Centrum voor Wiskunde en Informatica

REPORT*RAPPORT*

Bisimilarity in Term Graph Rewriting

Z.M. Ariola, J.W. Klop, D. Plump

Software Engineering (SEN)

**SEN-R9801 January 31, 1998**

Report SEN-R9801
ISSN 1386-369X

CWI
P.O. Box 94079
1090 GB Amsterdam
The Netherlands

# Bisimilarity in Term Graph Rewriting

Zena M. Ariola
*Computer and Information Science Department*
*University of Oregon, Eugene, OR 97401, USA*
ariola@cs.uoregon.edu

Jan Willem Klop
*CWI, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands*
jwk@cwi.nl

Detlef Plump
*Fachbereich Mathematik und Informatik*
*Universität Bremen, Postfach 33 04 40, 28334 Bremen, Germany*
det@informatik.uni-bremen.de

ABSTRACT

We present a survey of confluence properties of (acyclic) term graph rewriting. Results and counterexamples are given for different kinds of term graph rewriting: besides plain applications of rewrite rules, extensions with the operations of collapsing and copying, and with both operations together are considered. Collapsing and copying together constitute bisimilarity of term graphs. We establish sufficient conditions for—and counterexamples to—confluence, confluence modulo bisimilarity and the Church-Rosser property modulo bisimilarity. Moreover, we address rewriting modulo bisimilarity, that is, rewriting of bisimilarity classes of term graphs.

*1991 Computing Reviews Classification System:* F.1.1, F.4.1, F.4.2
*Keywords and Phrases:* term graph rewriting, bisimilarity, confluence, Church-Rosser property
*Note:* Work carried out under project SEN2.2, Data Manipulation. Part of the research of the third author was performed while he was on leave at CWI by a grant of the HCM network EXPRESS. This paper is a revised and extended version of [AKP97].

## 1. Introduction

Computations with term rewrite rules play an important role in areas like functional programming, symbolic computation and theorem proving. Such computations are commonly implemented on graph-like data structures for expressions. This makes it possible to *share* common subexpressions, thereby avoiding repeated evaluations of the same subexpression.

Term graph rewriting originates from the demand for a computational model that allows to reason about implementations with sharing. In this model, rewrite rules operate on graphs rather than on trees. Although term graph rewriting is closely related to term rewriting, the two models differ with respect to important properties like termination and confluence.

In this paper, we consider acyclic term graph rewriting according to the approach of [Plu93a, Plu93b]. The definition of rewrite steps in this setting is—as far as acyclic term graphs are concerned—equivalent to the corresponding definitions in [BvEG+87, KKSdV94, AK96]. We remark, however, that this equivalence fails for cyclic graphs. In particular, a "collapsing" term rewrite rule like $\mathtt{id(x)} \to \mathtt{x}$ yields, when applied to certain cyclic graphs, different results in the mentioned approaches (see [KKSdV94] and [CD97]).

We are mainly interested, in this paper, in confluence properties of term graph rewriting. We will address not only rewriting by applications of term rewrite rules, but also extensions with the operations of collapsing and copying, and with both operations together. These operations are important for completeness reasons: while collapsing allows to cope with term rewrite rules having repeated variables in their left-hand sides, copying allows to simulate certain term rewriting derivations that are otherwise prevented by sharing. Moreover, collapsing increases the degree of sharing and thus can, in certain cases, considerably speed up evaluation processes.

When collapsing and copying are present together, the (reflexive-transitive closure of the) term graph rewrite relation contains *bisimilarity* of term graphs. We call two term graphs bisimilar if they represent the same term. Equivalently, both graphs collapse to a common term graph or yield a common term graph by copying. We investigate, in addition to confluence, under which conditions term graph rewriting is confluent modulo bisimilarity or even Church-Rosser modulo bisimilarity. Moreover, we characterize confluence and termination of term graph rewriting modulo bisimilarity, that is, of rewriting bisimilarity classes.

The rest of this paper is organized as follows. In Section 2, we introduce term graphs, collapsing, copying and bisimilarity. Section 3 contains a review of term graph rewriting and motivates the use of collapsing and copying. The relation between confluence, confluence modulo bisimilarity and the Church-Rosser property modulo bisimilarity is clarified in Section 4. In Section 5, we recall some confluence results for non-overlapping rewrite rules and show that the full substitution strategy is cofinal. Examples demonstrate that the addition of collapsing or copying causes non-confluence. Orthogonal rewrite systems are treated in Section 6. It is shown that collapsing may still result in non-confluence, while plain term graph rewriting is shown to be confluent modulo bisimilarity. Section 7 is devoted to general systems with possibly overlapping rules. We present conditions under which confluence of term rewriting induces confluence of term graph rewriting, or even the Church-Rosser property modulo bisimilarity. In Section 8, rewriting of bisimilarity classes is addressed. Finally, in Section 9, we summarize our positive and negative results in two tables.

## 2. Term Graphs and Bisimilarity

Let $\Sigma$ be a set of *function symbols* where each $f \in \Sigma$ comes with a natural number $\mathrm{arity}(f) \geq 0$. Function symbols of arity 0 are called *constants*. We further assume that there is an infinite set X of *variables* such that $\mathrm{X} \cap \Sigma = \emptyset$, and we set $\mathrm{arity}(x) = 0$ for each variable $x$.

A *hypergraph* over $\Sigma \cup \mathrm{X}$ is a system $G = \langle \mathrm{V}_G, \mathrm{E}_G, \mathrm{lab}_G, \mathrm{att}_G \rangle$ consisting of two finite sets $\mathrm{V}_G$ and $\mathrm{E}_G$ of *nodes* and *hyperedges*, a labelling function $\mathrm{lab}_G \colon \mathrm{E}_G \to \Sigma \cup \mathrm{X}$, and an attachment function $\mathrm{att}_G \colon \mathrm{E}_G \to \mathrm{V}_G^*$ which assigns a string of nodes to a hyperedge $e$ such that the length of $\mathrm{att}_G(e)$ is $1 + \mathrm{arity}(\mathrm{lab}_G(e))$. In the following, we call hypergraphs and hyperedges simply graphs and edges.

Given a graph $G$ and an edge $e$ with $\mathrm{att}_G(e) = v\, v_1 \ldots v_n$, node $v$ is the *result node* of $e$ while $v_1, \ldots, v_n$ are the *argument nodes*. The result node $v$ is denoted by $\mathrm{res}(e)$. For each node $v$, $G[v]$ is the subgraph consisting of all nodes that are reachable from $v$ and all edges having these nodes as result nodes.

**Definition 2.1 (Term graph)** A graph $G$ is a *term graph* if

(1) there is a node $\mathrm{root}_G$ from which each node is reachable,

(2) $G$ is acyclic, and

(3) each node is the result node of a unique edge.

Figure 1 shows three term graphs with binary function symbols `f`, `g` and `h`, and a constant `a`. Edges are depicted as boxes with inscribed labels, and bullets represent nodes. A line connects each edge with its result node, while arrows point to the argument nodes. The order in the argument string is given by the left-to-right order of the arrows leaving the box.

Instead of using hypergraphs, term graphs can alternatively be defined as directed acyclic graphs consisting of a set of labelled nodes together with a successor function from nodes to tuples of nodes (see for example [BvEG+87, KKSdV94]). This kind of definition is equivalent to the present one since every term graph defined in that way can be easily transformed into a hypergraph conforming to Definition 2.1, and vice versa. In this paper, we use the hypergraph framework in order to be consistent with [Plu93a, Plu93b].

A *term* over $\Sigma \cup \mathrm{X}$ is a variable, a constant, or a string $f(t_1, \ldots, t_n)$ where $f$ is a function symbol of arity $n \geq 1$ and $t_1, \ldots, t_n$ are terms.

**Definition 2.2 (Term representation)** Let $v$ be a node in a term graph $G$, $e$ be the unique edge with result node $v$, and $\mathrm{att}_G(e) = v\, v_1 \ldots v_n$. Then

$$\mathrm{term}_G(v) = \begin{cases} \mathrm{lab}_G(e) & \text{if } n = 0, \\ \mathrm{lab}_G(e)(\mathrm{term}_G(v_1), \ldots, \mathrm{term}_G(v_n)) & \text{otherwise} \end{cases}$$

is the term represented by $v$. We write $\mathrm{term}(G)$ for $\mathrm{term}_G(\mathrm{root}_G)$.

A *graph morphism* $f\colon G \to H$ between two graphs $G$ and $H$ consists of two functions $f_\mathrm{V}\colon \mathrm{V}_G \to \mathrm{V}_H$ and $f_\mathrm{E}\colon \mathrm{E}_G \to \mathrm{E}_H$ that preserve labels and attachment to nodes, that is, $\mathrm{lab}_H \circ f_\mathrm{E} = \mathrm{lab}_G$ and $\mathrm{att}_H \circ f_\mathrm{E} = f_\mathrm{V}^* \circ \mathrm{att}_G$ (where $f_\mathrm{V}^*\colon \mathrm{V}_G^* \to \mathrm{V}_H^*$ maps a string $v_1 \ldots v_n$ to $f_\mathrm{V}(v_1) \ldots f_\mathrm{V}(v_n)$). The morphism $f$ is *injective* (*surjective*) if $f_\mathrm{V}$ and $f_\mathrm{E}$ are. If $f$ is injective and surjective, then it is an *isomorphism*. In this case $G$ and $H$ are *isomorphic*, which is denoted by $G \cong H$.

**Definition 2.3 (Collapsing and copying)** Given two term graphs $G$ and $H$, $G$ *collapses* to $H$ if there is a graph morphism $G \to H$ mapping $\mathrm{root}_G$ to $\mathrm{root}_H$. This is denoted by $G \succeq H$ or, if the morphism is non-injective, by $G \succ H$. The latter kind of collapsing is said to be *proper*. The inverse relation of collapsing is called *copying* and is denoted by $\preceq$. Proper copying, denoted by $\prec$, is the inverse relation of proper collapsing.

Two examples of collapsing and copying are given in Figure 1. It is easy to see that the collapse morphisms are the surjective graph morphisms between term graphs, and that the following fact holds.

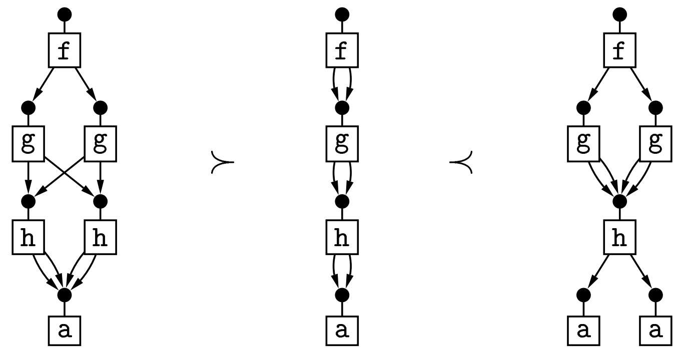


Figure 1: Collapsing and copying

**Fact 2.4** *For all term graphs $G$ and $H$, $G \succeq H$ implies* $\mathrm{term}(G) = \mathrm{term}(H)$.

In the following, we will frequently use term graphs with minimal or maximal sharing.

**Definition 2.5 (Tree and fully collapsed term graph)** A term graph $G$ is a *tree* if there is no $H$ with $H \succ G$, while $G$ is *fully collapsed* if there is no $H$ with $G \succ H$.

For example, the middle graph in Figure 1 is fully collapsed. The following is shown in [Plu93b].

**Lemma 2.6** *For every term graph $G$, there is a tree $\triangle G$ and a fully collapsed term graph $\nabla G$ such that*

$$\triangle G \succeq G \succeq \nabla G.$$

*Moreover, $\triangle G$ and $\nabla G$ are unique up to isomorphism.*

**Definition 2.7 (Bisimilarity)** Two term graphs $G$ and $H$ are *bisimilar*, denoted by $G \sim H$, if $\mathrm{term}(G) = \mathrm{term}(H)$.

The three graphs in Figure 1, for instance, are bisimilar. Note that the two outer graphs are neither related by collapsing nor by copying.

Originally, the notion of bisimilarity and bisimulation was formulated in the theory of concurrent or communicating systems, also called process algebra. As it turned out, the notion applies directly and elegantly to term graphs, in order to give an equivalent formulation of "tree equivalence", that is, identity of the possibly infinite trees arising after unwinding a (possibly cyclic) term graph (see [AK96]). Bisimilarity and bisimulations are in the term graph setting much simpler than in process algebra, because the sum operator (+) for processes is idempotent, associative, and commutative, so the objects are (edge-labelled) trees where the order of subtrees is irrelevant, other than in the case of term graphs. Our present setting of acyclic term graphs is even more simple, and enables us to define bisimilarity directly without mentioning the notion of bisimulation.

The uniqueness up to isomorphism of trees and fully collapsed term graphs characterizes bisimilarity as follows.

**Lemma 2.8** *For all term graphs $G$ and $H$, the following are equivalent:*

*(1)* $G \sim H$.

*(2)* $\triangle G \cong \triangle H$.

*(3)* $\nabla G \cong \nabla H$.

If we consider term graphs as "abstract graphs", i.e. as isomorphism classes of graphs, then $\succeq$ is a partial order and the equivalence class of a term graph $G$ with respect to $\sim$ (the "bisimilarity class" of $G$) is in fact a complete lattice with top element $\triangle G$ and bottom element $\nabla G$. This is shown in [AK96] (in a different technical framework) for possibly cyclic term graphs.

## 3. Term Graph Rewriting

In this section, we review how term graphs are transformed by applications of term rewrite rules, and we motivate the use of collapsing and copying in term graph rewriting.

A *term rewrite rule* $l \to r$ consists of two terms $l$ and $r$ over $\Sigma \cup \mathrm{X}$ such that $l$ is not a variable and all variables in $r$ occur also in $l$. A set $\mathcal{R}$ of term rewrite rules is a *term rewriting system*. We assume that the reader is familiar with basic concepts of term rewriting (see [DJ90, Klo92] for overviews). For the following we fix an arbitrary term rewiting system $\mathcal{R}$. The term rewrite relation associated with $\mathcal{R}$ is denoted by $\to$, its transitive closure by $\to^+$, and its reflexive-transitive closure by $\to^*$.

Given a term $t$, we write $\triangle t$ for the tree representing $t$. Moreover, $\Diamond t$ denotes the term graph representing $t$ such that only variables are shared, that is, each node $v$ with an indegree greater than one satisfies $\mathrm{term}_{\Diamond t}(v) \in \mathrm{X}$, and for each variable $x$ in $t$ there is a unique node $v$ with $\mathrm{term}_{\Diamond t}(v) = x$. The graph resulting from $\Diamond t$ after removing all edges labelled with variables is denoted by $\underline{\Diamond t}$. As an example, Figure 2 shows the graphs $\triangle\mathtt{f(x,x)}$, $\Diamond\mathtt{f(x,x)}$ and $\underline{\Diamond\mathtt{f(x,x)}}$.

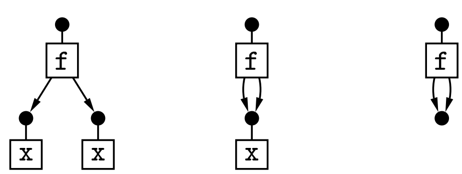


Figure 2: The graphs $\triangle\mathtt{f(x,x)}$, $\Diamond\mathtt{f(x,x)}$ and $\underline{\Diamond\mathtt{f(x,x)}}$

**Definition 3.1 (Instance and redex)** A term graph $T$ is an *instance* of a term $t$ if there is graph morphism $\underline{\Diamond t} \to T$ sending $\mathrm{root}_{\Diamond t}$ to $\mathrm{root}_T$. Given a node $v$ in a term graph $G$ and a rule $l \to r$ in $\mathcal{R}$, the pair $\langle v,\, l \to r\rangle$ is a *redex* if $G[v]$ is an instance of $l$.

**Definition 3.2 (Term graph rewriting)** Let $G$ and $H$ be term graphs, and $\langle v,\, l \to r\rangle$ be a redex in $G$. Then there is a *proper rewrite step* $G \Rightarrow_{v,\,l\to r} H$ if $H$ is isomorphic to the term graph $G_3$ constructed as follows:

(1) $G_1 = G - \{e\}$ is the graph obtained from $G$ by removing the unique edge $e$ satisfying $\mathrm{res}(e) = v$.

(2) $G_2$ is the graph obtained from the disjoint union $G_1 + \underline{\Diamond r}$ by

- identifying $v$ with $\mathrm{root}_{\Diamond r}$,
- identifying the image of $\mathrm{res}(e_1)$ with $\mathrm{res}(e_2)$, for each pair $\langle e_1, e_2 \rangle \in \mathrm{E}_{\Diamond l} \times \mathrm{E}_{\Diamond r}$ with $\mathrm{lab}_{\Diamond l}(e_1) = \mathrm{lab}_{\Diamond r}(e_2) \in \mathrm{X}$.

(3) $G_3 = G_2[\mathrm{root}_G]$ is the term graph obtained from $G_2$ by removing all nodes and edges not reachable from $\mathrm{root}_G$ ("garbage collection").

We denote such a rewrite step also by $G \Rightarrow_v H$ or simply by $G \Rightarrow H$, and we write $G \Rightarrow^* H$ if there are graphs $G_0, \ldots, G_n$ $(n \geq 0)$ such that $G \cong G_0 \Rightarrow G_1 \Rightarrow \ldots \Rightarrow G_n = H$.

Given a term graph rewrite step $G \Rightarrow H$ and a node $v$ in $G$, $v$ either has a unique image in $H$ or is removed by garbage collection. We use a partial function $\mathrm{tr}_{G \Rightarrow H} \colon \mathrm{V}_G \to \mathrm{V}_H$, the *track function* for $G \Rightarrow H$, to assign to each node in $G$ its corresponding node in $H$.

**Definition 3.3 (Track function)** Let $G \Rightarrow_{v, l \to r} H$ be a proper term graph rewrite step. Let, in the construction of Definition 3.2, $in \colon G_1 \to G_1 + \underline{\Diamond r}$ be the injective graph morphism associated with the disjoint union, $ident \colon G_1 + \underline{\Diamond r} \to G_2$ be the surjective morphism associated with the identification, and $i \colon G_3 \to H$ be the isomorphism between $G_3$ and $H$. Then the *track function* for this rewrite step is the partial function $\mathrm{tr}_{G \Rightarrow H} \colon V_G \to V_H$ defined as follows:

$$\mathrm{tr}_{G \Rightarrow H}(v) = \begin{cases} i(ident(in(v))) & \text{if } ident(in(v)) \in V_{G_3}, \\ \text{undefined} & \text{otherwise.} \end{cases}$$

We extend the track function to rewrite sequences as follows: If $G \Rightarrow^* H$ by an isomorphism $i \colon G \to H$, then $\mathrm{tr}_{G \Rightarrow^* H} = i_V$; if $G \Rightarrow M \Rightarrow^* H$, then $\mathrm{tr}_{G \Rightarrow M \Rightarrow^* H} = \mathrm{tr}_{M \Rightarrow^* H} \circ \mathrm{tr}_{G \Rightarrow M}$.

Term graph rewriting is sound with respect to term rewriting in the following sense.

**Theorem 3.4 (Soundness [BvEG+87, HP88])** *For all term graphs $G$ and $H$,*

$$G \Rightarrow H \text{ implies } term(G) \to^+ term(H).$$

In the sequel we consider not only term graph rewriting by $\Rightarrow$ but also extensions with collapsing and copying. To this end we introduce three extensions of $\Rightarrow$.

**Definition 3.5 ($\Rightarrow_{\mathrm{coll}}$, $\Rightarrow_{\mathrm{copy}}$, $\Rightarrow_{\mathrm{bi}}$)** The relations $\Rightarrow_{\mathrm{coll}}$, $\Rightarrow_{\mathrm{copy}}$ and $\Rightarrow_{\mathrm{bi}}$ on term graphs are defined as follows:

$$\begin{aligned} \Rightarrow_{\mathrm{coll}} &= \Rightarrow \cup \succ, \\ \Rightarrow_{\mathrm{copy}} &= \Rightarrow \cup \prec, \\ \Rightarrow_{\mathrm{bi}} &= \Rightarrow \cup \succ \cup \prec . \end{aligned}$$

We refer to $\Rrightarrow$, $\Rrightarrow_{\text{coll}}$, $\Rrightarrow_{\text{copy}}$ and $\Rrightarrow_{\text{bi}}$ as *plain term graph rewriting*, *term graph rewriting with collapsing*, *term graph rewriting with copying*, and *term graph rewriting with collapsing and copying*, respectively. By a *term graph rewrite relation* we mean any binary relation on term graphs.

Given a term graph rewrite relation $\Rrightarrow$, and term graphs $G$ and $H$, we write $G \Rrightarrow^{\lambda} H$ if $G \Rrightarrow H$ or $G \cong H$. The inverse of $\Rrightarrow$ is denoted by $\Lleftarrow$ and its symmetric closure by $\Lleftarrow\!\!\Rrightarrow$. The relation $\Rrightarrow^*$ is defined analogously to $\Rightarrow^*$.

Note that the relation $\Rrightarrow_{\text{bi}}^*$ contains bisimilarity since $G \sim H$ implies $G \preceq \triangle G \succeq H$ (see Lemma 2.6 and 2.8). Moreover, $\Rrightarrow_{\text{coll}}$, $\Rrightarrow_{\text{copy}}$ and $\Rrightarrow_{\text{bi}}$ are sound in the sense of Theorem 3.4 if we replace $\to^+$ by $\to^*$ (collapse and copy steps do not change the represented term).

Collapsing allows to cope with term rewrite rules having repeated variables in their left-hand sides. For instance, the rule $\mathtt{eq(x,x)} \to \mathtt{true}$ cannot be applied to the tree $\triangle\mathtt{eq(0,0)}$ because there is no graph morphism $\Diamond\mathtt{eq(x,x)} \to \triangle\mathtt{eq(0,0)}$ (see Figure 3). This problem is overcome by first collapsing $\triangle\mathtt{eq(0,0)}$ so that subsequently the rule can be applied.

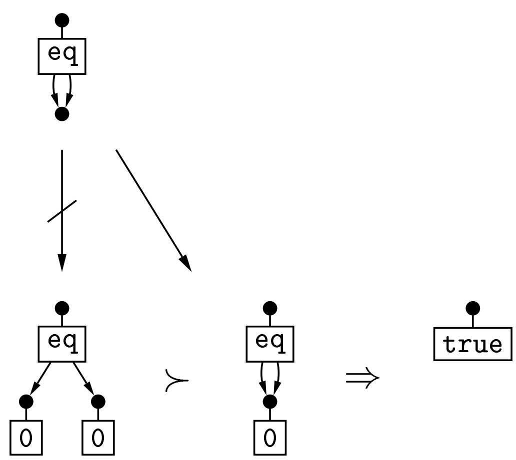


Figure 3: Collapsing to enable a rule application

Another advantage of collapsing is that, in certain cases, it can speed up evaluation processes drastically. A prime example is the specification of the Fibonacci function:

$$\begin{aligned} \mathtt{fib(0)} &\to \mathtt{0} \\ \mathtt{fib(s(0))} &\to \mathtt{s(0)} \\ \mathtt{fib(s(s(x)))} &\to \mathtt{fib(s(x))} + \mathtt{fib(x)} \end{aligned}$$

Using these three rules, evaluating a term of the form $\mathtt{fib(s}^n\mathtt{(0))}$ by term rewriting requires a number of rewrite steps exponential in $n$ (see [AS85]). One easily observes that the same number of steps is needed for plain term graph rewriting. After replacing $\Rightarrow$ by $\Rrightarrow_{\text{coll}}$, however, it is possible to evaluate $\mathtt{fib(s}^n\mathtt{(0))}$ in a linear number of steps. The evaluation strategy can be described as follows: (1) Collapse steps have priority over proper rewrite steps and produce fully collapsed term graphs. (2) Out of two `fib`-redexes, the one representing the greater number is reduced. See Figure 4 for an illustration of this strategy. It is not difficult to verify that, for $n \geq 2$, this procedure evaluates $\texttt{fib}(\texttt{s}^n(\texttt{0}))$ in $2n+1$ steps (viz. $n+1$ proper rewrite steps and $n$ collapse steps).

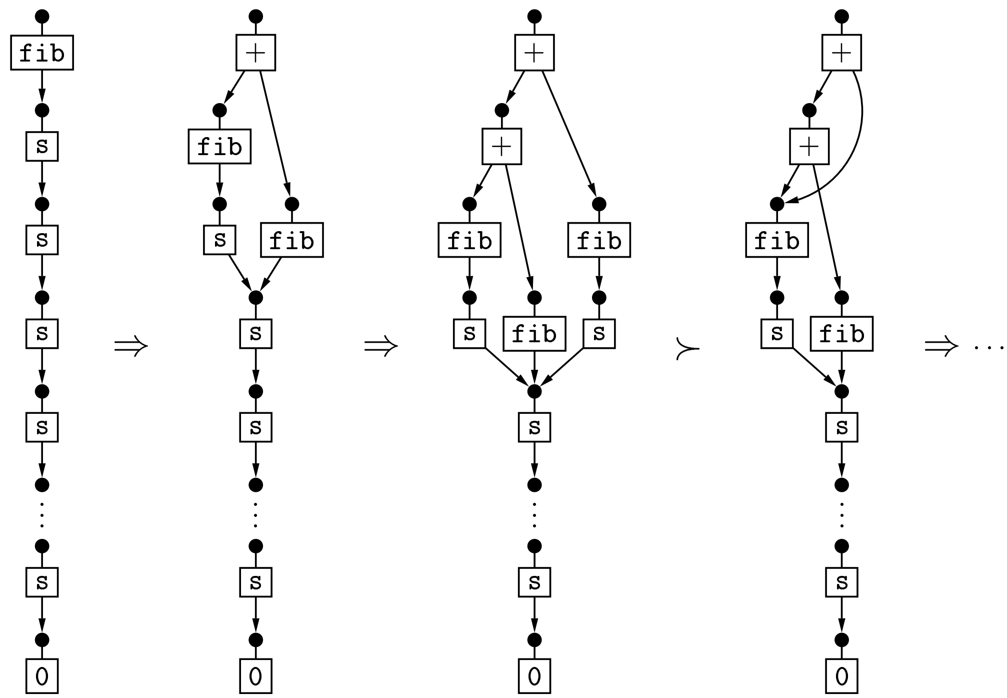


Figure 4: Collapsing to speed up evaluation

The benefit of copying is that it makes term graph rewriting complete with respect to term rewriting: Theorem 7.1 will show that every term rewriting sequence can be simulated if both collapsing and copying are present. Moreover, if there are no repeated variables in the left-hand sides of rules, then copying alone guarantees completeness (see Theorem 7.3).

## 4. Notions of Confluence

In this section we define confluence, confluence modulo bisimilarity and the Church-Rosser property modulo bisimilarity. It turns out that for $\Rightarrow_{\text{coll}}$, $\Rightarrow_{\text{copy}}$ and $\Rightarrow_{\text{bi}}$, these three properties are equivalent. For plain term graph rewriting, however, the Church-Rosser property modulo bisimilarity is strictly stronger than confluence modulo bisimilarity, and confluence is incomparable with the two other properties.

**Definition 4.1 (Confluence properties)** A term graph rewrite relation $\Rrightarrow$ is

(1) *confluent* if for every constellation $G_1 \Lleftarrow^* G \Rrightarrow^* G_2$ there is a term graph $G_3$ such that $G_1 \Rrightarrow^* G_3 \Lleftarrow^* G_2$,

(2) *confluent modulo bisimilarity* if whenever $G_1 \Lleftarrow^* G \sim H \Rrightarrow^* H_1$, there are term graphs $G_2$ and $H_2$ such that $G_1 \Rrightarrow^* G_2 \sim H_2 \Lleftarrow^* H_1$,

(3) *Church-Rosser modulo bisimilarity* if whenever $G \approx H$, there are term graphs $G_1$ and $H_1$ such that $G \Rrightarrow^* G_1 \sim H_1 \Lleftarrow^* H$. Here $\approx$ is the transitive closure of the relation $\Leftrightarrow \cup \sim$.

An important consequence of confluence is that rewriting yields deterministic results. Call a term graph $N$ a *normal form* with respect to $\Rrightarrow$ if there is no $N'$ with $N \Rrightarrow N'$. The relation $\Rrightarrow$ is *weakly normalizing* if for every term graph $G$ there is a normal form $N$ such that $G \Rrightarrow^* N$. *Uniqueness of normal forms* means that whenever $N_1 \Lleftarrow^* G \Rrightarrow^* N_2$ for normal forms $N_1$ and $N_2$, then $N_1 \cong N_2$. While confluence implies uniqueness of normal forms, confluence modulo bisimilarity implies uniqueness of normal forms up to bisimilarity.

From Definition 4.1 it is clear that "Church-Rosser modulo $\sim$" implies "confluent modulo $\sim$". The following lemma is a specialization of a lemma of Huet [Hue80] to the term graph setting.

**Lemma 4.2** *A weakly normalizing term graph rewrite relation is Church-Rosser modulo bisimilarity if and only if it is confluent modulo bisimilarity.*

For plain term graph rewriting, the Church-Rosser property modulo bisimilarity is strictly stronger than confluence modulo bisimilarity. This will become apparent by Example 6.2 in conjunction with Theorem 6.4. (The two outer graphs in Figure 9 are related by $\approx$ but cannot be reduced to bisimilar graphs by $\Rrightarrow$.)

The next two examples show that for plain term graph rewriting, confluence is in general incomparable with both the Church-Rosser property modulo bisimilarity and confluence modulo bisimilarity.

**Example 4.3** Consider the following system[1]:

$$\begin{array}{rcl} \mathtt{a} & \to & \mathtt{b} \\ \mathtt{b} & \to & \mathtt{a} \\ \mathtt{f(a,b)} & \to & \mathtt{c} \end{array}$$

It is easy to check that $\Rrightarrow$ is confluent, but Figure 5 shows that confluence modulo bisimilarity (and hence the Church-Rosser property modulo bisimilarity) fails.

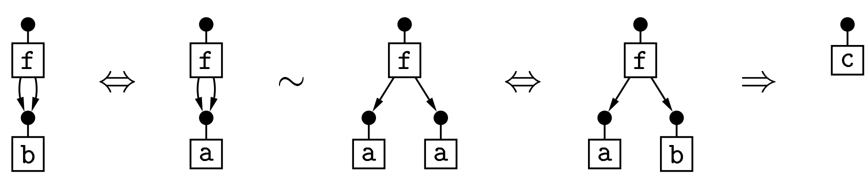


Figure 5: Confluence without confluence modulo $\sim$

[1] This system is used in [KKSdV94] for a different purpose.

**Example 4.4** Plain term graph rewriting may be Church-Rosser modulo bisimilarity (and hence confluent modulo bisimilarity) without being confluent. This is demonstrated by the following system:

$$\begin{array}{lcl} \mathtt{g(x)} & \to & \mathtt{f(x,x)} \\ \mathtt{g(a)} & \to & \mathtt{f(a,a)} \end{array}$$

Theorem 7.7 will show that $\Rightarrow$ is Church-Rosser modulo $\sim$, since $\Rightarrow$ is weakly normalizing and term rewriting is confluent. However, the term graph representing $\mathtt{g(a)}$ has two non-isomorphic normal forms (see Figure 6), hence $\Rightarrow$ is not confluent.

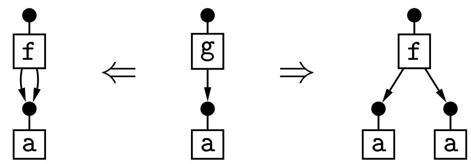


Figure 6: Church-Rosser property modulo $\sim$ without confluence

In the presence of collapsing or copying, the three confluence properties become equivalent. Thus, in particular, these properties are equivalent for the relations $\Rightarrow_{\mathrm{coll}}$, $\Rightarrow_{\mathrm{copy}}$ and $\Rightarrow_{\mathrm{bi}}$.

**Theorem 4.5** *Let $\Rrightarrow$ be a term graph rewrite relation such that $\succ \subseteq \Rrightarrow^*$ or $\prec \subseteq \Rrightarrow^*$. Then the following are equivalent:*

*(1) $\Rrightarrow$ is confluent.*

*(2) $\Rrightarrow$ is confluent modulo bisimilarity.*

*(3) $\Rrightarrow$ is Church-Rosser modulo bisimilarity.*

**Proof.** Suppose that $\succ \subseteq \Rrightarrow^*$; the case $\prec \subseteq \Rrightarrow^*$ is treated analogously. We show the implications (1) $\to$ (3) $\to$ (2) $\to$ (1).

(1) $\to$ (3): Consider term graphs $G$ and $H$ with $G \approx H$. Then there are term graphs $G_0, \ldots, G_n$, $n \geq 1$, such that $G_0 = G$, $G_n = H$ and for $i = 1, \ldots, n$, $G_{i-1} \Lleftarrow\!\!\Rrightarrow G_i$ or $G_{i-1} \sim G_i$. By Lemma 2.6 and 2.8, each constellation $G_{i-1} \sim G_i$ satisfies $G_{i-1} \preceq \Delta G_i \succeq G_i$. Hence $G_{i-1} \Lleftarrow^* \circ \Rrightarrow^* G_i$ for $i = 1, \ldots, n$. By induction on $n$, using confluence, we see that there is a term graph $M$ such that $G = G_0 \Rrightarrow^* M \Lleftarrow^* G_n = H$. Thus, $\Rrightarrow$ is Church-Rosser modulo $\sim$.

(3) $\to$ (2): Immediate consequence of Definition 4.1.

(2) $\to$ (1): Assume that $G_1 \Lleftarrow^* G \Rrightarrow^* G_2$. By confluence modulo $\sim$, there are term graphs $G_3$ and $G_4$ such that $G_1 \Rrightarrow^* G_3 \sim G_4 \Lleftarrow^* G_2$. Then $G_3 \succeq \nabla G_3 \preceq G_4$ and hence $G_1 \Rrightarrow^* \nabla G_3 \Lleftarrow^* G_2$. So $\Rrightarrow$ is confluent. $\Box$

## 5. Non-overlapping Systems

It is known that plain term graph rewriting is confluent if the left-hand sides of the given term rewrite rules do not overlap. After recalling this and a related result about the uniqueness of complete developments, we show that the reduction strategy of full substitution is cofinal. Then counterexamples are given demonstrating that confluence fails as soon as term graph rewriting is extended with copying or collapsing.

**Definition 5.1 (Non-overlapping)** A term $s$ *overlaps* a term $t$ in a subterm $u$ of $t$ if $u$ is not a variable and if there are substitutions[2] $\sigma$ and $\tau$ such that $\sigma(s) = \tau(u)$. The term rewriting system $\mathcal{R}$ is *non-overlapping* if for all rules $l_1 \to r_1$ and $l_2 \to r_2$ in $\mathcal{R}$, $l_1$ overlaps $l_2$ in a subterm $u$ only if $u = l_2$ and $(l_1 \to r_1) = (l_2 \to r_2)$.

**Theorem 5.2** *Let $\mathcal{R}$ be non-overlapping and $G$, $G_1$ and $G_2$ be term graphs such that $G_1 \Leftarrow_{v_1} G \Rightarrow_{v_2} G_2$. Then there is a term graph $G_3$ such that $G_1 \Rightarrow^{\lambda}_{\mathrm{tr}(v_2)} G_3 \Leftarrow^{\lambda}_{\mathrm{tr}(v_1)} G_2$, where $\mathrm{tr}_{G \Rightarrow G_1 \Rightarrow^{\lambda} G_3} = \mathrm{tr}_{G \Rightarrow G_2 \Rightarrow^{\lambda} G_3}$.*

**Proof.** A proof is already given in [Sta80], in a slightly different technical framework. A proof conforming to the present setting can be found in [Plu93b], as part of the proof of the so-called Critical Pair Lemma. □

Call the relation $\Rightarrow$ *subcommutative* if whenever $G_1 \Leftarrow G \Rightarrow G_2$, there is a term graph $G_3$ such that $G_1 \Rightarrow^{\lambda} G_3 \Leftarrow^{\lambda} G_2$. It is well known that subcommutativity implies confluence (for arbitrary binary relations; see [Klo92]).

**Corollary 5.3** *If $\mathcal{R}$ is non-overlapping, then $\Rightarrow$ is subcommutative.*

For the rest of this section we assume that $\mathcal{R}$ is an arbitrary non-overlapping system. The following property of subcommutative relations will be needed in showing that the full substitution strategy is cofinal.

**Corollary 5.4** *For all term graphs $G$, $G_1$ and $G_2$, $G_1 \Leftarrow G \Rightarrow^* G_2$ implies that there is a term graph $G_3$ such that $G_1 \Rightarrow^* G_3 \Leftarrow^{\lambda} G_2$.*

**Proof.** By induction on the length of $G \Rightarrow^* G_2$, using subcommutativity. □

We are going to show that complete developments of sets of redexes yield unique results. This fact allows to define the full substitution strategy. In the next section, the cofinality property of this strategy will be used to prove that $\Rightarrow$ is confluent modulo bisimilarity over orthogonal rewrite systems.

Since $\mathcal{R}$ is non-overlapping, every redex $\langle v,\, l \to r \rangle$ is uniquely determined by the node $v$. Hence, in this section, we treat redexes as nodes.

**Definition 5.5 (Residuals)** Let $\Pi$ be a set of redexes in a term graph $G$. The set $\rho(\Pi)$ of *residuals* of $\Pi$ with respect to a rewrite sequence $G \Rightarrow^* H$ is defined as follows. If $G \Rightarrow^* H$ has length 0, then $\rho(\Pi) = i(\Pi)$ for the unique isomorphism $i\colon G \to H$. If $G \Rightarrow^* H$ has the form $G \Rightarrow_v G' \Rightarrow^* H$, then $\rho(\Pi)$ is the set of residuals of $\mathrm{tr}_{G \Rightarrow G'}(\Pi - \{v\})$ with respect to $G' \Rightarrow^* H$.

[2] A *substitution* $\sigma$ is a mapping on the set of terms over $\Sigma \cup X$ such that $\sigma(c) = c$ for every constant $c$, and $\sigma(f(t_1, \ldots, t_n)) = f(\sigma(t_1), \ldots, \sigma(t_n))$ for every composite term $f(t_1, \ldots, t_n)$.

By the assumption that $\mathcal{R}$ is non-overlapping, the residuals of a redex set are again redexes. Note that this is different in term rewriting: there this property may fail when rules are present that have repeated variables in their left-hand sides.

**Definition 5.6 (Development)** A *development* of a set $\Pi$ of redexes in a term graph $G$ is either a derivation $G \Rightarrow^* H$ of length 0, or a derivation of the form $G \Rightarrow_{v,\, l \to r} G' \Rightarrow^* H$ such that $v \in \Pi$ and such that $G' \Rightarrow^* H$ is a development of the residuals of $\Pi$ in $G'$. The development is *complete* if $\Pi$ has no residuals in $H$.

**Theorem 5.7 (Uniqueness of developments)** [3] *Given a set $\Pi$ of redexes in a term graph $G$, all complete developments of $\Pi$ end (up to isomorphism) in the same term graph.*

**Proof.** Consider two complete developments $G \Rightarrow^* H_1$ and $G \Rightarrow^* H_2$ of $\Pi$. We proceed by induction on the number of redexes in $\Pi$. If $\Pi$ is empty, then $H_1 \cong G \cong H_2$ by the definition of complete development. Otherwise, there are (not necessarily distinct) nodes $v_1$ and $v_2$ in $\Pi$ such that for i=1,2, $G \Rightarrow^* H_i$ has the form $G \Rightarrow_{v_i} G_i \Rightarrow^* H_i$. By the proof of Theorem 5.2, there are steps $G_1 \Rightarrow^{\lambda}_{w_2} G' \Leftarrow^{\lambda}_{w_1} G_2$ such that $w_1$ and $w_2$ are residuals of $v_1$ and $v_2$, respectively, and such that the residuals of $\Pi$ with respect to $G \Rightarrow_{v_1} G_1 \Rightarrow^{\lambda}_{w_2} G'$ and $G \Rightarrow_{v_2} G_2 \Rightarrow^{\lambda}_{w_1} G'$ are the same (see [Plu93b]). Now let $G' \Rightarrow^* H$ be a complete development of the redex set $\Pi' = \mathrm{tr}_{G \Rightarrow G_1 \Rightarrow^{\lambda} G'}(\Pi - \{v_1, v_2\}) = \mathrm{tr}_{G \Rightarrow G_2 \Rightarrow^{\lambda} G'}(\Pi - \{v_1, v_2\})$. Then both $G_1 \Rightarrow^* H_1$ and $G_1 \Rightarrow^{\lambda}_{w_2} G' \Rightarrow^* H$ are complete developments of $\mathrm{tr}_{G \Rightarrow G_1}(\Pi - \{v_1\})$. Hence, by induction hypothesis, $H_1 \cong H$. Analogously one shows $H_2 \cong H$. Thus $H_1 \cong H_2$. □

Given a term graph $G$, we denote by $\mathrm{Cpl}(G)$ a term graph that results from a complete development of all redexes in $G$. The process of repeatedly developing all redexes is called the *full substitution* or *Gross-Knuth* strategy in the context of term rewriting systems (see [Klo92]). We show that this strategy is"cofinal" for term graph rewriting over non-overlapping systems.

**Theorem 5.8 (Cofinality)** *For all term graphs $G$ and $H$, $G \Rightarrow^* H$ implies that there is $n \geq 0$ such that $H \Rightarrow^* \mathrm{Cpl}^n(G)$.*

**Proof.** By induction on the length of $G \Rightarrow^* H$. Suppose that $G \Rightarrow^* H' \Rightarrow H$ for some term graph $H'$. By induction hypothesis, $H' \Rightarrow^* \mathrm{Cpl}^n(G)$ for some $n \geq 0$. Then, by Corollary 5.4, there is a term graph $H''$ such that $H \Rightarrow^* H'' \Leftarrow^{\lambda} \mathrm{Cpl}^n(G)$. Thus, by the definition of complete development, $H'' \Rightarrow^* \mathrm{Cpl}^{n+1}(G)$. It follows $H \Rightarrow^* \mathrm{Cpl}^{n+1}(G)$. □

Confluence of $\Rightarrow$ does no longer hold if collapsing or copying is added, as the following two counterexamples demonstrate. Moreover, the examples show that none of the four relations $\Rightarrow$, $\Rightarrow_{\mathrm{coll}}$, $\Rightarrow_{\mathrm{copy}}$ and $\Rightarrow_{\mathrm{bi}}$ is confluent modulo bisimilarity for non-overlapping systems in general.

**Example 5.9** Consider the following non-overlapping system of Huet [Hue80]:

$$\begin{array}{rcl} \mathtt{f(x,x)} & \to & \mathtt{a} \\ \mathtt{f(x,g(x))} & \to & \mathtt{b} \\ \mathtt{c} & \to & \mathtt{g(c)} \end{array}$$

[3] This result appears in [BvEG+87] without proof.

Figure 7 demonstrates that the tree representing f(c, c) has two non-isomorphic normal forms with respect to $\Rightarrow_{\text{coll}}$, so $\Rightarrow_{\text{coll}}$ and $\Rightarrow_{\text{bi}}$ are neither confluent nor confluent modulo bisimilarity. Note that the left-hand sides of the first two rewrite rules contain the variable x twice.

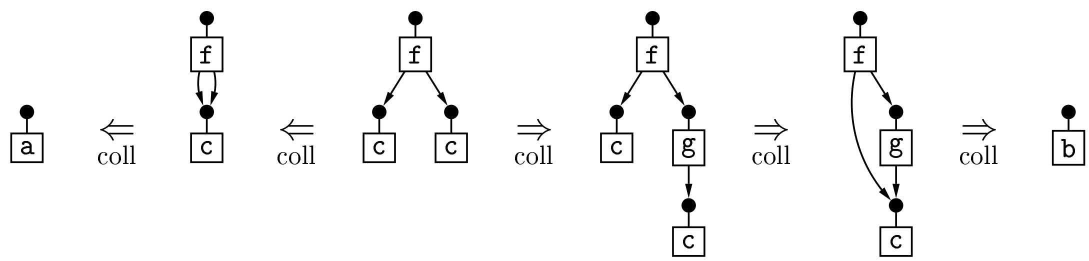


Figure 7: Non-confluence of $\Rightarrow_{\text{coll}}$ and $\Rightarrow_{\text{bi}}$

**Example 5.10** The rule

$$\texttt{f(x,x)} \quad \rightarrow \quad \texttt{a}$$

also contains two occurrences of x in its left-hand side. It shows that, for non-overlapping systems, $\Rightarrow_{\text{copy}}$ need neither be confluent nor confluent modulo bisimilarity. To see this, observe that the graph on the right in Figure 8 is a normal form with respect to $\Rightarrow_{\text{copy}}$.

Figure 8 also demonstrates that plain term graph rewriting is not confluent modulo bisimilarity for non-overlapping systems in general. This is because the rewrite step on the left is proper, and the graphs in the middle and on the right are bisimilar.

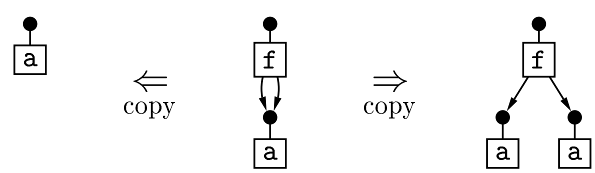


Figure 8: Non-confluence of $\Rightarrow_{\text{copy}}$

## 6. Orthogonal Systems

The counterexamples of the previous section show that for non-overlapping systems, $\Rightarrow$, $\Rightarrow_{\text{coll}}$, $\Rightarrow_{\text{copy}}$ and $\Rightarrow_{\text{bi}}$ need not be confluent modulo bisimilarity, and the last three relations need neither be confluent. In this section and the next it will become clear that this failure is caused, with the exception of $\Rightarrow_{\text{coll}}$, by rewrite rules with repeated variables in their left-hand sides.

**Definition 6.1 (Orthogonal)** The term rewriting system $\mathcal{R}$ is *left-linear* if for each rewrite rule $l \rightarrow r$ in $\mathcal{R}$, no variable occurs more than once in $l$. The system $\mathcal{R}$ is *orthogonal* if it is left-linear and non-overlapping.

The main result of this section is that for orthogonal systems, plain term graph rewriting is confluent modulo bisimilarity. As far as confluence is concerned, we know from the previous section that $\Rightarrow$ is confluent for non-overlapping systems and hence, in particular, for orthogonal systems. In the next section it is shown that $\Rightarrow_{\text{copy}}$ and $\Rightarrow_{\text{bi}}$ are confluent for classes of systems that properly include all orthogonal systems. In contrast, term graph rewriting with collapsing need not be confluent even for orthogonal systems.

**Example 6.2** Consider the single rule

$$\mathtt{c} \;\rightarrow\; \mathtt{g(c)}$$

and suppose that $\Sigma$ contains a binary function symbol $\mathtt{f}$. Figure 9 shows two $\Rightarrow_{\text{coll}}$-derivations starting from $\triangle\mathtt{f(c,c)}$ such that the resulting graphs do not have a common reduct under $\Rightarrow_{\text{coll}}$: the graphs derivable on the left represent the terms $\mathtt{f}(\mathtt{g}^n(\mathtt{c}), \mathtt{g}^n(\mathtt{c}))$, $n \geq 1$, while the graphs derivable on the right represent $\mathtt{f}(\mathtt{g}^n(\mathtt{c}), \mathtt{g}^{n+1}(\mathtt{c}))$, $n \geq 0$. Thus $\Rightarrow_{\text{coll}}$ is non-confluent. Notice also that $\Rightarrow$ is not Church-Rosser modulo $\sim$.

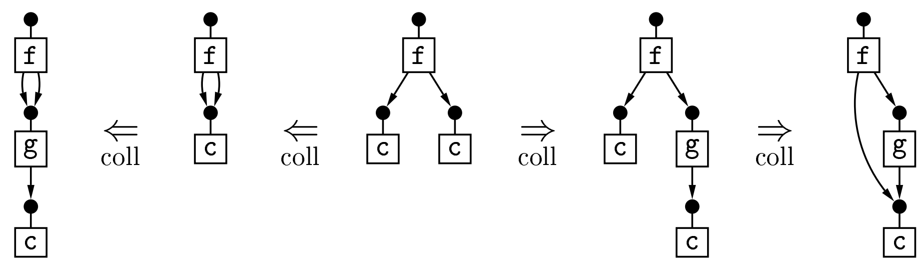


Figure 9: Non-confluence of $\Rightarrow_{\text{coll}}$

**Lemma 6.3** *Let $\mathcal{R}$ be orthogonal. Then for all term graphs $G$ and $H$, $G \sim H$ implies* $\text{Cpl}(G) \sim \text{Cpl}(H)$*.*

**Proof.** Given a complete development $T \Rightarrow^* \text{Cpl}(T)$ of all redexes in a term graph $T$, there is a corresponding complete development $\text{term}(T) \rightarrow^* \text{term}(\text{Cpl}(T))$ of all redexes in $\text{term}(T)$ (see [KKSdV94] for a proof in a slightly different technical setting). Since for orthogonal term rewriting systems, all complete developments of a set of redexes yield the same result [HL91], $\text{term}(G) = \text{term}(H)$ implies $\text{term}(\text{Cpl}(G)) = \text{term}(\text{Cpl}(H))$. □

It is worth mentioning that this lemma does not hold for non-overlapping systems. A counterexample is again $\mathcal{R} = \{\mathtt{f(x,x)} \rightarrow \mathtt{a}\}$: in Figure 8, the graph in the middle is bisimilar to the graph on the right which is a normal form with respect to $\Rightarrow$.

**Theorem 6.4** *If $\mathcal{R}$ is orthogonal, then $\Rightarrow$ is confluent modulo bisimilarity.*

**Proof.** Suppose that $G_1 \Leftarrow^* G \sim H \Rightarrow^* H_1$. By Theorem 5.8, there are $m, n \geq 0$ such that $G_1 \Rightarrow^* \text{Cpl}^m(G)$ and $H_1 \Rightarrow^* \text{Cpl}^n(H)$. Hence, choosing $p = max(m,n)$, we obtain $G_1 \Rightarrow^* \text{Cpl}^p(G)$ and $H_1 \Rightarrow^* \text{Cpl}^p(H)$. Now $\text{Cpl}^p(G) \sim \text{Cpl}^p(H)$ follows from Lemma 6.3. □

**Corollary 6.5** *If $\mathcal{R}$ is orthogonal and $\Rightarrow$ weakly normalizing, then $\Rightarrow$ is Church-Rosser modulo bisimilarity.*

**Proof.** Combine Theorem 6.4 and Lemma 4.2. □

Corollary 6.5 will be generalized by Theorem 7.7 where orthogonality is weakened to left-linearity in conjunction with confluence of $\to$.

## 7. General Systems

In this section we drop the assumption of the two previous sections that $\mathcal{R}$ is non-overlapping. Instead, we infer confluence of $\Rightarrow_{\text{bi}}$, $\Rightarrow_{\text{copy}}$ and $\Rightarrow_{\text{coll}}$ from confluence of term rewriting. In the case of $\Rightarrow_{\text{copy}}$ and $\Rightarrow_{\text{coll}}$, this requires suitable further conditions. Finally, we give sufficient conditions under which confluence of term rewriting makes $\Rightarrow$ Church-Rosser modulo bisimilarity.

We first show that term graph rewriting with collapsing and copying can simulate term rewriting, following the proof of the so-called Completeness Theorem in [Plu93b].

**Theorem 7.1** *For all term graphs $G$ and $H$:*

$$G \Rightarrow_{\text{bi}}^{*} H \text{ if and only if } term(G) \to^{*} term(H).$$

**Proof.** "Only if": By soundness of $\Rightarrow_{\text{bi}}$.
"If": Suppose that for every term rewrite step $t \to u$ there are term graphs $T$ and $U$ such that

$$\Delta t \succeq T \Rightarrow U \preceq \Delta u. \tag{7.1}$$

Then $term(G) \to^{*} term(H)$ implies $\Delta term(G) \Rightarrow_{\text{bi}}^{*} \Delta term(H)$, and with Lemma 2.6 follows $G \preceq \Delta term(G) \Rightarrow_{\text{bi}}^{*} \Delta term(H) \succeq H$. To show (7.1), let $l \to r$ be the rule applied in $t \to u$ and $\pi$ be the associated redex position in $t$. Let $v$ be the unique node in $\Delta t$ specified by $\pi$. Then there is a collapsing $\Delta t \succeq T$ such that $T[v']$ is fully collapsed, where $v'$ is the image of $v$ in $T$, and such that each node of $T$ not belonging to $T[v']$ has an indegree of at most one. By the structure of $T$, there is a step $T \Rightarrow_{v', l \to r} U$ such that $\text{term}(U) = u$. (Since $T[v']$ is fully collapsed, $l \to r$ is applicable at $v'$ even if $l$ contains repeated variables, and as there is a unique path from $\text{root}_T$ to $v'$, $T \Rightarrow_{v', l \to r} U$ simulates $t \to u$.) Hence $U \preceq \Delta u$. □

One should be aware that the generality of the relation $\Rightarrow_{\text{bi}}$ has to be payed with termination and efficiency problems. In particular, $\Rightarrow_{\text{bi}}$ is non-terminating[4] for every term graph representing a term containing two or more occurrences of some subterm. This is because such a graph admits an infinite sequence of alternating collapse and copy steps. In contrast, $\Rightarrow$, $\Rightarrow_{\text{coll}}$ and $\Rightarrow_{\text{copy}}$ are terminating whenever the term rewrite relation $\to$ is terminating. Apart from non-termination, the search space for computing a term normal form by $\Rightarrow_{\text{bi}}$ may be much larger than for $\Rightarrow$ or $\Rightarrow_{\text{coll}}$. (See [Plu91, Rao95] for conditions under which $\Rightarrow_{\text{coll}}$ suffices to compute term normal forms.)

[4] A binary relation $\to$ on a set $A$ is *terminating* (or *strongly normalizing*) if there does not exist an infinite sequence $a_1 \to a_2 \to \ldots$

**Corollary 7.2** *The relation $\Rightarrow_{\mathrm{bi}}$ is confluent if and only if $\to$ is confluent.*

**Proof.** "Only if": Suppose that $t_1 \leftarrow^* t \to^* t_2$ for some terms $t$, $t_1$ and $t_2$. Then $\Delta t_1 \Leftarrow^*_{\mathrm{bi}} \Delta t \Rightarrow^*_{\mathrm{bi}} \Delta t_2$ by Theorem 7.1. Since $\Rightarrow_{\mathrm{bi}}$ is confluent, there is a term graph $G$ such that $\Delta t_1 \Rightarrow^*_{\mathrm{bi}} G \Leftarrow^*_{\mathrm{bi}} \Delta t_2$. Hence $t_1 \to^* \mathrm{term}(G) \leftarrow^* t_2$ by Theorem 7.1.
"If": Given derivations $G_1 \Leftarrow^*_{\mathrm{bi}} G \Rightarrow^*_{\mathrm{bi}} G_2$, Theorem 7.1 yields $\mathrm{term}(G_1) \leftarrow^* \mathrm{term}(G) \to^* \mathrm{term}(G_2)$. Then, since $\to$ is confluent, there is a term $t$ such that $\mathrm{term}(G_1) \to^* t \leftarrow^* \mathrm{term}(G_2)$. With Theorem 7.1 follows $G_1 \Rightarrow^*_{\mathrm{bi}} \Delta t \Leftarrow^*_{\mathrm{bi}} G_2$, since $\mathrm{term}(\Delta t) = t$. □

In order to simulate term rewriting by $\Rightarrow_{\mathrm{copy}}$, the underlying system $\mathcal{R}$ has to be left-linear.

**Theorem 7.3** *If $\mathcal{R}$ is left-linear, then for all terms $t$ and $u$:*

$$\Delta t \Rightarrow^*_{\mathrm{copy}} \Delta u \text{ if and only if } t \to^* u.$$

**Proof.** "Only if": Immediate consequence of soundness of $\Rightarrow_{\mathrm{copy}}$.
"If": It suffices to show that for every term rewrite step $t \to u$ there is a term graph $U$ such that

$$\Delta t \Rightarrow U \preceq \Delta u.$$

Let $l \to r$ be the rule applied in $t \to u$. Then, since $\mathcal{R}$ is left-linear, $\Delta t \Rightarrow_{v, l \to r} U$ for some term graph $U$, where $v$ is the node corresponding to the redex position in $t$. As there is no sharing in $\Delta t$, we have $\mathrm{term}(U) = u$. Thus $U \preceq \Delta u$. □

**Corollary 7.4** *If $\mathcal{R}$ is left-linear, then $\Rightarrow_{\mathrm{copy}}$ is confluent if and only if $\to$ is confluent.*

**Proof.** "Only if": Easy consequence of Theorem 7.3 and soundness of $\Rightarrow_{\mathrm{copy}}$.
"If": Consider derivations $G_1 \Leftarrow^*_{\mathrm{copy}} G \Rightarrow^*_{\mathrm{copy}} G_2$. Then, by soundness of $\Rightarrow_{\mathrm{copy}}$, $\mathrm{term}(G_1) \leftarrow^* \mathrm{term}(G) \to^* \mathrm{term}(G_2)$. Confluence of $\to$ implies that there is a term $t$ such that $\mathrm{term}(G_1) \to^* t \leftarrow^* \mathrm{term}(G_2)$. With Theorem 7.3 follows $\Delta\mathrm{term}(G_1) \Rightarrow^*_{\mathrm{copy}} \Delta t \Leftarrow^*_{\mathrm{copy}} \Delta\mathrm{term}(G_2)$. Hence, using Lemma 2.6,

$$G_1 \preceq \Delta\mathrm{term}(G_1) \Rightarrow^*_{\mathrm{copy}} \Delta t \Leftarrow^*_{\mathrm{copy}} \Delta\mathrm{term}(G_2) \succeq G_2.$$

□

Corollary 7.4 implies that $\Rightarrow_{\mathrm{copy}}$ is confluent, in particular, for orthogonal systems. For it is well known that orthogonality implies confluence of term rewriting (see for example [DJ90, Klo92]).

An analogue to Corollary 7.4 for the case of $\Rightarrow_{\mathrm{coll}}$ can be obtained by replacing the condition of left-linearity with weak normalization of $\Rightarrow_{\mathrm{coll}}$.

**Theorem 7.5 ([Plu93a])** *If $\Rightarrow_{\mathrm{coll}}$ is weakly normalizing, then $\Rightarrow_{\mathrm{coll}}$ is confluent if and only if $\to$ is confluent.*

In general, weak normalization of $\Rightarrow_{\mathrm{coll}}$ neither implies nor follows from weak normalization of $\Rightarrow$. If all rules are left-linear, however, the two properties are equivalent.

**Lemma 7.6** *If $\mathcal{R}$ is left-linear, then $\Rightarrow_{\mathrm{coll}}$ is weakly normalizing if and only if $\Rightarrow$ is weakly normalizing.*

**Proof.** "If": Left-linearity implies that for every normal form $N$ with respect to $\Rightarrow$, $\nabla N$ is a normal form with respect to $\Rightarrow_{\mathrm{coll}}$.

"Only if": In [HP96] it is shown that every derivation $G \Rightarrow^*_{\mathrm{coll}} H$ can be transformed into a "minimally collapsing" derivation $G \Rightarrow^*_{\mathrm{coll}} H'$ such that $H' \succeq H$. If $\mathcal{R}$ is left-linear, this implies $G \Rightarrow^* H'$. Moreover, if $H$ is a normal form with respect to $\Rightarrow_{\mathrm{coll}}$, then $\mathrm{term}(H') = \mathrm{term}(H)$ is a normal form with respect to $\rightarrow$. Hence $H'$ is a normal form with respect to $\Rightarrow$. □

We conclude this section by giving conditions under which confluence of $\rightarrow$ guarantees that $\Rightarrow$ is Church-Rosser modulo bisimilarity. It turns out that both left-linearity of $\mathcal{R}$ and weak normalization of $\Rightarrow$ are needed.

**Theorem 7.7** *If $\mathcal{R}$ is left-linear, $\rightarrow$ confluent, and $\Rightarrow$ weakly normalizing, then $\Rightarrow$ is Church-Rosser modulo bisimilarity.*

**Proof.** By Lemma 4.2, it suffices to show that $\Rightarrow$ is confluent modulo bisimilarity. Given a constellation $G_1 \Leftarrow^* G \sim H \Rightarrow^* H_1$, consider normal forms $G_2$ and $H_2$ of $G_1$ and $H_1$, respectively. Then $\mathrm{term}(G_2) \leftarrow^* \mathrm{term}(G) = \mathrm{term}(H) \rightarrow^* \mathrm{term}(H_2)$ by soundness of $\Rightarrow$. Since $G_2$ and $H_2$ are normal forms and $\mathcal{R}$ is left-linear, $\mathrm{term}(G_2)$ and $\mathrm{term}(H_2)$ are normal forms with respect to $\rightarrow$. (Left-linearity implies that $\Diamond l$ is a tree for every term rewrite rule $l \rightarrow r$; hence, given a term graph $T$, there is a graph morphism $\underline{\Diamond l} \rightarrow T$ if and only if $l \rightarrow r$ is applicable to $\mathrm{term}(T)$.) Now confluence of $\rightarrow$ yields $\mathrm{term}(G_2) = \mathrm{term}(H_2)$, thus $G_2 \sim H_2$. □

The premise of Theorem 7.7 cannot be relaxed by dropping left-linearity or weak normalization, as is witnessed by Example 5.10 and 4.3, respectively. In these examples, $\Rightarrow$ is not even confluent modulo bisimilarity. Moreover, weak normalization of $\Rightarrow$ cannot be replaced by weak normalization of $\rightarrow$. We demonstrate this by a counterexample from [Plu93a].

**Example 7.8** Suppose that $\mathcal{R}$ consists of the following rules:

$$
\begin{array}{rcl}
\mathtt{f(x)} & \rightarrow & \mathtt{g(x,x)} \\
\mathtt{a} & \rightarrow & \mathtt{b} \\
\mathtt{g(a,b)} & \rightarrow & \mathtt{c} \\
\mathtt{g(b,b)} & \rightarrow & \mathtt{f(a)}
\end{array}
$$

Using structural induction on terms, it is easy to verify that every term has a unique normal form. Hence $\rightarrow$ is weakly normalizing and confluent. But Figure 10 shows that $\Rightarrow$ is neither confluent nor confluent modulo bisimilarity. (Notice that there is no graph rewrite step $\nabla\mathtt{g(a,a)} \Rightarrow \Delta\mathtt{g(a,b)}$ corresponding to the term rewrite step $\mathtt{g(a,a)} \rightarrow \mathtt{g(a,b)}$.)

We finally remark that the assumptions of Theorem 7.7 do not guarantee that $\Rightarrow$ is confluent. This can be seen from Example 4.4. There, $\Rightarrow$ is even terminating and $\mathcal{R}$ is "almost orthogonal", that is, every two overlapping term rewrite steps $t_1 \leftarrow t \rightarrow t_2$ satisfy $t_1 = t_2$ and the overlap occurs at the roots of the left-hand sides of the applied rules.

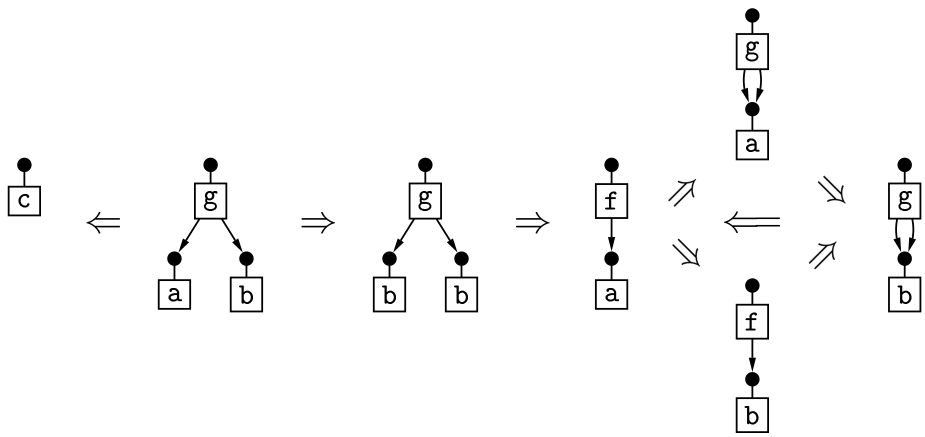


Figure 10: Non-confluence of $\Rightarrow$

8. Rewriting modulo Bisimilarity

We have seen that the relation $\Rightarrow_{\mathrm{bi}}$ behaves nicely with respect to confluence in that it is confluent if and only if term rewriting is confluent. A drawback of $\Rightarrow_{\mathrm{bi}}$, as remarked in the previous section, is that it is non-terminating whenever there is a term graph containing two different nodes representing the same term. We show that this problem disappears when moving from $\Rightarrow_{\mathrm{bi}}$ to rewriting of bisimilarity classes. It turns out that term graph rewriting modulo bisimilarity behaves with respect to confluence and termination exactly like term rewriting.

We call the equivalence class of a term graph $G$ with respect to $\sim$ the *bisimilarity class* of $G$ and denote it by $[G]$.

**Definition 8.1 ($\Rightarrow_\sim$)** The relation $\Rightarrow_\sim$ on bisimilarity classes is defined as follows: $[G] \Rightarrow_\sim [H]$ if $G' \Rightarrow H'$ for some $G' \in [G]$ and $H' \in [H]$. The reflexive-transitive closure of $\Rightarrow_\sim$ is denoted by $\Rightarrow_\sim^*$. We refer to $\Rightarrow_\sim$ as *term graph rewriting modulo bisimilarity.*

**Lemma 8.2** *For all term graphs $G$ and $H$,*

$$[G] \Rightarrow_\sim^* [H] \text{ if and only if } G \Rightarrow_{\mathrm{bi}}^* H.$$

**Proof.** "If": By a straightforward induction on the length of the derivation $G \Rightarrow_{\mathrm{bi}}^* H$.

"Only if": By induction on the length of $[G] \Rightarrow_\sim^* [H]$. If $[G] = [H]$, then $G \preceq \Delta G \succeq H$ and hence $G \Rightarrow_{\mathrm{bi}}^* H$. Suppose now that $[G] \Rightarrow_\sim^* [M] \Rightarrow_\sim [H]$ for some term graph $M$, where $G \Rightarrow_{\mathrm{bi}}^* M$. Then there are term graphs $M'$ and $H'$ such that $M \sim M' \Rightarrow H' \sim H$. It follows $M \Rightarrow_{\mathrm{bi}}^* M' \Rightarrow_{\mathrm{bi}} H' \Rightarrow_{\mathrm{bi}}^* H$, thus $G \Rightarrow_{\mathrm{bi}}^* H$. □

Notice that the above equivalence does not hold for the transitive closures of $\Rightarrow_\sim$ and $\Rightarrow_{\mathrm{bi}}$. For, if $G \Rightarrow_{\mathrm{bi}} H$ is a collapse or copy step, then $[G] \Rightarrow_\sim^+ [H]$ will not hold in general.

**Theorem 8.3** *The following are equivalent:*

*(1)* $\Rightarrow_\sim$ *is confluent.*

*(2)* $\Rightarrow_{\mathrm{bi}}$ *is confluent.*

*(3)* $\rightarrow$ *is confluent.*

**Proof.** The equivalence of (1) and (2) follows from Lemma 8.2, while (2) and (3) are equivalent by Corollary 7.2. □

Now we are going to show that rewriting of bisimilarity classes terminates if and only if term rewriting terminates. Actually, the next lemma says more: term graph rewriting modulo $\sim$ generalizes term rewriting in that every step corresponds to a non-empty sequence of term rewrite steps, while every term rewrite step corresponds to a class rewrite step.

**Lemma 8.4** *For all term graphs $G$ and $H$,*

*(1)* $[G] \Rightarrow_\sim [H]$ *implies* $\mathrm{term}(G) \rightarrow^+ \mathrm{term}(H)$, *and*

*(2)* $\mathrm{term}(G) \rightarrow \mathrm{term}(H)$ *implies* $[G] \Rightarrow_\sim [H]$.

**Proof.** (1) This holds by soundness of $\Rightarrow$, see Theorem 3.4.

(2) In the proof of Theorem 7.1 it is shown that for every term rewrite step $t \rightarrow u$ there are term graphs $T$ and $U$ such that $\triangle t \succeq T \Rightarrow U \preceq \triangle u$. Hence there are $G'$ and $H'$ such that $\triangle G \succeq G' \Rightarrow H' \preceq \triangle H$. Then $G \sim G'$ and $H \sim H'$, hence $[G] \Rightarrow_\sim [H]$. □

The following result is an immediate consequence of Lemma 8.4.

**Theorem 8.5** *The relation* $\Rightarrow_\sim$ *is terminating if and only if* $\rightarrow$ *is.*

## 9. Conclusion

Our positive and negative results on confluence, confluence modulo bisimilarity and the Church-Rosser property modulo bisimilarity are summarized in Table 1 and 2. In both tables, a "+" means that the respective confluence property holds under the given conditions, while a "−" indicates that there exists a counterexample. Exponents refer to the corresponding results and counterexamples.

The conditions for confluence considered in this paper either forbid overlaps between term rewrite rules or require confluence of the associated term rewrite relation. Another tool for analyzing confluence are *critical pairs* of term graph rewrite steps. We refer to [Plu94] for their definition and their use to decide confluence of $\Rightarrow_{\mathrm{coll}}$ in the presence of termination.

Table 1: Overview of confluence

| | $\Rightarrow$ | $\Rightarrow_{coll}$ | $\Rightarrow_{copy}$ | $\Rightarrow_{bi}$ | $\Rightarrow_{\sim}$ |
|---|---|---|---|---|---|
| $\mathcal{R}$ orthogonal | $+$ | $-^{6.2}$ | $+$ | $+$ | $+$ |
| $\mathcal{R}$ non-overlapping | $+^{5.3}$ | $-$ | $-^{5.10}$ | $-^{5.9}$ | $-$ |
| $\rightarrow$ confluent | $-$ | $-$ | $-$ | $+^{7.2}$ | $+$ |
| $\mathcal{R}$ left-linear, $\rightarrow$ confluent | $-$ | $-$ | $+^{7.4}$ | $+$ | $+$ |
| $\rightarrow$ confluent, $\Rightarrow_{coll}$ weakly normalizing | $-$ | $+^{7.5}$ | $-^{5.10}$ | $+$ | $+$ |
| $\mathcal{R}$ left-linear, $\rightarrow$ confluent & terminating | $-^{4.4}$ | $+$ | $+$ | $+$ | $+$ |

Table 2: Confluence properties modulo $\sim$

| | $\Rightarrow$ confluent modulo $\sim$ | $\Rightarrow$ Church-Rosser modulo $\sim$ |
|---|---|---|
| $\mathcal{R}$ orthogonal | $+^{6.4}$ | $-^{6.2}$ |
| $\mathcal{R}$ non-overlapping, $\rightarrow$ confluent & terminating | $-^{5.10}$ | $-$ |
| $\mathcal{R}$ left-linear, $\rightarrow$ confluent & weakly normalizing | $-^{7.8}$ | $-$ |
| $\mathcal{R}$ left-linear, $\rightarrow$ confluent, $\Rightarrow$ weakly normalizing | $+$ | $+^{7.7}$ |

## References

[AK96] Zena M. Ariola and Jan Willem Klop. Equational term graph rewriting. *Fundamenta Informaticae*, 26:207–240, 1996.

[AKP97] Zena M. Ariola, Jan Willem Klop, and Detlef Plump. Confluent rewriting of bisimilar term graphs. In *Proc. Fourth Workshop on Expressiveness in Concurrency*, volume 7 of *Electronic Notes in Theoretical Computer Science*. Elsevier, 1997.

[AS85] Harold Abelson and Gerald Jay Sussman. *Structure and Interpretation of Computer Programs.* The MIT Press, 1985.

[BvEG+87] Hendrik Barendregt, Marko van Eekelen, John Glauert, Richard Kennaway, Rinus Plasmeijer, and Ronan Sleep. Term graph rewriting. In *Proc. Parallel Architectures and Languages Europe*, volume 259 of *Lecture Notes in Computer Science*, pages 141–158. Springer-Verlag, 1987.

[CD97] Andrea Corradini and Frank Drewes. (Cyclic) term graph rewriting is adequate for rational parallel term rewriting. Technical Report TR-97-14, Universitá di Pisa, Dipartimento di Informatica, 1997.

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

[HL91] Gérard Huet and Jean-Jacques Lévy. Computations in orthogonal rewrite systems, I and II. In Jean-Louis Lassez and Gordon Plotkin, editors, *Computational Logic: Essays in Honor of Alan Robinson*, pages 395–443. MIT Press, 1991.

[HP88] Berthold Hoffmann and Detlef Plump. Jungle evaluation for efficient term rewriting. In *Proc. Algebraic and Logic Programming.* Mathematical Research 49, pages 191–203, Berlin, 1988. Akademie-Verlag. Also in Springer Lecture Notes in Computer Science 343, 191–203, 1989.

[HP96] Annegret Habel and Detlef Plump. Term graph narrowing. *Mathematical Structures in Computer Science*, 6:649–676, 1996.

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

[KKSdV94] Richard Kennaway, Jan Willem Klop, Ronan Sleep, and Fer-Jan de Vries. On the adequacy of term graph rewriting for simulating term rewriting. *ACM Transactions on Programming Languages and Systems*, 16(3):493–523, 1994.

[Klo92] Jan Willem Klop. Term rewriting systems. In S. Abramsky, Dov M. Gabbay, and T.S.E. Maibaum, editors, *Handbook of Logic in Computer Science,* volume 2, pages 1–116. Oxford University Press, 1992.

[Plu91] Detlef Plump. Graph-reducible term rewriting systems. In *Proc. Graph Grammars and Their Application to Computer Science*, volume 532 of *Lecture Notes in Computer Science*, pages 622–636. Springer-Verlag, 1991.

[Plu93a] Detlef Plump. Collapsed tree rewriting: Completeness, confluence, and modularity. In *Proc. Conditional Term Rewriting Systems*, volume 656 of *Lecture Notes* *in Computer Science*, pages 97–112. Springer-Verlag, 1993.

[Plu93b] Detlef Plump. Evaluation of functional expressions by hypergraph rewriting. Dissertation, Universität Bremen, Fachbereich Mathematik und Informatik, 1993.

[Plu94] Detlef Plump. Critical pairs in term graph rewriting. In *Proc. Mathematical Foundations of Computer Science 1994*, volume 841 of *Lecture Notes in Computer Science*, pages 556–566. Springer-Verlag, 1994.

[Rao95] Madala R.K. Krishna Rao. Graph reducibility of term rewriting systems. In *Proc. Mathematical Foundations of Computer Science 1995*, volume 969 of *Lecture Notes in Computer Science*, pages 371–381. Springer-Verlag, 1995.

[Sta80] John Staples. Computation on graph-like expressions. *Theoretical Computer Science*, 10:171–185, 1980.