Least Upper Bounds on the Size of Confluence and Church-Rosser Diagrams in Term Rewriting and λ-Calculus[1]

JEROEN KETEMA, Imperial College London
JAKOB GRUE SIMONSEN, University of Copenhagen

We study confluence and the Church-Rosser property in term rewriting and λ-calculus with explicit bounds on term sizes and reduction lengths. Given a system R, we are interested in the lengths of the reductions in the smallest valleys $t \to^* s' \,{}^*\!\leftarrow t'$ expressed as a function:

— for confluence a function $\mathrm{vs}_R(m,n)$ where the valleys are for peaks $t \,{}^*\!\leftarrow s \to^* t'$ with s of size at most m and the reductions of maximum length n, and
— for the Church-Rosser property a function $\mathrm{cvs}_R(m,n)$ where the valleys are for conversions $t \leftrightarrow^* t'$ with t and t' of size at most m and the conversion of maximum length n.

For confluent term rewriting systems (TRSs), we prove that vs_R is a total computable function, and for linear such systems that cvs_R is a total computable function. Conversely, we show that every total computable function is the lower bound on the functions $\mathrm{vs}_R(m,n)$ and $\mathrm{cvs}_R(m,n)$ for some TRS R: In particular, we show that for every total computable function $\varphi : \mathbb{N} \longrightarrow \mathbb{N}$ there is a TRS R with a single term s such that $\mathrm{vs}_R(|s|,n) \geq \varphi(n)$ and $\mathrm{cvs}_R(n,n) \geq \varphi(n)$ for all n.

For orthogonal TRSs R we prove that there is a constant k such that (a) $\mathrm{vs}_R(m,n)$ is bounded from above by a function exponential in k and (b) $\mathrm{cvs}_R(m,n)$ is bounded from above by a function in the fourth level of the Grzegorczyk hierarchy. Similarly, for λ-calculus, we show that $\mathrm{vs}_R(m,n)$ is bounded from above by a function in the fourth level of the Grzegorczyk hierarchy.

Categories and Subject Descriptors: F.4.1 [**Mathematical Logic and Formal Languages**]: Mathematical Logic—*Lambda calculus and related systems*

General Terms: Theory

Additional Key Words and Phrases: Term rewriting, lambda calculus, Church-Rosser property, confluence, upper bounds

ACM Reference Format:
Ketema, J., Simonsen, J.G. 2012. Least Upper Bounds on the Size of Confluence and Church-Rosser Diagrams in Term Rewriting and λ-Calculus. *ACM Trans. Comput. Logic* 0, 0, Article 0 (December 2012), 28 pages.
DOI: http://dx.doi.org/10.1145/0000000.0000000

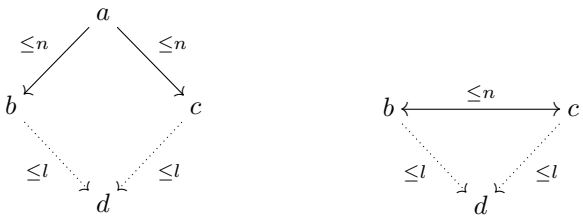
1. INTRODUCTION

Confluence is the property of some rewriting systems that any *peak* $t \,{}^*\!\leftarrow s \to^* t'$ has a corresponding *valley* $t \to^* s' \,{}^*\!\leftarrow t'$. The valley and the term s' are said to *complete* the diagram.

[1]This work is an extended and revised version of the conference paper [Ketema and Simonsen 2010].

Authors' addresses: J. Ketema, Department of Computing, Imperial College London, 180 Queen's Gate, London SW7 2BZ, United Kingdom; Jakob Grue Simonsen, Department of Computer Science, University of Copenhagen (DIKU), Njalsgade 126–128, 2300 Copenhagen S, Denmark

DOI: http://dx.doi.org/10.1145/0000000.0000000



(a) Confluence (b) Church-Rosser property

Fig. 1. Confluence and the Church-Rosser property of a rewrite system R with bounds on the lengths of the reductions. This paper is concerned with finding least upper bounds of l as a function of n.

A rewriting system is confluent iff it has the so-called Church-Rosser property; the property that any *conversion* $t \leftrightarrow^* t'$ has a corresponding valley $t \rightarrow^* s' \ {}^*\!\leftarrow t'$. As the two notions are equivalent, the terms 'confluence' and 'Church-Rosser property' are often used interchangeably in the literature; to the authors' knowledge, all contemporary research in rewriting systems considers confluence, rather than the classical Church-Rosser property, probably due to the fact that the former seems better suited to the range of standard proof methods available.

In functional programming, the Church-Rosser property ensures that different ways of evaluating a program always yield the same result (modulo non-termination): The result of running a program will be independent of the evaluation order or reduction strategy. In logic, if the rewrite system induced by the derivation relation of a deductive system has the Church-Rosser property and is strongly normalizing, the system will be consistent: No statement can both hold and not hold.

While the Church-Rosser property has been shown to hold for a variety of rewrite systems, there has, to the authors' knowledge, never been an investigation into the number of reduction steps in a valley that completes a peak or a conversion of a given size (see Figure 1). Succinctly: The question "How large is the valley as a function of the peak or conversion?" has apparently not been asked before.

We find the above question to be *intrinsically* interesting from a theoretical point of view as Church-Rosser-type results are ubiquitous. However, we also believe the practical implications in mainstream functional programming to be limited: Standard functional languages like ML and HASKELL employ a fixed evaluation strategy such as call-by-value or call-by-need, and there seems to be little interest in performing optimizations by switching strategies (modulo non-termination). However, for more specialized languages, like declarative DSLs where the evaluation order may not be fixed, there may be practical implications: If, for small peaks, the size of the smallest corresponding valley is so large that a term completing the Church-Rosser diagram cannot be computed using realistic resources, then it matters *very much* which reduction strategy is employed: Choosing the 'wrong' evaluation strategy (say, call-by-value) and performing just a few steps of a computation could result in a very long reduction before a result is reached—better to backtrack to the original term and try another strategy.

In this paper, we perform the first fundamental study of valley sizes for systems with the Church-Rosser property; specifically we study how the size of a peak or conversion affects the *size* of the smallest corresponding valley. We consider three very general settings: that of (arbitrary) first-order term rewriting systems, of orthogonal term rewriting systems (roughly corresponding to first-order functional programs that have no fixed evaluation order), and of untyped λ-calculus. We believe that these three

Table I. The bounds derived for the valley sizes and their tightness

	Confluence (vs)		Church-Rosser (cvs)	
	Bound	Tight?	Bound	Tight?
TRSs (Section 4)	Computable (Theorem 4.2)	Yes (Theorem 4.12)	Unknown	
Linear TRSs (Section 4)			Computable (Theorem 4.5)	Yes (Theorem 4.14)
Orthogonal TRSs (Section 5)	Exponential (Theorem 5.5)	Yes (Remark 5.6)	$\mathcal{E}_4$ (Theorem 5.9)	Unknown
λ-calculus (Section 6)	$\mathcal{E}_4$ (Theorem 6.5)	Unknown	Unknown	

areas cover most of the non-specialized areas where the Church-Rosser property occurs. The most significant area omitted is that of general higher-order rewrite systems (including higher-order functional programs and logics with bound variables)—we expect general upper bounds in that case to be very difficult to derive, as is foreshadowed by the difficulties we encounter in our treatment of λ-calculus in Section 6.

Valley sizes are measured by two functions, $\mathrm{vs} : \mathbb{N}^2 \longrightarrow \mathbb{N}$ for confluence, and $\mathrm{cvs} : \mathbb{N}^2 \longrightarrow \mathbb{N}$ for the Church-Rosser property. Roughly, $\mathrm{vs}(m, n)$ is the least number of steps required to complete a valley for a peak starting from a term of size at most m and with reductions of length at most n. Similarly, $\mathrm{cvs}(m, n)$ is the least number of steps required to complete a valley corresponding to a conversion involving at most n steps between two terms of size at most m.

Our results on upper bounds are summarized in Table I. In the table, the exponential upper bound for orthogonal TRSs depends on a constant dependent on the specific rewrite system and is thus tractable in some cases.

1.1. Related Work

For all the kinds of rewriting considered in this paper, the question of valley sizes in confluence and Church-Rosser diagrams does not appear to have been investigated before. However, in research on the 'efficiency' in λ-calculus, some related phenomena have been under scrutiny. Most pertinently, investigations have been performed concerning upper bounds on the length of developments by de Vrijer [1985] and concerning standard reductions by Xi [1999]. In typed systems, lower bounds for normalizing reductions have been studied by Statman [1979], upper bounds are treated Schwichtenberg [1982; 1991] and Beckmann [2001], and lower and upper bounds are discussed by Springintveld [1993]. A related vein of research considers lower bounds on the length of developments and reductions in λ-calculus, pioneered by Khasidashvili [1988] and later simplified by Sørensen [2007]; upper bounds on reductions are considered by Sørensen [1996].

In the present work, we employ distinct techniques to treat first-order term rewriting and λ-calculus. However, a body of literature exists that attempts to relate certain efficiency measures of the two. For example, Dal Lago and Martini [2009] showed that orthogonal constructor term rewrite systems and λ-calculus with weak call-by-value reduction can simulate each other with linear overhead.

Throughout the paper, we measure the size of diagrams by the number of rewrite steps performed. This is a measure quite different from the actual computational *work* employed in computing the diagram on a concrete machine. For λ-calculus, the relation between a single parallel β-step and the time and space complexity of performing it on abstract hardware such as a Turing machine has been investigated in depth in the 1990s, culminating in work by Lawall and Mairson [1996] showing that most earlier attempts at providing reasonable cost models with a polynomial overhead with respect to Turing machine simulation were not adequate; later work by Lawall and Mairson

[1997] attempted to use Lévy labeling to provide a cost model, but yielded no concrete algorithm to normalize arbitrary λ-terms with polynomial overhead. Dal Lago and Martini [2008] recently gave a cost model with polynomial time overhead for λ-calculus with weak call-by-value reduction, but it is as yet unclear whether the model can be amended to work with arbitrary β-steps.

2. PRELIMINARIES

We presuppose a working knowledge of Turing machines and a basic familiarity with term rewriting and λ-calculus. We give brief definitions below. The basic references for term rewriting are Baader and Nipkow [1998], Terese [2003], and Klop [1992]; for λ-calculus we refer the reader to Barendregt [1985], Terese [2003], and Fernández [2009]. For Turing machines, any introductory textbook on computability will do [Papadimitriou 1994; Jones 1997; Sipser 2006; Fernández 2009].

Sections 5 and 6 of the paper use the Grzegorczyk hierarchy; we refer the reader to Grzegorczyk [1953] and Odifreddi [1999] for definitions. Most important for our purposes is $\mathcal{E}_4$, the fourth level of the hierarchy, which roughly corresponds to limited recursion over exponential functions, i.e. *iterated exponentiation* or *tetration*—a typical function is:

$$n \mapsto \underbrace{2^{2^{\cdot^{\cdot^{2}}}}}_{n}.$$

2.1. Abstract Rewriting, Confluence, and the Church-Rosser Property

We introduce some basic notions related to abstract rewriting, confluence, and the Church-Rosser property.

Definition 2.1. An *abstract reduction system* (ARS) is a pair $(A, \to)$ with A a set of *objects* and $\to$ a *binary relation* over A where $(a, b) \in \to$ is written as $a \to b$. The ARS is said to be *finitely branching* if for every object a there are only finitely many objects b such that $a \to b$.

— A *reduction of length* n is a finite sequence of objects $\langle a_0, a_1, \ldots, a_n\rangle$ with $a_i \to a_{i+1}$ for all $i < n$, written as $a_0 \to a_1 \to \cdots \to a_{n-1} \to a_n$ or $a_0 \to^n a_n$, or even as $a_0 \to^* a_n$.
— A *conversion of length* n is a finite sequence objects $\langle a_0, a_1, \ldots, a_n\rangle$ with $a_i \to a_{i+1}$ or $a_i \leftarrow a_{i+1}$ for all $i < n$, written as $a_0 \leftrightarrow^n a_n$, or even as $a_0 \leftrightarrow^* a_n$.

Definition 2.2. A *peak* is a pair of reductions

$$(a_i \leftarrow a_{i-1} \leftarrow \cdots \leftarrow a_1 \leftarrow a,\ a \to a'_1 \to \cdots \to a'_{j-1} \to a'_j)$$

both starting from a. A *valley* is a pair of reductions

$$(b_0 \to b_1 \to \cdots \to b_{k-1} \to b,\ b \leftarrow b'_{l-1} \leftarrow \cdots \leftarrow b'_1 \leftarrow b'_0)$$

both ending in b. Below, a peak is usually written as $a_i\ {}^{i}\!\leftarrow a \to^j a'_j$ and a valley as $b_0 \to^k b\ {}^{l}\!\leftarrow b'_0$, occasionally replacing reduction lengths with the Kleene star $*$ when the lengths are unimportant.

Definition 2.3. An ARS $(A, \to)$ is *confluent* if for every peak $b\ {}^{*}\!\leftarrow a \to^* c$ there exists a valley $b \to^* d\ {}^{*}\!\leftarrow c$. Moreover, an ARS $(A, \to)$ has the *Church-Rosser property* if for every conversion $b \leftrightarrow^* c$ there exists a valley $b \to^* d\ {}^{*}\!\leftarrow c$.

It is a straightforward exercise to show that an ARS is confluent iff it has the Church-Rosser property (see e.g. Terese [2003, Proposition 1.1.10]).

Definition 2.4. Let $(A, \to)$ be an ARS and let $a \in A$. The *reduction graph* of a, denoted $\mathcal{G}(a)$, is the graph $(V_a, E_a) = (\bigcup_{n\geq 0} V_{a,n}, \bigcup_{n\geq 0} E_{a,n})$ with $V_{a,n}$ and $E_{a,n}$ inductively defined by

$$V_{a,n} = \begin{cases} \{a\} & \text{if } n = 0 \\ \{b : \exists a' \in V_{a,n-1}. a' \to b\} & \text{if } n > 0 \end{cases}$$

and

$$E_{a,n} = \begin{cases} \emptyset & \text{if } n = 0 \\ \{(a', b) : a' \in V_{a,n-1},\, b \in V_{a,n}.\, a' \to b\} & \text{if } n > 0 \end{cases}$$

Thus, $V_{a,n}$ is the set of objects b such that $a \to^n b$.

2.2. Term Rewriting Systems

Throughout, we assume a fixed, finite *signature* Σ with each function symbol of non-negative integer arity and a denumerable, infinite set of *variables* V. The set of *terms* over Σ and V, denoted $\mathrm{Ter}(\Sigma, V)$, is defined by induction, as usual. We assume the following, where $\mathbb{N}^*$ is the set of *finite strings over* $\mathbb{N}$ with ϵ the *empty string*.

Definition 2.5. Let s be a term.

— The term s is *ground* if no variables occur in s.
— The set of *positions* of s, denoted $\mathbf{pos}(s)$, is the subset of $\mathbb{N}^*$ inductively defined by $\mathbf{pos}(x) = \{\epsilon\}$ and $\mathbf{pos}(f(s_1, \ldots, s_n)) = \{\epsilon\} \cup (\bigcup_{i=1}^{n} i \cdot \mathbf{pos}(s_i))$.
— The set of *variables* of s, denoted $\mathbf{vars}(s)$, is the finite subset of V inductively defined by $\mathbf{vars}(x) = \{x\}$ and $\mathbf{vars}(f(s_1, \ldots, s_n)) = \bigcup_{i=1}^{n} \mathbf{vars}(s_i)$.
— The *size* of s, denoted $|s|$, is defined inductively by $|x| = 1$ and $|f(s_1, \ldots, s_n)| = 1 + |s_1| + \cdots + |s_n|$.

Positions are equipped with a (strict) partial order $\prec$ such that $p \prec q$ if p is a proper prefix of q. We write $s|_p$ for the *subterm* of a term s that occurs at position $p \in \mathbf{pos}(s)$.

Substitutions, written $\sigma : V \longrightarrow \mathrm{Ter}(\Sigma, V)$, are defined as usual. *Contexts* are terms over $\Sigma \uplus \{\Box\}$, written as $C[]$, where we say that that a context $C[]$ is a *k-hole context* if there are exactly k occurrences of $\Box$ in $C[]$.

Definition 2.6. A *rule* over Σ is a pair (l, r), invariably written $l \to r$, where l and r are terms over Σ such that $l \notin V$ and $\mathbf{vars}(r) \subseteq \mathbf{vars}(l)$. A term s *rewrites* to a term t by $l \to r$ if there exists a one-hole context $C[]$ and a substitution σ such that $s = C[\sigma(l)]$ and $t = C[\sigma(r)]$.

A *term rewriting system (TRS)* is a pair (Σ, R) with Σ a signature and R a finite set of rules over Σ.

We usually suppress explicit mention of the signature Σ and refer to the TRS (Σ, R) as R. Every TRS R gives rise to an ARS $(A, \to)$ in the obvious fashion: The objects of A are the terms and $\to$ is the above rewrite relation.

Definition 2.7. A term is *linear* if every variable occurs in it at most once. A rule $l \to r$ is *left-linear* if l is a linear term. Moreover, a rule is *linear* if both l and r are linear terms. A TRS R is *left-linear*, respectively *linear*, if all its rules are.

A rule $l_1 \to r_1$ is said to *overlap* a rule $l_2 \to r_2$ at position $p \in \mathbf{pos}(l_2)$ if $l_2|_p \notin V$ and there are substitutions σ, τ such that $\tau(l_1) = \sigma(l_2|_p)$. A TRS (Σ, R) is said to be *orthogonal* if R is left-linear and the only overlaps of rules in R are those where a rule overlaps itself at position ϵ.

Two TRSs (Σ_0, R_0) and (Σ_1, R_1) are said to be *mutually orthogonal* if they are left-linear and no rule of R_0 overlaps with a rule of R_1, and vice versa.

2.3. Combinator Systems

A *combinator system* [van Bakel and Fernández 2003] is a particular kind of ARS $(CS, \to)$. Assuming a finite set of constants C (often called *combinators*), the set CS of objects or *terms* M is inductively defined by

$$M ::= x \mid C \mid (M\, M)$$

Convention dictates that parentheses associate to the left and are dropped accordingly. Hence, $((x\, y)\, (x\, z))$ is usually written $x\, y\, (x\, z)$. The *size* of a term is defined inductively by $|x| = 1$, $|C| = 1$, and $|M\, N| = 1 + |M| + |N|$.

A rule is of the form

$$C\, x_1 \cdots x_n \to M$$

where C is a combinator, $x_1, \ldots, x_n$ are distinct variables, and such that every variable in M occurs among $x_1, \ldots, x_n$. A combinator system has *exactly one* rule per combinator. One-hole contexts, substitutions, and the relation $\to$ are defined as for TRSs, mutatis mutandis.

The standard example of a combinator system is *Combinatory Logic*, which has the combinators I, K, S with associated rules $I\, x \to x$, $K\, x\, y \to x$, and $S\, x\, y\, z \to x\, z\, (y\, z)$.

For our purposes, combinator systems—including Combinatory Logic—need only be treated superficially, as the upper bounds we shall derive for orthogonal term rewriting systems also hold for combinator systems due to the proposition below, which is folklore.

PROPOSITION 2.8. *For every combinator system $(CS, \to)$ with k combinators, there is an orthogonal rewriting system (Σ, R) with $k+1$ function symbols and k rules, and a bijective function ϕ mapping terms from CS to terms over Σ such that for every pair of terms M, N from CS:*

$$M \to_{CS} N \quad \text{iff} \quad \phi(M) \to_R \phi(N)$$

PROOF. Set $\Sigma = \{\mathrm{App}\} \uplus \{C_1, \ldots, C_k\}$, with App binary and all other function symbols nullary, and define ϕ inductively by $\phi(x) = x$, $\phi(C_i) = C_i$, and $\phi(M\, N) = \mathrm{App}(\phi(M), \phi(N))$. Let R be the set of all rewrite rules $\phi(l) \to \phi(r)$ with $l \to r$ a rule from the combinator system. Orthogonality of R follows by the assumption that all variables on the left-hand sides of combinator rules are distinct and by the assumption that there is exactly one rule per combinator C_i. Bijectivity of ϕ and the connection between $\to_{CS}$ and $\to_R$ now follow by an easy induction. □

Observe that as ϕ is a bijection, we also have for every pair (s, t) of terms over Σ that $s \to_R t$ iff $\phi^{-1}(s) \to_{CS} \phi^{-1}(t)$.

2.4. λ-Calculus

The *(untyped) λ-calculus* is the ARS $(\Lambda, \to_\beta)$ with Λ the set of objects or *λ-terms* M defined inductively by

$$M ::= x \mid \lambda x.M \mid M\, M$$

where $x \in V$ is a variable and with $\to_\beta$ the rewrite relation induced by the β-rule:

$$(\lambda x.M)\, N \to_\beta M\{N/x\}$$

where $M\{N/x\}$ equals M with N substituted for every free occurrence of x in M, working as usual modulo α-equivalence (see e.g. Barendregt [1985]). Contexts for λ-calculus are defined as for TRSs.[2] We assume the following.

[2]Observe that substitutions are only defined on terms and not on contexts; we make no use of substitutions on contexts.

Definition 2.9. Let M be a λ-term.

— The set of *positions* of M, denoted $\mathbf{pos}(M)$, is the subset of $\mathbb{N}^*$ inductively defined by $\mathbf{pos}(x) = \{\epsilon\}$, $\mathbf{pos}(\lambda x.M) = \{\epsilon\} \cup 0 \cdot \mathbf{pos}(M)$, and $\mathbf{pos}(M_1\, M_2) = \{\epsilon\} \cup 0 \cdot \mathbf{pos}(M_1) \cup 1 \cdot \mathbf{pos}(M_2)$.
— The *size* of M, denoted $|M|$, is defined inductively by $|x| = 1$, $|\lambda x.M| = 1 + |M|$, and $|M\, N| = 1 + |M| + |N|$.

Positions are equipped with a (strict) partial order $\prec$ such that $p \prec q$ if p is a proper prefix of q.

The notion of a *residual* of a β-redex across a reduction, i.e. the formalization of 'what happens' to a redex across a reduction, is defined as usual [Barendregt 1985]. The set of residuals of a β-redex u, respectively a set of redexes U, across a reduction $M \to_\beta^* N$ is denoted $u/(M \to_\beta^* N)$, respectively $U/(M \to_\beta^* N)$. Recall that a *development* of a set of β-redexes U of a λ-term M is a reduction starting from M contracting a residual of a redex in U in each step. As usual, a development $M \to_\beta^* N$ is *complete* if the set of residuals of redexes in U across $M \to_\beta^* N$ is empty, i.e. if $U/(M \to_\beta^* N) = \emptyset$. We have the following [Barendregt 1985].

THEOREM 2.10 (FINITE DEVELOPMENTS THEOREM). *Let M be a λ-term and U a set of redexes of M. All developments of U are finite and there is a unique λ-term N that is the final term of all complete developments of U.*

3. VALLEY SIZES IN ABSTRACT REDUCTION SYSTEMS

We now define the main objects of study in this paper: the functions vs_R and cvs_R.

Definition 3.1. Let $R = (A, \to)$ be a *finitely branching* and *confluent* ARS and let $|\cdot| : A \longrightarrow \mathbb{N}$ be a function ('size') such that $\{a \in A : |a| \leq m\}$ is finite for each $m \in \mathbb{N}$.

— The *valley size* $\mathrm{vs}_R : \mathbb{N}^2 \longrightarrow \mathbb{N}$ (see Figure 1(a) on page 2) is defined as $\mathrm{vs}_R(m, n) = l$ where l is the least non-negative integer such that for every object a with $|a| \leq m$ and every peak starting from a with reductions of length at most n there is a completing valley with reductions of length at most l. If there are no objects of size at most m, define $\mathrm{vs}_R(m, n) = 0$ for all n.

— The *conversion valley size* $\mathrm{cvs}_R : \mathbb{N}^2 \longrightarrow \mathbb{N}$ (see Figure 1(b) on page 2) is defined as $\mathrm{cvs}_R(m, n) = l$ where l is the least non-negative integer such that for every pair of objects b, c with $|b|, |c| \leq m$ and every conversion $b \leftrightarrow^* c$ of length at most n, there is a completing valley with reductions of length at most l. If there are no objects of size at most m, define $\mathrm{cvs}_R(m, n) = 0$ for all n.

The function $\mathrm{vs}_R(m, n)$ is well-defined: For each m, finiteness of the set $\{a \in A : |a| \leq m\}$ in combination with finite branching of $(A, \to)$ ensures that only finitely many peaks exist with reductions of length at most n; furthermore, the peaks all have valleys by confluence of $(A, \to)$. Observe that, if $(A, \to)$ were not finitely branching, there would exist an object a such that $a \to b$ for an infinite number of objects b. In this case, $\mathrm{vs}_R(|a|, 1)$ might not be bounded, as each for $n \in \mathbb{N}$ there could exist a peak $b_n \leftarrow a \to c_n$ such that one of the reductions in any completing valley has at least length n.

The function $\mathrm{cvs}_R(m, n)$ is also well-defined: For each m, finiteness of the set $\{a \in A : |a| \leq m\}$ ensures that there are only finitely many conversions of length at most n; the conversions all have valleys by confluence of $(A, \to)$. Observe that finite branching is redundant in this case. However, as ARSs will be finitely branching in all practical cases, we do not consider this additional restriction a severe one. In fact, from here onwards we assume *all* ARSs to be finitely branching.

The 'size' function $|\cdot|$ depends on the class of ARSs considered. In this paper, we are concerned solely with term rewriting systems, combinator systems, and λ-calculus where we consider terms equal (with respect to the size function) if one can be obtained from the other by renaming of (free) variables, as this will ensure that $\{a \in A : |a| \leq m\}$ is finite. Observe that for TRSs and combinator systems it is safe to make this assumption, as variables behave as nullary function symbols to which no rewrite rules apply. Similarly for λ-calculus, where free variables behave as nullary function symbols.

We employ $|a|, |b|, |c| \leq m$, and not $|a|, |b|, |c| = m$, in the definitions of vs_R and cvs_R to ensure that the functions are non-decreasing in both arguments. Replacing $|a|, |b|, |c| \leq m$ by $|a|, |b|, |c| = m$ gives less well-behaved functions; Example 3.2 below demonstrates this poor behavior: $\mathrm{vs}_R(2, 1)$ would be equal to 1 instead of being equal to $\mathrm{vs}_R(1, 1) = 2$.

A confluent ARS will usually have several (or even infinitely many) different valleys that complete a diagram. If an ARS is both confluent and terminating, a valley can always be found by reducing to normal form (but this may yield a valley with longer reductions than necessary); if an ARS has cycles, there may be an infinite number of different valleys.

Observe that the function $\mathrm{vs}_R(m, n)$ picks the *smallest* valley for each specific peak, but has to take into account all peaks with a starting term of size (at most) m and reductions of size (at most) n; thus, $\mathrm{vs}_R(m, n)$ may be larger than needed for 'most' peaks—it gives the least valley size that will *surely* work for all terms and peaks limited by m and n. The same holds for $\mathrm{cvs}_R(m, n)$, mutatis mutandis.

3.1. Computing Valley Sizes

We illustrate the definitions of vs_R and cvs_R by computing $\mathrm{vs}_R(2, 1)$ and $\mathrm{cvs}_R(2, 2)$ for a small TRS in the following example.

Example 3.2. Let R be the TRS with the rules

$$\left\{\begin{array}{lll} a \to b & b \to d & d \to e \\ a \to c & c \to a & g(x) \to h(a) \\ a \to e & d \to a & h(x) \to e \end{array}\right\}$$

This TRS is confluent[3] (and normalizing, but not terminating[4]).

Consider the peak $g(b) \leftarrow g(a) \to g(c)$. Some valleys completing this peak are:

$$g(b) \to h(a) \leftarrow g(c)$$
$$g(b) \to g(d) \to g(a) \to g(c)$$
$$g(b) \to h(a) \to e \leftarrow h(a) \leftarrow g(c)$$
$$g(b) \to g(d) \to h(a) \leftarrow g(c)$$

Observe there are an infinite number of valleys of the form $g(b) \to g(d) \to^* g(a) \to h(a) \leftarrow g(a) \ {}^*\!\leftarrow g(c)$ and, hence, there is no *largest* valley completing the diagram.

The *smallest* possible valley is the first one given above: Both reductions have length 1. Note that this valley does *not* involve normal forms, and that any valley with reductions to normal form involves strictly longer reductions.

By definition of the size of terms (Definition 2.5), the term $g(a)$ has size 2, and by inspection, we find that for any peak with reductions of length at most 1 starting from a term of size 2, there is a corresponding valley where each reduction has length at most 1. However, for terms of size 1, there is the peak $b \leftarrow a \to c$ whose smallest valleys involve reductions of length 2, e.g. $b \to d \to a \leftarrow c$. Thus, for peaks involving

[3]The system has the unique normal form property and is weakly confluent; see Terese [2003].

[4]Normalization and termination are also called, respectively, weak normalization and strong normalization.

terms of size *at most* 2 and reductions of length *at most* 1, the smallest corresponding valleys involve reductions of length *at most* 2, and there is a peak that needs a valley with reductions of length 2. Hence, $\mathrm{vs}_R(2,1) = 2$.

In case of $\mathrm{cvs}_R(2,2)$, the conversion will either be (a) a reduction of length at most 2, (b) a valley with two reductions of length 1, or (c) a peak with each reduction of length 1. In the case of (a) and (b), the considered reductions themselves are already valleys with reductions of length at most 2. In the case of (c), the valley size is equal to $\mathrm{vs}_R(2,1)$. Hence, $\mathrm{cvs}(2,2) = 2$.

As the following example shows, vs_R and cvs_R do not need to be computable for ARSs.

Example 3.3. Let $h : \mathbb{N} \longrightarrow \mathbb{N}$ be any non-computable, non-decreasing total function with $h(i) \geq 1$ for all $i \in \mathbb{N}$ (note that h must have unbounded range). Moreover, let $A = (\mathbb{N} \times \{*\}) \cup \mathbb{N}^2$ with $|(i,*)| = i$, for all $i \in \mathbb{N}$, and $|(i,j)| = i + j$, for all $(i,j) \in \mathbb{N}^2$

To see that vs_R does not need to be computable, define for every $m \geq 1$ and $n > 1$: $(m,*) \to_R (m,1)$, $(m,*) \to_R (m, h(m)+1)$, and $(m,n) \to (m,n-1)$. Then, $R = (A, \to)$ is confluent by the last rule, but $\mathrm{vs}_R(m,1) = h(m)$, as any peak with reductions of length 1 is either the peak $(m,1) \leftarrow (m,*) \to (m,h(m))$ or a reduction of length 1. Thus, we cannot compute $\mathrm{vs}_R(m,n)$.

To see that cvs_R does not need to be computable, define for every $m \geq 1$ and $n > 1$: $(m,*) \to_R (m,1)$, $(m,*) \to_R (m,0)$, $(m,0) \to_R (m, h(m)+1)$, and $(m,n) \to (m,n-1)$. Then, $R = (A, \to)$ has the Church-Rosser property by the last rule, but $\mathrm{cvs}_R(m+1,2) = h(m)+1$, as any conversion of length 2 is either the peak $(m,1) \leftarrow (m,*) \to (m,0)$, the valley $(m,*) \to (m,1) \leftarrow (m,2)$, or a reduction of length at most 2. Whence, we cannot compute $\mathrm{cvs}_R(m,n)$.

3.2. Relating vs_R and cvs_R

As an ARS is confluent iff it has the Church-Rosser property, it is natural to consider the relation between vs_R and cvs_R. The relation will be non-trivial, because the objects whose 'size' is restricted by vs_R and cvs_R are quite different: In the case of vs_R the size of the 'top of a peak' is bounded, while in the case of cvs_R the 'leftmost' and 'rightmost' objects of a conversion are bounded.

One might naïvely ponder dropping one of the object bounds from cvs_R to create a more symmetric situation. However, as the following example shows, this is not a viable option.

Example 3.4. Consider the ARS $R = (A, \to)$ with $A = \{a, c\} \cup \{b_1, b_2, \ldots\} \cup \{d_1, d_2, \ldots\}$, where $|a| = |c| = 1$ and $|b_i| = |d_i| = i$, and with the relation $\to$ defined by:

$$\left\{ \begin{array}{l} a \to d_1 \\ c \to a \end{array} \right\} \cup \left\{ \begin{array}{l} b_i \to d_i \\ c \to b_i \end{array} \;\middle|\; \text{for all } i \right\} \cup \{ d_i \to d_{i-1} \mid \text{for all } i > 1 \}$$

It is immediate that there is a conversion $a \leftrightarrow^* b_i$ of length 2 for all i, and that the only valley joining a and b_i is $a \to d_1 \leftarrow d_2 \leftarrow \cdots \leftarrow d_i \leftarrow b_i$, which has size i. Thus, if the rightmost object in the conversion were not bounded in the definition of cvs_R, then in current example $\mathrm{cvs}_R(1,2)$ would not be bounded from above by any i and, hence, not well-defined.

As an alternative, we could consider restricting the size of *all* the objects in the peaks of vs_R and the conversions of cvs_R. However, we consider this too restrictive as the growth rates of objects along reductions and conversions can be markedly different among different systems.

We now proceed by exposing an important difference between between vs_R and cvs_R. Thereafter, we derive an upper bound on vs_R given cvs_R, and vice versa.

3.2.1. Keeping the Object Size Constant. Assuming the object size m fixed, vs_R and cvs_R behave markedly different: $\mathrm{vs}_R(m,n)$ can grow unboundedly with n for some systems, while $\mathrm{cvs}_R(m,n)$ cannot. To see that $\mathrm{vs}_R(m,n)$ can grow unboundedly, consider the following example:

Example 3.5. Let R be the (obviously confluent) TRS with the rules

$$\left\{\begin{array}{c} a \to b \\ a \to f(a) \\ f(x) \to x \end{array}\right\}$$

and consider the term a of size 1.

For any $n \geq 1$, there exists a peak with a reduction of length n:

$$b \leftarrow a \to^n f^n(a)$$

Any completing valley of this peak is a reduction $f^n(a) \to^* b$. The reduction must erase all function symbols f from $f^n(a)$ and rewrite a to b. Hence, the reduction has length at least $n+1$. Besides a, the only other terms of size 1 are b and the variables, which are normal forms. Hence, $\mathrm{vs}_R(1,n) = n+1$ and vs_R thus grows unboundedly with n for fixed $m(=1)$.

For cvs_R, we have the following lemma.

LEMMA 3.6. *Let $R = (A, \to)$ be an ARS with the Church-Rosser property and let $|\cdot| : A \longrightarrow \mathbb{N}$ be a function ('size') such that for each $m \in \mathbb{N}$ the set $\{a \in A : |a| \leq m\}$ is finite. For each $m \in \mathbb{N}$ there is a number $n \in \mathbb{N}$ such that $\mathrm{cvs}_R(m,n') = \mathrm{cvs}_R(m,n)$ for all $n' \geq n$.*

PROOF. Define $l(a,b)$ as the minimal length of a conversion between a and b and set $l(a,b) = 0$ in case no conversion exists. Let $m \in \mathbb{N}$ and $A_m = \{a \in A : |a| \leq m\}$. Consider the set $B_m = \{l(a,b) : (a,b) \in A_m \times A_m\}$. As A_m is finite, B_m has an upper bound n. Hence, for any $n' \geq n$ the conversions considered in the computation of $\mathrm{cvs}_R(m,n')$ will be same ones as those considered in the computation of $\mathrm{cvs}_R(m,n)$. Whence, $\mathrm{cvs}_R(m,n') = \mathrm{cvs}_R(m,n)$ for all $n' \geq n$. □

Remark 3.7. The n from Lemma 3.6 is in general not computable, even for a fixed system R. To see this, consider any linear TRS R and assume n is computable. As we show below (Lemma 4.3), it is decidable for linear TRSs whether for two terms of size $\leq m$ a conversion of at most a certain fixed length exists in R. Hence, as we assume that n is computable, we can decide for all terms s and t in a linear TRS whether these terms are convertible: Compute $m = \max(|s|,|t|)$ and compute n for terms of size at most m; s and t are convertible whenever the pair (s,t) occurs in the finite set $h_c(j_R,m,n)$, where h_c is the computable function from Lemma 4.3 and where j_R encodes the TRS R as an integer. We now obtain a contradiction, as it is in general undecidable for linear TRSs with the Church-Rosser property whether two terms are convertible (consider e.g. Combinatory Logic).

3.2.2. Bounding vs_R by means of cvs_R. We will now derive an upper bound on vs_R given cvs_R. To mitigate the aforementioned problem regarding the difference in object sizes in the two definitions, we introduce the following function:

Definition 3.8. Let $R = (A, \to)$ be an ARS and let $|\cdot| : A \longrightarrow \mathbb{N}$ be a function ('size') such that for each $m \in \mathbb{N}$ the set $\{a \in A : |a| \leq m\}$ is finite. The *reduction upper bound* $\mathrm{rub}_R : \mathbb{N}^2 \longrightarrow \mathbb{N}$ is defined as $\mathrm{rub}_R(m,n) = l$ where l is the least non-negative integer such that for every object a with $|a| \leq m$ and every reduction $a \to^{\leq n} b$ the object b is of size at most l. If there are no objects of size at most m, define $\mathrm{rub}_R(m,n) = 0$ for all n.

The function rub_R is well-defined, as we assume all ARSs to be finitely branching. Moreover, rub_R is obviously non-decreasing in both arguments. We have the following lemma.

LEMMA 3.9. *Let $R = (A, \to)$ be an ARS with the Church-Rosser property and let $|\cdot| : A \longrightarrow \mathbb{N}$ be a function ('size') such that for each $m \in \mathbb{N}$ the set $\{a \in A : |a| \leq m\}$ is finite. Then, $\mathrm{vs}_R(m,n) \leq \mathrm{cvs}_R(\mathrm{rub}_R(m,n), 2n)$ for all $m, n \in \mathbb{N}$.*

PROOF. For all $|a| \leq m$ and all peaks $b \,{}^*\!\leftarrow a \to^* c$ with reductions of length at most n we have by definition of rub_R that b and c are of size at most $\mathrm{rub}_R(m,n)$. Moreover, $b \,{}^*\!\leftarrow a \to^* c$ is a conversion $b \leftrightarrow^* c$ of length at most $2n$. Hence, any valley $b \to^* d \,{}^*\!\leftarrow c$ will have reductions of length at most $\mathrm{cvs}_R(\mathrm{rub}_R(m,n), 2n)$, as required. □

As the following example shows, there exist ARSs for which the bound from the above lemma is tight.

Example 3.10. Consider the ARS $R = (A, \to)$ with $A = \{a_i : i \in \mathbb{Z}\}$, where $|a_i| = |i| + 1$, and with the relation $\to$ defined by:

$$\left\{ \begin{array}{l} a_i \to a_{i+1} \\ a_i \to a_{i-1} \end{array} \;\middle|\; \text{for all } i \right\}$$

Observe that there are valleys for all a_i and a_j, e.g. $a_i \to^* a_0 \,{}^*\!\leftarrow a_j$. Moreover, we obtain the smallest valley $a_i \to^* b \,{}^*\!\leftarrow a_j$ by meeting 'half way' between a_i and a_j, i.e. when one reduction is of length $\lceil |i-j|/2 \rceil$ and the other is of length $\lfloor |i-j|/2 \rfloor$. Hence, only the distance between a_i and a_j is relevant and $\mathrm{cvs}_R(m,n) = \lceil n/2 \rceil$. Moreover, as we have for all a_i that only the objects $a_{i-k}, \ldots, a_i, \ldots, a_{i+k}$ can occur occur in a peak with reductions of at most length k, it follows by exactly the same reasoning as for cvs_R that $\mathrm{vs}_R(m,n) = n$ (consider a_{i-k} and a_{i+k}). Consequently, $\mathrm{vs}_R(m,n) = \mathrm{cvs}_R(\mathrm{rub}_R(m,n), 2n)$.

It is also easy to construct an ARS with the Church-Rosser property for which the bound from the above lemma is not tight, viz. the following example.

Example 3.11. Consider the ARS $R = (A, \to)$ with $A = \{a_1, a_2, \ldots\}$, where $|a_i| = i$, and with the relation $\to$ defined by $\{a_i \to a_{i+1} \mid \text{for all } i\}$. For all objects a_i and a_j a valley exists and the smallest valley is $a_k \to^* a_l$ with $k = \min(i,j)$ and $l = \max(i,j)$. Hence, $\mathrm{cvs}_R(m,n) = \min(m-1, n)$ and $\mathrm{vs}_R(m,n) = n$. As $\mathrm{rub}_R(m,n) = m + n$, it follows for all $n > 0$ and $m > n$ that $\mathrm{vs}_R(m,n) = n = \mathrm{cvs}_R(m,n) < \mathrm{cvs}_R(m+n, 2n)$.

3.2.3. Bounding cvs_R by means of vs_R. Naïvely, one may conjecture that deriving a bound on cvs_R given vs_R is as simple as introducing the function that acts as a 'reverse' of rub_R, i.e. a function that yields the least non-negative integer l such that for every object b with $|b| \leq m$ and every reduction $a \to^{n\geq} b$ the object a is of size at most l. Unfortunately, this would require the *inverse* relation of the assumed ARS to be finitely branching, which in the opinion of the authors is unreasonable for the following reason: Any TRS with an erasing rewrite rule, i.e. a rule with a variable that occurs on its left-hand side but not on its right-hand side, induces an ARS whose inverse relation is *not* finitely branching. For example, given the rule $f(x) \to a$, we have for every term t that $f(t) \to a$.

Fortunately, it turns out that we do not have to consider all reductions $a \to^{n\geq} b$ with b of a given size; instead, we can employ the following function.

Definition 3.12. Let $R = (A, \to)$ be an ARS and let $|\cdot| : A \longrightarrow \mathbb{N}$ be a function ('size') such that for each $m \in \mathbb{N}$ the set $\{a \in A : |a| \leq m\}$ is finite. The *conversion bound* $\mathrm{cb}_R : \mathbb{N}^2 \longrightarrow \mathbb{N}$ is defined as $\mathrm{cb}_R(m,n) = l$ where l is the least non-negative integer such that for every pair of objects b, c with $|b|, |c| \leq m$ and every conversion $b \leftrightarrow^* c$ of

length at most n there exists a conversion $b \leftrightarrow^{\leq n} c$ such that $|d| \leq l$ for all objects d occurring in the conversion.

Observe that cb_R is well-defined, because $\{a \in A : |a| \leq m\}$ is finite for every m. Moreover, cb_R is obviously non-decreasing in both arguments. Intuitively, we can use cb_R to derive upper bounds, as we only need *one* conversion between a pair of objects to be able to construct a valley for the pair; that there can be many distinct conversions between two objects (or even infinitely many) is irrelevant.

We have the following.

LEMMA 3.13. *Let $R = (A, \to)$ be a confluent ARS and let $|\cdot| : A \longrightarrow \mathbb{N}$ be a function ('size') such that for each $m \in \mathbb{N}$ the set $\{a \in A : |a| \leq m\}$ is finite. Then, $\mathrm{cvs}_R(m, n) \leq h(\lfloor n/2 \rfloor, \mathrm{cb}_R(m, n), n)$ with $h(i, j, k)$ for all $i, j, k \in \mathbb{N}$ defined by:*

$$h(i, j, k) = \begin{cases} k & \text{if } i = 0 \\ h(i-1, j', k') & \text{if } i > 0 \end{cases}$$

such that in the second case $j' = \max(j, \mathrm{rub}_R(j, l'))$ and $k' = k + 2l'$ with $l' = \mathrm{vs}_R(j, k)$ and rub_R as defined in the previous section.

PROOF. Define a *half peak* as a peak with one reduction of length 0, and define a *full peak* as a peak with both reductions of positive length. We prove by induction on the number of *full peaks* p in a conversion $b \leftrightarrow^{\leq n} c$ that a valley exists with reductions of length at most $h(p, s, n)$ in case all objects in the conversion of length at most n are of size at most s. In the case $p = 0$, the conversion is either empty, of the form $\cdot \to^* \cdot$, or of the form $\cdot \to^* \cdot \ {}^*\!\leftarrow \cdot$. Hence, the conversion is already valley and the reductions in this valley have length at most n.

In the case $p = p' + 1$, consider an arbitrary full peak from the conversion $b \leftrightarrow^* c$ and observe that the reductions in this peak are of length at most n and start from an object of size at most s. By confluence, we may replace the peak by a valley with reductions of length at most $l' = \mathrm{vs}_R(s, n)$, yielding a new conversion $b \leftrightarrow^* c$ with p' full peaks. The new conversion is of length at most $n' = n + 2l'$ with each object of size at most $s' = \max(s, \mathrm{rub}_R(s, l'))$. By the induction hypothesis, a valley exists for the conversion $b \leftrightarrow^* c$ with p' peaks such that the reductions in the valley are of length at most $h(p', s', n')$, as required.

As vs_R and rub_R are non-decreasing in both the size of the objects and the length of reductions, and as $\max$ and addition over $\mathbb{N}$ are also non-decreasing in both arguments, it follows that h is non-decreasing in all arguments. Observe that any conversion $b \leftrightarrow^{\leq n} c$ has at most $\lfloor n/2 \rfloor$ full peaks, and, if $|b|, |c| \leq m$, we may assume that the objects d along any considered conversion $b \leftrightarrow^{\leq n} c$ are of size at most $\mathrm{cb}_R(m, n)$. Hence, by the first part of the current proof and as h is non-decreasing, $\mathrm{cvs}_R(m, n) \leq h(\lfloor n/2 \rfloor, \mathrm{cb}_R(m, n), n)$, as required. □

The bound derived in the above lemma cannot be tight, as $h(\lfloor n/2 \rfloor, \mathrm{cb}_R(m, n), n)$ grows unboundedly with n while cvs_R does not due to Lemma 3.6. An example from which this is apparent is the following.

Example 3.14. Consider the ARS from Example 3.11. Observe for any conversion $b \leftrightarrow^{\leq n} c$ with $|b|, |c| \leq m$ that the conversion involves objects of size at most $m + \lfloor n/2 \rfloor$ and, thus, $\mathrm{cb}_R(m, n) = m + \lfloor n/2 \rfloor$ (intuition: we may go 'beyond' a_m but we must leave enough 'room' to return to an object of size at most m). We now have

$$\begin{aligned} h(\lfloor n/2 \rfloor, \mathrm{cb}_R(m, n), n) &= h(\lfloor n/2 \rfloor, m + \lfloor n/2 \rfloor, n) \\ &= h(\lfloor n/2 \rfloor - 1, m + \lfloor n/2 \rfloor + n, 3n)\,. \end{aligned}$$

Furthermore, as $\max$ and addition are non-decreasing it follows that

$$h(\lfloor n/2 \rfloor - 1, m + \lfloor n/2 \rfloor + n, 3n) \geq \mathrm{vs}_R(m + \lfloor n/2 \rfloor + n, 3n) = 3n\,.$$

Hence, for all $n > 0$ and $m > n$, we have $\mathrm{cvs}_R(m,n) = n < 3n \leq h(\lfloor n/2 \rfloor, \mathrm{cb}_R(m,n), n)$.

4. VALLEY SIZES IN TERM REWRITING SYSTEMS

We proceed to treat first-order term rewriting systems. We start by showing that the behavior from Example 3.3 cannot be replicated in such systems: vs_R is computable for *arbitrary* term rewriting systems R; in fact it is *uniformly* computable: There is a program that, given an encoding of a confluent TRS R, returns another program computing vs_R. Similarly, there is a program that, given an encoding of a linear TRS R satisfying the Church-Rosser property, returns another program computing cvs_R. We give a formal account in the present section.

4.1. Computing Valley Sizes in Term Rewriting Systems

Recall that we consider only TRSs with a *finite* signature and a *finite* number of rules. As terms are inductively defined, it is clear that every TRS R can be recursively encoded and decoded as an integer j_R. In the remainder of the paper we assume fixed encodings and decodings of this kind.

Valley Sizes. For vs_R we have the following computability result.

LEMMA 4.1. *There is a (partial) computable function $g : \mathbb{N}^3 \longrightarrow \mathbb{N}$ such that if j_R encodes a confluent TRS R, then $\mathrm{vs}_R(m,n) = g(j_R, m, n)$ for all $m, n \in \mathbb{N}$.*

PROOF. Let P be a program that does the following: On input (j_R, m, n), P decodes j_R, builds all terms $t_1, \ldots, t_l$ of size at most m over Σ (modulo the renaming of variables), and stores them in memory. As R has a finite number of rules, each term has a finite number of one-step reducts. Thus, for each $t_i \in \{t_1, \ldots, t_l\}$, the program P may simply apply all rules from R in turn to obtain in finite time every t_i' such that $t_i \to^{\leq n} t_i'$. Next, for every pair (s_i, s_i') of such terms, P uses R to simultaneously build increasingly larger parts $(\bigcup_{0 \leq k \leq j} V_{s_i,k}, \bigcup_{0 \leq k \leq j} E_{s_i,k})$ and $(\bigcup_{0 \leq k \leq j} V_{s_i',k}, \bigcup_{0 \leq k \leq j} E_{s_i',k})$ of the reduction graphs of s_i and s_i'. As (Σ, R) is confluent, eventually a j is reached such that a term r_i exists in both $\bigcup_{0 \leq k \leq j} V_{s_i,k}$ and $\bigcup_{0 \leq k \leq j} V_{s_i',k}$. The program P stores the least such j for (s_i, s_i'). Clearly, this j is equal to the number of steps in the longest reduction of the smallest valley of s_i and s_i'. After iterating over every pair (s_i, s_i'), P takes the maximum of the stored lengths and returns it. This value is clearly $\mathrm{vs}_R(m,n)$. Thus, P computes a function $g(j_R, m, n)$ as desired. □

THEOREM 4.2. *If R is a confluent TRS, then vs_R is a total computable function.*

PROOF. By Lemma 4.1, we have $\mathrm{vs}_R(m,n) = g(j_R, m, n)$ for all $m, n \in \mathbb{N}$. That vs_R is a partial computable function follows immediately by the s-m-n Theorem [Rogers Jr. 1987]. That the function vs_R is total follows by the fact that vs_R is well-defined by the comments below Definition 3.1. □

Conversion Valley Sizes. For cvs_R, proving computability is harder than for vs_R. The main reason is that the presence of rules such as $f(x) \to a$ implies that, even though $\to$ is finitely branching, the relation $\leftarrow$ need not be finitely branching. In fact, we currently only know how to compute cvs_R for linear systems. We start by showing that we can compute the set of pairs of terms of size at most m that have a conversion of length at most n between them.

The proof below depends on the introduction of a fresh nullary function symbol $\bot$. Intuitively, $\bot$ is a placeholder that represents any possible term. Observe that the

introduction of $\bot$ entails existence of a decidable partial order on terms in the following way: $\bot \leq s$ for all s and $f(s_1, \ldots s_n) \leq f(t_1, \ldots, t_n)$ if $s_i \leq t_i$ for all $1 \leq i \leq n$. Given a rewrite rule $l \to r$, we say that s $\bot$-rewrites to t by $l \to r$, denoted $s \to_\bot t$, if there exists a one-hole context $C[]$ and a term $s' \neq \bot$ such that $s = C[s'] \leq C[\sigma(l)]$ and $t = C[\sigma(r)]$ with σ a substitution such that if $s \leq C[\tau(l)]$ for any substitution τ, then $\sigma(x) \leq \tau(x)$ for all $x \in \mathbf{vars}(l)$ (i.e. σ is the smallest substitution with respect to the partial order on terms). Observe that it is decidable whether a term s $\bot$-rewrites to a term t, and that σ is unique when restricted to $\mathbf{vars}(l)$.

LEMMA 4.3. *There is a (partial) computable function $h_{\mathrm{c}} : \mathbb{N} \longrightarrow \mathbb{N}$ such that if j_R encodes a linear TRS R, then $h_{\mathrm{c}}(j_R, m, n)$ is (an encoding as a natural number of) the finite set of pairs of terms (s, t) such that $s \leftrightarrow_R^{\leq n} t$ with s and t of size at most m.*

PROOF. Let P_{c} be a program that does the following: On input (j_R, m, n), P_{c} decodes j_R, builds all terms $t_1, \ldots, t_l$ of size at most m over Σ (modulo renaming of variables), and stores them in memory. Employing that Σ and R are finite sets, P_{c} extends Σ with a fresh nullary function symbol $\bot$ and computes the finite rule set

$$R_\bot = R \cup \{r \to \sigma_\bot^{l \to r}(l) \mid l \to r \in R\}$$

with the substitution $\sigma_\bot^{l \to r}$ defined by:

$$\sigma_\bot^{l \to r}(x) = \begin{cases} \bot & \text{if } x \in \mathbf{vars}(l) \setminus \mathbf{vars}(r) \\ x & \text{otherwise} \end{cases}$$

Observe that the definition of a rule is slightly relaxed here, as left-hand sides may be variables. Moreover, observe that each term has a finite number of one-step $\bot$-reducts, as $R_\bot$ has a finite number of rules. Using this fact, for each $t_i \in \{t_1, \ldots, t_l\}$, the program P_{c} brute-force applies all rules from $R_\bot$ under $\bot$-reduction to obtain in finite time every term t_i' over $\Sigma \cup \{\bot\}$ such that $t_i \to_\bot^{\leq n} t_i'$. Next, for every such term t_i', P_{c} computes each term $s_i \geq t_i'$ with $s_i \in \{t_1, \ldots, t_l\}$ and outputs (s_i, t_i).

This leaves to show that $s_i \leftrightarrow^{\leq n} t_i$ iff there exists a term $t_i' \leq s_i$ such that $t_i \to_\bot^{\leq n} t_i'$. Thus, suppose $s_i \leftrightarrow^k t_i$ for $k \leq n$. We prove by induction on k that there exists a term $t_i' \leq s_i$ such that $t_i \to_\bot^{\leq k} t_i'$. If $k = 0$, then $s_i = t_i$ and we may define $t_i' = t_i$, as $t_i \to_\bot^0 t_i$. If $k = k' + 1$, then either (a) $s_i \to s_i' \leftrightarrow^{k'} t_i$ or (b) $s_i \leftarrow s_i' \leftrightarrow^{k'} t_i$. In both cases, it follows by the induction hypothesis that there exists a term $t_i'' \leq s_i'$ such that $t_i \to_\bot^{\leq k'} t_i''$. Suppose that the hole of the one-hole context employed in $s_i \to s_i'$ or $s_i \leftarrow s_i'$ occurs at a position p and that the rule $l \to r$ is used. If $p \notin \mathbf{pos}(t_i'')$ or $t_i''|_p = \bot$, then $t_i'' \leq s_i$ and we can define $t_i' = t_i''$. Otherwise, $p \in \mathbf{pos}(t_i'')$ and $t_i''|_p \neq \bot$. In the case of (a), a $\bot$-redex for $r \to \sigma_\bot^{l \to r}(l)$ occurs at the position p in t_i'' and contracting this redex yields a term $t_i' \leq s_i$. In the case of (b), a $\bot$-redex for $l \to r$ occurs at the position p in t_i'' and contracting this redex yields a term $t_i' \leq s_i$. Hence, there exists a term $t_i' \leq s_i$ such that $t_i \to_\bot^{\leq k} t_i'$.

Now suppose there exists a term $t_i' \leq s_i$ such that $t_i \to_\bot^k t_i'$ for $k \leq n$. We prove by induction on k that $s_i \leftrightarrow^k t_i$. If $k = 0$, then $t_i' = t_i$ and, as no $\bot$ occurs in t_i, we have $s_i \leftrightarrow^0 t_i$. If $k = k' + 1$, then $t_i \to_\bot^k t_i'$ is of the form $t_i \to_\bot^{k-1} t_i'' \to_\bot t_i'$. The step $t_i'' \to_\bot t_i'$ employs either (a) $l \to r$ or (b) $r \to \sigma_\bot^{l \to r}(l)$. Suppose that the hole of the one-hole context employed in $t_i'' \to_\bot t_i'$ occurs at position p. In the case of (a), we have by linearity of R that there is a term $s_i' \geq t_i''$ such that $s_i' \to s_i$ (in fact, there are infinitely many such terms in case $\mathbf{vars}(l) \setminus \mathbf{vars}(r)$ is non-empty), where we contract the redex at position p using $l \to r$. In the case of (b), define the term $s_i' \geq t_i''$ as the result of contracting the redex at position p in s_i using $l \to r$, where the redex exists by

linearity of R. In both cases, $s'_i \leftrightarrow^{k'} t_i$ now follows by the induction hypothesis. Thus, we also have $s_i \leftrightarrow^{k} t_i$. □

We now have the following analogue of Lemma 4.1.

LEMMA 4.4. *There is a (partial) computable function* $h : \mathbb{N}^3 \longrightarrow \mathbb{N}$ *such that if* j_R *encodes a linear TRS* R *with the Church-Rosser property, then* $\mathrm{cvs}_R(m,n) = h(j_R, m, n)$ *for all* $m, n \in \mathbb{N}$.

PROOF. Let P be a program that does the following: On input (j_R, m, n), P decodes j_R and invokes the function h_c from Lemma 4.3 with arguments (j_R, m, n). Next, for every pair (s_i, s'_i) of terms output by h_c, the program P proceeds as the program from Lemma 4.1 and uses R to simultaneously build increasingly larger parts $(\bigcup_{0\leq k\leq j} V_{s_i,k}, \bigcup_{0\leq k\leq j} E_{s_i,k})$ and $(\bigcup_{0\leq k\leq j} V_{s'_i,k}, \bigcup_{0\leq k\leq j} E_{s'_i,k})$ of the reduction graphs of s_i and s'_i. As (Σ, R) has the Church-Rosser property, eventually a j is reached such that a term r_i exists in both $\bigcup_{0\leq k\leq j} V_{s_i,k}$ and $\bigcup_{0\leq k\leq j} V_{s'_i,k}$. The program P stores the least such j for (s_i, s'_i). Clearly, this j is equal to the number of steps in the longest reduction of the smallest valley of s_i and s'_i. After iterating over every pair (s_i, s'_i), P takes the maximum of the stored lengths and outputs it. This value is clearly $\mathrm{cvs}_R(m,n)$. Thus, P computes a function $h(j_R, m, n)$ as desired. □

THEOREM 4.5. *If* R *is a linear TRS with the Church-Rosser property, then* cvs_R *is a total computable function.*

PROOF. Identical to the proof of Theorem 4.2, but with g replaced by h. □

The above theorem also holds for *non-erasing* TRSs R, i.e. systems having no erasing rules. In the case of a non-erasing system R, define $R' = R \cup \{r \to l \mid l \to r \in R\}$ and apply the brute-force method from the first few lines of the proof of Lemma 4.1 with respect to R' to find every pair (s,t) with $s \leftrightarrow_R^{\leq n} t$ and $|s|, |t| \leq m$. By non-erasingness, all rules from R' are rules in the proper sense, except that left-hand sides may be variables, which does not pose a problem.

4.2. Majorizing Computable Functions by Valleys in Term Rewriting Systems

Above we showed for every confluent TRS R that vs_R and cvs_R are computable; thus, we have a computable upper bound on valley sizes. This is a step forwards compared to the much less restricted setting of ARSs where valley sizes are in general uncomputable, as we showed in Example 3.3.

However, naïvely, one might conjecture that an even tighter bound is obtainable for TRSs—e.g. that vs_R and cvs_R are always primitive recursive. We now proceed to show that this is not the case in a very strong sense: For every computable function $\varphi : \mathbb{N} \longrightarrow \mathbb{N}$, there is (a) a TRS R and a single term of some size $m \in \mathbb{N}$ such that $\mathrm{vs}_R(m,n) \geq \varphi(n)$ for all $n \geq 2$ and (b) a TRS R' and such that $\mathrm{cvs}_{R'}(n,n) \geq \varphi(n)$ for all $n \geq 4$. The results are obtained by encoding Turing machines as TRSs.

4.2.1. Encoding Turing Machines. We shall use the following (non-essential) constraints on the Turing machines we encode:

Definition 4.6. All Turing machines are deterministic, one-head, single-tape machines without auxiliary input and output tapes. There are no transitions to the initial state q_s, nor are there any transitions from the final state q_h. The input and tape alphabets of the Turing machines are $\{0, 1, \square\}$ where $\square$ is 'blank' as usual. All inputs are assumed to be given in unary; hence, $n \in \mathbb{N}$ is encoded as 0^n. The initial configuration of a Turing machine will always be in the initial state with the input starting in the tape cell immediately to the right of the read/write head. The machine is assumed

Rewrite rules induced by the transition rules of the Turing machine M ($\Delta_N(M)$)

(L/R)-move	Rewrite rules (for each $q \in Q$, $a \in \{0,1,\Box\}$)
$\delta(q,b) = (q',b',R)$	$q(x,by) \to q'(b'x,y)$
$\delta(q,b) = (q',b',L)$	$q(ax,by) \to q'(x,ab'y)$

Extra rules ($\Delta_E(M)$)

(L/R)-move	Extra rewrite rules (for each $q \in Q$, $a \in \{0,1,\Box\}$)
$\delta(q,\Box) = (q',b',R)$	$q(x,\rhd) \to q'(b'x,\rhd)$
$\delta(q,b) = (q',b',L)$	$q(\rhd,by) \to q'(\rhd,\Box b'y)$
$\delta(q,\Box) = (q',b',L)$	$q(ax,\rhd) \to q'(x,ab'\rhd)$ $q(\rhd,\rhd) \to q'(\rhd,\Box b'\rhd)$

$$\Delta(M) = \Delta_N(M) \cup \Delta_E(M)$$

Fig. 2. Basic encoding $\Delta(M)$ of a Turing machine M

Rule for transiting to T when the final state has been reached (†)

$$q_h(x,y) \to T$$

Rules for non-deterministic choice of a number $n \in \mathbb{N}$ ($\Delta_{\mathrm{ndt}}(M)$)

$r(x,\rhd) \to T$	$r(\rhd,y) \to q_s(\rhd,y)$
$r(x,0y) \to r(0x,y)$	$r(0x,y) \to r(x,0y)$
$r(x,0y) \to r(x,00y)$	$r(x,00y) \to r(x,0y)$

$$\Delta_C(M) = \Delta(M) \cup \{\dagger\} \cup \Delta_{\mathrm{ndt}}(M)$$

Fig. 3. Extra rules for non-deterministic choice and confluence

never to get stuck on any legal configuration, i.e. for every state $q \in Q \setminus \{q_h\}$ and every element $b \in \{0,1,\Box\}$, the transition $\delta(q,b)$ is defined.

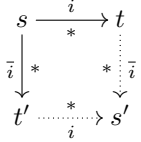
We give the standard encoding from Terese [2003]. In this encoding, the tape alphabet is modeled by unary function symbols 0, 1 and $\Box$, respectively. Moreover, both tape ends are modeled by the nullary function symbol $\rhd$. Hence, the string $01\Box1$ enclosed on the right by a tape end is represented by $0(1(\Box(1(\rhd))))$. For each state $q \in Q$ of the Turing machine we assume there exits a binary function symbol q. The position of the read/write head and tape extension are encoded in the TRS rules representing the Turing machine transitions. For a Turing machine M, the TRS $\Delta(M)$ induced by the transitions of M is given in Figure 2, omitting parentheses when no ambiguity arises.

For our purposes, we augment the signature of $\Delta(M)$ with a nullary function symbol T and a binary function symbol r. Moreover, we extend $\Delta(M)$ with the rules from Figure 3, where $\{\dagger\} \cup \Delta_{\mathrm{ndt}}(M)$ is the rule set defined by Endrullis et al. [2009, Section 5] extended with the rules $r(x,0y) \to r(x,00y)$ and $r(x,00y) \to r(x,0y)$. The result is the TRS $\Delta_C(M)$.

To prove confluence of $\Delta_C(M)$ in case where M halts on all inputs, we first state a general fact concerning mutually orthogonal systems. In the lemma, $\bar{i} = (i+1) \bmod 2$ for $i \in \{0,1\}$.

LEMMA 4.7. *Let R_0 and R_1 be mutually orthogonal TRSs. If for each $i \in \{0,1\}$ and for each peak $t \,{}^{*}_{i}\!\leftarrow s \to_i^* t'$, there exists a corresponding valley $t \to_i^* s' \,{}^{*}_{i}\!\leftarrow t'$ or $t \to_{\bar{i}}^* s' \,{}^{*}_{\bar{i}}\!\leftarrow t'$, then $R_0 \cup R_1$ is confluent.*

PROOF. As R_0 and R_1 are mutually orthogonal, R_0-reductions and R_1-reductions commute, i.e. the following diagram commutes for every $i \in \{0,1\}$:



By the conditions of the lemma, for each peak $t \,{}^{*}_{i}\!\leftarrow s \to_i^* t'$, at least one of the two diagrams below commutes for every $i \in \{0,1\}$:

Consider the relation $\to_Q = \to_0^* \cup \to_1^*$. Then, $\to_Q^* = (\to_0 \cup \to_1)^* = \to_{0\cup1}^*$, and it suffices to prove that $\to_Q$ is confluent. We make a stronger claim from which confluence will follow: $\to_Q$ has the diamond property. To see this, observe that if $t \,{}_{Q}\!\leftarrow s \to_Q t'$, then there are the four possibilities: (a) $s \to_0^* t$ and $s \to_0^* t'$, (b) $s \to_0^* t$ and $s \to_1^* t'$, (c) $s \to_1^* t$ and $s \to_0^* t'$, (d) $s \to_1^* t$ and $s \to_1^* t'$. By the assumptions, for each peak there is a corresponding valley by the one of the three diagrams above. As each reduction in a valley is either a $\to_0^*$- or a $\to_1^*$-reduction, it is in particular a $\to_Q$-step; hence, $\to_Q$ has the diamond property and $\to_{0\cup1}$ is thus confluent. □

PROPOSITION 4.8. *If the Turing machine M halts on all inputs, then the systems $R_0 = \Delta(M) \cup \{\dagger\}$ and $R_1 = \Delta_{\mathrm{ndt}}(M)$ satisfy the conditions of Lemma 4.7.*

PROOF. Both systems are left-linear and clearly no left-hand side of a rule of R_0 overlaps with a left-hand side of a rule of R_1, and vice versa; whence, the two systems are mutually orthogonal. As M is assumed to be deterministic, R_0 is orthogonal and, hence, confluent.

Observe that two rules from of $\Delta_{\mathrm{ndt}}(M)$ can only overlap at the root. As there are no collapsing rules in $\Delta_{\mathrm{ndt}}(M)$ we thus obtain confluence if every peak $t \,{}^{*}_{1}\!\leftarrow r(s,s') \to_1^* t'$ has a corresponding valley. By inspection of the rules from $\Delta_{\mathrm{ndt}}(M)$, it is seen that if $r(s,s') \to_1^* r(t,t')$, then $r(t,t') \to_1^* r(s,s')$. Thus, the only peaks of R_1 that do not have corresponding valleys in R_1 are the ones of the form

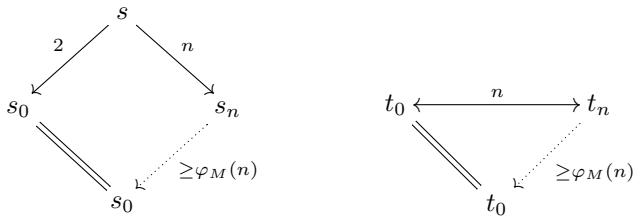
$$T \,{}^{*}_{1}\!\leftarrow r(s,s') \to_1^* q_s(\triangleright, t)\,,$$

where $s = 0^n\triangleright$ and $s' = 0^{n'}\triangleright$ for $n, n' \in \mathbb{N}$. By inspection of the rules of $\Delta_{\mathrm{ndt}}(M)$, we see that such a peak is only possible if $t = 0^n\triangleright$. As M halts on all inputs, we obtain $q_s(\triangleright, t) \to_0^* q_h(t', t) \to_0 T$, concluding the proof. □

COROLLARY 4.9. *If M halts on all inputs, then $\Delta_C(M)$ is confluent.*

4.2.2. Majorizing Computable Functions by Valleys in Term Rewriting Systems.

Valley Sizes. We now show that for every computable function $\varphi_M : \mathbb{N} \longrightarrow \mathbb{N}$, there exists a confluent TRS R and a term s such that there is a peak of size n with the



(a) Confluence (b) Church-Rosser property

Fig. 4. Majorizing a computable function φ_M by a valley

smallest corresponding valley of size at least $\varphi_M(n)$. Thus, $\mathrm{vs}_R(m,n) \geq \varphi_M(n)$ for all $m \geq |s|$.

We first prove a general lemma regarding Turing machines and employ this lemma to construct the desired TRS R.

LEMMA 4.10. *Let $\varphi_M : \mathbb{N} \longrightarrow \mathbb{N}$ be a total computable function. There is a Turing machine M' that (a) halts on all inputs and (b) halts on input 0^n in at least $\varphi_M(n)$ steps.*

PROOF. Let M' be the Turing machine containing an inlined copy of the Turing machine M. On input 0^n, let M' compute $k = \varphi_M(n)$ and then perform k 'idle steps' before halting. As M halts on all inputs, so does M' and, by construction, M' runs for at least $\varphi_M(n)$ steps before halting. □

LEMMA 4.11. *For every total computable function $\varphi_M : \mathbb{N} \longrightarrow \mathbb{N}$, there exists a confluent TRS R, a ground term s, and a ground normal form s_0 such that, for every natural number $n \geq 1$, there exists a term s_n with (a) $s_0 \,{}^2\!\leftarrow s \rightarrow^n s_n$, (b) $s_n \rightarrow^* s_0$, and (c) every $s_n \rightarrow^* s_0$ of length at least $\varphi_M(n)$ (see Figure 4(a)).*

PROOF. Let M' be the Turing machine obtained by applying Lemma 4.10 to φ_M. Then, M' halts on all inputs and halts in at least $\varphi_M(n)$ steps on input 0^n for all $n \in \mathbb{N}$. We set $R = \Delta_C(M')$, $s = r(\triangleright, 0\triangleright)$, $s_0 = T$, and $s_n = q_s(\triangleright, 0^n\triangleright)$. For all $n \in \mathbb{N}$, we then have $s \rightarrow r(0\triangleright, \triangleright) \rightarrow T$ and $s \rightarrow^n s_n$. Observe that R is confluent by Corollary 4.9 and that s is ground. By the fact that each step of $\Delta(M')$ simulates exactly one step of M', we obtain that $q_s(\triangleright, 0^n\triangleright) \rightarrow^m q_h(t,t')$ (for terms t, t') where $m \geq \varphi_M(n)$. As M' is deterministic, this is the only possible reduction from $q_s(\triangleright, 0^n\triangleright)$ to $q_h(t,t')$. Finally, we use the †-rule to obtain $q_h(t,t') \rightarrow T = s_0$. Hence, $s_n \rightarrow^* s_0$ and all such reductions are of length at least $\varphi_M(n)$. □

By Lemma 4.11, we thus have the following theorem showing that the valley size as a function of the peak size can be made to majorize any total computable function.

THEOREM 4.12. *For every total computable function $\varphi_M : \mathbb{N} \longrightarrow \mathbb{N}$, there is an explicitly constructible, confluent TRS R and an explicitly constructible ground term s of R such that $\mathrm{vs}_R(m,n) \geq \varphi_M(n)$ for all $m \geq |s|$ and $n \geq 2$.*

Conversion Valley Sizes. Analogously to the above, we next show that for every computable function $\varphi_M : \mathbb{N} \longrightarrow \mathbb{N}$, there exists a confluent TRS R' such that there is a conversion of size n between terms of size at most n with the smallest corresponding valley of size at least $\varphi_M(n)$. Thus, $\mathrm{cvs}_{R'}(n,n) \geq \varphi_M(n)$.

LEMMA 4.13. *For every total computable function $\varphi_M : \mathbb{N} \longrightarrow \mathbb{N}$, there exists a TRS R' with the Church-Rosser property, and a ground normal form t_0 of size 1 such that,*

for every natural number $n \geq 4$, there exists a term t_n of size n with (a) $t_0 \leftrightarrow^n t_n$, (b) $t_n \to^* t_0$, and (c) every $t_n \to^* t_0$ of length at least $\varphi_M(n)$ (see Figure 4(b)).

PROOF. To start, observe that similarly to the construction in Lemma 4.10, we can construct from the Turing machine M of φ_M a Turing machine M' that (a) halts on all inputs and (b) halts on input 0^n using at least $\varphi_M(n+3)$ steps: Let M' compute $\varphi_M(n+3)$ instead of $\varphi_M(n)$.

Define $R' = R'_0 \cup R'_1$ with

$$R'_0 = \Delta(M') \cup \{q_h(x,y) \to T', T \to T'\}$$
$$R'_1 = \Delta_{\mathrm{ndt}}(M')$$

where T' is a fresh nullary function symbol. Hence, compared to $\Delta_C(M')$, the right-hand side of the †-rule has been changed, and a new rule $T \to T'$ has been added. By reasoning similar to that of Proposition 4.8, it follows that R'_0 and R'_1 satisfy the conditions of Lemma 4.7, where the valley constructed for the peak $T \,{}^*_1\!\leftarrow r(s,s') \to^*_1 q_s(\triangleright, t)$ is $T \to_0 T' \,{}_0\!\leftarrow q_h(t',t) \,{}^*_0\!\leftarrow q_s(\triangleright, t)$. Hence, R' has the Church-Rosser property.

Let $n \geq 4$. Set $t_0 = T'$ and $t_n = q_s(\triangleright, 0^{n-3}\triangleright)$. We have $|t_0| = 1$ and $|t_n| = n$. Moreover, for $s = r(\triangleright, 0\triangleright)$ we have $s \to r(0\triangleright, \triangleright) \to T \to T'$ and $s \to^{n-3} t_n$. Hence, $t_0 \leftrightarrow^n t_n$. As each step of $\Delta(M')$ simulates exactly one step of M', we obtain that $q_s(\triangleright, 0^{n-3}\triangleright) \to^m q_h(t,t')$ (for terms t, t') where $m \geq \varphi_M(n-3+3) = \varphi_M(n)$. As M' is deterministic, this is the only possible reduction from $q_s(\triangleright, 0^{n-3}\triangleright)$ to $q_h(t,t')$. Finally, we use the rule $q_h(x,y) \to T'$ to obtain $q_h(t,t') \to T' = t_0$. Hence, $t_n \to^* t_0$ and all such reductions are of length at least $\varphi_M(n)$. □

We now immediately have the following theorem.

THEOREM 4.14. *For every total computable function $\varphi_M : \mathbb{N} \longrightarrow \mathbb{N}$, there is an explicitly constructible TRS R' with the Church-Rosser property such that $\mathrm{cvs}_{R'}(n,n) \geq \varphi_M(n)$ for all $n \geq 4$.*

In contrast to Theorem 4.12 where a fixed term s of fixed size was employed, the term size cannot be fixed in Theorem 4.14 because of Lemma 3.6. Observe that the TRS R' from the above theorem is linear. This implies that the result from Theorem 4.5 for linear systems cannot be improved.

5. UPPER BOUNDS ON VALLEY SIZES IN ORTHOGONAL TERM REWRITING SYSTEMS

Valley Sizes. For *orthogonal* TRSs, much better bounds can be obtained than those presented in Section 4. We shall prove existence, for every TRS R, of a constant μ_R such that $\mathrm{vs}_R(m,n) \leq n \cdot (\mu_R)^n$, where μ_R and, hence, vs_R is independent of the term size m. The constant, called the *multiplicity* of R, is defined as follows.

Definition 5.1. The *multiplicity* of a finite TRS R, denoted μ_R, is defined as:

$$\max_{l \to r \in R} \max_{x \in \mathbf{vars}(l)} (1, \text{number of occurrences of } x \text{ in } r)$$

Thus, the multiplicity of a system is simply the maximum number of times that a variable can occur on a right-hand side of a rule of R.

Example 5.2. Let $R = \{f(x,y) \to g(x,x,y),\ g(x,y,z) \to f(x,z)\}$ Then $\mu_R = 2$, as x occurs twice on the right-hand side of $f(x,y) \to g(x,x,y)$ and as no variable occurs more often.

We can now derive the touted exponential bound for orthogonal systems by enriching the standard confluence proof for orthogonal systems with reduction lengths. To do so, we first define the—standard—notion of parallel rewriting.

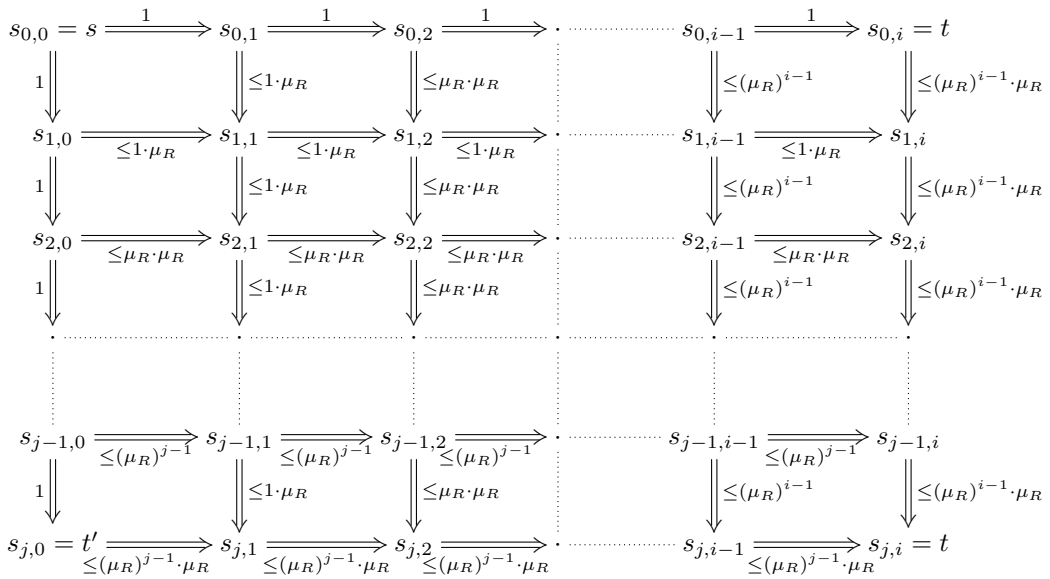


Fig. 5. Tiling diagram annotated with reduction lengths for the proof of Theorem 5.5

Definition 5.3. Let R be a TRS. The *parallel rewrite relation* $\Rightarrow$ is defined as follows: $s \Rightarrow^k t$ if there is a k-hole context such that (a) $s = C[s_1, \ldots, s_k]$, (b) $t = C[t_1, \ldots, t_k]$, and (c) for all $1 \leq i \leq k$, $s_i \to t_i$.

It is easily shown that $\Rightarrow^* = \to^*$.

LEMMA 5.4 (PARALLEL MOVES LEMMA). *Let R be an orthogonal TRS and let s be a term. If $t \; {}^{m}\!\Leftarrow s \Rightarrow^n t'$ is a peak, then there exists a valley $t \Rightarrow^{\leq n \cdot \mu_R} s' \; {}^{m \cdot \mu_R \geq}\!\Leftarrow t'$.*

PROOF. Existence of a valley follows by the standard Parallel Moves Lemma [Baader and Nipkow 1998]. The reduction $t \Rightarrow s'$ consists of a parallel contraction of the residuals of the redexes contracted in $s \Rightarrow^n t'$ across contraction of the m redexes in $s \Rightarrow^m t$, and vice versa for $t' \Rightarrow s'$. The step $s \Rightarrow^m t$ consists of m separate $\to$-steps, each contracting a single redex parallel to the other $m - 1$ redexes. By the definition of the rewrite relation $\to$, every single step using a rule $l \to r$ may copy each of its subterms by as many times as a variable occurs in r. Each of the n parallel redexes contracted in $s \Rightarrow^n t'$ may, or may not, occur inside one of the subterms copied by a redex in $s \Rightarrow^m t$. The total number of copies occurring in t is, hence, bounded from above by n times the maximum number of times that a single variable can occur in the right-hand side of a rule, hence $n \cdot \mu_R$. The case for $m \cdot \mu_R$ is symmetrical. □

THEOREM 5.5. *Let R be an orthogonal TRS and let s be a term. If $t \; {}^{j}\!\leftarrow s \to^i t'$, then there exists a valley $t \to^{\leq j \cdot (\mu_R)^i} s' \; {}^{i \cdot (\mu_R)^j \geq}\!\leftarrow t'$. Hence,* $\mathrm{vs}_R(m, n) \leq n \cdot (\mu_R)^n$.

PROOF. As every $\to$-reduction is also a $\Rightarrow$-reduction and as $\Rightarrow^* = \to^*$, repeated application of Lemma 5.4 allows us to erect the tiling diagram in Figure 5. The result now follows by tallying the number of steps on the right-most and bottom-most sides of the diagram. □

Remark 5.6. The bounds from Theorem 5.5 are tight for every non-erasing TRS R in the following sense: There is an infinite number of terms s such that $\mathrm{vs}_R(|s|, n) =$

$n \cdot (\mu_R)^n$. Let $l \to r$ be a rule such that there is a variable x in l that occurs μ_R times in r. For $j \geq 0$ let s_j be the term defined inductively by $s_0 = l$ and $s_{j+1} = l[s_j]_{p_x}$ where p_x is the (unique, by left-linearity) position of the variable x in l. For every $n \geq 1$, consider the term s_{2n} and the peak obtained by performing (a) a complete development of the n outermost redexes, and (b) the n innermost redexes, where in both cases the redexes are contracted in inside-out fashion (i.e. by iteratively contracting redexes that have no redexes below them that also need to be contracted); observe that both reductions are of length precisely n. The (a)-reduction copies the 'inner' term s_n a total of $(\mu_R)^n$ times ending in some term t. The (b)-reduction leaves exactly one copy of each of the top n redexes, ending in some term t'. To complete the confluence diagram, one needs to reach the term obtained by a complete development of all redexes in s_{2n}. From the term t', a total of n steps is required to reach this term. From the term t, reaching the final term requires the contraction of n redexes in $(\mu_R)^n$ parallel subterms, for a total of $n \cdot (\mu_R)^n$ steps.

Remark 5.7. By Proposition 2.8 any bound on $\mathrm{vs}_R(m,n)$ derived for orthogonal TRSs immediately carries over to combinator systems, which are always orthogonal. For the specific case of Combinatory Logic, inspection of the rules shows that $\mu_{CL} = 2$, as the variable x occurs twice in the right-hand side of rule $S\,x\,y\,z \to x\,z\,(y\,z)$; whence, $\mathrm{vs}_{CL}(m,n) \leq n \cdot 2^n$.

A simple construction along the lines of Remark 5.6 shows that the above bound is tight: Define $M_0 = S\,x\,y\,z$ and $M_{j+1} = S\,M_j\,y\,z$ for every $j \geq 0$. For every $n \geq 1$, consider the term M_{2n} and the peak obtained by performing (a) a complete development of the n outermost redexes, and (b) the n innermost redexes. As in Remark 5.6, both reductions have exactly n steps. Note that the only combinator occurring in any M_j is S, and that the rule associated with S is non-erasing, whence completion of the confluence diagram requires reaching the term obtained by performing a complete development of all redexes in M_{2n}; for the term at the end of the (a)-reduction, this requires contraction of $n \cdot 2^n$ redexes.

Conversion Valley Sizes. In case R is orthogonal, we can employ the bound on vs_R to derive a bound for cvs_R.

LEMMA 5.8. *Let R be an orthogonal TRS. Consider the family $s \leftrightarrow^{\leq n} t$ of conversions (for $n \geq 0$) and write $i = \lfloor n/2 \rfloor$. Then, the length $\mathtt{vl}_R(i,n)$ of the reductions in the valleys $s \to^* \cdot\, {}^*\!\leftarrow t$ of $s \leftrightarrow^{\leq n} t$ constructed as in Lemma 3.13 satisfies the following recursion inequality:*

$$\mathtt{vl}_R(i,n) \leq \begin{cases} n & \text{if } i = 0 \\ \mathtt{vl}_R(i-1, n + 2n \cdot (\mu_R)^n) & \text{if } i > 0 \end{cases}$$

PROOF. Immediate by Theorem 5.5 and Lemma 3.13, employing that Theorem 5.5 does not depend on term sizes. □

We now have the following upper bound on conversion valleys.

THEOREM 5.9. *Let R be an orthogonal TRS. There exists a function $g : \mathbb{N}^2 \longrightarrow \mathbb{N}$ such that $\mathrm{cvs}_R(m,n) \leq g(m,n)$ and*

— if $\mu_R = 1$, then g is in $\mathcal{E}_3$, the third level of the Grzegorczyk hierarchy, and
— if $\mu_R > 1$, then g is in $\mathcal{E}_4$, the fourth level of the Grzegorczyk hierarchy.

PROOF. We prove the two cases in turn.

— In case $\mu_R = 1$, observe for all $n \in \mathbb{N}$ that $n + 2n \cdot (\mu_R)^n = 3n$ and $n \cdot (\mu_R)^n = n$. Hence, by Lemma 5.8, we have $\mathrm{cvs}_R(m,n) \leq \mathtt{vl}_R(\lfloor n/2 \rfloor, n) \leq g'(\lfloor n/2 \rfloor, n)$ with

$$g'(i,n) = \begin{cases} n & \text{if } i = 0 \\ 3 \cdot g'(i-1,n) & \text{if } i > 0 \end{cases}$$

Obviously, the right-hand side of g' involves multiplication applied to limited recursion. Hence, as multiplication is in the second level of the Grzegorczyk hierarchy, we have that g' is in $\mathcal{E}_3$. The result follows by defining $g(m,n) = g'(\lfloor n/2 \rfloor, n)$.

— In case $\mu_R > 1$, observe for all $n \in \mathbb{N}$ that

$$n + 2n \cdot (\mu_R)^n \leq 3n \cdot (\mu_R)^n \leq (\mu_R)^{3n} \cdot (\mu_R)^n = (\mu_R)^{4n}$$

and

$$n \cdot (\mu_R)^n \leq (\mu_R)^n \cdot (\mu_R)^n \leq (\mu_R)^{4n}$$

Hence, by Lemma 5.8, we have $\mathrm{cvs}_R(m,n) \leq \mathtt{vl}_R(\lfloor n/2 \rfloor, n) \leq g'(\lfloor n/2 \rfloor, n)$ with

$$g'(i,n) = \begin{cases} (\mu_R)^{4n} & \text{if } i = 0 \\ (\mu_R)^{4 \cdot g'(i-1,n)} & \text{if } i > 0 \end{cases}$$

Obviously, the right-hand side of g' involves composition of multiplication and exponentiation, applied to limited recursion. Hence, as multiplication and exponentiation are, respectively, in the second and third level of the Grzegorczyk hierarchy, g' is in $\mathcal{E}_4$. The result follows by defining $g(m,n) = g'(\lfloor n/2 \rfloor, n)$. □

A natural notion of tightness of the bound in Lemma 5.8 would be that for some orthogonal TRSs $\mathrm{cvs}_R(m,n) = \mathtt{vl}_R(\lfloor n/2 \rfloor, n + 2n \cdot (\mu_R)^n)$ for all or most $m, n \in \mathbb{N}$. However, due to the observation from Lemma 3.6, the bound is unlikely to be tight in this sense. Moreover, as in the general case described in Remark 3.7, the n from Lemma 3.6 cannot be computed for (linear) orthogonal TRSs, as convertibility of two terms is undecidable even within this subclass (see Terese [2003]).

Although the derived bound will in general not be tight, it is possible that tighter bounds for $\mu_R = 1$ and $\mu_R > 1$ could exist in $\mathcal{E}_3$ and $\mathcal{E}_4$, respectively. We currently do not know whether this is the case.

6. UPPER BOUNDS ON VALLEY SIZES IN λ-CALCULUS

In λ-calculus we cannot expect valley sizes to be independent of term sizes as in the case of orthogonal TRSs (cf. Theorem 5.5), because the growth rate of terms across β-steps depends on the number of occurrences of bound variables in the original term, and hence on its size. Thus, the size of the valleys is determined by the number of copies of redexes, and $\mathrm{vs}_\Lambda(m,n)$ and $\mathrm{cvs}_\Lambda(m,n)$ must thus depend on m.

As the proof of Theorem 4.2 trivially extends to λ-calculus, vs_Λ is clearly a total computable function. However, following the approach from the previous section, an explicit upper bound on vs_Λ can be obtained by enriching one of the standard confluence proofs for λ-calculus with reduction lengths; we derive this bound below.

Unfortunately, knowing a bound on vs_Λ is insufficient to obtain a bound on cvs_Λ by means of Lemma 3.13. As vs_Λ and cvs_Λ depend on term sizes, employing the lemma requires us to establish an upper bound on the function cb_Λ from Definition 3.12. We currently do not know how to derive such an upper bound and, consequently, we do not know how to establish an upper bound on cvs_Λ either. In fact, as it seems that Lemma 4.1 does not immediately extend to λ-calculus, we do not even know whether cvs_Λ is computable. For these reasons, we focus on vs_Λ in the remainder of this section.

Of the many available methods of proving λ-calculus confluent, we believe that the one most amenable to analysis of reduction lengths is the method of 'tiling peaks' with

commuting squares of so-called *complete developments* of sets of redexes in a single term; the construction is essentially the same as the one depicted in Figure 5 (indeed, the figure is often called a *tiling diagram* [Terese 2003]), except that for λ-calculus, the 'parallel reduction' relation used in each square is replaced by a complete development of a set of redexes in a single term. An analysis of this proof reveals $\mathrm{vs}_\Lambda(m,n)$ to be bounded from above by a function in $\mathcal{E}_4$, the fourth level of the Grzegorczyk hierarchy. Indeed, considering the special case of the so-called Strip Lemma where one reduction in the peak has length 1 and the other length k (see Lemma 6.3), naïve analysis yields a bound $|M_{i,0}|^{2^{2\cdot|M_{i,0}|^{2^k}+k}}$ for the length of the reduction $M_{i+1,0} \to^* M_{i+1,k}$. We give a somewhat better bound in the present section; this bound is still in $\mathcal{E}_4$, but much less than the bound obtained by naïve analysis: $|M_{i,0}|^{2^{2^k}+k+2}$ for the Strip Lemma.

6.1. Upper Bounds for the Strip Lemma

To establish upper bounds on the length of the reductions in the Strip Lemma, we first prove two auxiliary lemmas concerning positions in λ-terms.

In the proofs of the lemmas, we write $p \parallel q$ if p and q are incomparable with respect to the prefix order on positions. Moreover, if $C[]$ is a one-hole context with the hole occurring at position q, then we write $C[]_q$. Finally, if $P = C[(\lambda x.M)\,N]_q \to_\beta C[M\{N/x\}]_q = Q$, then we say that there is a *redex* at position q in P.

LEMMA 6.1. *Let $M_0 \to M_1 \to \cdots M_{n-1} \to M_n$ be a reduction of length $n \geq 0$, and let u be a redex in M_0. For each position $p \in \mathbf{pos}(M_n)$, at most 2^n residuals of u occur in M_n at prefix positions of p.*

PROOF. We prove the result by induction on n. If $n = 0$, then only a single copy of u occurs in $M_n = M_0$, and the result follows.

If $n = n'+1$, let the redex contracted in $M_{n'} \to_\beta M_n$ be $C[(\lambda x.M)\,N]_q \to_\beta C[M\{N/x\}]_q$ and let $p \in \mathbf{pos}(M_n)$. If $p \preceq q$ or $p \parallel q$, there are at most $2^{n'} < 2^n$ residuals of u above p by the induction hypothesis. If $q \prec p$, then either the number of residuals of u above p is bounded by the number of residuals of u above (a) the position $q \cdot 0 \cdot 0 \cdot q'$ with $q' \in \mathbf{pos}(M)$ and $p = q \cdot q'$ in case $M|_{q'} \neq x$ or (b) the positions $q \cdot 0 \cdot 0 \cdot q'$ and $q \cdot 1 \cdot p'$ with $q' \in \mathbf{pos}(M)$, $p' \in \mathbf{pos}(N)$, and $p = q \cdot q' \cdot p'$ in case $M|_{q'} = x$. By the induction hypothesis, we have in the case of (a) that the number of residuals is at most $2^{n'} < 2^n$ and in the case of (b) that the number is at most $2^{n'} + 2^{n'} = 2 \cdot 2^{n'} = 2^{n'+1} = 2^n$ (observe that, although there may be many copies of N in $M\{N/x\}$, no copy of N will occur above any another copy of N in $M\{N/x\}$). □

LEMMA 6.2. *Let M be a term and U a set of redexes in M. Suppose for each $p \in \mathbf{pos}(M)$ that at most $i \geq 0$ redexes from U occur at prefix positions of p. Then, contracting all redexes in U yields a term of at most size $|M|^{2^{2\cdot(i+1)}}$.*

PROOF. We prove the result by induction on i, observing that there are at most $|M|$ redexes in U.

If $i = 0$, then for every $p, q \in U$, we have $p \parallel q$. Furthermore, contracting a single redex can produce a term of size at most $|M|^2$. Hence, the total size of the term obtained by contracting all redexes in U is $|M| \cdot |M|^2$, and as there are at most $|M|$ positions above or parallel to the redexes of U, we obtain a term of size at most $|M| + |M| \cdot |M|^2 \leq |M|^{2^2}$.

If $i = i' + 1$, then by the Finite Developments Theorem, we may contract the redexes of U in any order to obtain the unique final term. In particular, we may contract the redexes in an inside-out fashion (i.e. by iteratively contracting redexes that have no redexes below them that also need to be contracted). Hence, consider the subterms of M immediately below the outermost redexes in U. By the induction hypothesis, perform-

ing an inside-out reduction of redexes in these subterms yields terms of size at most $|M|^{2^{2\cdot(i'+1)}}$. Contracting (the residual of) an outermost redex in U after reducing all of the subterms below it can thus yield a term of size at most $(|M|^{2^{2\cdot(i'+1)}})^2 = |M|^{2^{2\cdot(i'+1)+1}}$.

There are at most $|M|$ outermost redexes in U, and there are at most $|M|$ positions of M parallel to or above all redexes of U. Hence, the total size of the term obtained after contracting all redexes of U is at most

$$|M| + |M| \cdot |M|^{2^{2\cdot(i'+1)+1}} \leq |M| \cdot |M| \cdot |M|^{2^{2\cdot(i'+1)+1}} \leq |M|^{2^{2\cdot(i'+1)+2}} = |M|^{2^{2\cdot(i+1)}},$$

concluding the proof. □

We can now establish our result for the Strip Lemma.

LEMMA 6.3 (STRIP LEMMA). *Let $k \geq 1$ and consider the peak*

$$M_{i+1,0} \;{}_\beta\!\leftarrow M_{i,0} \to_\beta M_{i,1} \to_\beta M_{i,2} \to_\beta \cdots M_{i,k-1} \to_\beta M_{i,k}\,.$$

Then, a valley may be obtained by tiling the peak using the Finite Developments Theorem in the following way:

$$\begin{array}{ccccccccc}
M_{i,0} & \xrightarrow{1} & M_{i,1} & \xrightarrow{1} & M_{i,2} & \cdots & M_{i,k-1} & \xrightarrow{1} & M_{i,k} \\
\downarrow{\scriptstyle 1} & & \vdots{\scriptstyle *} & & \vdots{\scriptstyle *} & & \vdots{\scriptstyle *} & & \vdots{\scriptstyle *} \\
M_{i+1,0} & \overset{*}{\cdots\!\to} & M_{i+1,1} & \overset{*}{\cdots\!\to} & M_{i+1,2} & \cdots & M_{i+1,k-1} & \overset{*}{\cdots\!\to} & M_{i+1,k}
\end{array}$$

where the following holds for $1 \leq j \leq k$:

(1) $|M_{i,j}| \leq |M_{i,0}|^{2^j}$ and $|M_{i+1,j}| \leq |M_{i,0}|^{2^{2^{j+1}+j+2}}$, and
(2) the reduction $M_{i+1,j-1} \to_\beta^* M_{i+1,j}$ has length at most $|M_{i,0}|^{2^{2^j+j+1}}$.

Moreover, the reduction $M_{i+1,0} \to_\beta^* M_{i+1,k}$ has length at most $|M_{i,0}|^{2^{2^k+k+2}}$.

PROOF. If $P \to_\beta Q$, then $|Q| \leq |P|^2$. Hence, straightforward induction shows that $|M_{i,j}| \leq |M_{i,0}|^{2^j}$. Let u be the redex contracted in $M_{i,0} \to_\beta M_{i+1,0}$. By Lemma 6.1, the number of residuals of u along any path from the root of $M_{i,j}$ to a leaf is at most 2^j.

Observe that any reduction $M_{i,j} \to_\beta^* M_{i+1,j}$ is a complete development of $U = u/(M_{i,0} \to_\beta^* M_{i,j})$. Then, Lemma 6.2 and the first part of the current lemma yield

$$|M_{i+1,j}| \leq (|M_{i,0}|^{2^j})^{2^{2\cdot(2^j+1)}} = |M_{i,0}|^{2^j \cdot 2^{2^{j+1}+2}} = |M_{i,0}|^{2^{2^{j+1}+j+2}}.$$

The reduction $M_{i+1,j-1} \to_\beta^* M_{i+1,j}$ is a complete development of a set of residuals of the single redex contracted in $M_{i,j-1} \to_\beta M_{i,j}$, and an inside-out development has length bounded from above by the size of $M_{i+1,j-1}$; by the above, that size is at most $|M_{i,0}|^{2^{2^j+j+1}}$. Hence, the length of the entire bottom reduction $M_{i+1,0} \to_\beta^* M_{i+1,k}$ is bounded from above by

$$\sum_{j=1}^{k} |M_{i,0}|^{2^{2^j+j+1}} \leq 2 \cdot |M_{i,0}|^{2^{2^k+k+1}} \leq |M_{i,0}|^{2^{2^k+k+2}},$$

where the first inequality follows once we observe that $2 \cdot |M_{i,0}|^{2^{2^{j'}+j'+1}} \leq |M_{i,0}|^{2^{2^j+j+1}}$ in case $j' = j-1$. □

6.2. Valley Sizes in λ-Calculus are in $\mathcal{E}_4$

Using the Strip Lemma, we can now prove the following lemma.

LEMMA 6.4. *Consider the following family of peaks (for $l, k \geq 0$):*

$$M_{l,0} \ {}_\beta\!\leftarrow \cdots {}_\beta\!\leftarrow M_{1,0} \ {}_\beta\!\leftarrow M_{0,0} \rightarrow_\beta M_{0,1} \rightarrow_\beta \cdots \rightarrow_\beta M_{0,k}$$

and write $m = |M_{0,0}|$. Then, in the tiling of the peak with complete developments, the length $\mathtt{bl}\,(l,k,m)$ of the bottom side of the tiling diagram satisfies the following recursion inequality:

$$\mathtt{bl}\,(l,k,m) \leq \begin{cases} k & \text{if } l = 0 \\ m^{2^{2^{\mathtt{bl}(l-1,k,m)+\mathtt{bl}(l-1,k,m)+l+1}}} & \text{if } l > 0 \end{cases}$$

PROOF. The tiling diagram may be viewed as l copies of the Strip Lemma (horizontal tiling) stacked on top of each other. The result now follows by a simple induction employing Lemma 6.3 (observe for $1 \leq i \leq l$ that the upper left term in the ith copy of the Strip Lemma has size $|M_{i,0}| \leq m^{2^{2^{i-1}}}$). □

We can now prove:

THEOREM 6.5. *There exists a function $g : \mathbb{N}^2 \longrightarrow \mathbb{N}$ in $\mathcal{E}_4$, the fourth level of the Grzegorczyk hierarchy, such that $\mathrm{vs}_\Lambda(m,n) \leq g(m,n)$.*

PROOF. The right-hand side of the recurrence equation of Lemma 6.4 involves composition of addition, multiplication, and exponentiation, applied to limited recursion on the function $\mathtt{bl}\,(l,k,m)$ being defined. As addition, multiplication and exponentiation are at the first, second, and third levels of the Grzegorczyk hierarchy, hence a fortiori in $\mathcal{E}_3$, the function g, defined as $g(m,n) = \mathtt{bl}\,(n,n,m)$, is in $\mathcal{E}_4$. □

The reader should note that while performing *projections* of the reductions in a peak across each other (i.e. by repeatedly applying the Strip Lemma as in the above theorem) may yield reductions of extreme length, the projections will often be reductions between terms that also have *short* reductions between them and, hence, will only give rise to 'small' values of vs_Λ. Therefore, it does not automatically follow from Theorem 6.5 that reductions exist that saturate the above bound.

6.3. Attempting to Saturate the Upper Bound

It is interesting to investigate whether the inequality from Lemma 6.4 is tight. We followed two approaches to investigate this question. Firstly, we wrote an implementation of the program from the proof of Theorem 4.2 for λ-calculus. Unfortunately, due to the combinatorial explosion in the number of terms with growing term sizes and the number of reductions with increasing reduction lengths, the program quickly ran out of memory for any non-trivial combination of term size and reduction length.

Secondly, we attempted to manually construct terms and reductions that approach the computed bound. The construction presented below is the best example we were able to devise that comes 'close' to the upper bound from the lemma. Observe that the example is only doubly exponential, which leads us to believe that the inequality from Lemma 6.4 is, in fact, *not* tight.

To start, define for any variable x and positive integer k the term x^k as follows:

$$x^k = \begin{cases} x & \text{if } k = 1 \\ x\,x^{k-1} & \text{if } k > 1 \end{cases}$$

Let $\{x_0, x_1, \ldots\}$ be a denumerably infinite set of distinct variables. Define for integers $n \geq 0$ and $k \geq 1$ the term $\mathrm{M}_{k,n}$ as follows:

$$\mathrm{M}_{k,n} = \begin{cases} x_0^k & \text{if } n = 0 \\ (\lambda z.z\,(z\,x_n))\,(\lambda x_{n-1}.\mathrm{M}_{k,n-1}) & \text{if } n > 0 \end{cases}$$

Observe that $|\mathrm{M}_{k,0}| = 2k - 1$ and $|\mathrm{M}_{k,0}| = 8 + |\mathrm{M}_{k,n-1}|$ for $n > 0$, whence $|\mathrm{M}_{k,n}| = 8n + 2k - 1$ for $n \geq 0$.

We claim that $\mathrm{M}_{k,n} \to_\beta^{3n} t$ where t is a term that contains x_n as a free variable at k^{2^n} distinct positions. To prove the claim, use induction on n: In case $n = 0$, we have $\mathrm{M}_{k,0} = x_0^k$, which clearly contains k free copies of x_0. In case $n > 0$, observe that by the induction hypothesis

$$\mathrm{M}_{k,n} = (\lambda z.z\,(z\,x_n))\,(\lambda x_{n-1}.\mathrm{M}_{k,n-1}) \to_\beta^{3(n-1)} (\lambda z.z\,(z\,x_n))\,(\lambda x_{n-1}.N')$$

where N' contains $k^{2^{n-1}}$ free copies of x_{n-1}. Now,

$$\begin{aligned}(\lambda z.z\,(z\,x_n))\,(\lambda x_{n-1}.N') &\to_\beta (\lambda x_{n-1}.N')\,((\lambda x_{n-1}.N')\,x_n) \\ &\to_\beta (\lambda x_{n-1}.N')\,(N'\{x_n/x_{n-1}\}) \\ &\to_\beta N'\{(N'\{x_n/x_{n-1}\})/x_{n-1}\} = N\end{aligned}$$

Thus, N has $k^{2^{n-1}} \cdot k^{2^{n-1}} = k^{2\cdot 2^{n-1}} = k^{2^n}$ free copies of x_n, and

$$\mathrm{M}_{k,n} \to_\beta^{3(n-1)} (\lambda z.z\,(z\,x_n))\,(\lambda x_{n-1}.t') \to_\beta^{3} N\,,$$

concluding the proof of the claim.

Consider the term $(\lambda x_n.\mathrm{M}_{k,n})\,((\lambda y.y)\,y')$ and the peak

$$(\lambda x_n.\mathrm{M}_{k,n})\,y' \;{}_\beta\!\leftarrow (\lambda x_n.\mathrm{M}_{k,n})\,((\lambda y.y)\,y') \to_\beta \mathrm{M}_{k,n}\{(\lambda y.y)\,y'/x_n\} \to_\beta^{3n} N\{(\lambda y.y)\,y'/x_n\}\,,$$

where N is as above. By construction, k^{2^n} copies of $(\lambda y.y)\,y'$ occur in $N\{(\lambda y.y)\,y'/x_n\}$ and, as these are the only redexes that occur in this term, any valley that completes the peak must contract all these redexes. The other reduction of the completing valley is

$$(\lambda x_n.\mathrm{M}_{k,n})\,y' \to_\beta \mathrm{M}_{k,n}\{y'/x_n\} \to_\beta^{3n} N\{y'/x_n\}\,.$$

Hence, for any $k \geq 2$, the reductions in the valley are bounded by k^{2^n}, which is significantly smaller than $\mathtt{bl}\,(1, 3n+1, |\mathrm{M}_{k,n}| + 6) = \mathtt{bl}\,(1, 3n+1, 8n+2k+5)$.

7. CONCLUSION AND CONJECTURES

We have performed the first fundamental study of the size of confluence and Church-Rosser diagrams in term rewriting and λ-calculus, and have derived bounds for the valleys in confluence diagrams for general and orthogonal systems.

The work reported in the paper has raised a number of questions and conjectures:

— The dependence on term size $|s|$ in the bound given for arbitrary TRSs in Section 4 can possibly be removed; we are currently unable to do so.

— Our inability to construct terms in λ-calculus that saturate the upper bounds derived in Section 6 suggests that vs_Λ may be in $\mathcal{E}_3$. Similarly, we conjecture that conversion valley sizes for orthogonal TRSs are in $\mathcal{E}_3$.

— We conjecture that conversion valley sizes for arbitrary TRSs and λ-calculus are computable. For conversion valley sizes in λ-calculus, the derivation of conversion valley sizes for orthogonal TRSs, combined with the $\mathcal{E}_4$ bound on valley sizes in λ-calculus, suggests a bound in at least $\mathcal{E}_5$, the fifth level of the Grzegorczyk hierarchy.

— The question of valley sizes for higher-order rewriting systems must be investigated; bounds for such systems will automatically lead to bounds for deduction systems in first- and higher order logics, as well as for higher-order functional programs.

ACKNOWLEDGMENTS

The authors would like to thank the following people: Andrzej Filinski and Richard Statman for providing answers to our inquiries, Clemens Grabmayer for pointing out a small error, and Vincent van Oostrom and the anonymous reviewers for comments.

References

Franz Baader and Tobias Nipkow. 1998. *Term Rewriting and All That*. Cambridge University Press, Cambridge.

Hendrik Pieter Barendregt. 1985. *The Lambda Calculus: Its Syntax and Semantics* (rev. ed.). Studies in Logic and the Foundations of Mathematics, Vol. 103. North-Holland, Amsterdam.

Arnold Beckmann. 2001. Exact Bounds for Lengths of Reductions in Typed λ-Calculus. *Journal of Symbolic Logic* 66, 3 (2001), 1277–1285.

Ugo Dal Lago and Simone Martini. 2008. The weak lambda calculus as a reasonable machine. *Theoretical Computer Science* 398, 1–3 (2008), 32–50.

Ugo Dal Lago and Simone Martini. 2009. On Constructor Rewrite Systems and the Lambda-Calculus. In *Proceedings of the 36th International Colloquium on Automata, Languages and Programming (ICALP 2009), Part II (Lecture Notes in Computer Science)*, Vol. 5556. Springer, Berlin, 163–174.

Roel de Vrijer. 1985. A Direct Proof of the Finite Developments Theorem. *Journal of Symbolic Logic* 50, 2 (1985), 339–343.

Jörg Endrullis, Herman Geuvers, and Hans Zantema. 2009. Degrees of Undecidability in Term Rewriting. In *Proceedings of the 23rd International Workshop on Computer Science Logic (CSL 2009) (Lecture Notes in Computer Science)*, Vol. 5771. Springer, Berlin, 255–277.

Maribel Fernández. 2009. *Models of Computation: An Introduction to Computability Theory*. Springer, New York.

Andrzej Grzegorczyk. 1953. *Some Classes of Recursive Functions*. Rozprawy Matematyczne (= Dissertationes Mathematicae), Vol. 4. Polish Academy of Sciences, Warsaw.

Neil D. Jones. 1997. *Computability and Complexity from a Programming Perspective*. The MIT Press, Cambridge, MA.

Jeroen Ketema and Jakob Grue Simonsen. 2010. Least Upper Bounds on the Size of Church-Rosser Diagrams in Term Rewriting and λ-Calculus. In *Proceedings of the 10th International Symposium on Functional and Logic Programming (FLOPS 2010) (Lecture Notes in Computer Science)*, Vol. 6009. Springer, Berlin, 272–287.

Zurab Khasidashvili. 1988. β-reductions and β-developments of λ-terms with the least number of steps. In *Proceedings of the International the Conference on Computer Logic (COLOG-88) (Lecture Notes in Computer Science)*, Vol. 417. Springer, Berlin, 105–111.

Jan Willem Klop. 1992. Term Rewriting Systems. In *Handbook of Logic in Computer Science*, S. Abramsky, D. Gabbay, and T. Maibaum (Eds.). Vol. 2. Oxford University Press, Oxford, 1–116.

Julia L. Lawall and Harry G. Mairson. 1996. Optimality and Inefficiency: What Isn't a Cost Model of the Lambda Calculus?. In *Proceedings of the 1st ACM SIGPLAN International Conference on Functional Programming (ICFP'96)*. ACM, New York, 92–101.

Julia L. Lawall and Harry G. Mairson. 1997. On Global Dynamics of Optimal Graph Reduction. In *Proceedings of the 2nd ACM SIGPLAN International Conference on Functional Programming (ICFP'97)*. ACM, New York, 188–195.

Piergiorgio Odifreddi. 1999. *Classical Recursion Theory, volume II*. Studies in Logic and the Foundations of Mathematics, Vol. 143. North-Holland, Amsterdam.

Christos Papadimitriou. 1994. *Computational Complexity*. Addison-Wesley, Reading, MA.

Hartley Rogers Jr. 1987. *Theory of Recursive Functions and Effective Computability*. The MIT Press, Cambridge, MA.

Helmut Schwichtenberg. 1982. Complexity of Normalization in the Pure Typed Lambda-Calculus. In *The L.E.J. Brouwer Centenary Symposium: Proceedings of the Conference held in Noordwijkerhout (Studies in Logic and the Foundations of Mathematics)*, Vol. 110. North-Holland, Amsterdam, 453–457.

Helmut Schwichtenberg. 1991. An upper bound for reduction sequences in the typed λ-calculus. *Archive for Mathematical Logic* 30, 5–6 (1991), 405–408.

Michael Sipser. 2006. *Introduction to the Theory of Computation* (2nd ed.). Thomson Course Technology, Boston.

Morten Heine Sørensen. 1996. Effective Longest and Infinite Reduction Paths in Untyped λ-Calculi. In *Proceedings of the 21st International Colloquium on Trees and Algebra in Programming (CAAP'96) (Lecture Notes in Computer Science)*, Vol. 1059. Springer, Berlin, 287–301.

Morten Heine Sørensen. 2007. A Note on Shortest Developments. *Logical Methods in Computer Science* 3, 4:2 (2007), 8 pages.

Jan Springintveld. 1993. Lower and Upper Bounds for Reductions of Types in $\lambda\underline{\omega}$ and λP (Extended abstract). In *Proceedings of the 1st International Conference on Typed Lambda Calculi and Applications (TLCA'93) (Lecture Notes in Computer Science)*, Vol. 664. Springer, Berlin, 391–405.

Richard Statman. 1979. The Typed Lambda Calculus is not Elementary Recursive. *Theoretical Computer Science* 9 (1979), 73–81.

Terese. 2003. *Term Rewriting Systems*. Cambridge Tracts in Theoretical Computer Science, Vol. 55. Cambridge University Press, Cambridge.

Steffen van Bakel and Maribel Fernández. 2003. Normalization, approximation, and semantics for combinator systems. *Theoretical Computer Science* 290, 1 (2003), 975–1019.

Hongwei Xi. 1999. Upper bounds for standardizations and an application. *Journal of Symbolic Logic* 64, 1 (1999), 291–303.

Received ?; revised ?; accepted ?