Proceedings Paper:
Plump, DETLEF (1993) Collapsed Tree Rewriting: Completeness, Confluence, and Modularity. In: Proceedings Conditional Term Rewriting Systems (CTRS 92), Third International Workshop. Lecture Notes in Computer Science. Springer, pp. 97-112.

https://doi.org/10.1007/3-540-56393-8_7

Collapsed Tree Rewriting: Completeness, Confluence, and Modularity

Detlef Plump*
Universität Bremen

Abstract

Collapsed trees are (hyper)graphs which represent functional expressions such that common subexpressions can be shared. Rewrite steps with collapsed trees include applications of term rewrite rules as well as "folding steps" which identify common subexpressions. Different aspects of this model of computation are considered: (1) It is shown that collapsed tree rewriting is complete with respect to equational validity in the same sense as term rewriting is. (2) Collapsed tree rewriting and term rewriting are compared with respect to confluence. (3) Termination and convergence of collapsed tree rewriting turn out to be modular properties, in contrast to the situation for term rewriting.

1 Introduction

The representation of functional expressions by *collapsed trees* (also known as *directed acyclic graphs*) is well-known from implementations of term rewriting and functional programming languages. The advantage over trees is that common subexpressions can be shared, so that evaluation costs are cut down both in time and space. However, while for term rewriting there is a comprehensive theory which addresses various operational and semantical aspects, comparatively little is known about collapsed tree rewriting. This appears unsatisfactory as the two models of computation do behave different in several respects. For instance, there are non-terminating term rewriting systems that become terminating under collapsed tree rewriting, and confluence may get lost by passing from terms to collapsed trees.

This paper makes three contributions to the theory of collapsed tree rewriting. First of all, it is shown that collapsed tree rewriting is complete with respect to equational validity in the same sense as term rewriting is. This result requires the "collapsing" of collapsed trees during evaluation, and hence does not hold

*Author's address: Fachbereich Mathematik und Informatik, Universität Bremen, Postfach 33 04 40, 2800 Bremen 33, Germany. e-mail: det@informatik.uni-bremen.de. Research supported by Deutsche Forschungsgemeinschaft and by ESPRIT Basic Research Working Group 3299, COMPUGRAPH.

in most of the existing approaches to function evaluation by graph rewriting ([BvEG+87, FW91, CR92] to mention a few).

Then the relationship between term rewriting and collapsed tree rewriting with respect to confluence is clarified. It turns out that term rewriting is confluent whenever collapsed tree rewriting is, but the converse does not hold. Sufficient conditions for confluence are presented.

Finally, collapsed tree rewriting is shown to behave modular when terminating or convergent (i.e. terminating and confluent) systems are combined; in both cases the combined systems are allowed to share certain function symbols. In contrast, it is known that even the disjoint union of term rewriting systems may destroy termination and convergence.

2 Collapsed Tree Rewriting

Collapsed tree rewriting is similar to *jungle evaluation* as it is described in [HP91]. Jungles are acyclic hypergraphs which represent collections of many-sorted terms, and collapsed trees are single-rooted jungles over unsorted signatures. While the restriction to the unsorted case is immaterial (and could be easily dropped at the costs of more complex definitions), a significant difference is that the rewrite steps used here include "garbage collection". The latter is natural from the perspective of term rewriting and simplifies the formulation of confluence results, as otherwise in most cases the garbage in jungles must be ignored explicitly.

Let X be a fixed set the elements of which are called *variables*. A *signature* Σ is a set of function symbols disjoint from X. Each function symbol f comes with an integer $arity(f) \geq 0$; f is called a *constant* if $arity(f) = 0$. For each variable x, define $arity(x) = 0$. From now on Σ stands for an arbitrary signature.

Definition 2.1 (hypergraph) A *hypergraph* G over Σ is a system $\langle V, E, s, t, l\rangle$, where V, E are finite sets of *nodes* and *hyperedges* (or *edges* for short) , $s\colon E \to V$, $t\colon E \to V^*$ are mappings that assign a source node and a string of target nodes to each hyperedge, and $l\colon E \to \Sigma \cup X$ is a mapping that labels each hyperedge e such that $arity(l(e))$ is the length of $t(e)$.

The components of a hypergraph G are referred to as V_G, E_G, s_G, t_G, and l_G. A node v is a *predecessor* of a node v' if there is an edge e with source v such that v' occurs in $t_G(e)$. The relations $>_G$ and $\geq_G$ are the transitive respectively reflexive-transitive closure of the predecessor relation. G is *acyclic* if there is no node v with $v >_G v$. For each node v, G/v is the subhypergraph[1] of G consisting of all nodes v' with $v \geq_G v'$ and of all edges outgoing from these nodes.

Definition 2.2 (collapsed tree) A hypergraph C over Σ is a *collapsed tree* if

(1) there is a node $root_C$ such that $root_C \geq_C v$ for each node v,

[1] Given hypergraphs U, G, U is a *subhypergraph* of G if $V_U \subseteq V_G$, $E_U \subseteq E_G$, and if s_U, t_U, l_U are restrictions of the corresponding mappings of G.

(2) C is acyclic, and

(3) each node has a unique outgoing edge, that is, s_C is bijective.

Example 2.3 Figure 1 shows a collapsed tree with function symbols $+$, $\times$, 0, and a variable x, where $arity(+) = arity(\times) = 2$ and $arity(0) = 0$. Hyperedges are depicted as boxes with inscribed labels, and circles represent nodes. A line connects each edge with its source node, while arrows point to target nodes. The order in a target string is given by the left-to-right order of the arrows leaving a box, unless it is indicated by numbered arrows.

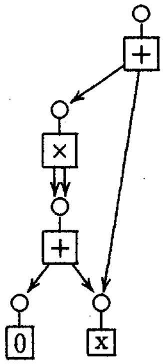

Figure 1: Collapsed tree

As usual, a *term* over Σ and X is a string that consists of a single variable or constant, or has the form $f(t_1, \ldots, t_n)$ for a function symbol f with arity n and terms $t_1, \ldots, t_n$. $T(\Sigma, X)$ is the set of all terms over Σ and X. In examples, terms with binary function symbols are written in infix notation if convenient.

Each node v in a collapsed tree represents a term which can be obtained by traversing all edges "below" v, analogously to a preorder tree traversal.

Definition 2.4 (from nodes to terms) Let C be a collapsed tree. Then the mapping $term_C \colon V_C \to T(\Sigma, X)$ is defined by

$$term_C(v) = \begin{cases} l_C(e) & \text{if } t_C(e) = \lambda,^2 \\ l_C(e)\,(term_C(v_1), \ldots, term_C(v_n)) & \text{if } t_C(e) = v_1 \ldots v_n, \end{cases}$$

where e is the unique edge with source v.

In the following $term(C)$ stands for $term_C(root_C)$.

As an example, $term(\mathrm{C}) = ((0 + \mathrm{x}) \times (0 + \mathrm{x})) + \mathrm{x}$ for the collapsed tree C in Figure 1.

[2] λ is the empty string.

A *rewrite rule* $l \to r$ over Σ consists of two terms $l, r \in T(\Sigma, X)$ such that l is not a variable and all variables in r occur also in l. Given a set R of rewrite rules, $\mathcal{R} = \langle \Sigma, R \rangle$ is a *term rewriting system*. $\to_{\mathcal{R}}$ is the rewrite relation on $T(\Sigma, X)$ associated with $\mathcal{R}$, and $\to^+_{\mathcal{R}}, \to^*_{\mathcal{R}}$ are the transitive respectively reflexive-transitive closure of $\to_{\mathcal{R}}$. (The reader is assumed to be familiar with basic concepts of term rewriting; for an introduction, see for example [DJ90, HO80, Klo90].) Until the end of section 4, $\mathcal{R}$ denotes an arbitrary term rewriting system over Σ.

Given some rewrite rule $l \to r$ and a collapsed tree C, the matching of l with a subterm of $term(C)$ is realized by a hypergraph morphism $\underline{\Diamond l} \to C$, where $\underline{\Diamond l}$ is a particular hypergraph representing l.

Definition 2.5 (hypergraph morphism) Let G,H be hypergraphs. A *hypergraph morphism* $g\colon G \to H$ is a pair of mappings $\langle g_V\colon V_G \to V_H, g_E\colon E_G \to E_H \rangle$ that preserve sources, targets, and labels; that is, $s_H \circ g_E = g_V \circ s_G$, $t_H \circ g_E = g_V^* \circ t_G$, and $l_H \circ g_E = l_G$.[3]

Definition 2.6 (tree with shared variables) A collapsed tree C is a *tree with shared variables* if (1) for each node v, $indegree_C(v) > 1$ implies $term_C(v) \in X$, and (2) for all nodes v,v', $term_C(v) = term_C(v') \in X$ implies $v = v'$.[4]

For every term t, $\Diamond t$ is a tree with shared variables such that $term(\Diamond t) = t$.

To model the matching of l in C, $\Diamond l$ is not yet the right representation of l because hypergraph morphisms preserve labels. The solution is simple: cut off all edges labelled with variables.

For every collapsed tree C, let $\underline{C}$ be the hypergraph that is obtained from C by removing all edges labelled with variables.

Definition 2.7 (evaluation step) Let C, D be collapsed trees. Then there is an *evaluation step* from C to D, denoted by $C \Rightarrow_{\mathcal{E}} D$, if there is a rewrite rule $l \to r$ in $\mathcal{R}$ and a hypergraph morphism $g\colon \underline{\Diamond l} \to C$ such that D is isomorphic[5] to the collapsed tree constructed as follows:

(1) Remove the hyperedge outgoing from $g(root_{\Diamond l})$, yielding a hypergraph C'.

(2) Build the disjoint union $C' + \underline{\Diamond r}$ and

- identify $g(root_{\Diamond l})$ with $root_{\Diamond r}$,
- for each pair $\langle v, v' \rangle \in V_{\Diamond l} \times V_{\Diamond r}$ with $term_{\Diamond l}(v) = term_{\Diamond r}(v') \in X$, identify $g(v)$ with v'.

Let C'' be the resulting hypergraph.

(3) Remove garbage, that is, the resulting collapsed tree is $C''/root_C$.

[3]Given a mapping $f\colon A \to B$, $f^*\colon A^* \to B^*$ sends λ to λ and $a_1 \ldots a_n$ to $f(a_1) \ldots f(a_n)$.

[4]$indegree_C(v)$ is defined as $\sum_{e \in E_C} \#(v, t_C(e))$, where $\#(v, t_C(e))$ is the number of occurrences of v in $t_C(e)$.

[5]Two hypergraphs G, H are *isomorphic*, denoted by $G \cong H$, if there is a hypergraph morphism $g\colon G \to H$ with g_V and g_E bijective.

Example 2.8 Figure 2 shows an evaluation step based on the rewrite rule $x \times (y + z) \to (x \times y) + (x \times z)$. Note that the morphism locating $\underline{\Diamond x \times (y + z)}$ identifies the nodes representing x and $(y + z)$.

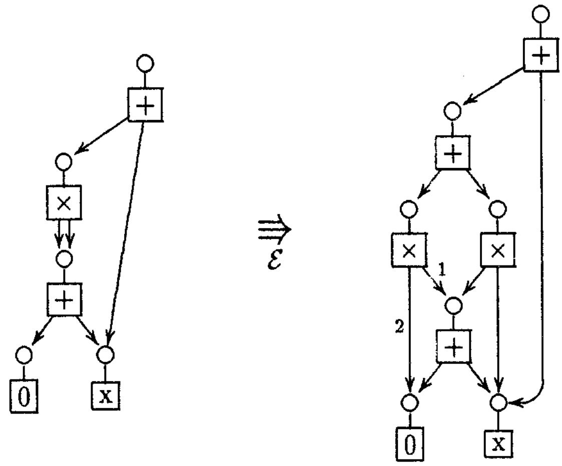

Figure 2: Evaluation step

Evaluation steps as defined above work well for *left-linear* rewrite rules: if no variable occurs more than once in a term l, then there is a morphism $\underline{\Diamond l} \to C$ into a collapsed tree C whenever l can be matched with a subterm of $term(C)$ (see Lemma 3.8 in the next section). But if l is non-linear, then no such morphism needs to exist. (For example, there is no morphism into the tree representing l.) This problem is overcome by "folding" collapsed trees prior to evaluation steps.

Definition 2.9 (folding step) Let C, D be collapsed trees. Then there is a *folding step* $C \Rightarrow_{\mathcal{F}} D$ if there are distinct edges e, e' in C with $l_C(e) = l_C(e')$ and $t_C(e) = t_C(e')$, and if D is isomorphic to the collapsed tree that is obtained from C by identifying e with e' and $s_C(e)$ with $s_C(e')$.

As an example, Figure 3 shows a folding step.

It is easy to check that folding steps preserve represented terms.

Lemma 2.10 *For every folding step $C \Rightarrow_{\mathcal{F}} D$, $term(C) = term(D)$.*

3 Soundness and Completeness

In this section it is shown that collapsed tree rewriting is sound and complete for function evaluation in the same sense as term rewriting is. Soundness is obtained by showing that collapsed tree rewriting implements a special kind of parallel

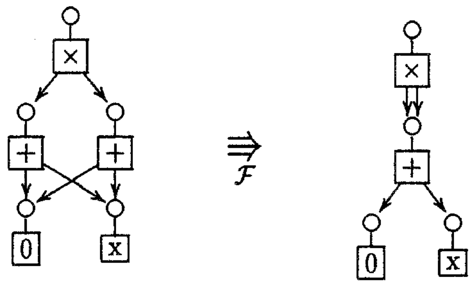

Figure 3: Folding step

term rewriting. Completeness (with respect to equational validity) follows from the fact that for every term rewrite step $t \rightarrow_{\mathcal{R}} u$, the trees representing t and u are convertible by collapsed tree rewriting.

In order to state precisely the effect of an evaluation step $C \Rrightarrow_{\mathcal{E}} D$ on $term(C)$, some notions are needed for relating positions in terms with nodes in collapsed trees.

Given a term t, the set $Pos(t)$ of *positions* in t is the subset of $\mathbb{N}^*$ defined by (1) $Pos(t) = \{\lambda\}$ if t is a variable or a constant, and (2) $Pos(t) = \{\lambda\} \cup \{i.\pi \mid 1 \leq i \leq n \text{ and } \pi \in Pos(t_i)\}$[6] if t is a composite term $f(t_1, \ldots, t_n)$. For each $\pi \in Pos(t)$, the subterm t/π is given by (1) $t/\pi = t$ if $\pi = \lambda$ and (2) $t/\pi = t_i/\rho$ if $\pi = i.\rho$ and $t = f(t_1, \ldots, t_n)$.

Definition 3.1 (from positions to nodes) Let C be a collapsed tree. Then the mapping $node_C\colon Pos(term(C)) \rightarrow V_C$ is defined by $node_C(\lambda) = root_C$ and $node_C(i.\pi) = node_{C/v_i}(\pi)$, where $v_1 \ldots v_n$ is the target string of the edge with source v.

For a node v, the number n of positions in $node_C^{-1}(v)$ is a measure for the degree of sharing at v: C/v represents n occurrences of $term_C(v)$ in $term(C)$. n is the number of paths from $root_C$ to v.

Definition 3.2 (path) Given two nodes v, v' in a collapsed tree C, a *path* from v to v' is a (possibly empty) sequence of pairs $\langle e_1, i_1\rangle, \ldots, \langle e_n, i_n\rangle$ such that $e_1, \ldots, e_n \in E_C$, $v = s_C(e_1)$, $v' = t_C(e_n)|_{i_n}$, and $t_C(e_j)|_{i_j} = s_C(e_{j+1})$ for $j = 1, \ldots, n-1$.[7]

Lemma 3.3 *Let C be a collapsed tree. Then*

(1) *for each position π in $term(C)$, $term(C)/\pi = term_C(node_C(\pi))$, and*

(2) *for each node v in C, $node_C^{-1}(v)$ contains as many positions as there are paths from $root_C$ to v.*

[6] $i.\pi$ denotes the left addition of i to the string π.

[7] For a string w, $w|_i$ is the i-th element in w.

Now the application of a rewrite rule $l \to r$ to a collapsed tree C can be explained as the parallel application of $l \to r$ at all positions in $term(C)$ that are represented by the image of $root_{\Diamond l}$ in C. To this end, let $t \rightrightarrows_\Delta u$ denote the parallel application of a rewrite rule $l \to r$ to a term t at all positions in Δ; more precisely, there is a substitution σ such that $t/\pi = \sigma(l)$ for each $\pi \in \Delta$, and u is obtained by simultaneously replacing $\sigma(l)$ with $\sigma(r)$ at all positions in Δ.

Theorem 3.4 (Soundness Theorem) *Let $C \Rrightarrow_\mathcal{E} D$ be an evaluation step based on a rewrite rule $l \to r$ and a morphism $g: \underline{\Diamond l} \to C$. Then*

$$term(C) \underset{\Delta}{\rightrightarrows} term(D)$$

through $l \to r$, where $\Delta = node_C^{-1}(g(root_{\Diamond l}))$.

Proof By a straightforward adaptation of the corresponding proof in [HP91]. □

As argued in the previous section, evaluation steps have to be combined with folding steps to cope with arbitrary term rewriting systems. A priori, there should be no restriction for mixing evaluation and folding steps; therefore the following deals with properties of the union of both relations.

Definition 3.5 $\Rrightarrow_\mathcal{R} = \Rrightarrow_\mathcal{E} \cup \Rrightarrow_\mathcal{F}$.

Given collapsed trees C, D, $C \Rrightarrow_\mathcal{R}^* D$ expresses that there is a sequence $C \Rrightarrow_\mathcal{R} \ldots \Rrightarrow_\mathcal{R} D$ or that $C \cong D$.

Corollary 3.6 *For all collapsed trees C, D,*

$$C \Rrightarrow_\mathcal{R}^* D \text{ implies } term(C) \to_\mathcal{R}^* term(D).$$

Proof By the Soundness Theorem and Lemma 2.10, $term(C) \to_\mathcal{R}^* term(D)$ whenever $C \Rrightarrow_\mathcal{R}^+ D$. Moreover, $C \cong D$ implies $term(C) = term(D)$. □

Corollary 3.6 establishes the soundness of collapsed tree rewriting in the sense of first-order logic. For it is well-known that whenever $t \to_\mathcal{R}^* u$ for two terms t, u, then t and u are equal in every model of $\mathcal{R}$, where the rules in $\mathcal{R}$ are considered as universally quantified equations. By the completeness theorem of Birkhoff [Bir35], term rewriting is also complete in the sense that if two terms t, u are equal in all models, then t and u are equivalent with respect to the equivalence generated by $\to_\mathcal{R}$. In the rest of this section it is shown that collapsed tree rewriting is complete in the same sense: two collapsed trees are equivalent under the equivalence generated by $\Rrightarrow_\mathcal{R}$ whenever their represented terms are equivalent.

Definition 3.7 Let $=_\mathcal{R}$ and $\equiv_\mathcal{R}$ be the reflexive-symmetric-transitive closure of $\to_\mathcal{R}$ and $\Rrightarrow_\mathcal{R}$, respectively.

While soundness of collapsed tree rewriting follows from the fact that every sequence of $\Rrightarrow_{\mathcal{R}}$-steps corresponds to a sequence of $\rightarrow_{\mathcal{R}}$-steps, there is no such direct correspondence from which completeness could be obtained. Indeed, the converse of Corollary 3.6 does not hold. As a simple counterexample, consider the rewrite rules $\mathrm{f(x)} \rightarrow \mathrm{g(x,x)}$ and $\mathrm{a} \rightarrow \mathrm{b}$, where x is a variable and a, b are constants. Then $\mathrm{f(a)} \rightarrow_{\mathcal{R}} \mathrm{g(a,a)} \rightarrow_{\mathcal{R}} \mathrm{g(a,b)}$, but there is no corresponding sequence of $\Rrightarrow_{\mathcal{R}}$-steps because of the sharing introduced by an evaluation step with the first rule.

Nevertheless, collapsed tree rewriting is complete with respect to equational validity. The point is that $\equiv_{\mathcal{R}}$ comprises "reversed folding steps" which allow to unfold collapsed trees between evaluation steps. As a consequence, every rewrite step $t \rightarrow_{\mathcal{R}} u$ can be lifted to a $\equiv_{\mathcal{R}}$-conversion of the trees representing t and u, leading straightforward to the completeness result.

Before the "Lifting Lemma" can be stated, the different kinds of matching for collapsed trees and terms have to be related. The following lemma is proved similar to Theorem 3.12 in [HP91].

Lemma 3.8 (morphisms and substitutions) *Let C be a collapsed tree and l be a term. Then there is a morphism $g\colon \underline{\Diamond l} \rightarrow C$ with $g(root_{\Diamond l}) = root_C$ if and only if there is a substitution σ such that*

(1) *$term(C) = \sigma(l)$, and*

(2) *for all positions π, ρ in l, $l/\pi = l/\rho \in X$ implies $node_C(\pi) = node_C(\rho)$.*

In the simulation of a single term rewrite step by collapsed tree rewriting, two distinguished representations of terms are particularly useful.

Definition 3.9 (tree and fully collapsed tree) A collapsed tree C is a *tree* if no node has an indegree greater than one. C is a *fully collapsed tree* if each two different nodes represent different subterms, that is, if $term_C$ is injective.

For every term t, $\vartriangle t$ and $\blacktriangle t$ denote a tree respectively a fully collapsed tree representing t.

It can be shown that $\vartriangle t$ and $\blacktriangle t$ are unique up to isomorphism. Moreover, $\vartriangle t$ can be transformed by folding steps into each collapsed tree representing t, which in turn can be transformed into $\blacktriangle t$.

Lemma 3.10 (existence of foldings) *For every collapsed tree C,*

$$\vartriangle term(C) \overset{*}{\underset{\mathcal{F}}{\Rrightarrow}} C \overset{*}{\underset{\mathcal{F}}{\Rrightarrow}} \blacktriangle term(C).$$

Lemma 3.11 (Lifting Lemma) *For every term rewrite step $t \rightarrow_{\mathcal{R}} u$ there are collapsed trees C, D such that*

$$\vartriangle t \overset{*}{\underset{\mathcal{F}}{\Rrightarrow}} C \underset{\mathcal{E}}{\Rrightarrow} D \overset{*}{\underset{\mathcal{F}}{\Lleftarrow}} \vartriangle u.$$

Proof Let $l \to r$ be the rule applied in $t \to_{\mathcal{R}} u$, σ be the associated substitution, and π be the position in t where $\sigma(l)$ is replaced by $\sigma(r)$. Then $\Delta t/node_{\Delta t}(\pi) \cong \Delta\sigma(l)$ and the folding $\Delta\sigma(l) \Rrightarrow^*_{\mathcal{F}} \blacktriangle\sigma(l)$ can be embedded in Δt, yielding a folding $\Delta t \Rrightarrow^*_{\mathcal{F}} C$. Then $C/node_C(\pi) \cong \blacktriangle\sigma(l)$ and each variable in l corresponds to a unique node in $\blacktriangle\sigma(l)$. Hence, by Lemma 3.8, there is a morphism from $\underline{\Diamond l}$ to $C/node_C(\pi)$ that maps $root_{\Diamond l}$ to $node_C(\pi)$; let $g\colon \underline{\Diamond l} \dashrightarrow C$ be the extension of this morphism to C. Then there is an evaluation step $C \Rrightarrow_{\mathcal{E}} D$ based on $l \to r$ and g. As there is only one path from $root_C$ to $node_C(\pi)$, the Soundness Theorem 3.4 yields $term(C) \rightrightarrows_{\{\pi\}} term(D)$; thus $term(D) = u$. Finally, $\Delta u \Rrightarrow^*_{\mathcal{F}} D$ holds by Lemma 3.10. □

For simplicity, the folding $\Delta t \Rrightarrow^*_{\mathcal{F}} C$ in the proof of the Lifting Lemma makes $\Delta t/node_{\Delta t}(\pi)$ fully collapsed. But it is often possible to do with a non-complete folding, as Lemma 3.8 requires only that multiple occurrences of a variable in l correspond to the same node in C. In particular, this folding can be omitted if l is a linear term.

Theorem 3.12 (Completeness Theorem) *For all collapsed trees C, D,*

$$C \equiv_{\mathcal{R}} D \text{ if and only if } term(C) =_{\mathcal{R}} term(D).$$

Proof If $C \equiv_{\mathcal{R}} D$, then $term(C) =_{\mathcal{R}} term(D)$ follows from Corollary 3.6 by induction on the number of $\Rrightarrow_{\mathcal{R}}$-steps constituting the equivalence.

Conversely, assume that $term(C) =_{\mathcal{R}} term(D)$. Then, by induction on the number of $\to_{\mathcal{R}}$-steps, the Lifting Lemma 3.11 yields $\Delta term(C) \equiv_{\mathcal{R}} \Delta term(D)$. With Lemma 3.10 follows $C \equiv_{\mathcal{R}} \Delta term(C) \equiv_{\mathcal{R}} \Delta term(D) \equiv_{\mathcal{R}} D$. □

It is important to notice that completeness fails when folding steps are discarded from $\Rrightarrow_{\mathcal{R}}$. This is demonstrated by the following example.

Example 3.13 Let $\mathcal{R}$ contain the following rules:

$$\begin{aligned} &\mathrm{a} \to \mathrm{f(a)} \\ &\mathrm{g(x)} \to \mathrm{p(x, f(x))} \\ &\mathrm{h(x)} \to \mathrm{p(f(x), x)} \end{aligned}$$

Then $\mathrm{g(a)} =_{\mathcal{R}} \mathrm{h(a)}$ since both terms are rewritable to $\mathrm{p(f(a), f(a))}$. But $\Delta\mathrm{g(a)}$ and $\Delta\mathrm{h(a)}$ are not convertible by evaluation steps alone, as can be easily checked.

4 Confluence

By the Completeness Theorem 3.12 it is possible to prove equality of terms by collapsed tree rewriting, but in general such proofs are impractical since they involve "reversed" rewrite and folding steps. As for term rewriting, the remedy is to restrict to confluent systems; then, as a consequence of completeness, equality can be proved by searching for common reducts of collapsed trees.

It turns out, though, that $\Rrightarrow_{\mathcal{R}}$ needs not to be confluent when $\rightarrow_{\mathcal{R}}$ is. The main result in this section shows that this phenomenon is ruled out when $\Rrightarrow_{\mathcal{R}}$ is weakly terminating. As a result, confluence of collapsed tree rewriting can be established for two important classes of confluent term rewriting systems.

Recall that $\rightarrow_{\mathcal{R}}$ is *confluent* if for all terms t, t_1, t_2 with $t_1 \;{}_{\mathcal{R}}^{*}\!\!\leftarrow t \rightarrow_{\mathcal{R}}^{*} t_2$, there is a term t_3 such that $t_1 \rightarrow_{\mathcal{R}}^{*} t_3 \;{}_{\mathcal{R}}^{*}\!\!\leftarrow t_2$. Confluence of $\Rrightarrow_{\mathcal{R}}$ is defined analogously.

Corollary 4.1 *If $\Rrightarrow_{\mathcal{R}}$ is confluent, then the following holds:*

(1) *For all collapsed trees C, D, $term(C) =_{\mathcal{R}} term(D)$ if and only if there is a collapsed tree E such that $C \Rrightarrow_{\mathcal{R}}^{*} E \;{}_{\mathcal{R}}^{*}\!\!\Lleftarrow D$.*

(2) *$\rightarrow_{\mathcal{R}}$ is confluent.*

Proof (1) It is well-known that confluence is equivalent to the *Church-Rosser property*, the latter meaning here that $C \equiv_{\mathcal{R}} D$ if and only if $C \Rrightarrow_{\mathcal{R}}^{*} E \;{}_{\mathcal{R}}^{*}\!\!\Lleftarrow D$ for some collapsed tree E. So the proposition follows from the Completeness Theorem.

(2) Let $t_1 \;{}_{\mathcal{R}}^{*}\!\!\leftarrow t \rightarrow_{\mathcal{R}}^{*} t_2$ for some terms t, t_1, t_2, and consider collapsed trees T_1, T_2 with $term(T_i) = t_i$, for $i = 1, 2$. Then $t_1 =_{\mathcal{R}} t_2$ and hence, by (1), there is a collapsed tree T_3 such that $T_1 \Rrightarrow_{\mathcal{R}}^{*} T_3 \;{}_{\mathcal{R}}^{*}\!\!\Lleftarrow T_2$. With Corollary 3.6 follows $t_1 \rightarrow_{\mathcal{R}}^{*} term(T_3) \;{}_{\mathcal{R}}^{*}\!\!\leftarrow t_2$. Thus $\rightarrow_{\mathcal{R}}$ is confluent. □

The following counterexample reveals that the converse of Corollary 4.1(2) does not hold.

Example 4.2 Suppose that $\mathcal{R}$ contain the following rules:

$$\begin{aligned} &\mathrm{f(x)} \rightarrow \mathrm{g(x,x)} \\ &\mathrm{a} \rightarrow \mathrm{b} \\ &\mathrm{g(a,b)} \rightarrow \mathrm{c} \\ &\mathrm{g(b,b)} \rightarrow \mathrm{f(a)} \end{aligned}$$

Based on the following rewrite steps, structural induction shows that every term has a unique normal form:

$$\begin{tikzcd} \mathrm{g(a,b)} \arrow[r] \arrow[d] & \mathrm{g(b,b)} \arrow[r] & \mathrm{f(a)} \arrow[d] \\ \mathrm{c} & & \mathrm{g(a,a)} \arrow[llu] \end{tikzcd}$$

So $\rightarrow_{\mathcal{R}}$ is confluent, but $\Rrightarrow_{\mathcal{R}}$ is not. To see this, consider the rewrite steps $\Delta\mathrm{c} \Lleftarrow \Delta\mathrm{g(a,b)} \Rrightarrow \Delta\mathrm{g(b,b)} \Rrightarrow \Delta\mathrm{f(a)} \Rrightarrow \blacktriangle\mathrm{g(a,a)}$. There is no step from $\blacktriangle\mathrm{g(a,a)}$ back to $\Delta\mathrm{g(a,b)}$, and hence the only collapsed trees derivable from $\blacktriangle\mathrm{g(a,a)}$ are $\blacktriangle\mathrm{g(b,b)}$, $\Delta\mathrm{f(a)}$, and $\blacktriangle\mathrm{g(a,a)}$ itself. Thus $\Rrightarrow_{\mathcal{R}}$ is non-confluent.

Call a term t a *normal form* if there is no rewrite step $t \rightarrow_{\mathcal{R}} t'$. Analogously, a collapsed tree C is a normal form if there is no step $C \Rrightarrow_{\mathcal{R}} C'$. $\rightarrow_{\mathcal{R}}$ respectively

$\Rrightarrow_{\mathcal{R}}$ is *weakly terminating* if every term respectively collapsed tree is rewritable to a normal form.

It turns out that a counterexample as above is only possible if some collapsed trees are not reducible to a normal form. The following lemma is needed for the proof of this result.

Lemma 4.3 (1) *A collapsed tree C is fully collapsed if and only if there is no folding step $C \Rrightarrow_{\mathcal{F}} D$.*

(2) *A term t is a normal form if and only if $\blacktriangle t$ is a normal form.*

Proof See Theorems 4.7 and 6.3 in [HP91]. □

Theorem 4.4 *If $\Rrightarrow_{\mathcal{R}}$ is weakly terminating, then $\Rrightarrow_{\mathcal{R}}$ is confluent if and only if $\rightarrow_{\mathcal{R}}$ is confluent.*

Proof One direction of the proposition follows directly from Corollary 4.1. For the other direction, assume that $\rightarrow_{\mathcal{R}}$ is confluent and $\Rrightarrow_{\mathcal{R}}$ weakly terminating. Let C, D, E be collapsed trees such that $D \,{}^{*}_{\mathcal{R}}\!\Lleftarrow C \Rrightarrow^{*}_{\mathcal{R}} E$. Then there are normal forms D', E' with $D \Rrightarrow^{*}_{\mathcal{R}} D'$ and $E \Rrightarrow^{*}_{\mathcal{R}} E'$. Corollary 3.6 yields $term(D') \,{}^{*}_{\mathcal{R}}\!\leftarrow term(C) \rightarrow^{*}_{\mathcal{R}} term(E')$. By Lemma 4.3(1), $D' \cong \blacktriangle term(D')$ and $E' \cong \blacktriangle term(E')$. Hence, according to Lemma 4.3(2), $term(D')$ and $term(E')$ are normal forms. So $term(D') = term(E')$ since $\rightarrow_{\mathcal{R}}$ is confluent. It follows $D' \cong E'$ by the uniqueness of fully collapsed trees. Thus $\Rrightarrow_{\mathcal{R}}$ is confluent. □

It is interesting that the above proof fits into Curien and Ghelli's schema for showing confluence of weakly terminating systems [CG91] (although the proof was obtained prior to the publication of [CG91]). The interpretation function used in their method is here the *term*-function which assigns to every collapsed tree the represented term.

With Theorem 4.4 collapsed tree rewriting can be shown to be confluent for two important classes of confluent term rewriting systems. Call $\mathcal{R}$ *terminating* if there is no infinite sequence $t_1 \rightarrow_{\mathcal{R}} t_2 \rightarrow_{\mathcal{R}} \ldots$ Furthermore, $\mathcal{R}$ is *orthogonal* if all rules are left-linear and if left-hand sides of distinct rules do not overlap. (Two terms l and l' *overlap* if there are substitutions σ, σ' such that $\sigma(l/\pi) = \sigma'(l')$ for some position π in l with $l/\pi \notin X$.)

Corollary 4.5 *$\Rrightarrow_{\mathcal{R}}$ is confluent in the following two cases:*

(1) *$\rightarrow_{\mathcal{R}}$ is confluent and terminating.*

(2) *$\mathcal{R}$ is orthogonal and $\rightarrow_{\mathcal{R}}$ is weakly terminating.*

Proof (1) Let $\rightarrow_{\mathcal{R}}$ be confluent and terminating and suppose there were an infinite sequence $C_1 \Rrightarrow_{\mathcal{R}} C_2 \Rrightarrow_{\mathcal{R}} \ldots$ Then this sequence contains infinitely many evaluation steps because $\Rrightarrow_{\mathcal{F}}$ is terminating. Since folding preserves terms, the Soundness Theorem implies the existence of an infinite sequence of $\rightarrow_{\mathcal{R}}$-steps, contradicting the assumption. So $\Rrightarrow_{\mathcal{R}}$ is terminating, and hence confluent by Theorem 4.4.

(2) Let C be some collapsed tree and $\Omega = \{\omega_1, \ldots, \omega_n\}$ be a set of outermost redexes in $term(C)$ such that (i) for each outermost redex ω in $term(C)$ there is $\omega_i \in \Omega$ with $term(C)/\omega = term(C)/\omega_i$, and (ii) $term(C)/\omega_i \neq term(C)/\omega_j$ for $i \neq j$. Then there is a sequence $term(C) = t_1 \rightrightarrows_{\Delta_1} t_2 \rightrightarrows_{\Delta_2} \ldots \rightrightarrows_{\Delta_n} t_{n+1}$ with $\Delta_i = \{\pi \mid t_i/\pi = t_i/\omega_i\}$ (this sequence is well-defined because $\mathcal{R}$ is orthogonal). By the Soundness Theorem and Lemma 3.10, there is a sequence

$$C = C_1 \Rrightarrow^*_{\mathcal{F}} \overline{C_1} \Rrightarrow_{\mathcal{E}} C_2 \Rrightarrow^*_{\mathcal{F}} \overline{C_2} \Rrightarrow_{\mathcal{E}} \ldots \Rrightarrow_{\mathcal{E}} C_{n+1} \Rrightarrow^*_{\mathcal{F}} \overline{C_{n+1}}$$

such that for $i = 1, \ldots, n+1$, $\overline{C_i}$ is fully collapsed and $term(\overline{C_i}) = t_i$. Define C' as $\overline{C_{n+1}}$. Then the sequence $C \Rrightarrow^*_{\mathcal{R}} C' \Rrightarrow^*_{\mathcal{R}} C'' \Rrightarrow^*_{\mathcal{R}} \ldots$ terminates in a normal form whenever $term(C)$ has a normal form; this is because $t_1 \rightrightarrows_{\Delta_1} t_2 \rightrightarrows_{\Delta_2} \ldots$ is *eventually outermost* in the sense of O'Donnell, see Theorem 10 in [O'D77]. Hence, if $\rightarrow_{\mathcal{R}}$ is weakly terminating, then so is $\Rrightarrow_{\mathcal{R}}$. Now the proposition follows by Theorem 4.4 and the well-known fact that orthogonal term rewriting systems are confluent. □

In point (1) of Corollary 4.5, termination cannot be relaxed to weak termination since $\rightarrow_{\mathcal{R}}$ is weakly terminating in Example 4.2. In (2), weak termination cannot be dropped as otherwise folding steps may cause non-confluence. This is demonstrated by the following example.

Example 4.6 Let $\mathcal{R}$ contain the single rule $\mathrm{a} \rightarrow \mathrm{f(a)}$ and assume that there is a binary function symbol g. Then

$$\blacktriangle\mathrm{g(f(a), f(a))} \underset{\mathcal{E}}{\Lleftarrow} \blacktriangle\mathrm{g(a, a)} \underset{\mathcal{F}}{\Lleftarrow} \Delta\mathrm{g(a, a)} \underset{\mathcal{E}}{\Rrightarrow} \Delta\mathrm{g(a, f(a))} \underset{\mathcal{F}}{\Rrightarrow} \blacktriangle\mathrm{g(a, f(a))},$$

but the outer collapsed trees do not have a common reduct.

5 Modularity

This section is concerned with modular properties of collapsed tree rewriting, that is, properties that are preserved when different term rewriting systems are combined (see [Mid90] for an excellent survey on modular properties of term rewriting). The first result establishes the modularity of termination, and can be proved by adapting the proof of the corresponding result for jungle evaluation in [Plu91].

Definition 5.1 Two term rewriting systems $\langle \Sigma_0, R_0 \rangle$ and $\langle \Sigma_1, R_1 \rangle$ are *crosswise disjoint* if the function symbols in the left-hand sides of R_i do not occur in the right-hand sides of R_{1-i}, for $i = 0, 1$.

Theorem 5.2 (modularity of termination) *Let $R = \langle \Sigma_0 \cup \Sigma_1, R_0 \cup R_1 \rangle$ and $\mathcal{R}_i = \langle \Sigma_i, R_i \rangle$, $i = 0, 1$, be term rewriting systems such that $\mathcal{R}_0$ and $\mathcal{R}_1$ are crosswise disjoint. Then $\Rrightarrow_{\mathcal{R}}$ is terminating if and only if $\Rrightarrow_{\mathcal{R}_0}$ and $\Rrightarrow_{\mathcal{R}_1}$ are terminating.*

This result does not hold for term rewriting, even if Σ_0 and Σ_1 are disjoint [Toy87]. Even worse, if both systems are terminating and confluent, their disjoint union still needs not to terminate [Toy87, Dro89].

In the rest of this section it is shown that confluence together with termination is a modular property of collapsed tree rewriting, provided the given systems are crosswise disjoint and their left-hand sides do not mutually overlap. This result is not obvious as confluence alone may get lost already by signature extensions (see the example below Lemma 5.8).

At first an excursion to jungle evaluation is made in order to show that evaluation steps with non-overlapping rules commute. Call a hypergraph G a *jungle* if conditions (2) and (3) of Definition 2.2 are satisfied, and write $G \Rightarrow H$ if H is isomorphic to the jungle constructed in steps (1) and (2) of Definition 2.7 (with C standing for G).

The following lemma is an immediate consequence of the definition of $\Rrightarrow_{\mathcal{E}}$.

Lemma 5.3 *For every evaluation step $C \Rrightarrow_{\mathcal{E}} D$ there is a jungle $\overline{D}$ such that $C \Rightarrow \overline{D}$ and $D = \overline{D}/root_C$.*

For collapsed trees C, D, write $C \Rrightarrow_{\mathcal{E}}^{\lambda} D$ if $C \Rrightarrow_{\mathcal{E}} D$ or $C \cong D$.

Lemma 5.4 *Let G, H be jungles such that $G \Rightarrow H$. Then $G/v \Rrightarrow_{\mathcal{E}}^{\lambda} H/v$ for each node v in G.*

Proof Let $l \to r$ and $g: \underline{\Diamond l} \to G$ be the rewrite rule and the morphism associated with $G \Rightarrow H$.

Case 1: $g(root_{\Diamond l}) \in V_{G/v}$. Then g can be restricted to a morphism $\underline{\Diamond l} \to G/v$, so $G/v \Rightarrow X$ for some jungle X. By the construction of $G \Rightarrow H$, the removal of $G - G/v$ from H yields a well-defined jungle X'. Moreover, this removal commutes with the two construction steps that transform G into H. Hence $X = X'$. Then $X/v = (H - (G - G/v))/v = H/v$ because the nodes and edges in $G - G/v$ are unreachable from v in H. Thus $G/v \Rrightarrow_{\mathcal{E}} X/v = H/v$.

Case 2: $g(root_{\Diamond l}) \notin V_{G/v}$. Then the edge removed by $G \Rightarrow H$ is in $G - G/v$. Also, none of the inserted nodes and edges is reachable from v. Hence $G/v \cong H/v$. □

Theorem 5.5 (commutation of evaluation steps) *Let $C \Rrightarrow_{\mathcal{E}} D_i$ be an evaluation step based on a rewrite rule $l_i \to r_i$ and a morphism $g_i: \underline{\Diamond l_i} \to C$, and let e_i be the edge removed in C ($i = 0, 1$). If $e_i \notin g_{1-i}(\underline{\Diamond l_{1-i}})$, then there is a collapsed tree E such that $D_0 \Rrightarrow_{\mathcal{E}}^{\lambda} E \;{}_{\mathcal{E}}^{\lambda}\!\Lleftarrow D_1$.*

Proof By Lemma 5.3 there are jungles $\overline{D}_0, \overline{D}_1$ such that $C \Rightarrow \overline{D}_i$ and $D_i = \overline{D}_i/root_C$, for $i = 0, 1$. By choosing suitable hypergraph rules, these steps become direct derivations in the "Berlin approach" to (hyper)graph rewriting, see [HP91]. In particular, the above condition guarantees that the two derivations are "parallel independent" in the sense of [EK76]; then the Church-Rosser Theorem I in that paper[8] shows that there is a jungle F such that $\overline{D}_0 \Rightarrow E \Leftarrow \overline{D}_1$. With Lemma 5.4 follows $D_0 \Rrightarrow_{\mathcal{E}}^{\lambda} E/root_C \;{}_{\mathcal{E}}^{\lambda}\!\Lleftarrow D_1$. □

[8] The result is formulated for graphs, but the category-theoretic proof allows a straightforward adaptation to the hypergraph case.

Definition 5.6 Two term rewriting systems $\langle \Sigma_0, R_0 \rangle$ and $\langle \Sigma_1, R_1 \rangle$ are *non-interfering* if no left-hand side of R_i overlaps a left-hand side of R_{1-i}, for $i = 0, 1$.

Given a term rewriting system $\mathcal{R}$, $\Rrightarrow_\mathcal{R}$ is *locally confluent* if for all collapsed trees C, D_0, D_1 with $D_0 \ {}_\mathcal{R}\!\Lleftarrow C \Rrightarrow_\mathcal{R} D_1$, there is a collapsed tree E such that $D_0 \Rrightarrow^*_\mathcal{R} E \ {}^*_\mathcal{R}\!\Lleftarrow D_1$. $\Rrightarrow_\mathcal{R}$ is *convergent* if it is terminating and confluent.

Theorem 5.5 implies that local confluence of collapsed tree rewriting is preserved by the union of non-interfering term rewriting systems over the same signature.

Corollary 5.7 *Let $\mathcal{R} = \langle \Sigma, R_0 \cup R_1 \rangle$ be a term rewriting system such that $\mathcal{R}_0 = \langle \Sigma, R_0 \rangle$ and $\mathcal{R}_1 = \langle \Sigma, R_1 \rangle$ are non-interfering. If $\Rrightarrow_{\mathcal{R}_0}$ and $\Rrightarrow_{\mathcal{R}_1}$ are locally confluent, then so is $\Rrightarrow_\mathcal{R}$.*

Proof Let $\Rrightarrow_{\mathcal{R}_0}, \Rrightarrow_{\mathcal{R}_1}$ be locally confluent and C, D_0, D_1 be collapsed trees such that $D_0 \ {}_\mathcal{R}\!\Lleftarrow C \Rrightarrow_\mathcal{R} D_1$. By assumption, D_0 and D_1 have a common reduct if one of the two steps is a folding step, or if both steps are applications of rewrite rules from either $\mathcal{R}_0$ or $\mathcal{R}_1$. Assume therefore $C \Rrightarrow_\mathcal{E} D_i$, based on a rule $l_i \to r_i$ from $\mathcal{R}_i$ and a morphism $g_i \colon \underline{\Diamond l_i} \to C$, for $i = 0, 1$. Then, by Theorem 5.5, $D_0 \Rrightarrow^\lambda_\mathcal{E} E \ {}^\lambda_\mathcal{E}\!\Lleftarrow D_1$ for some collapsed tree E, provided that the edge removed by $C \Rrightarrow_\mathcal{E} D_i$ is not in $g_{1-i}(\underline{\Diamond l_{1-i}})$, for $i = 0, 1$. The latter holds as otherwise l_0 and l_1 would overlap by the substitutions induced by g_0 and g_1. $\square$

Lemma 5.8 *Let $\mathcal{R} = \langle \Sigma, R \rangle$ and $\mathcal{R}' = \langle \Sigma', R \rangle$ be term rewriting systems such that $\Sigma \subseteq \Sigma'$. If $\Rrightarrow_\mathcal{R}$ is convergent, then so is $\Rrightarrow_{\mathcal{R}'}$.*

Proof Let $\Rrightarrow_\mathcal{R}$ be convergent. As $\mathcal{R}'$ is the union of $\mathcal{R}$ and $\langle \Sigma', \emptyset \rangle$, Theorem 5.2 shows that $\Rrightarrow_{\mathcal{R}'}$ is terminating (since collapsed tree rewriting for $\langle \Sigma', \emptyset \rangle$ consists only of folding steps). Furthermore, $\to_\mathcal{R}$ is confluent by Corollary 4.1. Then $\to_{\mathcal{R}'}$ is also confluent, see Proposition 3.1.2 in [Mi 190], so Theorem 4.4 yields the confluence of $\Rrightarrow_{\mathcal{R}'}$. $\square$

In contrast to the situation for term rewriting, Lemma 5.8 does not hold for confluence alone. This can be seen with the one rule system $\mathrm{a} \to \mathrm{f(a)}$ over the signature $\{\mathrm{a}, \mathrm{f}\}$, for which collapsed tree rewriting is clearly confluent. But Example 4.6 demonstrates that an additional binary function symbol causes non-confluence.

Theorem 5.9 (modularity of convergence) *Let $\mathcal{R} = \langle \Sigma_0 \cup \Sigma_1, R_0 \cup R_1 \rangle$ and $\mathcal{R}_i = \langle \Sigma_i, R_i \rangle$, $i = 0, 1$, be term rewriting systems such that $\mathcal{R}_0$ and $\mathcal{R}_1$ are crosswise disjoint and non-interfering. If $\Rrightarrow_{\mathcal{R}_0}$ and $\Rrightarrow_{\mathcal{R}_1}$ are convergent, then so is $\Rrightarrow_\mathcal{R}$.*

Proof Let $\Rrightarrow_{\mathcal{R}_0}$ and $\Rrightarrow_{\mathcal{R}_1}$ be convergent. By Theorem 5.2, $\Rrightarrow_\mathcal{R}$ is terminating. Hence, by Newman's Lemma (see for example [Hue80]), it remains to show that $\Rrightarrow_\mathcal{R}$ is locally confluent. Let $\mathcal{R}'_i = \langle \Sigma_0 \cup \Sigma_1, R_i \rangle$, for $i = 0, 1$; by Lemma 5.8, $\Rrightarrow_{\mathcal{R}'_0}$ and $\Rrightarrow_{\mathcal{R}'_1}$ are convergent. Then Corollary 5.7 shows that $\Rrightarrow_\mathcal{R}$ is locally confluent. $\square$

6 Conclusion

Collapsed tree rewriting is an efficient model of function evaluation which is sound and complete in the same sense as term rewriting is. Folding steps that identify common subexpressions are necessary to achieve completeness and to cope with non-left-linear rewrite rules. Rewriting collapsed trees rather than terms or trees makes a difference for operational properties like confluence and termination. The class of confluent term rewriting systems properly includes those systems which are confluent under collapsed tree rewriting; but to the latter belong all convergent and all weakly terminating orthogonal systems. In contrast to confluence, collapsed tree rewriting terminates for more rule systems than term rewriting does; in particular, termination is preserved by combinations of systems under suitable conditions. This allows also to establish modularity of convergence. Both modularity results do not hold for term rewriting.

References

[Bir35] Garrett Birkhoff. On the structure of abstract algebras. *Proceedings of the Cambridge Philosophical Society*, 31:433–454, 1935.

[BvEG+87] 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. In *Proc. Parallel Architectures and Languages Europe*, pages 141–158. Springer Lecture Notes in Computer Science 259, 1987.

[CG91] Pierre-Louis Curien and Giorgio Ghelli. On confluence for weakly normalizing systems. In *Proc. Rewriting Techniques and Applications*, pages 215–225. Springer Lecture Notes in Computer Science 488, 1991.

[CR92] Andrea Corradini and Francesca Rossi. Hyperedge replacement jungle rewriting for term rewriting systems and logic programming. *Theoretical Computer Science*, 1992. To appear.

[DJ90] Nachum Dershowitz and Jean-Pierre Jouannaud. Rewrite systems. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science*, vol. B, chapter 6. Elsevier, 1990.

[Dro89] Klaus Drosten. *Termersetzungssysteme*. Informatik-Fachberichte 210. Springer-Verlag, 1989.

[EK76] Hartmut Ehrig and Hans-Jörg Kreowski. Parallelism of manipulations in multidimensional information structures. In *Proc. Mathematical Foundations of Computer Science*, pages 284–293. Springer Lecture Notes in Computer Science 45, 1976.

[FW91] William M. Farmer and Ronald J. Watro. Redex capturing in term graph rewriting. In *Proc. Rewriting Techniques and Applications*, pages 13–24. Springer Lecture Notes in Computer Science 488, 1991.

[HO80] Gérard Huet and Derek C. Oppen. Equations and rewrite rules, a survey. In Ronald V. Book, editor, *Formal Language Theory: Perspectives and Open Problems*, pages 349–405. Academic Press, 1980.

[HP91] Berthold Hoffmann and Detlef Plump. Implementing term rewriting by jungle evaluation. *RAIRO Theoretical Informatics and Applications*, 25(5):445–472, 1991.

[Hue80] Gérard Huet. Confluent reductions: Abstract properties and applications to term rewriting systems. *Journal of the ACM*, 27(4):797–821, 1980.

[Klo90] Jan Willem Klop. Term rewriting systems — from Church-Rosser to Knuth-Bendix and beyond. In *Proc. Automata, Languages, and Programming*, pages 350–369. Springer Lecture Notes in Computer Science 443, 1990.

[Mid90] Aart Middeldorp. Modular properties of term rewriting systems. Dissertation, Vrije Universiteit, Amsterdam, 1990.

[O'D77] Michael J. O'Donnell. *Computing in Systems Described by Equations*. Lecture Notes in Computer Science 58. Springer-Verlag, 1977.

[Plu91] Detlef Plump. Implementing term rewriting by graph reduction: Termination of combined systems. In *Proc. Conditional and Typed Rewriting Systems*, pages 307–317. Springer Lecture Notes in Computer Science 516, 1991.

[Toy87] Yoshihito Toyama. Counterexamples to termination for the direct sum of term rewriting systems. *Information Processing Letters*, 25:141–143, 1987.