.R. SLEEP, Term graph rewriting, Proc. PARLE'87 Conference II, LNCS **259**, 141-158, (Springer-Verlag 1987).

[Der90] N. DERSHOWITZ and J.P. JOUANNAUD, Rewrite systems, in *Handbook of Theoretical Computer Science, volume B*, ed. J. van Leeuwen, (Elsevier, Amsterdam, 1990).

[vEB90] P. VAN EMDE BOAS, Machine models and simulations, in Algorithms and Complexity, in *Handbook of Theoretical Computer Science, volume A*, ed. J. van Leeuwen, (Elsevier, Amsterdam, 1990).

[Far89] W.M. FARMER and R.J. WATRO, Redex capturing in term graph rewriting, in *Computing with the Curry Chip,* Report M89-59, The MITRE Corporation, Massachusetts, (1989). A concise version has appeared in Proc. Conference on Rewriting Techniques and Applications, ed. R.V. Book, LNCS **488**, (Springer-Verlag 1991).

[Far90] W.M. FARMER, J.D. RAMSDELL, and R.J. WATRO, A correctness proof for combinator reduction with cycles, *ACM Trans. on Programming Languages and Systems*, **12** (1), 123–134, (1990).

[Hof88] B. HOFFMAN and D. PLUMP, Jungle evaluation for efficient term rewriting, in J. Grabowski, P. Lescanne, and W. Weckler (eds.), *Proc. Int. Workshop on Algebraic and Logic Programming*, *Mathematical Research* vol. 49, pp. 191–203 (Akademie-Verlag, 1988)

[Hue91] G. HUET and J.-J. LÉVY, Computations in orthogonal rewriting systems: I and II, in J.-L. Lassez and G.D. Plotkin, eds., *Computational Logic: Essays in Honor of Alan Robinson*, 394–443 (MIT Press, 1991)

[Ken87] J.R. KENNAWAY, On 'On graph rewritings', *Th. Comp. Sci.*, **52**, 37–58, 1987.

[Ken93a] KENNAWAY, J.R., KLOP, J.W., SLEEP, M.R., and DE VRIES, F.J., An infinitary Church-Rosser property for non-collapsing orthogonal term rewriting systems, chapter 4 of [Sle93].

[Ken93b] KENNAWAY, J.R., KLOP, J.W., SLEEP, M.R., and DE VRIES, F.J., Transfinite reductions in orthogonal term rewriting systems, Technical Report, CWI, Amsterdam, and submitted to *Information and Computation*, (1993).

[Klo92] KLOP, J.W., Term rewriting systems, in vol.2 of [Abr92] (1992).

[Mit91] MITCHELL J.C., On abstraction and the expressive power of programming languages, to appear in Proc. Int. Conf. on Theoretical aspects of Computer Software, (Sendai, Japan, 1991).

[ODo85] O'DONNELL, M.J., *Equational Logic as a Programming Language*, (MIT Press, 1985).

[Pey85] S.L. PEYTON JONES, *The Implementation of Functional Programming Languages*, (Prentice-Hall, 1987).

[Phi92] I.C.C. PHILLIPS, Recursion theory, in vol.1 of [Abr92] (1992).

[Rao84] J. C. Raoult, On graph rewritings, *Th. Comp. Sci.*, **32**, 1–24, 1984.

[Sle93] M.R. SLEEP, M.J. PLASMEIJER, and M.C.J.D. VAN EEKELEN (eds.), *Term Graph Rewriting: Theory and Practice*, (Wiley, 1993).

[Smy92] M.B. SMYTH, Topology, in vol.1 of [Abr92] (1992).

[Sta80] J. STAPLES, Computation on graph-like expressions, Th. Comp. Sci., v.10, 171–185, (1980).

[Wad71] C.P. WADSWORTH, *Semantics and Pragmatics of the Lambda-Calculus*, D.Phil. thesis, University of Oxford, 1971.