Properties of a First-Order Functional Language with Sharing

Zena Ariola & Arvind

CIS-TR-96-10
May 1996

Also in *Theoretical Computer Science 146, (1995) 69-108*

Department of Computer and Information Science
University of Oregon

Properties of a First-order Functional Language with Sharing

Zena M. Ariola

Computer & Information Science Department

University of Oregon

Arvind

Laboratory for Computer Science

Massachusetts Institute of Technology

A calculus and a model for a first-order functional language with sharing is presented. In most implementations of functional languages, argument subexpressions in a function application are shared to avoid their repeated evaluation. Recursive functions are typically implemented using graphs with cycles. Compilers for these languages sometimes employ non-left-linear and left-cyclic rules for optimizations. A Graph Rewriting System (GRS) to address these concerns is developed. It is shown that a GRS without interfering rules is confluent. Along the lines of Lévy's term model for the λ-calculus, a semantics of such a GRS is also presented. An application of the term model to compiler optimizations is discussed.

1. Introduction

Sharing of subexpressions is of utmost importance in the implementation of functional languages. Consider the function definition $\mathsf{F}\ x = x+x$ and the expression $\mathsf{F}(2+3)$. Any decent implementation, independently of the evaluation strategy (normal-order or applicative-order) it employs, will evaluate the subexpression $2+3$ only once. Several compiler optimizations are about increasing the sharing of subexpressions to avoid their repeated evaluation. In this paper, we discuss the syntactic and semantic properties of a calculus, which is adequate for capturing the sharing of subexpressions in first-order functional languages. The results of this paper are also relevant to compiling higher-order functional languages, because compilers of such languages often employ a technique known as "lambda-lifting" [13]. The program that results after lambda-lifting is in a "supercombinatory" form, and is treated as a first-order program [12].

A way to capture sharing is to represent the expression as a graph instead of a linear text string or tree. This allows sharing of identical terms through pointers, and avoids repeated evaluation of identical terms as it is commonly done in normal-order reduction. Graph reduction for the λ-calculus was proposed by Wadsworth in order to bring together the advantages of both the applicative and the normal order evaluation [20]. Wadsworth also formally proved the correctness of his graph reduction technique. As an aside, Wadsworth also showed that his graph reduction did not capture enough sharing to lead to an optimal interpreter. More recently a new graph structure, which allows sharing of "*contexts*", has been proposed in [14, 17]. This latter technique leads to provably optimal interpreters for the λ-calculus [18]. In this paper, however, we are not concerned with optimality questions, and we restrict our attention to "argument sharing" in a language which is simpler than the λ-calculus.

Much of the past work on graph rewriting has been to prove its correctness with respect to either the λ-calculus [20] or Term Rewriting Systems [7, 8, 9, 16]. In contrast, this paper explores graph rewriting as a system in its own right, and makes no attempt to prove the correctness of a graph implementation with respect to a "tree (or unshared) view" of the computation. Motivated by what we have observed in real implementations of functional languages, we explore syntactic and semantic properties of graphs with cycles, and rewriting rules that recognize or create cycles. In this respect our calculus goes farther than either [20] or [8] where only acyclic graphs are considered. Without cyclic graphs some important implementation ideas are ruled out. More recently, Klop *et al.* [15] have extended the Barendregt's graph rewriting system to deal with cycles. However, their approach is significantly different from ours in that they model cyclic graph rewriting as "transfinite reduction" of infinitary graph terms.

In the following, we formally introduce a Graph Rewriting System (GRS). The basic feature of a GRS is the *block* construct, *i.e., letrec*. A block in a GRS is not treated as syntactic sugar for application; it is central to expressing the sharing of subexpressions. Our GRS includes cyclic terms and permits both *non-left-linear rules* and *left-cyclic rules*. We prove that in the absence of *interfering rules* a GRS is confluent. This is a more general result than the confluence theorem in [15]. We think that our approach also leads to a simpler proof of confluence than in [15].

We also develop a term model for a GRS without interfering rules along the lines of Lévy's term model for the λ-calculus. We introduce the notion of "information content" associated to a term, and show that the information content defines a congruence on the set of terms. This result can be applied in a straightforward manner to show the partial correctness of those optimizations that simply increase the sharing in a term. An example of such an optimization is the common subexpression elimination. Moreover, the result implies the partial correctness of the cyclic implementation of the Y-rule.

The paper is organized as follows: In Section 2, we introduce graph rewriting taking combinatory logic as an example, and compare our notation with that of Barendregt's [8]. We formally describe GRSs, in Section 3. We introduce an ordering on terms based on sharing of subterms, and formulate the notion of a redex using this ordering. In Section 4, we introduce GRSs without interfering rules and prove their confluence. Section 5 gives a term model for non-interfering GRSs. In Section 6, we apply the results of Section 5 to prove the correctness of some compiler optimizations, while in Section 7 we discuss directions for future work.

2. An Example: A Graph Rewriting System for Combinatory Logic

Given the TRS rule $\mathsf{F}(x) \longrightarrow \mathsf{G}(x, x)$, the term $\mathsf{F}(+(2,3))$ can be rewritten to $\mathsf{G}(+(2,3), +(2,3))$. That is, the term $+(2,3)$ is substituted for each occurrence of the variable x on the right-hand side of the above rule, and is thus, *duplicated.* A graph rewriting system avoids this duplication of work by substituting a *pointer* to $+(2,3)$ for each reference to variable x, as depicted below:

$$\begin{tikzcd}
& \mathsf{F} \arrow[rr] \arrow[d] & & \mathsf{G} \arrow[dll, bend left] \arrow[dll, bend left=50] \\
& + \arrow[dl] \arrow[d] & & \\
2 & 3 & &
\end{tikzcd}$$

Our formalism for expressing graph rewriting is based on the observation that a natural way to represent a graph textually is to associate an identifier to each node of the graph, and then write down all the interconnections as a *recursive let-block*. Equivalently we can say that we associate a name to each subexpression of a term. For example, the above term $\mathsf{F}(+(2,3))$ will be expressed as

$$\begin{aligned} \{\ t_1 &= +(2,3); \\ t_2 &= \mathsf{F}(t_1); \\ &\mathsf{In}\ t_2\}\ . \end{aligned}$$

In applying the above rule, the name t_1, and not the expression $+(2,3)$, will be substituted for each occurrence of x, leading to the term

$$\begin{aligned} \{\ t_1 &= +(2,3); \\ t_2 &= \mathsf{G}(t_1, t_1); \\ &\mathsf{In}\ t_2\}\ . \end{aligned}$$

The substitution of an expression such as $+(2,3)$ is not permitted to avoid duplication of work. Only when $+(2,3)$ becomes a *value*, *i.e.*, 5, it can be substituted for each free occurrence of t_1. Therefore, we think that an *essential feature of a language to model sharing is a recursive let-block construct with a suitable notion of substitutable values.*

The syntax of GRS terms is given in Figure 1. Superscript on a function symbol indicates its "arity" *i.e.*,

SE	$\in$	Simple Expression
E	$\in$	Expression
F^k	$\in$	$\mathcal{F}^k$
Constant	$\in$	$\mathcal{F}^0$
SE	::=	*Variable* | *Constant*
E	::=	SE
	|	$\mathsf{F}^k\,(SE_1, \cdots, SE_k)$
	|	*Block*
	|	Ω
Block	::=	$\{\ [Binding;]^*\ \mathsf{In}\ SE\}$
Binding	::=	$Variable = E$
Term	::=	E

Figure 1. Syntax of terms of a GRS with signature $\mathcal{F}$

the number of arguments it is supposed to have; constants are assumed to be function symbols of arity 0. Throughout this paper we consider constants to be (in implementation parlance) "unboxed". Thus, they are never shared and are freely substitutable. "Boxed" values can be modeled in a straightforward manner by wrapping a function symbol of arity one (say, called Box) around a value. The textual order of bindings in a block is not relevant, and the variable names on the left-hand side in a block must be pairwise distinct. Furthermore, for technical convenience we make a stronger assumption, that is, no variable name can be defined more than once regardless of its lexical level. Ω is a special term whose significance will become clear when we formally define the notion of a redex in Section 3. Next we informally present the GRS for combinatory logic, and relate our GRS notation to Barendregt's [8].

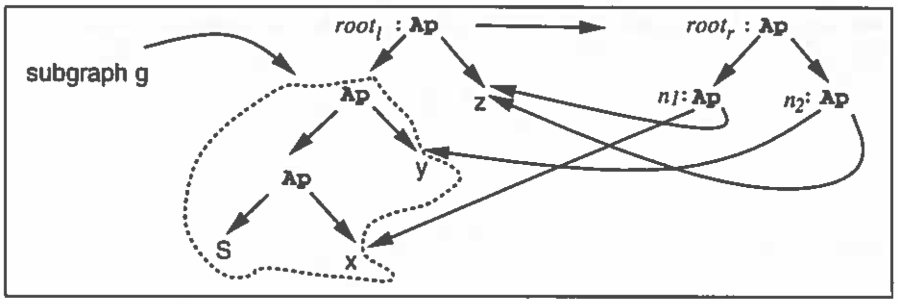


Figure 2. Graph rule for the S combinator

Consider the S rule:

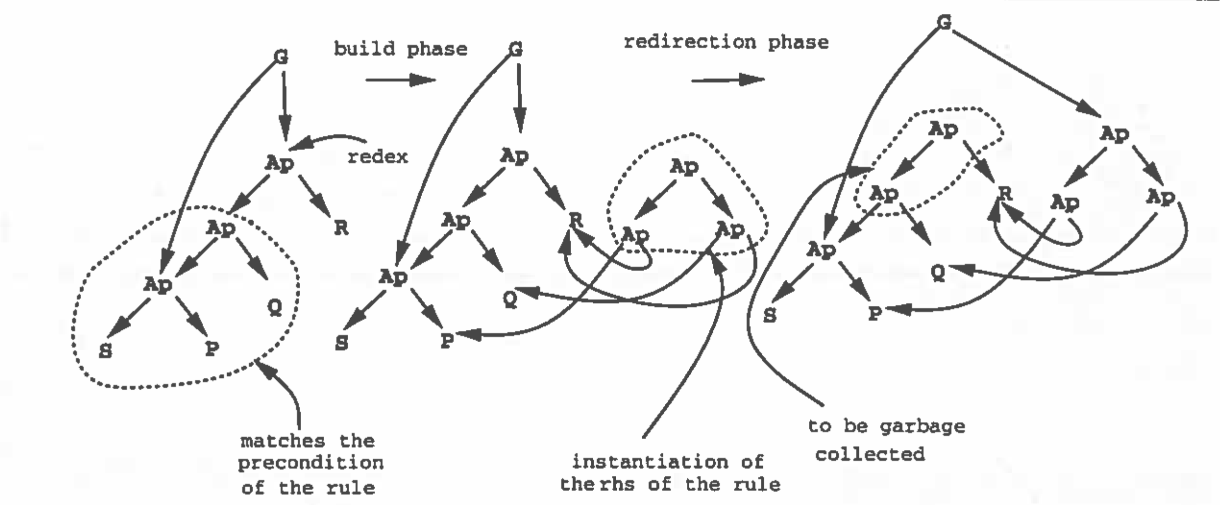
$$\mathsf{Ap}\ (\mathsf{Ap}\ (\mathsf{Ap}\ (\mathsf{S},\ x),\ y),\ z) \longrightarrow \mathsf{Ap}\ (\mathsf{Ap}\ (x,\ z),\ \mathsf{Ap}\ (y,\ z))$$

which is shown in Barendregt *et al.* graph notation in Figure 2. Intuitively, applying this rule consists of allocating three new nodes, $root'_r$, n'_1 and n'_2, corresponding to $root_r$, n_1 and n_2, respectively, (*build phase*) and redirecting all the pointers to the redex node (*i.e.*, the node matching $root_l$) to the node $root'_r$ (*redirection phase*). Notice that the redirection phase does not affect the graph matching the subgraph $g \equiv$ Ap (Ap (S, x), y) (included in dotted lines in Figure 2), thus we call the subgraph g the *precondition* of the above rule. In order to represent the right-hand side (*i.e.*, rhs) of the S rule textually we simply write down the graph rooted at $root_r$ as a recursive let-block. Thus, the $\mathsf{S_g}$ rule, that is, the S rule in GRS notation, can be written as follows:

$$\frac{x_1 = \mathsf{Ap}\ (x_2,\ z_2)\ ;\ x_2 = \mathsf{Ap}\ (\mathsf{S},\ z_3)}{x = \mathsf{Ap}\ (x_1,\ z_1) \longrightarrow x = \{\begin{array}{lcl} t_1 & = & \mathsf{Ap}\ (z_3,\ z_1); \\ t_2 & = & \mathsf{Ap}\ (z_2,\ z_1); \\ t & = & \mathsf{Ap}\ (t_1,\ t_2); \\ \multicolumn{3}{l}{\ \mathsf{In}\ t\}} \end{array}}\ .$$

Variables, such as t_1, t_2 and t, that occur on the rhs of the rule but not on the lhs or in the precondition, generate new corresponding variables for each application of the rule.
Similarly the $\mathsf{K_g}$ rule, that is, the K rule in GRS notation is expressed as follows:

$$\frac{x_1 \ = \ \mathsf{Ap}\ (\mathtt{K},\ z_2)}{x = \mathsf{Ap}\ (x_1,\ z_1) \longrightarrow x = z_2}\ .$$


build phase
redirection phase
G
Ap
redex
R
Q
S
P
matches the precondition of the rule
instantiation of the rhs of the rule
to be garbage collected

$$\left\{\ \boxed{\begin{array}{l} w_1 = \mathsf{Ap}\ (\mathsf{S},\ \mathsf{P}); \\ w_2 = \mathsf{Ap}\ (w_1,\ \mathsf{Q}); \end{array}} \right. \atop \begin{array}{l} w_3 = \underbrace{\mathsf{Ap}\ (w_2,\ \mathsf{R});}_{\rho} \\ w_4 = \mathsf{G}\ (w_1,\ w_3); \\ \mathsf{In}\ w_4\} \end{array} \xrightarrow[w_3]{} M_1 \equiv \begin{array}{l} \{w_1 = \mathsf{Ap}\ (\mathsf{S},\ \mathsf{P}); \\ \ w_2 = \mathsf{Ap}\ (w_1,\ \mathsf{Q}); \\ \ w_3 = \{\ t'_1 = \mathsf{Ap}\ (\mathsf{P},\ \mathsf{R}); \\ \qquad\quad t'_2 = \mathsf{Ap}\ (\mathsf{Q},\ \mathsf{R}); \\ \qquad\quad t' = \mathsf{Ap}\ (t'_1,\ t'_2); \\ \qquad\quad \mathsf{In}\ t'\} \\ \ w_4 = \mathsf{G}\ (w_1,\ w_3); \\ \ \mathsf{In}\ w_4\} \end{array} \equiv_\alpha \begin{array}{l} \{w_1 = \mathsf{Ap}\ (\mathsf{S},\ \mathsf{P}); \\ \ w_2 = \mathsf{Ap}\ (w_1,\ \mathsf{Q}); \\ \ t'_1 = \mathsf{Ap}\ (\mathsf{P},\ \mathsf{R}); \\ \ t'_2 = \mathsf{Ap}\ (\mathsf{Q},\ \mathsf{R}); \\ \ t' = \mathsf{Ap}\ (t'_1,\ t'_2); \\ \ w_4 = \mathsf{G}\ (w_1,\ t'); \\ \ \mathsf{In}\ w_4\} \end{array} \equiv_\alpha \begin{array}{l} \{w_1 = \mathsf{Ap}\ (\mathsf{S},\ \mathsf{P}); \\ \ t'_1 = \mathsf{Ap}\ (\mathsf{P},\ \mathsf{R}); \\ \ t'_2 = \mathsf{Ap}\ (\mathsf{Q},\ \mathsf{R}); \\ \ t' = \mathsf{Ap}\ (t'_1,\ t'_2); \\ \ w_4 = \mathsf{G}\ (w_1,\ t'); \\ \ \mathsf{In}\ w_4\} \end{array}$$

Figure 3. Graph reduction and its corresponding GRS reduction

In Figure 3 we show the graph reduction in Barendregt notation and our notation using the following term:

$$M \equiv \left\{\ \boxed{\begin{array}{lcl} w_1 & = & \mathsf{Ap}\ (\mathsf{S},\ \mathsf{P}); \\ w_2 & = & \mathsf{Ap}\ (w_1,\ \mathsf{Q}); \end{array}} \right. \atop \begin{array}{lcl} w_3 & = & \underbrace{\mathsf{Ap}\ (w_2,\ \mathsf{R});}_{\rho} \\ w_4 & = & \mathsf{G}\ (w_1,\ w_3); \\ \mathsf{In}\ w_4\} & . & \end{array}$$

Intuitively, the two bindings in the box inside M match the precondition and the subterm ρ matches the lhs of the $\mathsf{S_g}$ rule according to the following substitution:

$$x = w_3,\ x_1 = w_2,\ x_2 = w_1,\ z_3 = \mathsf{P},\ z_2 = \mathsf{Q},\ z_1 = \mathsf{R}\ .$$

Since w_3 corresponds to x, it is called the root of the redex. Using the above substitution an instance of the rhs of the $\mathsf{S_g}$ rule is created. It introduces fresh copies for the bound variables of the rhs of the $\mathsf{S_g}$ rule. This step corresponds to the *build phase* of the Barendregt system. Subsequently, variable w_3 is bound to the newly instantiated term. Thus obtaining M_1. This rebinding of w_3 corresponds to the *redirection phase* in Barendregt. M_1 is then *canonicalized* by flattening blocks and substituting variables and constants in M_1. The introduction of fresh variables during the instantiation of the rhs of a rule removes the need to rename variables when blocks are flattened. There is one more step called the *garbage collection*, that is, the deletion of nodes that are not reachable from the root. Thus, the definition of variable w_2 is eliminated because it is no longer reachable from node w_4. The term M is said to rewrite to the final term so obtained. Notice that the final term is indeed the term corresponding to the rightmost graph in Figure 3.

There is, however, a subtle difference between the two systems which shows up in the presence of "projection" rules and cyclic graphs. For example, given the rule $x = \mathsf{I}(y) \longrightarrow x = y$, and the cyclic term $M \equiv \{t = \mathsf{I}(t);\ \mathsf{In}\ t\}$, $M \longrightarrow M$, following the Barendregt system, while $M \longrightarrow \{t = t;\ \mathsf{In}\ t\}$, following our system. As explained later $\{t = t;\ \mathsf{In}\ t\}$ becomes $\odot$, a symbol which represents a "meaningless" term. This difference has a strong impact on the confluence of GRSs. We will further clarify these issues after introducing GRSs formally in Section 3.

3. GRS: Terms, Rules and Reduction

3.1. GRS terms

Definition 3.1. (GRS term) *A GRS term over signature $\mathcal{F} = \mathcal{F}^0 \cup \mathcal{F}^1 \cup \cdots$ is defined inductively as follows:*

(i) a variable x is a term;

(ii) a constant c, where $c \in \mathcal{F}^0$, is a term;

(iii) Ω is a term;

(iv) $\mathsf{F}^k(y_1, \cdots, y_k)$ is a term if $\forall i, 1 \leq i < k$, y_i is either a variable or a constant, and $\mathsf{F}^k \in \mathcal{F}^k$;

(v) $\{x_1 = e_1; \cdots; x_p = e_p;\ \mathsf{In}\ x\}$ is a term if

(v.1) $\forall i, 1 \leq i \leq p$, x_i is a variable and e_i is a term;

(v.2) x is either a constant or a variable;

(v.3) $\forall i, j, 1 \leq i < j \leq p$, $x_i \not\equiv x_j$.

The order of bindings in a block term is irrelevant.

Clause (v.3) prevents multiple definitions of a variable; for example, $\{x = 3;\ x = 4; \cdots \mathsf{In}\ z\}$ is not a legal term. For technical convenience we assume that if the main term M is of the form $\mathsf{F}^k(y_1, \cdots, y_k)$ or Ω then M has a name associated to it, that is, $M \equiv \{t = M;\ \mathsf{In}\ t\}$. We also assume that $\{\mathsf{In}\ x\} \equiv x$.

Definition 3.2. (Root of a term) *Given a GRS term M, the root of M, $\mathtt{Root}(M)$, is :*

(i) M, if M is either a constant or a variable;

(ii) x, if M is $\{x_1 = e_1; \cdots x_p = e_p;\ \mathsf{In}\ x\}$.

Definition 3.3. (Free Variables of a GRS term) *The set of free variables of a GRS term M, $\mathtt{FV}(M)$, is defined inductively as follows:*

(i) $\mathtt{FV}(x) = \{x\}$;

(ii) $\mathtt{FV}(c) = \emptyset$;

(iii) $\mathtt{FV}(\Omega) = \emptyset$;

(iv) $\mathtt{FV}(\mathsf{F}^k(y_1, \cdots, y_k)) = \bigcup\{\mathtt{FV}(y_i) \mid 1 \leq i \leq k\}$;

(v) $\texttt{FV}(\{x_1 = e_1; \cdots; x_p = e_p;\ \mathsf{In}\ x\}) = (\bigcup\{\texttt{FV}(e_i) \mid 1 \leq i \leq p\} \cup \texttt{FV}(x)) - \{x_1, \cdots, x_p\}$, *where $-$ is the set difference operator.*

DEFINITION 3.4. (BOUND VARIABLES OF A GRS TERM) *The set of bound variables of a GRS term M, $\texttt{BV}(M)$, is defined inductively as follows:*

(i) $\texttt{BV}(x) = \emptyset$;

(ii) $\texttt{BV}(c) = \emptyset$;

(iii) $\texttt{BV}(\Omega) = \emptyset$;

(iv) $\texttt{BV}(\mathsf{F}^k(y_1, \cdots, y_k)) = \emptyset$;

(v) $\texttt{BV}(\{x_1 = e_1; \cdots; x_p = e_p;\ \mathsf{In}\ x\}) = \{x_1, \cdots, x_p\} \cup (\bigcup\{\texttt{BV}(e_i) \mid 1 \leq i \leq p\})$.

DEFINITION 3.5. (VARIABLES OF A GRS TERM) *The set of variables of a GRS term M, $\texttt{Var}(M)$, is defined as $\texttt{FV}(M) \cup \texttt{BV}(M)$.*

DEFINITION 3.6. (CONSTANTS OF A GRS TERM) *The set of constants of a GRS term M, $\texttt{Constants}(M)$, is defined inductively as follows:*

(i) $\texttt{Constants}(x) = \emptyset$;

(ii) $\texttt{Constants}(c) = \{c\}$;

(iii) $\texttt{Constants}(\Omega) = \emptyset$;

(iv) $\texttt{Constants}(\mathsf{F}^k(y_1, \cdots, y_k)) = \bigcup\{\texttt{Constants}(y_i) \mid 1 \leq i \leq k\}$;

(v) $\texttt{Constants}(\{x_1 = e_1; \cdots; x_p = e_p;\ \mathsf{In}\ x\}) = (\bigcup\{\texttt{Constants}(e_i) \mid 1 \leq i \leq p\}) \cup \texttt{Constants}(x)$.

For technical convenience we will assume the following variable convention.

VARIABLE CONVENTION:

i.) all bound variables of a term are distinct;

ii.) all bound and free variables of a term are distinct.

The following two terms are illegal because of this variable convention.

$$\begin{array}{ll}
\{\ x = \{\ y = +(w, w); & \qquad \{\ x = \{\ x = +(w, w); \\
\qquad\qquad \mathsf{In}\ y\}; & \qquad\qquad\qquad \mathsf{In}\ x\}; \\
\quad w = +(y, x); & \qquad\quad w = +(y, x); \\
\quad \mathsf{In}\ w\} & \qquad\quad \mathsf{In}\ w\}
\end{array}$$

DEFINITION 3.7. (SUBSTITUTION OPERATION) *Given a GRS term M, $y \in (Variable \cup \mathcal{F}^0)$, and $z \in Variable$ such that $\{y, z\} \not\subseteq \texttt{BV}(M)$, the substitution of y for each free occurrence of z in M, written as $M[y/z]$, is defined inductively as follows:*

(i) $z[y/z] = y$;

(ii) $x[y/z] = y$, *if* $x \not\equiv z$;

(iii) $c[y/z] = c$;

(iv) $\Omega[y/z] = \Omega$;

(v) $\mathsf{F}^k(y_1, \cdots, y_k)[y/z] = \mathsf{F}^k(y_1[y/z], \cdots, y_k[y/z])$;

(vi) $\{x_1 = e_1; \cdots; x_p = e_p;\ \mathsf{In}\ x\}[y/z] = \{x_1 = e_1[y/z]; \cdots; x_p = e_p[y/z];\ \mathsf{In}\ x[y/z]\}$.

3.2. Canonical forms of terms

Consider the following terms:

$$\begin{array}{lll}
\{\ x = 8; & \{\ x = 8; & \{\ w = +(8,8); \\
\quad z = \{\ y = x; & \quad y = x; & \quad \mathsf{In}\ w\} \\
\qquad\quad w = +(x,y); & \quad w = +(x,y); & \\
\qquad\quad \mathsf{In}\ w\}; & \quad \mathsf{In}\ w\}; & \\
\quad \mathsf{In}\ z\} & &
\end{array}$$

These terms have different syntactic structure, however, we consider this difference merely *syntactic noise.* While the following terms:

$$\begin{array}{ll}
\{\ w = +(x,y); & \{\ w = +(x_1,y_1); \\
\quad \mathsf{In}\ w\}; & \quad \mathsf{In}\ w\}
\end{array}$$

which differ only in the reference to the free variables will not be considered the same. We also consider the following two terms to be distinct because Ω is not a "substitutable value":

$$\begin{array}{ll}
\{\ x = \mathsf{F}(y,z); & \{\ x = \mathsf{F}(y,y); \\
\quad y = \Omega; & \quad y = \Omega; \\
\quad z = \Omega; & \quad \mathsf{In}\ x\} \\
\quad \mathsf{In}\ x\} &
\end{array}$$

We introduce the following rules to compute the canonical form of a term.

Block Flattening rule:

$$\begin{array}{lcl}
\{x\ =\ \{\ y_1 = e_1';\cdots\ y_m = e_m'; & & \{x\ =\ y; \\
\qquad\qquad \mathsf{In}\ \ y\}; & \longrightarrow & \quad y_1 = e_1';\cdots y_m = e_m'; \\
\quad x_1 = e_1;\ \cdots\ \ x_n = e_n; & & \quad x_1 = e_1;\ \cdots\ \ x_n = e_n; \\
\quad \mathsf{In}\ z\} & & \quad \mathsf{In}\ z\}
\end{array}$$

Substitution rules:

$$\begin{array}{lcll}
\{x_1 = e_1;\cdots x = c;\cdots x_n = e_n;\ \mathsf{In}\ z\} & \longrightarrow & \{x_1 = e_1[c/x];\cdots x_n = e_n[c/x];\ \mathsf{In}\ z[c/x]\} & c \in \mathcal{F}^0 \\
\{x_1 = e_1;\cdots x = y;\cdots x_n = e_n;\ \mathsf{In}\ z\} & \longrightarrow & \{x_1 = e_1[y/x];\cdots x_n = e_n[y/x];\ \mathsf{In}\ z[y/x]\} & x \not\equiv y \\
\{x_1 = e_1;\cdots x = x;\cdots x_n = e_n;\ \mathsf{In}\ z\} & \longrightarrow & \{x_1 = e_1[\odot/x];\cdots x_n = e_n[\odot/x];\ \mathsf{In}\ z[\odot/x]\} &
\end{array}$$

where $\odot$ is a special constant. These rules formalize the notion of a substitutable expression, and say that only constants and variables (provided x and y are distinct variables) can be substituted freely. Moreover, note that a binding like $x = y$ or $x = c$ is deleted after the substitution. If we encounter a *degenerate binding* like $x = x$ then we substitute the special constant $\odot$ for x. Disallowing such bindings does not help because they can arise as a consequence of doing a reduction. In fact, as pointed out in Section 2, given the rule $x = \mathsf{I}(y) \longrightarrow x = y$, the term $\{x = \mathsf{I}(x);\ \mathsf{In}\ x\}$ will go to $\odot$.

Definition 3.8. (Canonical form) *Given a GRS term M, the canonical form of M is computed using the following two steps:*

1. *Flatten all blocks in M using the Block Flattening rule. Then apply the Substitution rules to the term as many times as they apply. Let the result be $\overline{M}$.*
2. *Eliminate the garbage, that is, the subterms of $\overline{M}$ that are unreachable from the root of M. Let the result be* $\texttt{GC}(\overline{M})$.

PROPOSITION 3.9. *The canonical form of a GRS term M always exists and is unique.*

PROOF. Since there are only a finite number of blocks in M, and an application of the Block Flattening rule eliminates one block without creating a new one, the Block Flattening rule can be applied only a finite number of times. The final flattened term is unique since the Block flattening rule has the *diamond property.*

There are only a finite number of bindings of the form $x = y$ or $x = c$ or $x = x$ in the flattened term. Each of these bindings can be eliminated in one substitution step without creating a new binding. Furthermore, because of the third substitution rule (*i.e.*, the introduction of $\odot$), the substitution rules also have the *diamond property.* Hence $\overline{M}$ exits and is unique. $\texttt{GC}(\overline{M})$ is unique by the definition of garbage collection. □

Renaming rule:

$$\{x_1 = e_1; \cdots; x_p = e_p;\ \mathsf{In}\ x\} \longrightarrow \{x_1 = e_1[x_j'/x_j]; \cdots; x_j' = e_j[x_j'/x_j]; \cdots; x_p = e_p[x_j'/x_j]\ \mathsf{In}\ x[x_j'/x_j]\}$$

where x_j' is a new variable. The renaming rule is similar to the α-renaming in the λ-calculus.

DEFINITION 3.10. (α-EQUIVALENCE) *Given GRS terms M and N, M and N are said to be α-equivalent, written as $M \equiv_\alpha N$, iff* $\texttt{GC}(\overline{M}) \equiv \texttt{GC}(\overline{N})$ *up to renaming.*

The canonicalization rules plus renaming do not affect the "graph" associated to a term, in other words, all α-equivalent terms will correspond to rooted isomorphic graphs.

3.3. GRS RULES

DEFINITION 3.11. (GRS RULE) *A GRS rule τ is a set of preconditions, $x_1 = e_1, \cdots, x_n = e_n$, and a left-hand side, l, and a right-hand side, r, and is written as:*

$$\frac{x_1 = e_1; \cdots; x_n = e_n}{x = l \longrightarrow x = r}$$

where:
(i) the pattern of rule τ, $\{x_1 = e_1; \cdots x_n = e_n;\ x = l;\ \mathsf{In}\ x\}$, is a block-term in canonical form, and $l \not\equiv \Omega$;
(ii) r is a term such that $\texttt{FV}(r) \subseteq \texttt{Var}(\{x_1 = e_1; \cdots x_n = e_n;\ x = l;\ \mathsf{In}\ x\})$.
The pattern of rule τ is denoted by $\mathcal{P}(\tau)$. The term r is called the right-hand side of the rule and is denoted by $\mathcal{RHS}(\tau)$. The free variables of the pattern of τ are called the meta-variables of rule τ.

Notice that restriction (*i*) makes it impossible to give a GRS rule to rewrite a constant or a variable. In general, a GRS rule does not contain any Ωs.

DEFINITION 3.12. (GRS) *A GRS is a structure $(A(\mathcal{F}), R)$, where $A(\mathcal{F})$ is the set of GRS terms defined over signature $\mathcal{F}$, and R is a set of GRS rules.*

Definition 3.13. (Left-linear rule) *A GRS rule τ is said to be left-linear iff $\forall y \in \mathtt{Var}(\mathcal{P}(\tau))$, y is referenced at most once in $\mathcal{P}(\tau)$.*

Definition 3.14. (Left-acyclic rule) *A GRS rule τ is said to be left-acyclic iff $\forall y \in \mathtt{BV}(\mathcal{P}(\tau))$, y is not reachable from itself in $\mathcal{P}(\tau)$. Otherwise the rule is said to be a left-cyclic rule.*

The rule, $x = \mathsf{A}(x) \longrightarrow x = 0$, is an example of a left-cyclic rule. Notice that a left-cyclic rule is always non-left-linear.

3.4. Identifying redexes and ω-ordering

There are subtle issues involved in identifying redexes in a term. Consider the following two rules:

$$\tau_1 : \frac{x_1 = \mathsf{F}(0);\ x_2 = \mathsf{F}(0)}{x = \mathsf{G}(x_1, x_2) \ \longrightarrow\ x = 0} \qquad\qquad \tau_2 : \frac{x_1 = \mathsf{F}(0)}{x = \mathsf{G}(x_1, x_1) \ \longrightarrow\ x = 0}$$

and the following two terms:

$$M \equiv \{\ t_1 = \mathsf{F}(0);\ t_2 = \mathsf{F}(0);\ t_3 = \mathsf{G}(t_1, t_2);\ \mathsf{In}\ t_3\} \qquad\qquad N \equiv \{\ t_1 = \mathsf{F}(0);\ t_2 = \mathsf{G}(t_1, t_1);\ \mathsf{In}\ t_2\}$$

On the basis of the intuitive description given in Section 2, we can undoubtedly say that τ_1 matches M, with substitution "$x = t_3, x_1 = t_1, x_2 = t_2$", and τ_2 matches N with substitution "$x = t_2, x_1 = t_1$". Does rule τ_1 applies to N? or does rule τ_2 applies to M? Rule τ_1 does indeed apply to the term N by matching both the preconditions to the same binding, that is, by the substitution "$x_1 = t_1, x_2 = t_1, x = t_2$". However, there is no variable substitution that can make τ_2 applicable to M. Thus, the preconditions of a rule can be satisfied by overlapping bindings, moreover, the left-hand side of a rule can also overlap its precondition, as shown in the following example. Consider the rule

$$\frac{x_1 = \mathsf{G}(y)}{x = \mathsf{G}\,(x_1) \ \longrightarrow\ x = 0}$$

and the term $M \equiv \{t = \mathsf{G}(t);\ \ \mathsf{In}\ t\}$. The substitution "$x = t, x_1 = t, y = t$" makes $\mathsf{G}(t)$ both a redex and its precondition! We can capture the notion of a redex in terms of an ordering on terms.

Definition 3.15. (ω-ordering: $\leq_\omega$) *Given GRS terms M and N in canonical form, $M \leq_\omega N$ iff $\exists$ a function σ :*

$$\mathtt{Var}(M) \cup \mathtt{Constants}(M) \rightarrow \mathtt{Var}(N) \cup \mathtt{Constants}(N)$$

such that:

(i) $\forall c \in \mathtt{Constants}(M)$, $\sigma(c) = c$;

(ii) $\forall x \in \mathtt{FV}(M)$, $\sigma(x) = x$;

(iii) $\forall x \in \mathtt{BV}(M)$, *if* $x = \mathsf{F}^k(y_1, \cdots, y_k)$ *in* M *then* $\sigma(x) \in \mathtt{BV}(N)$ *and* $\sigma(x)$ *is bound to* $\mathsf{F}^k(\sigma(y_1), \cdots, \sigma(y_k))$ *in* N;

(iv) $\sigma(\mathtt{Root}(M)) = \mathtt{Root}(N)$.

σ_{MN} is called the substitution function induced by the ordering.

Notice that according to condition *(iii)* if x is bound to Ω then x can be mapped to any variable or constant in N. Intuitively, $M \leq_\omega N$ if N can be obtained from M by replacing Ω with any other term or by increasing the sharing in M. This is a generalization of the ordering on TRS terms introduced by Huet and Lévy [11].

PROPOSITION 3.16. *The ω-ordering is a partial order with Ω as the least element.*

PROOF. See [5]. □

The following examples may enhance the reader's intuition about ω-ordering.
Consider the following acyclic terms:

$$\begin{aligned}
M_1 &\equiv \{x_1 = \mathsf{G}(x_2, x_2);\ x_2 = \mathsf{F}(0);\ \mathsf{In}\ x_1\} \\
M_2 &\equiv \{x_1 = \mathsf{G}(x_2, x_3);\ x_2 = \mathsf{F}(0);\ x_3 = \mathsf{F}(0);\ \mathsf{In}\ x_1\} \\
M_3 &\equiv \{x_1 = \mathsf{G}(x_2, x_2);\ x_2 = \Omega;\ \mathsf{In}\ x_1\} \\
M_4 &\equiv \{x_1 = \mathsf{G}(x_2, x_3);\ x_2 = \Omega;\ x_3 = \Omega;\ \mathsf{In}\ x_1\}
\end{aligned}$$

Notice that $M_2 \leq_\omega M_1$, $M_3 \leq_\omega M_1$, and M_4 is $\leq_\omega M_1$, M_2 and M_3. However, M_2 and M_3 are not related.
Now consider the following cyclic terms:

$$\begin{aligned}
N_1 &\equiv \{x_1 = \mathsf{F}(x_1);\ \mathsf{In}\ x_1\} \\
N_2 &\equiv \{x_1 = \mathsf{F}(x_2);\ x_2 = \mathsf{F}(x_1);\ \mathsf{In}\ x_1\} \\
N_3 &\equiv \{x_1 = \mathsf{F}(x_2);\ x_2 = \mathsf{F}(x_3);\ x_3 = \mathsf{F}(x_1);\ \mathsf{In}\ x_1\}
\end{aligned}$$

Notice that $N_2 \leq_\omega N_1$ and $N_3 \leq_\omega N_1$ but N_2 and N_3 are not related.

We use ω-ordering as follows in defining a redex. We bind all meta-variables of a rule to Ω. Such a term is called the *closure of a rule*. If term p is the closure of rule τ then a term M is said to be a τ-redex if $p \leq_\omega M$.

DEFINITION 3.17. (CLOSURE OF A RULE) *Given a GRS rule τ, where $\mathcal{P}(\tau) \equiv \{x_1 = e_1; \cdots x_n = e_n;\ x = l;\ \mathsf{In}\ x\}$, the closure of τ, written as $Cl(\tau)$, is the term $\{y_1 = \Omega;\ \cdots y_m = \Omega;\ x_1 = e_1; \cdots x_n = e_n; x = l;\ \mathsf{In}\ x\}$, where $\{y_1, \cdots, y_m\} = \mathtt{FV}(\mathcal{P}(\tau))$.*

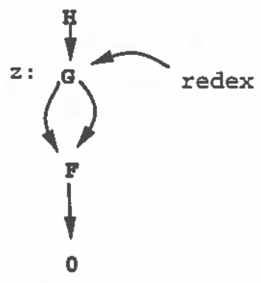
DEFINITION 3.18. (SUBTERM ROOTED AT x_i) *Given a GRS term $M \equiv \{x_1 = e_1; \cdots x_n = e_n;\ \mathsf{In}\ x\}$ in canonical form, and $x_i \in \mathtt{BV}(M)$, the subterm of M rooted at x_i, written as $M@x_i$, is the term $\mathtt{GC}(\{x_1 = e_1; \cdots x_n = e_n;\ \mathsf{In}\ x_i\})$.*

DEFINITION 3.19. (REDEX) *A redex in a GRS term M in canonical form is a triple (τ, z, σ) such that:*
(i) τ is a GRS rule;
(ii) $z \in \mathtt{BV}(M)$, such that $Cl(\tau) \leq_\omega M@z$;
(iii) σ is the substitution induced by $Cl(\tau) \leq_\omega M@z$.
z is said to be the root of the redex. If $z = \mathtt{Root}(M)$ then M itself is said to be a redex.
The set containing the roots of all redexes in M is denoted by $\Re(M)$.

For example, consider the following rule:

$$\tau : \; x = \mathsf{G}(y, y) \longrightarrow x = 0$$

and the term $M \equiv \{z = \mathsf{G}(z_1, z_1);\ z_1 = \mathsf{F}(0);\ z_2 = \mathsf{H}(z);\ \mathsf{In}\ z_2\}$ which has the following graph:



$M@z$ is a redex because $Cl(\tau) \equiv \{y = \Omega;\ x = \mathsf{G}(y, y);\ \ \mathsf{In}\ x\} \leq_\omega M@z$.

DEFINITION 3.20. (DISTINCT REDEXES) *Two redexes $\rho_1 = (\tau_1, z_1, \sigma_1)$ and $\rho_2 = (\tau_2, z_2, \sigma_2)$ in a GRS term M in canonical form are said to be distinct if $\tau_1 \neq \tau_2$ or $z_1 \neq z_2$.*

3.5. REDUCTION

In Section 2 we have explained informally that a GRS reduction consists of first making a copy of the right-hand side of the rule, and then replacing the root of the redex with that copy. Therefore, we introduce the notion of an *instance* of a term M for a given substitution σ, and the operation of *replacement*. An instance of M is created by substituting $\sigma(x)$ for each free occurrence of variable x in M, and by renaming each bound variable of M.

DEFINITION 3.21. (INSTANCE OF A TERM) *Given a GRS term M in canonical form and a substitution σ, an instance of M, written as M^σ, is defined inductively as follows:*

(i) $x^\sigma = \sigma(x)$, *where x is either a constant or a variable;*

(ii) $\Omega^\sigma = \Omega$;

(iii) $\mathsf{F}^k(y_1, \cdots, y_k)^\sigma = \mathsf{F}^k(y_1^\sigma, \cdots, y_k^\sigma)$;

(iv)
$$\left\{ \begin{array}{l} x_1 = e_1; \\ \vdots \\ x_p = e_p; \\ \mathsf{In}\ x \end{array} \right\}^\sigma = \left\{ \begin{array}{l} x_1' = (e_1[x_1'/x_1]\cdots[x_p'/x_p])^\sigma; \\ \vdots \\ x_p' = (e_p[x_1'/x_1]\cdots[x_p'/x_p])^\sigma; \\ \mathsf{In}\ (x[x_1'/x_1]\cdots[x_p'/x_p])^\sigma \end{array} \right\}$$

where x_i', $1 \leq i \leq p$, are new variables.

The above definition does not depend on the order of the substitutions because $x_i' \notin \mathtt{Var}(M)$.

Given a rule τ, and a redex (τ, z, σ) in M, the reduction step consists of *replacing* the term bound to z by $(\mathcal{RHS}(\tau))^\sigma$ (*i.e.*, an instance of the right-hand side of τ).

Definition 3.22. (Replacement) *Given GRS terms M and N, where N is in canonical form and* $\mathtt{BV}(N) \cap \mathtt{Var}(M) = \emptyset$, *the replacement of the term bound to z in M by N, written as $M[z \leftarrow N]$, is defined as follows:*
(i) M, *if* $z \notin \mathtt{BV}(M)$;
(ii) $\{x_1 = e_1; \cdots; z = N; \cdots; x_n = e_n;\ \mathsf{In}\ x\}$, *if* $M \equiv \{x_1 = e_1; \cdots; z = e_i \cdots; x_n = e_n;\ \mathsf{In}\ x\}$.

Notice that no renaming occurs during the replacement, thus the free variables of N can get captured. Moreover, the bound variables of M, which are different from z, are not affected by the replacement.

Notation: $M[\mathcal{Z} \leftarrow \Omega]$, *where* $\mathcal{Z} = \{z_1, \cdots, z_n\}$ *will stand for* $M[z_1 \leftarrow \Omega] \cdots [z_n \leftarrow \Omega]$.

Now we state certain useful properties of replacement.

Proposition 3.23. *Given GRS terms M, N_1 and N_2 in canonical form such that* $\mathtt{BV}(N_1) \cap \mathtt{BV}(N_2) = \emptyset$, $(\mathtt{BV}(N_1) \cup \mathtt{BV}(N_2)) \cap \mathtt{Var}(M) = \emptyset$, *and variables z_1, z_2 such that $z_1 \not\equiv z_2$:*

$$M[z_1 \leftarrow N_1][z_2 \leftarrow N_2] \equiv M[z_2 \leftarrow N_2][z_1 \leftarrow N_1].$$

Proof. By cases on the existence of z_1 and z_2 in $\mathtt{BV}(M)$. □

The following proposition shows that under appropriate circumstances the garbage collection step can be postponed.

Proposition 3.24. *Given GRS terms M and N, such that N is in canonical form. If* $\mathtt{BV}(N) \cap \mathtt{Var}(M) = \emptyset$ *and* $\mathtt{FV}(N) \subseteq \mathtt{Var}(\mathtt{GC}(\overline{M}))$ *then*

$$\mathtt{GC}(\overline{\mathtt{GC}(\overline{M})[z \leftarrow N]}) \equiv \mathtt{GC}(\overline{\overline{M}[z \leftarrow N]}).$$

Proof. The garbage collection does not affect N. The condition $\mathtt{FV}(N) \subseteq \mathtt{Var}(\mathtt{GC}(\overline{M}))$ guarantees that the garbage not picked up by the inner $\mathtt{GC}$ is not affected by N, and thus is collected by the outer $\mathtt{GC}$. □

Definition 3.25. (Reduction) *Given a GRS term M in canonical form and a rule τ, M reduces to N by doing the τ-redex at z in M, written as $M \xrightarrow[z]{\tau} N$, iff (τ, z, σ) is a redex in M, and $N \equiv \mathtt{GC}(\overline{M[z \leftarrow (\mathcal{RHS}(\tau))^\sigma]})$. The reflexive and transitive closure of $\longrightarrow$ is denoted by $\twoheadrightarrow$.*

The redex (τ, z, σ) is often given the name ρ, and sometimes we will also use the notation $M \xrightarrow[\rho]{} N$ to show the reduction of redex ρ. Note that each replacement is followed by a canonicalization step.

3.6. Descendant of a redex

Definition 3.26. (Descendant of a redex) *Given two distinct redexes $\rho_1 = (\tau_1, z_1, \sigma_1)$ and $\rho_2 = (\tau_2, z_2, \sigma_2)$ in a GRS term M in canonical form, the descendant of ρ_2 with respect to the reduction $M \xrightarrow[\rho_1]{} M_1$ (written as $\rho_2 \setminus \rho_1$):*
(i) does not exist, if $z_2 \notin \mathtt{BV}(M_1)$;

(ii) is the triple $\rho'_2 = (\tau_2, z_2, \sigma'_2)$, *if* $z_2 \in \mathtt{BV}(M_1)$ *and* σ'_2 *is:*

$$\sigma'_2(x) = \begin{cases} \sigma_2(x) & \sigma_2(x) \neq z_1 \\ \mathtt{Root}(r_1^{\sigma_1}) & \sigma_2(x) = z_1 \text{ and } \mathtt{Root}(r_1^{\sigma_1}) \neq z_1 \\ \odot & \sigma_2(x) = z_1 \text{ and } \mathtt{Root}(r_1^{\sigma_1}) = z_1 \end{cases}$$

The descendant of a redex does not necessarily exist, and even if it exists, it is not necessarily a redex, as illustrated by the following example. Consider the following two rules:

$$\tau_1 : \quad \frac{x_1 = \mathsf{G}(y)}{x = \mathsf{F}(x_1) \longrightarrow x = x_1}$$

$$\tau_2 : \quad x = \mathsf{G}(0) \longrightarrow x = 0$$

and the following reduction:

$$\begin{tikzcd} & \{\ t_1 = \mathsf{F}(t_2); t_2 = \mathsf{G}(0);\ \mathsf{In}\ t_1\} \arrow[dl, "\tau_2"'] \arrow[dr, "\tau_1"] & \\ \{\ t_1 = \mathsf{F}(0);\ \mathsf{In}\ t_1\} & & \{\ t_2 = \mathsf{G}(0);\ \mathsf{In}\ t_2\} \end{tikzcd}$$

The two redexes in the above term are $\rho_1 = (\tau_1, t_1, \sigma_1)$, where σ_1 is "$x = t_1,\ x_1 = t_2,\ y = 0$", and $\rho_2 = (\tau_2, t_2, \sigma_2)$, where σ_2 is "$x = t_2$", respectively. $\rho_2 \setminus \rho_1$ is (τ_2, t_2, σ'_2), where σ'_2 is "$x = t_2$", and is still a redex. On the other hand, $\rho_1 \setminus \rho_2 = (\tau_1, t_1, \sigma'_1)$, where σ'_1 is "$x = t_1,\ x_1 = 0,\ y = 0$", is no longer a redex.

REMARK 3.27. *A GRS is non-duplicative, that is, the descendant of a redex, if it exists, is unique.*

4. Confluence of a GRS without interfering rules

Not all GRSs are confluent, however, we can show that for a restricted class, namely GRSs without interfering rules, confluence is guaranteed. We introduce the notion of *compatible terms* [11] which will be used, among other things, to define the notion of interference among rules. The idea is that terms which are not ordered, may still have a common upper bound. As we shall see later, such terms potentially interfere with each other.

DEFINITION 4.1. (COMPATIBLE TERMS) *Given GRS terms* M_1 *and* M_2 *in canonical form,* M_1 *and* M_2 *are said to be compatible, written as* $M_1 \uparrow_\omega M_2$, *iff* $\exists M_3$ *such that* $M_1 \leq_\omega M_3$ *and* $M_2 \leq_\omega M_3$.

For example, consider the following terms:

$$M_1 \equiv \{x_1 = \mathsf{G}(x_2, x_2);\ x_2 = \Omega;\ \mathsf{In}\ x_1\},$$
$$M_2 \equiv \{x_1 = \mathsf{G}(x_2, x_3);\ x_2 = \mathsf{F}(0);\ x_3 = \mathsf{F}(0);\ \mathsf{In}\ x_1\},$$
$$M_2 \equiv \{x_1 = \mathsf{G}(x_2, x_3);\ x_2 = \mathsf{F}(0);\ x_3 = \mathsf{H}(0);\ \mathsf{In}\ x_1\}\ .$$

M_1 and M_2 are compatible with least upper bound: $\{x_1 = \mathsf{G}(x_2, x_2);\ x_2 = \mathsf{F}(0);\ \mathsf{In}\ x_1\}$. On the other hand, M_3 is not compatible either with M_1 or M_2.

DEFINITION 4.2. (INTERFERENCE) *Given two distinct GRS rules* τ_1 *and* τ_2, τ_1 *is said to interfere with* τ_2 *iff* $\exists x \in \mathtt{BV}(\mathcal{P}(\tau_1))$ *such that* $Cl(\tau_1)@x \uparrow_\omega Cl(\tau_2)$.

DEFINITION 4.3. (SELF-INTERFERENCE) *Given a GRS rule τ, τ is said to be self-interfering iff* $\exists x \in \mathtt{BV}(\mathcal{P}(\tau_1)) - \mathtt{Root}(\mathcal{P}(\tau))$ *such that* $Cl(\tau)@x \uparrow_\omega Cl(\tau)$.

DEFINITION 4.4. ($\mathrm{GRS_{NI}}$) *If all rules in a GRS are non-self-interfering and pairwise non-interfering then the GRS is called a* GRS_{NI}.

For example, the following rule:

$$\tau: \ \frac{x_1 = \mathsf{L}(y)}{x = \mathsf{L}(x_1) \longrightarrow x = 0}$$

interferes with itself because $Cl(\tau)@x_1 \uparrow_\omega Cl(\tau)$. The following rules:

$$\tau_1: \ x = \mathsf{Or}(y, \mathsf{True}) \longrightarrow x = \mathsf{True}$$

$$\tau_2: \ x = \mathsf{Or}(\mathsf{True}, y) \longrightarrow x = \mathsf{True}$$

also interfere because $Cl(\tau_1) \uparrow_\omega Cl(\tau_2)$, with $\mathsf{Or}(\mathsf{True}, \mathsf{True})$ being the upper-bound.

DEFINITION 4.5. (ORDERING ON REDEXES) *Given two distinct redexes* $\rho_1 = (\tau_1, z_1, \sigma_1)$ *and* $\rho_2 = (\tau_2, z_2, \sigma_2)$ *in a GRS term M in canonical form:*
(i) ρ_1 *occurs inside* ρ_2, *written as* $\rho_1 \preceq \rho_2$, *iff* $z_1 \in \{\sigma_2(x) \mid x \in \mathtt{BV}(\mathcal{P}(\tau_2))\}$;
(ii) ρ_2 *occurs inside* ρ_1, *written as* $\rho_2 \preceq \rho_1$, *iff* $z_2 \in \{\sigma_1(x) \mid x \in \mathtt{BV}(\mathcal{P}(\tau_1))\}$;
(iii) ρ_1 *and* ρ_2 *are disjoint iff* $\rho_1 \not\preceq \rho_2$, *and* $\rho_2 \not\preceq \rho_1$.
Furthermore, if $\rho_1 \preceq \rho_2$ *and* $\rho_2 \preceq \rho_1$ *then* ρ_1 *and* ρ_2 *are said to overlap at the root.*

PROPOSITION 4.6. *Given a* GRS_{NI} *term M in canonical form, any two distinct redexes* ρ_1 *and* ρ_2 *in M are mutually disjoint.*

PROOF. Let $\rho_1 = (\tau_1, z_1, \sigma_1)$ and $\rho_2 = (\tau_2, z_2, \sigma_2)$. Suppose $\rho_1 \preceq \rho_2$. Then by Definition 4.5, $\exists x \in \mathtt{BV}(\mathcal{P}(\tau_2))$, say x_2, such that $z_1 = \sigma_2(x_2)$. Therefore, we have $Cl(\tau_2)@x_2 \uparrow_\omega Cl(\tau_1)$, with $M@z_1$ being the upper bound. Since rules τ_1 and τ_2 are non-interfering, we have reached a contradiction. Similarly, if $\rho_2 \preceq \rho_1$. Hence ρ_1 and ρ_2 must be disjoint. □

PROPOSITION 4.7. *In a* GRS_{NI} *reduction, the descendant of a redex, except for the redex being reduced, if it exists, is always a redex.*

PROOF. Let $M \xrightarrow[\rho]{} M_1$, where $\rho = (\tau, z, \sigma)$. Let $\rho_1 = (\tau_1, z_1, \sigma_1)$ be a distinct redex occurring in M, such that $z_1 \in \mathtt{BV}(M_1)$. We want to show that $\rho_1 \setminus \rho = (\tau_1, z_1, \sigma_1')$ is a redex, that is, σ_1' is a substitution function from $Cl(\tau_1)$ to $M_1@z_1$. According to the definition of descendant of a redex (Definition 3.26), σ_1' differs from σ_1 only at z, the root of redex ρ. Therefore, there are only two reasons why (τ_1, z_1, σ_1') may not be a redex:
(1) the new definition of z;
(2) the canonicalization of $M[z \leftarrow \mathcal{RHS}(\tau)^\sigma]$ removes some elements in the range of σ_1.
The first reason is not possible because, by Proposition 4.6, all distinct redexes in M are disjoint, and therefore, if $\exists x \in \mathtt{Var}(\mathcal{P}(\tau_1))$, such that $\sigma_1(x) = z$ then $x \in \mathtt{FV}(\mathcal{P}(\tau_1))$. The second reason is also not possible

because if a variable or a constant in the range of σ_1 is accessible from z_1 in M then it has to be accessible from z_1 in M_1. □

THEOREM 4.8. (SUBCOMMUTATIVITY OF A $\mathrm{GRS_{NI}}$) *Given a GRS_{NI} term M in canonical form, and two distinct redexes ρ_1 and ρ_2 in M, if $M \xrightarrow[\rho_1]{} M_1$ and $M \xrightarrow[\rho_2]{} M_2$, then $\exists\ M_3$ such that $M_2 \xrightarrow[\rho_1 \backslash \rho_2]{0/1} M_3$ and $M_1 \xrightarrow[\rho_2 \backslash \rho_1]{0/1} M_3$.*

PROOF. Let $\rho_1 = (\tau_1, z_1, \sigma_1)$, and $\rho_2 = (\tau_2, z_2, \sigma_2)$. Then, $M_1 \equiv \mathtt{GC}(\overline{M[z_1 \leftarrow r_1^{\sigma_1}]})$, where $r_1 = \mathcal{RHS}(\tau_1)$, and $M_2 \equiv \mathtt{GC}(\overline{M[z_2 \leftarrow r_2^{\sigma_2}]})$, where $r_2 = \mathcal{RHS}(\tau_2)$.

The proof is by cases on the existence of the descendant of redexes ρ_1 and ρ_2, respectively.

1: $\rho_2 \setminus \rho_1 = (\tau_2, z_2, \sigma_2')$ and $\rho_1 \setminus \rho_2 = (\tau_1, z_1, \sigma_1')$.
By Proposition 4.7, (τ_2, z_2, σ_2') and (τ_1, z_1, σ_1') must be redexes. Thus, we want to show the following:

$$\mathtt{GC}(\overline{M_1[z_2 \leftarrow r_2^{\sigma_2'}]}) \equiv \mathtt{GC}(\overline{M_2[z_1 \leftarrow r_1^{\sigma_1'}]})$$

We have

$$\begin{array}{lll}
 & \mathtt{GC}(\overline{M_2[z_1 \leftarrow r_1^{\sigma_1'}]}) & \\
\equiv & \mathtt{GC}(\overline{\mathtt{GC}(\overline{M[z_2 \leftarrow r_2^{\sigma_2}]})[z_1 \leftarrow r_1^{\sigma_1'}]}) & \\
\equiv & \mathtt{GC}(\overline{\overline{M[z_2 \leftarrow r_2^{\sigma_2}]}[z_1 \leftarrow r_1^{\sigma_1'}]}) & \text{(by Proposition 3.24)} \\
\equiv & \mathtt{GC}(\overline{\overline{M[z_2 \leftarrow r_2^{\sigma_2}]}[z_1 \leftarrow r_1^{\sigma_1}]}) & \text{(implied by the definition of the descen-} \\
 & & \text{dant of a redex (Definition 3.26) and of} \\
 & & \text{the canonicalization procedure (Defini-} \\
 & & \text{tion 3.2))} \\
\equiv & \mathtt{GC}(\overline{\overline{M[z_1 \leftarrow r_1^{\sigma_1}]}[z_2 \leftarrow r_2^{\sigma_2}]}) & \text{(by Proposition 3.23)} \\
\equiv & \mathtt{GC}(\overline{\overline{M[z_1 \leftarrow r_1^{\sigma_1}]}[z_2 \leftarrow r_2^{\sigma_2'}]}) & \text{(Definition 3.26 and 3.2)} \\
\equiv & \mathtt{GC}(\overline{\mathtt{GC}(\overline{M[z_1 \leftarrow r_1^{\sigma_1}]})[z_2 \leftarrow r_2^{\sigma_2'}]}) & \text{(by Proposition 3.24)} \\
\equiv & \mathtt{GC}(\overline{M_1[z_2 \leftarrow r_2^{\sigma_2'}]}) &
\end{array}$$

2: $\rho_1 \setminus \rho_2 = (\tau_1, z_1, \sigma_1')$ and $\rho_2 \setminus \rho_1$ does not exist.
We need to show

$$M_1 \equiv \mathtt{GC}(\overline{M_2[z_1 \leftarrow r_1^{\sigma_1'}]})$$

We have

$$\begin{array}{rll}
 & \mathtt{GC}(\overline{M_2[z_1 \leftarrow r_1^{\sigma_1'}]}) & \\
\equiv & \mathtt{GC}(\overline{M[z_1 \leftarrow r_1^{\sigma_1}][z_2 \leftarrow r_2^{\sigma_2}]}) & \text{(following the argument in Case 1)} \\
\equiv & \mathtt{GC}(\overline{\overline{M[z_1 \leftarrow r_1^{\sigma_1}]}[z_2 \leftarrow r_2^{\sigma_2'}]}) & \\
\equiv & \mathtt{GC}(\mathtt{GC}(\overline{M[z_1 \leftarrow r_1^{\sigma_1}]})) & \text{(since } z_2 \text{ is not reachable in } \overline{M[z_1 \leftarrow r_1^{\sigma_1}]}\text{)} \\
\equiv & \mathtt{GC}(\overline{M[z_1 \leftarrow r_1^{\sigma_1}]}) & \\
\equiv & M_1 &
\end{array}$$

3: $\rho_2 \setminus \rho_1 = (\tau_2, z_2, \sigma_2')$ and $\rho_1 \setminus \rho_2$ does not exist.
The same as Case 2.
4: Both $\rho_2 \setminus \rho_1$ and $\rho_1 \setminus \rho_2$ do not exist.
If $\rho_1 \setminus \rho_2$ does not exist then there must be a path from z_2 to z_1 in M. Analogously, if $\rho_2 \setminus \rho_1$ does not exist then there must be a path from z_1 to z_2 in M. Therefore, it must be the case that there exists a cycle involving z_1 and z_2. Without loss of generality, let us assume there exists a path from the root of M to z_1 without visiting z_2. This implies that independently of the reduction of ρ_2, $z_1 \in \mathtt{BV}(M_2)$, that is, $\rho_1 \setminus \rho_2$ exists in M_2. We reach a contradiction, therefore, this case cannot arise.

□

Corollary 4.9. *A GRS_{NI} is confluent up to α-equivalence.*

Consider the following projection rules:

$$\begin{array}{rlcl}
\tau_1 : & x = \mathsf{I}(y) & \longrightarrow & x = y \\
\tau_2 : & x = \mathsf{J}(y) & \longrightarrow & x = y
\end{array}$$

and the following term M:

$$\begin{array}{l}
\{ \; x = \mathsf{I}(y); \\
\quad y = \mathsf{J}(x); \\
\quad \mathsf{In}\ x \}
\end{array}$$

then M will have the following reduction:

$$\begin{tikzcd}
 & M \arrow[dl] \arrow[dr] & \\
M_1 \equiv \{x = \mathsf{I}(x);\ \mathsf{In}\ x\} \arrow[dr] & & M_2 \equiv \{x = \mathsf{J}(x);\ \mathsf{In}\ x\} \arrow[dl] \\
 & \odot &
\end{tikzcd}$$

Notice that if both M_1 and M_2 are not reduced to $\odot$, the confluence property will be lost, as was observed in [15]. Barendregt's graph reduction system is not confluent precisely because of the absence of such a reduction.

Hereafter, we will use the notation GRS_{C} to denote a confluent GRS.

5. A Graph Model for GRS

We are interested in defining an equality on the set of terms, such that the equality is useful in analyzing the correctness of compiler optimizations. If we want equal terms to be freely substitutable for each other, then the equality must be preserved by terms formed by putting equal terms in the same context, which is defined as follows.

DEFINITION 5.1. (CONTEXT) *Let the definition of a GRS term (Definition 3.1) be extended to include a special symbol* $\square$. *A context is then a GRS term containing* $\square$.

NOTATION: *We write* $C[\square]$ *for an arbitrary context, such as* $\{x_1 = e_1; \cdots x_n = e_n;\ z = \square;\ \mathsf{In}\ x\}$, *and* $C[M]$ *for the term* $\{x_1 = e_1; \cdots x_n = e_n;\ z = \square;\ \mathsf{In}\ x\}[z \leftarrow M] \equiv \{x_1 = e_1; \cdots x_n = e_n;\ z = M;\ \mathsf{In}\ x\}$. As in any replacement operation $\mathsf{BV}(M) \cap \mathsf{BV}(C[\square]) = \emptyset$, for $C[M]$ to be a legal term. Observe that the free variables of a term M can get bound in $C[M]$.

For a compiler a useful equality must have the property $M = N \Longrightarrow \forall C[\square], C[M] = C[N]$. This means that the equality has to be a *congruence* with respect to the formation rules of terms. An optimization will be considered correct if it preserves equality.

An example of an equivalence relation is *convertibility.* If two terms M_1 and M_2 are convertible and the GRS is confluent, then it follows that there will not be any context that can distinguish between them. Thus, convertibility is a congruence. Therefore, independent of the meaning or observations we associate to a term, a minimal requirement that will have to be satisfied is that all terms in the set $\{M' \mid M \twoheadrightarrow M'\}$ have the same meaning. It follows that all optimization rules drawn from the set of rewriting rules will be automatically meaning preserving. However, as pointed out in [20], convertibility makes too fine a distinction to be interesting; it does not capture the computational behavior of a term. For example, consider the following two rules:

$$\tau_1 :\ x = \mathsf{F}(y) \longrightarrow x = \mathsf{F}(y) \qquad\qquad \tau_2 :\ x = \mathsf{G}(y) \longrightarrow x = \{t = \mathsf{G}(t_1); t_1 = \mathsf{G}(y);\ \mathsf{In}\ t\}$$

and the terms

$$M \equiv \{y = \mathsf{F}(z);\ \mathsf{In}\ y\} \qquad\qquad N \equiv \{y = \mathsf{G}(z);\ \mathsf{In}\ y\}\ .$$

M is not convertible to N or vice versa. Yet from a computational point of view, we would like to consider them as producing no information. We may be tempted to extend convertibility by equating all terms without normal form. However, it has been shown by Wadsworth in [20] that this will lead to an inconsistent theory. For example, if terms M and N do not have normal forms, then both the following terms in SK-combinatory logic:

$$\begin{aligned} M_1 \equiv \{\ & t = \mathsf{Ap}(x, \mathsf{K}); \\ & t_1 = M; \\ & t_2 = \mathsf{Ap}(t, t_1); \\ & \mathsf{In}\ t_2\} \end{aligned} \qquad\qquad \begin{aligned} N_1 \equiv \{\ & t = \mathsf{Ap}(x, \mathsf{S}); \\ & t_1 = N; \\ & t_2 = \mathsf{Ap}(t, t_1); \\ & \mathsf{In}\ t_2\} \end{aligned}$$

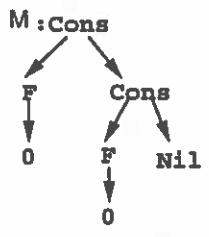
will not have normal forms either, and thus, will be equated. However, by plugging both of them in the context $\{x = \mathsf{K};\ z = \Box;\ \mathsf{In}\ z\}$ we can derive $\mathsf{K} = \mathsf{S}$. As shown in [6], this will immediately lead to an inconsistent theory, and will cause all terms to be equated.

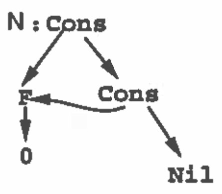
The notion of *head normal form* (hnf) was introduced by Wadsworth to syntactically characterize the class of terms that cannot be equated. Intuitively, a term does not have a head normal form if no information can be extracted by reducing that term in any context, that is, the term is totally undefined. For example, the two terms M and N, introduced earlier, are totally undefined, while the above terms M_1 and N_1 have some information contained in them. For example, we know that whichever term M_1 reduces to will have a stable prefix of the form: $\{\ t = \mathsf{Ap}(x, \mathsf{K}); t_1 = \Box; t_2 = \mathsf{Ap}(t, t_1);\ \ \mathsf{In}\ t_2\}$. Similarly for N_1, $\{t = \mathsf{Ap}(x, \mathsf{S}); t_1 = \Box; t_2 = \mathsf{Ap}(t, t_1);\ \mathsf{In}\ t_2\}$ constitutes a stable prefix. Informally the hnf of a term embodies its maximum stable prefix. Since the hnf's of M_1 and N_1 are different they cannot be equated. Furthermore, if two terms M and N do not contain any information, *i.e.*, do not have hnf's, it intuitively implies that the terms $C[M]$ and $C[N]$ will exhibit the same behavior and thus, can be equated. In this manner, we have performed a further classification of the terms without normal forms into ones that contain some information and those which contain no information. It is therefore legitimate to ask which terms in head normal forms can be equated. Convertibility may still be too restrictive, as shown by the following example, where Cons is the usual list contructor:

$$M \equiv \{\ x = \mathsf{Cons}(y, z);\quad y = \mathsf{F}(0);\quad z = \mathsf{Cons}(w, \mathsf{Nil});\quad w = \mathsf{F}(0);\quad \mathsf{In}\ x\}$$

$$N \equiv \{\ x = \mathsf{Cons}(y, z);\quad y = \mathsf{F}(0);\quad z = \mathsf{Cons}(y, \mathsf{Nil});\quad \mathsf{In}\ x\}$$

Graphs for M and N may be drawn as follows:





M and N are in normal form but clearly not inter-convertible. However, if internal representation of lists is ignored by an observer then both the terms represent the same unfolded list, $\mathsf{F}(0) : \mathsf{F}(0) : \mathsf{Nil}$. If the GRS containing these terms has a non-left-linear rule, it may be possible to distinguish between such terms. Thus, such terms cannot be equated without disallowing non-left-linear rules.

We should also notice that $M \leq_\omega N$, *i.e.*, M has "less sharing" than N in the above example. Does it mean that M is "less defined" than N in the sense that one can *compute less* with M than with N?. We would like to answer the above question without delving into heavy duty model theory. We have carefully said "compute" to emphasize that we are interested in studying what a term represents from an operational point of view. In particular, we are interested in observing the gradual *syntactic building up of the final term.*

We introduce a function ω to compute the stable part of a term, that is, the part of the term that will not change as more reductions are performed on it. The ω function captures what Lévy has called the ***direct approximation*** of a λ-calculus term [18], and Welch has called the ***instantaneous semantics*** of a term [21]. Take the following term:

$$\begin{aligned} P \equiv \{\; & t = \mathsf{Ap}(\mathsf{Y}, f); \\ & \mathsf{In}\ t\} \end{aligned}$$

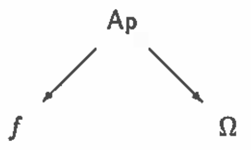
where the rule for the Y combinator is expressed as

$$x = \mathsf{Ap}(\mathsf{Y}, f) \longrightarrow x = \{t = \mathsf{Ap}(f, t_1);\ t_1 = \mathsf{Ap}(\mathsf{Y}, f);\ \mathsf{In}\ t\}\ .$$

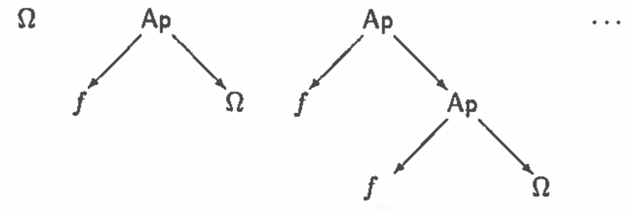
Since $\mathsf{Ap}(\mathsf{Y}, f)$ occurring in P is a redex, we cannot predict what $\mathsf{Ap}(\mathsf{Y}, f)$ will produce without reducing it. Thus, without any prior knowledge of the reduction, we will have to assert that no information is associated to P, otherwise, a wrong assumption may lead to a situation where we have to retract what was printed earlier. Thus, we will say that $\omega(P) = \Omega$, where Ω stands for no information. Let's perform one-step reduction on P to obtain the term:

$$\begin{aligned} P_1 \equiv \{\; & t = \mathsf{Ap}(f, t_1); \\ & t_1 = \mathsf{Ap}(\mathsf{Y}, f); \\ & \mathsf{In}\ t\}\ . \end{aligned}$$

At this point notice that all further reductions of P_1 will produce terms with context: $\{t = \mathsf{Ap}(f, \square);\ \mathsf{In}\ t\}$. Thus, we can safely say that $\omega(P_1)$ is



In general, we will have



Notice that as more reductions are performed the stable part should get larger, that is, $\omega(P) \leq_\omega \omega(P_1) \leq_\omega \omega(P_2) \cdots$. We remind the reader that $\leq_\omega$ is the syntactic ordering on terms which captures both the sharing and the fact that Ω is less than any other term. Assuming that this chain has a limit, then, even though the term P does not have a normal form, it may still have a precise meaning.

We collect all the (stable) information gathered by reducing P in a set, called $W^*(P)$, and say that it represents the ***information content*** of P. We can now formulate our original question regarding the impact of sharing on a program's behavior as follows: ***if M has less sharing than N then is $W^*(M)$ contained in $W^*(N)$***? As we shall see shortly, this is indeed the case in the absence of interfering rules.

It is also interesting to analyze if M "less defined" then N implies that for all context $C[\square]$, $C[M]$ is "less defined" than $C[N]$? That is, is the equality induced by W^* a ***congruence***? Later we will see that, in the absence of interfering rules, the equality introduced by information content is also a congruence. Thus, the collection of stable information contained in GRS terms does indeed turn out to be a model for GRS without interfering rules.

We will need the following definition shortly.

DEFINITION 5.2. (INITIAL Ω-FORM OF A TERM, M_Ω) *Given a GRS term M in canonical form, M_Ω is the term* $\mathsf{GC}(M[\Re(M) \leftarrow \Omega])$.

5.1. INSTANT SEMANTICS

The instant semantics of a GRS term M is obtained by computing its *stable part*, where stable part means the part of M which will not change by further reductions. A solution that comes to mind is to replace each redex in a term by an Ω, that is, to treat M_Ω as stable. Intuitively, this seems right because a redex can become any expression, and Ω is less than all expressions. Furthermore, it seems that if $M \rightarrow N$ then $M_\Omega \leq_\omega N_\Omega$. Unfortunately, this solution has a problem, as shown by the following example. Consider the rules

$$\tau_1 : x = \mathsf{F}(y, y) \longrightarrow x = 4 \qquad \tau_2 : x = \mathsf{I}(y) \longrightarrow x = y$$

and the following reduction:

$$\begin{array}{lcl} M \equiv \{\ t = \mathsf{F}(t_1, t_2); & \xrightarrow[t_2]{\tau_2} & M_1 \equiv \{\ t = \mathsf{F}(t_1, t_1); \\ \qquad t_1 = \mathsf{A}(0); & & \qquad t_1 = \mathsf{A}(0); \\ \qquad t_2 = \mathsf{I}(t_1); & & \qquad \mathsf{In}\ t\} \\ \qquad \mathsf{In}\ t\} & & \end{array}$$

The only redex in M is rooted at t_2, and thus, its stable part is the following term M_2:

$$\begin{tikzcd} & \mathsf{F} \arrow[dl] \arrow[dr] & \\ \mathsf{A} \arrow[d] & & \Omega \\ 0 & & \end{tikzcd}$$

However, since M_1 is a redex, M_1 does not contain any stable information. Thus, the information contained in M_1 is less than the information contained in M. This is contrary to our intuition that the information should not decrease with reduction. The problem is due to the presence of rule τ_2 which can introduce sharing. If we want to compute the instant semantics of a term without analyzing the rhs of rules then we have to assume that the arc pointing to Ω in M_2 can be redirected to point to the node of label A, and thus making M_2 a τ_1-redex. In other words, we should not treat M_2 as being in stable form. However, M_Ω does give the stable part of a term M in a Recursive Program Schema (RPS)!

5.1.1. Stable information in RPS and TRS

We remind the reader that a Recursive Program Schema (RPS) is a TRS which has a finite set of function symbols $\{F_1, \cdots, F_n\}$ (the "unknown" or user-defined functions) where F_i has arity m_i, and a disjoint set of basic function symbols $\{G_1, \cdots, G_m\}$ (the "known" functions), where G_j has arity l_j. The rewrite rules of a RPS have the form $\mathsf{F}_i^{m_i}(x_1, \cdots, x_{m_i}) \longrightarrow t_i, 1 \leq i \leq n$, where all variables $x_i, 1 \leq i \leq m_i$, are pairwise distinct and t_i is an arbitrary term built from variables and functions, $F_i, \cdots, G_j, \cdots$. Furthermore, for each F_i there is exactly one rule.

Lemma 5.3. *For any RPS term M, M_Ω is the stable prefix of M.*

Proof. If $M_\Omega = \Omega$ then M_Ω is obviously stable. If $M_\Omega \neq \Omega$ then $\forall N$ such that $M_\Omega \leq_\omega N$, N cannot be a redex at the root, in other words, M_Ω is stable. □

However, M_Ω is not necessarily the stable prefix for a TRS term M[†]. Consider the following rules:

$$\begin{aligned} \tau_1 :\ & \mathsf{A}(x) && \longrightarrow && \mathsf{B}(x) \\ \tau_2 :\ & \mathsf{F}(\mathsf{B}(x)) && \longrightarrow && 0 \end{aligned}$$

and the reduction

$$M \equiv \mathsf{F}(\mathsf{A}(x)) \longrightarrow N \equiv \mathsf{F}(\mathsf{B}(x))$$

M_Ω is $\mathsf{F}(\Omega)$, while N_Ω is Ω, and $M_\Omega \not\leq_\omega N_\Omega$. Thus, we erroneously assumed that $\mathsf{F}(\Omega)$ is a stable prefix. A TRS differs from a RPS in that even though the TRS term $F(\Omega)$ is not a redex it can become one by increasing its information, for example, by substituting $\mathsf{B}(x)$ for Ω. This phenomena is called the *upward creation of redexes.* Reduction of a term in the λ-calculus can also result in the upward creation of redexes. To cope with this problem in the λ-calculus, Wadsworth[20] and Lévy[18] had introduced the notion of an ω-rule, which reduces any term that can become a redex (by upward creation) to Ω. The ω-rule for the λ-calculus is

$$\Omega\, M \longrightarrow \Omega\ .$$

The ω-rule associated to τ_1 and τ_2 will be $\mathsf{F}(\Omega) \longrightarrow \Omega$. However, in the presence of non-left-linear rules it is difficult to generate ω-rules for a TRS as shown by the following example. We remind the reader that non-left-linearity in a TRS is interpreted as tree equivalence on terms. Now consider the TRS rule $\mathsf{F}(y, y) \longrightarrow y$. Suppose we generate the following ω-rules:

$$\mathsf{F}(\Omega, y) \longrightarrow \Omega$$
$$\mathsf{F}(y, \Omega) \longrightarrow \Omega$$

then the term $M \equiv \mathsf{F}(\mathsf{A}(0), \mathsf{A}(\Omega))$ will be in stable form. However, it should not be because by replacing Ω by 0, M becomes a redex.

[†] Private communication with Jean-Jacques Lévy.

5.1.2. ω-REDEX

Non-left-linear rules constitute a problem for a GRS also. Since we want to include them in our analysis, we abandon the idea of generating ω-rules, and instead introduce a new notion of redex, called *ω-redex* [11]. A ω-redex captures our intuition about why a term should be rewritten to Ω. It consists of analyzing a term to see if it can become a redex by either replacing Ω with some other term or by increasing the sharing in the term. The stable part of a term M will be computed by first replacing all redexes in M by Ω, and then by reducing all ω-redexes to Ω .

Notice that we have taken a conservative approach in determining which terms can become redexes; we may reduce to Ω some terms, which may never become a redex. However, we also need to guarantee that not too much information is lost. Since the stable part will be a part of our criteria for equating terms, we may end up equating far too many terms and lose congruence. Clearly if we reduce everything to Ω, all terms will get equated and we will immediately have an inconsistent model. As we will see in Section 5.6, a way of guaranteeing that not too much information is lost is by showing that the behavior of a term $C[M]$ can be inferred from the observations about M. Consider the rules

$$\tau_1: \quad x = \mathsf{F}(y,y) \longrightarrow x = 4$$

$$\tau_2: \quad \frac{x_1 = \mathsf{F}(0,1)}{x = \mathsf{G}(x_1) \longrightarrow x = 3}$$

and terms $M \equiv \mathsf{F}(0,1)$ and $N \equiv \mathsf{F}(2,1)$. Suppose we compute the stable information of term M by applying the (incorrect) rule: $x = \mathsf{F}(y_1, y_2) \longrightarrow x = \Omega$. Then both $\omega(M)$ and $\omega(N)$ will be Ω and as such they will be equated. Now consider the context $C[\Box] \equiv \{t_1 = \mathsf{G}(t_2);\ \ t_2 = \Box;\ \mathsf{In}\ t_1\}$. $C[M]$ produces 3 as a possible observation, while $C[N]$ does not. Thus, we erroneously equate terms which are not "extensionally equal". It seems that we cannot discard any information that can be used to build terms. By treating $\omega(M)$ as Ω we discard too much information as can be seen by the fact that by plugging the observations of M in $C[\Box]$ we will not be able to observe 3.

5.2. ω-REDUCTIONS AND ITS PROPERTIES

DEFINITION 5.4. (ω-REDEX) *A ω-redex in a GRS term M in canonical form is a pair (τ, z) such that*
(i) τ is a rule;
(ii) $z \in \mathtt{BV}(M)$ and z is not bound to Ω;
(iii) $Cl(\tau) \uparrow_\omega M@z$ and $Cl(\tau) \not\leq_\omega M@z$.
z is called the root of the ω-redex. If z is also the root of the term M then M is said to be a ω-redex. The set containing the roots of all ω-redexes is denoted by $\Re_\omega(M)$.

Notice that because of condition (*iii*), a ω-redex cannot be an ordinary redex. For the example given at the beginning of Section 5.1, we have that $Cl(\tau_1) \not\leq_\omega M_2$ and $Cl(\tau_1) \uparrow_\omega M_2$, thus, M_2 is a ω-redex, and will be reduced to Ω in computing the stable part of M_2.

DEFINITION 5.5. (DISTINCT ω-REDEXES) *Two ω-redexes $\rho_1 = (\tau_1, z_1)$ and $\rho_2 = (\tau_2, z_2)$ in a GRS term M in canonical form are said to be distinct if either $\tau_1 \neq \tau_2$ or $z_1 \neq z_2$.*

PROPOSITION 5.6. *Given GRS terms M_1 and M_2 in canonical form, if $M_1 \leq_\omega M_2$ then*
(i) if M_2 is a ω-redex then M_1 is either a ω-redex or Ω;
(ii) if M_2 is a redex then M_1 is either a redex, a ω-redex or Ω.

PROOF. (*i*) From the definition of ω-redex we have that $\exists\ \tau$ such that $M_2 \uparrow_\omega Cl(\tau)$. Therefore, if M_1 is not Ω it must be a ω-redex. Analogously for point (*ii*). □

DEFINITION 5.7. (ω-REDUCTION) *Given a GRS term M in canonical form, M ω-reduces to N by doing the τ ω-redex at z, written as $M \xrightarrow[z]{\tau}_\omega N$, iff (τ, z) is a ω-redex in M and $N \equiv \mathtt{GC}(M[z \leftarrow \Omega])$. A term is said to be in ω-normal form when it has no ω-redexes.*

Note that since Ω is not a substitutable value, $M[z \leftarrow \Omega]$ is in canonical form.

DEFINITION 5.8. (DESCENDANT OF A ω-REDEX WITH RESPECT TO ω-REDUCTION) *Given two distinct ω-redexes $\rho_1 = (\tau_1, z_1)$ and $\rho_2 = (\tau_2, z_2)$ in a GRS term M in canonical form, the descendant of ρ_2 with respect to the reduction $M \xrightarrow[\rho_1]{}_\omega M_1$ (written as $\rho_2 \setminus \rho_1$):*
(i) does not exist, if $z_2 \notin \mathtt{BV}(M_1)$;
(ii) is the pair $\rho_2' = (\tau_2, z_2)$, if $z_2 \in \mathtt{BV}(M_1)$.

The following proposition shows that, given two distinct ω-redexes ρ_1 and ρ_2, even in the presence of interfering rules $\rho_2 \setminus \rho_1$ is an ω-redex if it exists.

PROPOSITION 5.9. *Given ω-redexes $\rho_1 = (\tau_1, z_1)$ and $\rho_2 = (\tau_2, z_2)$ in a GRS term M in canonical form, if $M \xrightarrow[\rho_1]{}_\omega M_1$ and $z_2 \in \mathtt{BV}(M_1)$ then either $\rho_2 \setminus \rho_1$ is an ω-redex or z_2 is bound to Ω in M_1.*

PROOF. Since $M_1 \equiv \mathtt{GC}(M[z_1 \leftarrow \Omega])$, $M_1 \leq_\omega M$. Therefore, if $z_2 \in \mathtt{BV}(M_1)$ then $M_1@z_2 \leq_\omega M@z_2$. Therefore, by Proposition 5.6, $\rho_2 \setminus \rho_1$ is either a ω-redex or Ω. □

The stable part of a term M, *i.e.*, $\omega(M)$, will then be computed by first replacing all distinct redexes occurring in M by Ω and then computing the ω-normal form of the term so obtained. Before giving the formal definition of the ω-function we introduce some properties of ω-redexes and ω-reduction.

THEOREM 5.10. (SUBCOMMUTATIVITY OF $\longrightarrow_\omega$) *Given two distinct ω-redexes ρ_1 and ρ_2 in a GRS term M in canonical form, if $M \xrightarrow[\rho_1]{}_\omega M_1$ and $M \xrightarrow[\rho_2]{}_\omega M_2$, then $\exists\ M_3$ such that $M_2 \xrightarrow[\rho_1\setminus\rho_2]{0/1}_\omega M_3$ and $M_1 \xrightarrow[\rho_2\setminus\rho_1]{0/1}_\omega M_3$.*

PROOF. Let $\rho_1 = (\tau_1, z_1)$, $\rho_2 = (\tau_2, z_2)$, and $M_1 \equiv \mathtt{GC}(M[z_1 \leftarrow \Omega])$, $M_2 \equiv \mathtt{GC}(M[z_2 \leftarrow \Omega])$. If $z_1 \equiv z_2$ then $M_1 \equiv M_2 \equiv M_3$. If $z_1 \not\equiv z_2$ then the proof, analogous to Theorem 4.8, is by cases on the existence of $\rho_1 \setminus \rho_2$ and $\rho_2 \setminus \rho_1$. We show the first two cases only.

1: $\rho_2 \setminus \rho_1 = (\tau_2, z_2)$ and $\rho_1 \setminus \rho_2 = (\tau_1, z_1)$.
By Proposition 5.9 $\rho_1 \setminus \rho_2$ and $\rho_2 \setminus \rho_1$ must be ω-redexes. Thus, we want to show the following:

$$\mathtt{GC}(M_1[z_2 \leftarrow \Omega]) \equiv \mathtt{GC}(M_2[z_1 \leftarrow \Omega]).$$

We have

$$\begin{array}{lll} & \texttt{GC}(M_1[z_2 \leftarrow \Omega]) & \\ \equiv & \texttt{GC}(\texttt{GC}(M[z_1 \leftarrow \Omega])[z_2 \leftarrow \Omega]) & \text{(by Proposition 3.24)} \\ \equiv & \texttt{GC}(M[z_1 \leftarrow \Omega][z_2 \leftarrow \Omega]) & \text{(by Proposition 3.23)} \\ \equiv & \texttt{GC}(M[z_2 \leftarrow \Omega][z_1 \leftarrow \Omega]) & \text{(by Proposition 3.24)} \\ \equiv & \texttt{GC}(\texttt{GC}(M[z_2 \leftarrow \Omega])[z_1 \leftarrow \Omega]) & \\ \equiv & \texttt{GC}(M_2[z_1 \leftarrow \Omega]) & \end{array}$$

2: $\rho_1 \setminus \rho_2 = (\tau_1, z_1)$ and $\rho_2 \setminus \rho_1$ does not exist.
We need to show

$$M_1 \equiv \texttt{GC}(M_2[z_1 \leftarrow \Omega]) \ .$$

We have

$$\begin{array}{lll} & \texttt{GC}(M_2[z_1 \leftarrow \Omega]) & \\ \equiv & \texttt{GC}(\texttt{GC}(M[z_2 \leftarrow \Omega])[z_1 \leftarrow \Omega]) & \\ \equiv & \texttt{GC}(\texttt{GC}(M[z_1 \leftarrow \Omega])[z_2 \leftarrow \Omega]) & \text{(following the argument in Case 1)} \\ \equiv & \texttt{GC}(M[z_1 \leftarrow \Omega]) & \text{(since } z_2 \notin M_1 \text{ and GC is idempotent)} \\ \equiv & M_1 & \end{array}$$

□

COROLLARY 5.11. $\twoheadrightarrow_\omega$ *is confluent.*

PROPOSITION 5.12. $\twoheadrightarrow_\omega$ *is strongly normalizing.*

PROOF. Suppose there are n function symbols in a term M. Each ω-reduction gets rid off one function symbol and does not introduce any new ones. Thus, the lenght of ω-reductions on M is bounded by n. □

Since ω-reductions are strongly normalizing and confluent we can now define the ω-function as follows.

DEFINITION 5.13. (ω-FUNCTION) *Given a GRS term M, $\omega(M)$ is the ω-normal form of M_Ω.*

5.3. INTERACTION OF ω-REDUCTIONS AND NORMAL REDUCTIONS

In the following, we analyze the effect of ω-reductions and normal reductions on ω-redexes and redexes, respectively.

DEFINITION 5.14. (DESCENDANT OF A REDEX WITH RESPECT TO ω-REDUCTION) *Given an ω-redex $\rho_1 = (\tau_1, z_1)$ and a redex $\rho_2 = (\tau_2, z_2, \sigma_2)$ in a GRS term M in canonical form, the descendant of ρ_2 with respect to the reduction $M \underset{\rho_1}{\longrightarrow}_\omega M_1$ (written as $\rho_2 \setminus \rho_1$):*
(i) does not exist, if $z_2 \notin \texttt{BV}(M_1)$;

(ii) is the triple $\rho'_2 = (\tau_2, z_2, \sigma'_2)$, if $z_2 \in \mathtt{BV}(M_1)$, and σ'_2 is:

$$\sigma'_2(x) = \begin{cases} \sigma_2(x) & \sigma_2(x) \neq z_1 \\ \Omega & \sigma_2(x) = z_1 \end{cases}$$

The following proposition shows that an ω-reduction cannot destroy a redex in the absence of interfering rules.

PROPOSITION 5.15. *Given an ω-redex $\rho_1 = (\tau_1, z_1)$ and a redex $\rho_2 = (\tau_2, z_2, \sigma_2)$ in a GRS_{NI} term M in canonical form, if $M \xrightarrow[\rho_1]{}_\omega M_1$ and $z_2 \in \mathtt{BV}(M_1)$ then $\rho_2 \setminus \rho_1$ is a redex.*

PROOF. If $z_2 \in \mathtt{BV}(M_1)$, let $\rho_2 \setminus \rho_1 = (\tau_2, z_2, \sigma'_2)$. If $\exists x, \sigma_2(x) \neq \sigma'_2(x)$, by non-interference it must be the case that $x \notin \mathtt{BV}(\mathcal{P}(\tau_2))$. Moreover, suppose $\exists x \in \mathtt{BV}(\mathcal{P}(\tau_2)), \sigma'_2(x)$ does not occur in M_1, it means that all paths from z_2 to $\sigma'_2(x)$ go through z_1. This means that $z_1 \in \{\sigma_2(y) \mid y \in \mathtt{BV}(\mathcal{P}(\tau_2))\}$. We reached a contradiction; therefore, $\forall x \in \mathtt{BV}(\mathcal{P}(\tau_2)), \sigma'_2(x)$ occurs in M_1. Since the new definition of z_1 and the garbage collection do not affect the substitution σ'_2, $\rho_2 \setminus \rho_1$ is a redex. □

DEFINITION 5.16. (DESCENDANT OF AN ω-REDEX WITH RESPECT TO REDUCTION) *Given a redex ρ_1 and an ω-redex $\rho_2 = (\tau_2, z_2)$ in a GRS term M in canonical form, the descendant of ρ_2 with respect to the reduction $M \xrightarrow[\rho_1]{} M_1$ (written as $\rho_2 \setminus \rho_1$):*
(i) does not exist, if $z_2 \notin \mathtt{BV}(M_1)$;
(ii) is the pair $\rho'_2 = (\tau_2, z_2)$, if $z_2 \in \mathtt{BV}(M_1)$.

A reduction, however, can destroy an ω-redex even in the absence of interfering rules, as shown in the example below. Consider the rules

$$\begin{array}{lll} \tau_1: & x = \mathsf{F}(y,y,y) & \longrightarrow \quad x = y \\ \tau_2: & x = \mathsf{G}(0) & \longrightarrow \quad x = 1 \end{array}$$

and the reduction

$$\begin{array}{llll} M \equiv \{ & t_1 = \mathsf{F}(t_2, t_3, t_4); & \xrightarrow[t_2]{} M_1 \equiv \{ & t_1 = \mathsf{F}(1, t_3, t_4); \\ & t_2 = \mathsf{G}(0); & & t_3 = \mathsf{G}(\Omega); \\ & t_3 = \mathsf{G}(\Omega); & & t_4 = \mathsf{G}(0); \\ & t_4 = \mathsf{G}(0); & & \mathsf{In}\ t_1\} \\ & \mathsf{In}\ t_1\} & & \end{array}$$

Note that the ω-redex rooted at t_1 in M is no longer an ω-redex in M_1. However, it can get restored by reducing the redex rooted at t_4 and the ω-redex rooted at t_3. Equivalently, by setting t_3 and t_4 to Ω, respectively.

Before stating the property that ω-redexes can be restored we need the following proposition.

PROPOSITION 5.17. *Given GRS terms M and N in canonical form. If $M \leq_\omega N$, let $\rho = (\tau, z, \sigma)$ be a redex in M and $\rho_1 = (\tau, z_1, \sigma_1)$ be a redex in N such that $\sigma_{MN}(z) = z_1$ then*

$$\mathtt{GC}(\overline{M[z \leftarrow \mathcal{RHS}(\tau)^\sigma][\mathcal{Z} \leftarrow \Omega]}) \leq_\omega \mathtt{GC}(\overline{N[z_1 \leftarrow \mathcal{RHS}(\tau)^{\sigma_1}]})$$

where $\mathcal{Z} = \{s \mid \sigma_{MN}(s) = z_1, s \not\equiv z\}$.

Proof. We first show that the function σ_{MN} restricted to the domain of variables occurring in $\mathtt{GC}(M[z \leftarrow \Omega][\mathcal{Z} \leftarrow \Omega])$ induces the ordering:

$$\mathtt{GC}(M[z \leftarrow \Omega][\mathcal{Z} \leftarrow \Omega]) \leq_\omega \mathtt{GC}(\overline{N[z_1 \leftarrow \mathcal{RHS}(\tau)^{\sigma_1}]}) \ .$$

Suppose it is not, then it must be the case that for an element, say x, occurring in the left-hand side term, $\sigma_{MN}(x)$ does not occur in the right-hand side term. This means that the replacement of the term bound to z_1 in N by $\mathcal{RHS}(\tau)^{\sigma_1}$ has removed the path to $\sigma_{MN}(x)$. However, this means that the corresponding path in M also gets removed by setting z and all the elements of $\mathcal{Z}$ to Ω. Contradiction. Therefore it must be the case that $\sigma_{MN}(x)$ occurs in $\mathtt{GC}(\overline{N[z_1 \leftarrow \mathcal{RHS}(\tau)^{\sigma_1}]})$.

Moreover, for all elements x in $\mathtt{BV}(\mathcal{RHS}(\tau)^{\sigma})$, let x' be its renamed version. The following function:

$$\sigma'(x) = \begin{cases} \sigma_{MN}(x) & x \in \mathtt{GC}(M[z \leftarrow \Omega][\mathcal{Z} \leftarrow \Omega]) \\ x' & x \in \mathtt{BV}(\mathcal{RHS}(\tau)^{\sigma}) \end{cases}$$

determines the ordering:

$$\mathtt{GC}(\overline{M[z \leftarrow \mathcal{RHS}(\tau)^{\sigma}]}[\mathcal{Z} \leftarrow \Omega]) \leq_\omega \mathtt{GC}(\overline{N[z_1 \leftarrow \mathcal{RHS}(\tau)^{\sigma_1}]}) \ .$$

□

Proposition 5.18. *Given a redex $\rho_1 = (\tau_1, z_1, \sigma_1)$ and an ω-redex $\rho = (\tau, z)$ in a GRS_{NI} term M in canonical form, if $M \xrightarrow[\rho_1]{} M_1$ and $z \in \mathtt{BV}(M_1)$ then $\rho \setminus \rho_1$ is either an ω-redex or it can become so after having perfomed some ω-reductions on $M_{1\Omega}$.*

Proof. Since ρ is an ω-redex, $\exists N, M \leq_\omega N$ such that $\sigma_{MN}(z)$ is the root of a τ-redex in N. Let $z_1' = \sigma_{MN}(z_1)$ be the root of the redex $(\tau_1, z_1', \sigma_1')$, and let N_1 be such that $N \xrightarrow[z_1']{} N_1$. Let $\mathcal{Z} = \{z' \mid \sigma_{MN}(z') = z_1', z'$ is not bound to $\Omega, z' \not\equiv z_1\}$. By Proposition 5.17 we have

$$\mathtt{GC}(M_1[\mathcal{Z} \leftarrow \Omega]) \leq_\omega \mathtt{GC}(\overline{N[z_1' \leftarrow \mathcal{RHS}(\tau_1^{\sigma_1'})]}) \equiv N_1$$

Moreover, by Proposition 4.7 the descendant of the redex rooted at $\sigma_{MN}(z)$ in N must be a redex in N_1. Therefore, by Proposition 5.6 we have that $\rho \setminus \rho_1$ must be either a redex, an ω-redex or Ω in $\mathtt{GC}(M_1[\mathcal{Z} \leftarrow \Omega])$. By non-interference it follows that the only condition applicable is for $\rho \setminus \rho_1$ to be an ω-redex in $\mathtt{GC}(M_1[\mathcal{Z} \leftarrow \Omega])$. The result then follow from the fact that $\mathcal{Z} \subseteq \Re(M) \cup \Re_\omega(M)$. □

We now discuss two additional properties of the ω-function, that are, its monotonicity with respect to $\leq_\omega$, and with respect to reduction. These will be very useful in later proofs.

Proposition 5.19. (Monotonicity of ω-function with respect to $\leq_\omega$) *Given GRS terms M and N in canonical form, if $M \leq_\omega N$ then $\omega(M) \leq_\omega \omega(N)$.*

Proof. Let $\mathcal{Z} = \{z \mid \sigma_{MN}(z) \in \Re(N) \cup \Re_\omega(N), z$ is not bound to $\Omega\}$. From Proposition 5.17,

$$\mathtt{GC}(M[\mathcal{Z} \leftarrow \Omega]) \leq_\omega \omega(N_\Omega) = \omega(N) \ .$$

By Proposition 5.6, $\mathcal{Z} \subseteq (\Re(M) \cup \Re_\omega(M))$. Therefore, we have

$$\omega(M) \leq_\omega \mathtt{GC}(M[\mathcal{Z} \leftarrow \Omega]) \leq_\omega \omega(N).$$

□

Proposition 5.20. (Monotonicity of ω-function with respect to reduction) *Given GRS terms M and N in canonical form, if $M \twoheadrightarrow N$ then $\omega(M) \leq_\omega \omega(N)$.*

Proof. The proof is by induction on the number of reduction steps. Suppose $M \xrightarrow[\rho]{} N$, where $\rho = (\tau, z, \sigma)$, then $N \equiv \mathtt{GC}(\overline{M[z \leftarrow P]})$, for some term P. Since $\omega(M) = \omega(\mathtt{GC}(M[z \leftarrow \Omega]))$ and $\mathtt{GC}(M[z \leftarrow \Omega]) \leq_\omega \mathtt{GC}(\overline{M[z \leftarrow P]})$, then from the monotonicity of the ω-function with respect to $\leq_\omega$ (Proposition 5.19) we have that $\omega(M) \leq_\omega \omega(N)$. The result then follows by induction. □

5.4. Meaning of a GRS term

We collect all observable information about GRS terms in a set called ω-graphs.

Definition 5.21. (ω-graphs: Set of observations) *Given a GRS, the set of all observations, called ω-graphs, is defined as:*

$$\omega\text{-}graphs = \{\omega(M) \mid \forall \text{ GRS terms } M\}.$$

We would like to define the meaning of a term M as the limit of the set of our observations, that is, $\{\omega(M') \mid M \twoheadrightarrow M'\}$. However, *$\omega$-graphs* does not guarantee the existence of such a limit. Therefore, we apply the *ideal completion* method to turn *ω-graphs* into a complete partial order [10]. Given a partial order $(A, \leq)$, a subset D of A is an *ideal* iff (i) D is non-empty. (ii) $\forall a, b \in D, \exists c \in D, a \leq c$ and $b \leq c$, (iii) $\forall c \in D$, if $\exists d \in A, d \leq c$ then $d \in D$.

Definition 5.22. (ω-graphs$^\infty$: Domain of observations) *Given a GRS, the domain of observations, called ω-graphs$^\infty$, is the ideal completion of ω-graphs, that is:*

$$\omega\text{-}graphs^\infty = \{I \mid I \subseteq \omega\text{-}graphs \text{ and } I \text{ is an ideal }\}.$$

To reflect this change in our domain of observations, we include all *ω-graph* terms smaller than $\omega(M)$ in our observations as follows.

Definition 5.23. (W^*: The information content of a GRS term) *Given a GRS term M in canonical form, $W^*(M) = \{a \mid a \in \omega\text{-}graphs, a \leq_\omega \omega(M'), M \twoheadrightarrow M'\}$.*

We need to define under what conditions $W^*(M)$ is an element of *ω-graphs$^\infty$*, that is, $W^*(M)$ is an ideal.

Proposition 5.24. ($W^*(M)$ is an ideal for $\mathrm{GRS}_\mathbb{C}$) *For all $\mathrm{GRS}_\mathbb{C}$ terms M, $W^*(M)$ is in ω-graphs$^\infty$.*

Proof.

(i) $W^*(M)$ is not empty since it contains at least Ω;

(ii) We show that $W^*(M)$ is a directed set, that is, $\forall a, b \in W^*(M)$, $\exists c$, such that $a \leq_\omega c$ and $b \leq_\omega c$. If $a \in W^*(M)$, it means that $\exists M_1$, such that $M \twoheadrightarrow M_1$ and $a \leq_\omega \omega(M_1)$. Analogously, $\exists M_2, M \twoheadrightarrow M_2$ and $b \leq_\omega \omega(M_2)$. Follows from the confluence of $\twoheadrightarrow$ that $\exists M_3$, $M_1 \twoheadrightarrow M_3$ and $M_2 \twoheadrightarrow M_3$. Moreover, from the monotonicity of the ω-function with respect to $\twoheadrightarrow$ (Proposition 5.20) we have

$$a \leq_\omega \omega(M_3) \text{ and } b \leq_\omega \omega(M_3)$$

that is, $\omega(M_3)$ is the element we were looking for.

(iii) $W^*(M)$ is closed downwards by definition.

□

Let's now define a semantic ordering based on the information content.

Definition 5.25. ($\sqsubseteq_{\mathbf{g}}$: Information ordering) *Given GRS terms M and N in canonical form, $M \sqsubseteq_{\mathbf{g}} N$ iff $W^*(M) \subseteq W^*(N)$. If $M \sqsubseteq_{\mathbf{g}} N$ and $N \sqsubseteq_{\mathbf{g}} M$ then $M \equiv_{\mathbf{g}} N$.*

If we want W^* to be our interpretation function, W^* will have to satisfy some properties, that is, the meaning will have to be preserved by reduction, and it will have to be compositional. In other words,

$$\begin{array}{llll} \textbf{Soundness}: & M \twoheadrightarrow N & \Longrightarrow & M \equiv_{\mathbf{g}} N \\ \textbf{Congruence}: & M \equiv_{\mathbf{g}} N & \Longrightarrow & C[M] \equiv_{\mathbf{g}} C[N] \ . \end{array}$$

We will first show the soundness, and then later come back to a discussion of congruence after a digression on the impact of sharing on the meaning function.

Proposition 5.26. (Soundness of $\equiv_{\mathbf{g}}$) *Given $GRS_{\mathcal{C}}$ terms M and N in canonical form, if $M \twoheadrightarrow N$ then $M \equiv_{\mathbf{g}} N$.*

Proof. Trivially, $N \sqsubseteq_{\mathbf{g}} M$. Next, we prove that $M \sqsubseteq_{\mathbf{g}} N$. Let $P \in W^*(M)$, it means that $\exists M_1, M \twoheadrightarrow M_1$ and $P \leq_\omega \omega(M_1)$. From the confluence of $\twoheadrightarrow$ we know that $\exists M_2, M_1 \twoheadrightarrow M_2$ and $N \twoheadrightarrow M_2$. From the monotonicity of the ω-function with respect to reduction (Proposition 5.20), we have that $P \leq_\omega \omega(M_1) \leq_\omega \omega(M_2)$. Since $\omega(M_2) \in W^*(N)$, then $P \in W^*(N)$ as well. □

5.5. Impact of sharing on program behavior

What is the impact of sharing on program behavior, that is,

$$M \leq_\omega N \Longrightarrow M \sqsubseteq_{\mathbf{g}} N \ ?$$

It turns out that in the presence of interfering rules the above will not hold. Consider the following GRS which has *confluent* but interfering rules:

$$\tau_1: \quad \frac{x_1 = \mathsf{B}(0);\ x_2 = \mathsf{C}(0)}{x = \mathsf{A}(x_1, x_2) \longrightarrow x = 0}$$

$$\tau_2: \quad x = \mathsf{C}(0) \longrightarrow x = \mathsf{B}(0)$$

$$\tau_3: \quad x = \mathsf{B}(0) \longrightarrow x = \mathsf{C}(0)$$

and the following terms:

$$M \equiv \{\ t_1 = \mathsf{A}(t_2, t_3);\ t_2 = \mathsf{B}(0);\ t_3 = \mathsf{B}(0);\ \mathsf{In}\ t_1\} \qquad N \equiv \{\ t_1 = \mathsf{A}(t_2, t_2);\ t_2 = \mathsf{B}(0);\ \mathsf{In}\ t_1\}$$

then, it is easy to see that $W^*(M) = \{\Omega, 0\}$, while $W^*(N) = \{\Omega, \{x_1 = \Omega;\ x_2 = \mathsf{A}(x_1, x_1);\ \mathsf{In}\ x_2\}\}$. Therefore, $M \leq_\omega N$ and $M \not\sqsubseteq_{\mathsf{g}} N$. However, there is no such problem for a GRS without interfering rules.

THEOREM 5.27. (MONOTONICITY OF $\sqsubseteq_{\mathsf{g}}$ WITH RESPECT TO $\leq_\omega$) *Given GRS_{NI} terms M and N in canonical form, if $M \leq_\omega N$ then $M \sqsubseteq_{\mathsf{g}} N$.*

PROOF. We prove that if $M \twoheadrightarrow M'$, then $\exists N'$ such that $N \twoheadrightarrow N'$ and $\omega(M') \leq_\omega \omega(N')$. The proof is by induction on the number of reduction steps of $M \twoheadrightarrow M'$.

Let $M \xrightarrow[\rho]{} M_1$, where $\rho = (\tau, z, \sigma)$. Then $\rho' = (\tau, z', \sigma')$, where $z' = \sigma_{MN}(z)$, and $\sigma' = \sigma_{MN} \circ \sigma$, is a redex in N. Let $N \xrightarrow[\rho']{} N_1$, and $\mathcal{Z} = \{s \mid \sigma_{MN}(s) = z', s \text{ not bound to } \Omega, s \not\equiv z\}$. That is, $\mathcal{Z}$ is the set of all variables in M, distinct from z, which are mapped into z'. From Proposition 5.17,

$$\mathsf{GC}(\overline{M[z \leftarrow \mathcal{RHS}(\tau)^\sigma]}[\mathcal{Z} \leftarrow \Omega]) \leq_\omega \mathsf{GC}(\overline{N[z' \leftarrow \mathcal{RHS}(\tau)^{\sigma'}]}) \equiv N_1.$$

By Proposition 5.6, $\mathcal{Z} \subseteq \{\Re(M) \cup \Re_\omega(M)\}$. Therefore,

$$\omega(M_1) = \omega(\mathsf{GC}(\overline{M[z \leftarrow \mathcal{RHS}(\tau)^\sigma]}[\mathcal{Z} \leftarrow \Omega])).$$

Therefore, by monotonicity of the ω-function with respect to $\leq_\omega$ (Proposition 5.19) we have

$$\omega(M_1) \leq_\omega \omega(N_1).$$

Notice that any redex in M_1, that does not belong to $\mathcal{Z}$, is a redex in $\mathsf{GC}(M_1[\mathcal{Z} \leftarrow \Omega])$ (Proposition 4.7), and therefore it must be a redex in N_1. The result then follows by induction. □

This theorem leads to a simple but powerful result regarding the behavior of terms obtained by plugging terms related by sharing into any context.

COROLLARY 5.28. *Given GRS_{NI} terms M and N, if $M \leq_\omega N$ then $\forall C[\Box]$, $C[M] \sqsubseteq_{\mathsf{g}} C[N]$.*

PROOF. It follows from the fact that if $M \leq_\omega N$ then $C[M] \leq_\omega C[N]$. □

It will be shown later this result can be applied in a straightforward manner to show the partial correctness of those compiler optimizations that simply increase the sharing in a term.

5.6. Congruence

Consider the rule

$$\tau: \; x = \mathsf{F}(y) \longrightarrow x = \{t = \mathsf{F}(y);\; t_1 = \mathsf{Cons}(y, t);\; \mathsf{In}\; t_1\}$$

and the following two terms:

$$M \equiv \{x = \mathsf{Cons}(1, x);\; \mathsf{In}\; x\} \qquad\qquad N \equiv \{x = \mathsf{F}(1);\; \mathsf{In}\; x\}\;.$$

Notice that $N \not\leq_\omega M$, but given the chain:

$$N \longrightarrow N_1 \longrightarrow N_2 \longrightarrow \cdots \longrightarrow N_i \longrightarrow \cdots$$

where $N_i \equiv \{t_0 = \mathsf{F}(1); t_1 = \mathsf{Cons}(1, t_0); \cdots; t_i = \mathsf{Cons}(1, t_{i-1});\; \mathsf{In}\; t_i\}$, we have that

$$\omega(N) \leq_\omega \omega(N_1) \leq_\omega \cdots \leq_\omega \omega(N_i) \leq_\omega \cdots \leq_\omega M.$$

In fact, using induction on i we can show that $N \sqsubseteq_{\mathbf{g}} M$, and one would correctly guess that $\forall C[\Box],\;\; C[N] \sqsubseteq_{\mathbf{g}} C[M]$. However, this does not follow from Corollary 5.28. We need to show that $\sqsubseteq_{\mathbf{g}}$ is a congruence relation. Note that terms like M and N arise in the cyclic implementation of recursive rules, such as the Y-rule.

A way of assuring that $\sqsubseteq_{\mathbf{g}}$ is a congruence is to show that the behavior of $C[M]$, for any context $C[\Box]$, can be inferred from the observations about M, that is,

$$\forall C[\Box],\;\; C[M] \equiv_{\mathbf{g}} \bigsqcup\{C[P] \mid P \in W^*(M)\}.$$

In other words, the context should be a *continuous* operation with respect to our observations. (Notice that $W^*(\bigsqcup\{C[P] \mid P \in W^*(M)\}) = \bigcup\{W^*(C[P]) \mid P \in W^*(M)\}$.)

We will first prove

$$\forall C[\Box],\;\; \bigsqcup\{C[P] \mid P \in W^*(M)\} \sqsubseteq_{\mathbf{g}} C[M]$$

that is, by plugging the observations of M in the context we do not produce more information than the one contained in $C[M]$.

Lemma 5.29. *Given a* GRS_{NI} *term* M *and context* $C[\Box]$, $\bigsqcup\{C[P] \mid P \in W^*(M)\} \sqsubseteq_{\mathbf{g}} C[M]$.

Proof. Suppose $M \twoheadrightarrow M'$, then $\forall P,\; P \leq_\omega \omega(M')$ we have

$$\begin{array}{lll} P & \leq_\omega M' & \\ C[P] & \sqsubseteq_{\mathbf{g}} C[M'] & \text{(by monotonicity of } \sqsubseteq_{\mathbf{g}} \text{ with respect to } \leq_\omega \text{ (Corollary 5.28))} \\ & \equiv_{\mathbf{g}} C[M] & \text{(by soundness of } \equiv_{\mathbf{g}} \text{ (Proposition 5.26))}\;. \end{array}$$

□

The other direction

$$\forall C[\Box],\;\; C[M] \sqsubseteq_{\mathbf{g}} \bigsqcup\{C[P] \mid P \in W^*(M)\}$$

requires some more machinery. We want to show that each observation of $C[M]$ can be obtained by first observing M and then plugging those observations instead of M in the context. The proof has two basic steps. First we show that each reduction sequence $C[M] \twoheadrightarrow N$ can be reordered, such that, if the sequence contains any redexes that are descendants of redexes in M then in the reordered sequence these redexes are performed before all others. Second, we show that the ω-function does not lose too much information. Since the ω-function involves two steps – first computing M_Ω and then applying ω-reductions – we will analyze the impact of these steps separately.

In the following, we will use the notation $M \;{}_{\mathcal{Z}(M)}\!\!\not\twoheadrightarrow N$, where $\mathcal{Z}(M) \subseteq \Re(M)$, to mean that no descendant of redexes in $\mathcal{Z}(M)$ is reduced.

Proposition 5.30. *Given a GRS_{NI} term M in canonical form, if $C[M] \twoheadrightarrow N$ then $\exists M'$, $M \twoheadrightarrow M'$ and $C[M'] \;{}_{\Re(M')}\!\!\not\twoheadrightarrow N$.*

Proof. Let $\mathcal{Z}(M)$ be the set of redexes in M contracted during the reduction $C[M] \twoheadrightarrow N$, that is, $\mathcal{Z}(M) = \{z \in \mathtt{BV}(M) \mid \exists i, C[M] \twoheadrightarrow N_i \xrightarrow[z]{} N_{i+1} \twoheadrightarrow N\}$. Then by the subcommutative property of $\mathrm{GRS}_{\mathsf{NI}}$ (Theorem 4.8), $\exists M', C[M] \xrightarrow[cd(\mathcal{Z}(M))]{} C[M']$ and $C[M'] \;{}_{\Re(M')}\!\!\not\twoheadrightarrow N$, where $cd(\mathcal{Z}(M))$ denotes a complete development with respect to $\mathcal{Z}(M)$. □

Proposition 5.31. *Given a GRS_{NI} term M in canonical form, if $M \;{}_{\mathcal{Z}(M)}\!\!\not\twoheadrightarrow N$ then $\mathtt{GC}(M[\mathcal{Z}(M) \leftarrow \Omega]) \twoheadrightarrow \mathtt{GC}(N[\mathcal{Z}(M) \leftarrow \Omega])$.*

Proof. Follows from Proposition 4.7. □

Proposition 5.32. *Given GRS_{NI} terms M and N, if $M \twoheadrightarrow_\omega N$ then $M \equiv_{\mathbf{g}} N$.*

Proof. Since $N \leq_\omega M$, and by monotonicity of $\sqsubseteq_{\mathbf{g}}$ with respect to $\leq_\omega$ (Theorem 5.27), we have that $N \sqsubseteq_{\mathbf{g}} M$. In order to prove the other direction let $M \twoheadrightarrow M_1$ by reducing redexes in $\mathcal{Z}$. From Proposition 5.15, ω-reductions do not destroy redexes, therefore, $\exists N_1, N \twoheadrightarrow N_1$ by reducing the descendants of redexes in $\mathcal{Z}$. The result then follows from Proposition 5.18. □

Lemma 5.33. *Given GRS_{NI} terms M and N, and context $C[\square]$, if $C[M] \;{}_{\Re(M)}\!\!\not\twoheadrightarrow N$ then $\omega(N) \sqsubseteq_{\mathbf{g}} C[\omega(M)]$.*

Proof. By Proposition 5.31,

$$C[\mathtt{GC}(M[\Re(M) \leftarrow \Omega])] \twoheadrightarrow \mathtt{GC}(N[\Re(M) \leftarrow \Omega]) \ .$$

Moreover, by the soundness of reduction (Proposition 5.26),

$$C[\mathtt{GC}(M[\Re(M) \leftarrow \Omega])] \equiv_{\mathbf{g}} \mathtt{GC}(N[\Re(M) \leftarrow \Omega]) \ .$$

Since $\omega(N) \leq_\omega \mathtt{GC}(N[\Re(M) \leftarrow \Omega])$, from the monotonicity of $\sqsubseteq_{\mathbf{g}}$ with respect to $\leq_\omega$ (Proposition 5.27),

$$\omega(N) \sqsubseteq_{\mathbf{g}} \mathtt{GC}(N[\Re(M) \leftarrow \Omega]) \equiv_{\mathbf{g}} C[\mathtt{GC}(M[\Re(M) \leftarrow \Omega])] \ .$$

Since $C[\mathrm{GC}(M[\Re(M) \leftarrow \Omega])] \twoheadrightarrow_\omega C[\omega(M)]$, and by Proposition 5.32 ω-reductions preserve the meaning

$$C[\mathrm{GC}(M[\Re(M) \leftarrow \Omega])] \equiv_{\mathbf{g}} C[\omega(M)].$$

Therefore, $\omega(N) \sqsubseteq_{\mathbf{g}} \mathrm{GC}(N[\Re(M) \leftarrow \Omega]) \equiv_{\mathbf{g}} C[\mathrm{GC}(M[\Re(M) \leftarrow \Omega])] \equiv_{\mathbf{g}} C[\omega(M)]$.
Pictorially we have

$$\begin{tikzcd}
C[M] \arrow[r, phantom, "\overset{\rightharpoonup}{\Re(M)}"] \arrow[d] & N \arrow[d, dashed] \\
C[\mathrm{GC}(M[\Re(M) \leftarrow \Omega])] \arrow[r, dashed] \arrow[d, "\omega"] & \mathrm{GC}(N[\Re(M) \leftarrow \Omega]) \arrow[d, dashed, "\omega"] \\
C[\omega(M)] \arrow[r, dashed] & \omega(N)
\end{tikzcd}$$

□

LEMMA 5.34. *Given a GRS$_{\mathsf{NI}}$ term M and context $C[\square]$, $C[M] \sqsubseteq_{\mathbf{g}} \bigsqcup\{C[P] \mid P \in W^*(M)\}$.*

PROOF. Since

$$C[M] \equiv_{\mathbf{g}} \bigsqcup\{Q \mid Q \in W^*(C[M])\}$$

we want to show that

$$\bigsqcup\{Q \mid Q \in W^*(C[M])\} \sqsubseteq_{\mathbf{g}} \bigsqcup\{C[P] \mid P \in W^*(M)\}$$

that is, $\forall Q \in W^*(C[M]), \exists P \in W^*(M)$ such that $Q \sqsubseteq_{\mathbf{g}} C[P]$. If $Q \in W^*(C[M])$ it means that $C[M] \twoheadrightarrow N$ and $Q \leq_\omega \omega(N)$, and, therefore, by monotonicity of $\sqsubseteq_{\mathbf{g}}$ with respect to $\leq_\omega$ (Theorem 5.27), we have $Q \sqsubseteq_{\mathbf{g}} \omega(N)$. By Proposition 5.30, given the reduction $C[M] \twoheadrightarrow N$, $\exists M', M \twoheadrightarrow M'$ and $C[M'] \overset{\rightharpoonup}{\Re(M')} N$. Therefore, by Lemma 5.33 we have

$$\omega(N) \sqsubseteq_{\mathbf{g}} C[\omega(M')] \ .$$

Since $\omega(M') \in W^*(M)$, by letting $P \equiv \omega(M')$ we have that $Q \sqsubseteq_{\mathbf{g}} C[P]$. □

THEOREM 5.35. *Given a GRS$_{\mathsf{NI}}$ term M and context $C[\square]$, $C[M] \equiv_{\mathbf{g}} \bigsqcup\{C[P] \mid P \in W^*(M)\}$.*

PROOF. Follows from Lemma 5.29 and Lemma 5.34. □

THEOREM 5.36. (CONGRUENCE OF $\equiv_{\mathbf{g}}$ WITH RESPECT TO THE CONTEXT OPERATION) *Given GRS$_{\mathsf{NI}}$ terms M and N, and context $C[\square]$, if $M \equiv_{\mathbf{g}} N$ then $C[M] \equiv_{\mathbf{g}} C[N]$.*

PROOF.
$$\begin{aligned}
C[M] &\equiv_{\mathbf{g}} \bigsqcup\{C[P] \mid P \in W^*(M)\} && \text{(by Theorem 5.35)}\\
&\equiv_{\mathbf{g}} \bigsqcup\{C[Q] \mid Q \in W^*(N)\} && \text{(by the hypothesis)}\\
&\equiv_{\mathbf{g}} C[N] && \text{(by Theorem 5.35)}\ .
\end{aligned}$$

□

We stressed again that in the presence of interfering rules, $\equiv_{\mathbf{g}}$ is not guaranteed to be a congruence.

Consider the rules

$$\tau_1 : \quad x = \mathsf{A}(y) \longrightarrow x = \mathsf{A}(y)$$

$$\tau_2 : \quad x = \mathsf{B}(y) \longrightarrow x = \mathsf{B}(y)$$

$$\tau_3 : \quad \frac{x_1 = \mathsf{A}(y)}{x = \mathsf{F}(x_1) \longrightarrow x = 2}$$

$$\tau_4 : \quad \frac{x_1 = \mathsf{B}(y)}{x = \mathsf{F}(x_1) \longrightarrow x = 3}$$

and the following two terms:

$$M \equiv \{\ t = \mathsf{A}(x); \ \mathsf{In}\ t\} \qquad N \equiv \{\ t = \mathsf{B}(x); \ \mathsf{In}\ t\}$$

Notice that $M \equiv_{\mathsf{g}} N$, but the context $\{t_1 = \mathsf{F}(t_2);\ t_2 = \square;\ \mathsf{In}\ t_1\}$ can distinguish between M and N.

6. Correctness of Optimizations

If we let $(A(\mathcal{F}), R_{opt})$ be a GRS, where $A(\mathcal{F})$ represents the set of terms over signature $\mathcal{F}$, and R_{opt} a set of optimization rules, then correctness can be formulated as follows.

Definition 6.1. (Total correctness) *An optimizer $(A(\mathcal{F}), R_{opt})$ is totally correct with respect to a GRS_{NI} $(A(\mathcal{F}), R)$ iff $\forall\ M \in A(\mathcal{F})$, if $M \twoheadrightarrow N$ using rules in R_{opt} then $M \equiv_{\mathsf{g}} N$.*

In many situations it is acceptable if optimizations produce a more defined program.

Definition 6.2. (Partial correctness) *An optimizer $(A(\mathcal{F}), R_{opt})$ is partially correct with respect to a GRS_{NI} $(A(\mathcal{F}), R)$ iff $\forall\ M \in A(\mathcal{F})$, if $M \twoheadrightarrow N$ using rules in R_{opt} then $M \sqsubseteq_{\mathsf{g}} N$.*

Fact 6.3. *If $M \longrightarrow M_1$ by applying the common subexpression elimination rule then $M \leq_{\omega} M_1$.*

Proposition 6.4. *The common subexpression elimination is partially correct with respect to a GRS_{NI}.*

Proof. Directly from the above fact and Corollary 5.28. $\square$

Let $\mathrm{GRS}^{\mathsf{Y}}$ be a non-interfering GRS containing the following Y-rule:

$$\text{Y-rule :} \qquad x = \mathsf{Ap}(\mathsf{Y}, f) \longrightarrow x = \{\ t = \mathsf{Ap}(f, t_1);\ t_1 = \mathsf{Ap}(\mathsf{Y}, f);\ \mathsf{In}\ t\}$$

We can avoid the redex rooted at t_1 on the right hand side of the above rule by using the following cyclic Y-rule:

$$\mathsf{Y_c}\text{-rule :} \qquad x = \mathsf{Ap}(\mathsf{Y}, f) \longrightarrow x = \mathsf{Ap}(f, x)$$

Proposition 6.5. *The cyclic Y-rule is partially correct with respect to a GRS^{Y}.*

PROOF. Suppose $M \longrightarrow M_1$ using the Y-rule and $M \longrightarrow M_2$ using the $\mathsf{Y_C}$-rule. Then $M_1 \sqsubseteq_g M_2$ and the result follows from Theorem 5.36. □

If we want to prove the total correctness of the cyclic implementation of the Y rule or the common subexpression elimination rule, then we need to impose further restrictions on the GRS rules. In particular, non-left-linear rules (which include left-cyclic rules) will have to be disallowed. Furthermore, sharing will have to be ignored from the observations as well. The reader may refer to [1], where such a term model based on Böhm trees is presented.

Surprisingly, the following algebraic rules:

$$*(x, 0) \longrightarrow 0$$
$$*(x, 1) \longrightarrow x$$

are not partially correct according to our model. Consider the following reduction:

$$M_1 \equiv \{x = \mathsf{G}(7);\ y = *(x, 1);\ \mathsf{In}\ y\} \longrightarrow M_1' \equiv \{x = \mathsf{G}(7);\ \mathsf{In}\ x\}.$$

As long as G(7) is a redex, $\omega(M_1) \leq_\omega \omega(M_1')$. However, if G(7) is not a redex, then there is no relationship between $\omega(M_1)$ and $\omega(M_1')$. Such situations can arise either because of *type errors* or *deadlocks*. We illustrate the deadlock situation by the following example:

$$M_2 \equiv \{x = *(y, 5); y = *(x, 1);\ \mathsf{In}\ x\} \longrightarrow M_2' \equiv \{x = *(x, 5);\ \mathsf{In}\ x\}.$$

Notice M_2 is type correct but it cannot produce an integer as an answer. However, in our term model, without the optimization M_2 will produce itself as an answer, and with the optimization it will produce M_2', which is not related to M_2. We believe that cycles involving "strict operators" should not be observable, that is, the observations of both M_2 and M_2' should be Ω.

7. Future directions

We have defined a class of GRSs which can express sharing of terms, including cyclic terms. Our GRS admits both non-left-linear and left-cyclic rules. We prove the confluence of $\mathrm{GRS_{NI}}$, that is, a GRS with non interfering rules, and develop a term model a la Lévy for such GRSs. Furthermore, we apply the semantic relation to show the partial correctness of important optimizations such as the common subexpression elimination and the cyclic implementation of the Y-rule. The main advantage of our approach is its simplicity because it avoids infinitary terms and associated transfinite reduction.

The motivation for this work came from a desire to formalize the compilation process of Id, an implicitly parallel language [19]. Id has a purely functional, higher-order, non-strict core. In addition Id also contains logical variables in the form of I-structures and mutable variables in the form of M-structures. The compiler of Id is expressed as a series of translations into simpler and simpler languages. In our prior work we have introduced the Kid (Kernel id) language [3] and the P-TAC (Parallel Three Address Code) language [2], and provided the translation of Id into Kid and of Kid into P-TAC [4]. We have formalized many optimizations in the Id compiler in terms of source-to-source transformations on these intermediate languages.

Functional subset of P-TAC can be seen as an example of the GRS presented in this paper. Functional

Kid, however, is more general due to the presence of λ-abstraction. A model for a GRS with λ-abstraction will provide a sound mathematical basis for the functional subset of the Id language. However, even proving confluence for the λ-calculus with sharing is a difficult problem.

Another direction of work would be to incorporate a notion of types, and the distinction between strict and non-strict operators in our GRS framework. This will allow one to prove the correctness of compiler optimizations, such as the algebraic rules.

Yet another direction would be to incorporate I-structures and M-structures in our GRS. Elsewhere, we have shown the confluence of a GRS with I-structures [2]. I-structures operations can be expressed using "multi-rooted rules". However, a term model for a GRS with I-structures would require a very different treatment of unconnected subterms and garbage collection.

Acknowledgements

This work was done at the Laboratory for Computer Science at MIT, and at the University of Oregon. Funding for this work has been provided in part by the Advanced Research Projects Agency of the U.S. Department of Defense under the Office of Naval Research contracts N00014-92-J-1310 (MIT). The research of the first author has been supported by NSF grant CCR-94-10237.

We would like to express immense gratitude to Jean-Jacques Lévy, and Jan Willem Klop for insightful discussions.

References

1. Z. Ariola. Relating Graph and Term Rewriting via Böhm Models. In *Proc. RTA '93, Montreal, Canada, June*, 1993.
2. Z. Ariola and Arvind. P-TAC: A Parallel Intermediate Language. In *Proc. ACM Conference on Functional Programming Languages and Computer Architecture, London, UK*, September 1989.
3. Z. Ariola and Arvind. A Syntactic Approach to Program Transformations. In *Proc. ACM SIGPLAN Symposium on Partial Evaluation and Semantics Based Program Manipulation, New Haven, CT*, June 1991.
4. Z. Ariola and Arvind. Compilation of Id. In *Proc. of the Fourth Workshop on Languages and Compilers for Parallel Computing, Santa Clara, CA, Springer-Verlag LNCS 589*, August 1991.
5. Z. Ariola and J. W. Klop. Equational Term Graph Rewriting. Technical Report Draft, 1993.
6. H. Barendregt. *The Lambda Calculus: Its Syntax and Semantics.* North-Holland, Amsterdam, 1984.
7. H. Barendregt, T. Brus, M. van Eekelen, J. Glauert, J. Kennaway, M. van Leer, M. Plasmeijer, and M. R. Sleep. Towards an Intermediate Language based on Graph Rewriting. In *Proceedings of the PARLE Conference, Eindhoven, The Netherlands, Springer-Verlag LNCS 259*, June 1987.
8. H. Barendregt, M. van Eekelen, J. Glauert, J. Kennaway, M. Plasmeijer, and M. Sleep. Term Graph Rewriting. In *Proceedings of the PARLE Conference, Eindhoven, The Netherlands, Springer-Verlag LNCS 259*, June 1987.
9. T. Brus, M. van Eekelen, M. vam Leer, and M. Plasmeijer. Clean - A language for Functional Graph Rewriting. In *Proc. ACM Conference on Functional Programming Languages and Computer Architecture, Portland, OR, Springer-Verlag LNCS 274*, 1987.
10. M. Hennessy. *Algebraic Theory of Processes.* MIT Press, 1988.
11. G. Huet and J.-J. Lévy. Computations in orthogonal rewriting systems 1 and 2. In *Computational logic. Essays in Honor of Alan Robinson. Ed. J.-L. Lassez & G.D. Plotkin*, 1991.
12. R. J. M. Hughes. Super-combinators. In *Proceedings of Lisp and Functional Programming, Pittsburg, PA*, 1982.
13. T. Johnsson. Lambda Lifting: Transforming Programs to Recursive Equations. In *Proceedings of Functional Programming Languages and Computer Architecture, Nancy, France, Springer-Verlag LNCS 201*, September 1985.
14. V. Kathail. *Optimal Interpreters for Lambda-calculus Based Funtional Languages.* PhD thesis, Dept. of Electrical Engineering and Computer Science, MIT, Cambridge, MA, May 1990.
15. J. Kennaway, J. Klop, M. Sleep, and F. de Vries. Transfinite Reductions in Orthogonal Term Rewriting Systems. In *Proc. RTA '91, Springer-Verlag LNCS*, 1991.
16. R. Kennaway. Implementing Term Rewrite Languages in Dactl. *Theoretical Computer Science*, Vol. 1, 1990.
17. J. Lamping. An algorithm for optimal lambda calculus reduction. In *Proc. ACM Conference on Principles of Programming Languages, San Francisco, CA*, January 1990.
18. J.-J. Lévy. *Réductions Correctes et Optimales dans le Lambda-Calcul.* PhD thesis, Universite Paris VII, Paris, France, October 1978.
19. R. S. Nikhil. Id (Version 90.0) Reference Manual. Technical Report CSG Memo 284-a, Laboratory for Computer Science, MIT, Cambridge, MA, USA, July 1990.
20. C. Wadsworth. *Semantics And Pragmatics Of The Lambda-Calculus.* PhD thesis, University of Oxford, Oxford, UK, September 1971.
21. P. Welch. Continuous Semantics and Inside-out Reductions. In *λ-Calculus and Computer Science Theory, Italy, Springer-Verlag LNCS 37*, March 1975.

Table of Contents