Transformational Approach to Inverse Computation in Term Rewriting

January 2004

Naoki Nishida

Graduate School of Engineering, Nagoya University

submitted to Nagoya University
for the degree of Doctor of Engineering

Abstract

In this thesis, we approach by program transformation to inverse computation in term rewriting.

Inverse computation of a given program (or function) is the calculation of the possible input of the program for a given output. Inverse interpreters are algorithmic approaches to inverse computation, which computes the possible input from a given program and a given output. Inverse compilers are transformational approaches, which generates from a given program a program computing the possible input of the given program for a given output. Such a generated program is called an inverse program of the given program. Once the inverse program is generated for the given program, inverse computation can be done by the inverse program inputing a given output of the original program.

In this thesis, we propose inverse compilers for term rewriting systems (TRSs, for short). As the first step, we treat a class of TRSs, called PT-TRSs. PT-TRSs can define pure treeless functions which are often used in program transformations and have the feature that each argument of defined symbols appearing in the right-hand side is a variable. While they can be easily treated in program transformation, their syntax is very restrictive and the class of functions defined as PT-TRSs is narrow. Hence we extend the inverse compiler for PT-TRSs to that for constructor TRSs. Constructor TRSs includes PT-TRSs properly, and it is known that their computation power is equivalent to that of Turing machines.

Both of the inverse compilers proposed in this thesis are consisting of two parts. The first part is an actual inverse compiler that generates from a given TRS a conditional TRS (CTRS) which computes inverse of the given TRS. We show the correctness of our inverse compilers by proving that the generated CTRSs are inverse programs of the given TRSs. The second part is a program transformation of the generated CTRS into a TRS with extra variables, which is considered to be equivalent to the generated CTRS. TRSs with extra variables are called EV-TRSs, which can contain extra variables in the right-hand sides of

their rewrite rules. Extra variables are variables which appear in the conditional part or the right-hand side of a conditional rewrite rule but not in the left-hand side. In order to show the correctness of the program transformation in the second part, we show soundness and completeness of the generated EV-TRSs with respect to the terms which are treated by the input CTRSs.

We also show the general relationships between the syntactic properties of the input TRS and its inverse EV-TRS generated by our inverse compilers. Three of such relationships are very important. One is that the generated EV-TRSs are constructor systems. A second is that the generated EV-TRS is a TRS if the input TRS is non-erasing. The other is that the generated EV-TRS is right-linear (left-linear) if the input is left-linear (right-linear, respectively).

EV-TRSs contain two serious problems for practical use of them. One is that their one-step of rewrite relation by a rule which contains an extra variable is infinitely branching even if renamed terms are regarded as the same. The other is that they are not terminating at all if extra variables are contained. Therefore, EV-TRSs in general cannot be executed efficiently. Unfortunately, the inverse programs generated by our inverse compilers are EV-TRSs. Therefore a simulation method of rewrite relation of EV-TRSs is necessary.

To simulate rewrite relation of EV-TRSs, we show that narrowing substituting a fresh variable for each extra variable can simulate, as a narrowing sequence starting from ground terms, a rewrite sequence of a given EV-TRS if the EV-TRS is right-linear or any redex which is reduced in the rewrite sequence is not introduced by means of extra variables. Since narrowing is finitely branching up to renaming and those starting from ground terms are sometimes terminating, the above problems of EV-TRSs are avoided. Since the narrowing of TRSs on ground terms is equal to the rewrite relation of them and every rewrite sequence of TRSs can be considered as a ground rewrite sequence, the computation of TRSs can be also done by the narrowing. Hence, the input and output programs of our inverse compilers can be worked on the same interpretation.

Termination is one of the important properties of computation. Since the computation of our inverse EV-TRSs is simulated by narrowing, to guarantee the termination of the inverse computation by our approaches, we propose two termination proof methods of narrowing, especially starting from ground terms. One is based on the notion of active chains, and another is based on the dependency pair method which is a termination proof method for rewrite relations of TRSs.

Solutions of inverse computation are not always unique, and hence all solution search is necessary. However, it is known that all solution search is expensive. For the purpose of improving efficiency of inverse computation, we show that basic narrowing is sufficient to simulate rewrite sequences of right-linear EV-TRSs. Since the left-linearity is often assumed in first-order functional programs and inverse EV-TRSs of linear constructor TRSs are right-linear, this result is useful for the efficience of our inverse computation. We also show that for linear constructor EV-TRSs, all normal forms of a given terminating term can be obtained by innermost narrowing.

To generate innermost-terminating inverse TRSs, we give a condition on the relationship between the depth of the both hand sides of each rewrite rules in the input non-erasing TRSs. The condition for each rule is that

- the root symbol of the right-hand side is not a defined symbol,
- the depth of the left-hand side is two or less if the right-hand side is a variable, or less than the number of constructors in the shortest path from the root to variables or defined symbols in the right-hand side.

Since every redex is innermost in rewrite sequences representing inverse computation, innermost termination is sufficient as termination of inverse computation.

As an extension of the proposed inverse compilers, we tackle partial inverse computation. Partial inverse computation is the calculation of some of the possible unknown input for a program, a given output and the known input. We extend the inverse compiler for constructor TRSs into a compiler which generate a CTRS computing partial inverses of the input TRS.

Finally, we give some examples as applications of the generated inverse compilers. These shows that inverse compilers are advantageous over inverse interpreters.

Acknowledgment

I would like to thank my principal advisor Professor Toshiki Sakabe of Nagoya University from the bottom of my heart for his kind guidance and encouragement. I am deeply grateful to my advisor Professor Masahiko Sakai of Nagoya University for his helpful discussions and suggestions during this work. I would like to express my gratitude to my advisor Lecturer Keiichirou Kusakari for his useful

discussions. I also thank Professor Naofumi Takagi of Nagoya University for his advises.

I wish to thank Professor Yoshihito Toyama of Tohoku University, Professor Aart Middeldorp of the University of Innsbruck, Associate Professor Robert Glück of the University of Copenhagen, and Associate Professor Zhenjiang Hu of the University of Tokyo, for their advises.

I am thankful to all of my colleagues.

Contents

List of Figures

Chapter 1

Introduction

Inverse computation is an important and useful concept in many fields, and an *inverse operator* is one of the primitive operators in programming.

In this thesis, we approach to inverse computation in term rewriting from a transformational approach, that is, we propose inverse compilers for term rewriting systems (TRSs). We also analyze syntactic properties of generated programs. That is, we show some relationships between syntactic properties of input TRSs and generated programs.

As an inverse program, our inverse compilers generate a TRS, called an EV-TRS, which may have extra variables that are variables appearing only in the right-hand sides of rewrite rules. This fact causes serious problems for practical inverse computation; The reduction by rewrite relation of EV-TRSs are infinitely branching even if renamed terms are regarded as the same, and they are not terminating if containing at least one extra variable.

To solve such problems, we show that computation of the generated programs can be simulated by narrowing sequences starting from ground terms. It is known that narrowing (even on EV-TRSs) is finitely branching if the renaming terms are regarded as the same. On the other hand, simulating ground rewrite relations of EV-TRSs by narrowing brings the termination for computation by EV-TRSs. Although narrowing is not terminating even on TRSs, narrowing starting from ground terms is terminating for some EV-TRSs. Hence, it is said that the problems — infinitely branching and non-termination — can be solved.

In the next section, we introduce a problem which motivates us to approach by program transformation to inverse computation.

1.1 Motivation

In term rewriting theories, we sometimes treat programs which defines functions by rewrite rules with left-hand sides in which function symbols are nested, while such a definition is not allowed in first-order functional programs. Such programs are difficult to execute automatically for both human beings and computers, even if they are correct in the sense of the semantics. The reason is that a certain kind of reasoning is required.

Consider a program (function) *gcd* which computes the greatest common divisor (G.C.D.) of two natural numbers. The function *gcd* is often defined as follows;

$$\begin{cases} gcd(\, x+y, \quad y \quad) = gcd(\, x, y\,) \\ gcd(\quad y, \quad x+y\,) = gcd(\, x, y\,) \\ gcd(\quad x, \quad\; 0 \quad) = \quad x \\ gcd(\quad 0, \quad\; x \quad) = \quad x\,. \end{cases}$$

The above definition is correct in the sense that it is based on Euclid's algorithm. However, it is not executable automatically. For example, $gcd(4, 2)$ cannot output the desirable solution 2. It is because $gcd(4, 2)$ does not syntactically match with the left-hand side $g(x + y, x)$ of the first rule which should be applied.

On the other hand, by decomposing 4 into 2 and 2, we can match $gcd(x+y, x)$ with $gcd(4, 2)$. Such matching is called *semantic matching*. From the example above, it is easily known that semantic matching generally requires to solve equations, for instance, $x + y = 4$ and $x = 2$. Computation for finding solutions of given equations is sometimes called *inverse computation*. However, describing such reasoning precisely as an algorithm which works automatically is not easy. Thus, an algorithm for inverse computation is required in semantic matching.

The semantic unification of 4 and $x + y$ in the above example seems to an easy task because almost all programming languages have subtraction operator as a primitive function. However, finding an algorithm for inverse computation of general functions (including user-defined functions) is not an easy task. Consider the following example which shows a definition of a palindrome of a list [14]:

$$\begin{cases} palindrome(\, append(\, x, \quad reverse(x) \quad)\,) = True \\ palindrome(\, append(\, x, cons(y, reverse(x))\,)\,) = True \end{cases}$$

where a list consists of *cons* and *nil*, and *append* and *reverse* on lists are defined as usual. To use the above definition within a functional language, matching patterns of the form either $append(x, reverse(x))$ or $append(x, cons(y, reverse(x)))$

against lists, such as $cons(1, cons(2, cons(1, nil)))$. To perform such semantic matching for every function, an algorithm for inverse computation (which is incorporable into other programs) is required.

1.2 Background

In the previous section, we have described that *inverse computation* is a certain reasoning which is to solve equations, such as $x + y = 4$.

More generally, *inverse computation* is the calculation of the possible inputs of a program for a given output, while standard computation is the calculation of the output of a program for given inputs. Advances of inverse computation have been achieved in the area of logic programming, based on solutions emerging from logic and proof theory. Inverse computation can be also considered as a special case of equational unification which is one of the most important problems in functional languages. Inverse computation is used in solving equational unification, semantic matching and so on.

Formally speaking, *inverse computation* is considered as a way to solve *inverse problem* formalized as follows [44];

> Let $\mathtt{P} : D^n \to D$ be a computable program (or function) where D is a set of data values, and $\mathtt{P}$ is given in a language L. For a given output v and inputs $v_{k+1}, \ldots, v_n$ which are already known, find (all of) the possible inputs $v_1, \ldots, v_k$ satisfying
>
> $$\mathtt{P}(v_1, \ldots, v_k, v_{k+1}, \ldots, v_n) = v. \tag{1.1}$$

Determining inputs data $v_1, \ldots, v_k$ for program $\mathtt{P}$, its input data $v_{k+1}, \ldots, v_n$ and its output data v such that $\mathtt{P}(v_1, ..., v_n) = v$, is inverse computation.

Approaches to inverse computation are distinguished between two types. One is an algorithmic approach called an *inverse interpreter* that performs inverse computation, and the other is a transformational approach called an *inverse compiler* (a *inverse translator*, or a *program inverter*) that generates a program $\mathtt{P}^{-1}$ which performs inverse computation for a given program, that is, $\mathtt{P}^{-1}(v, v_{k+1}, \ldots, v_n) = (v_1, \ldots, v_k)$ [3].

An inverse problem is either of two types, a *full inverse problem* or a *partial inverse problem* [45]. A full inverse problem is the case that $k = n$ in equation (1.1), and a partial inverse problem is the remaining case, that is, $1 \leq k < n$. In this thesis, the word 'inverse' means 'full inverse'. Approaches to these

inverse problems from an inverse interpreter and an inverse compiler are shown in Figure 1.1 and 1.2, respectively.

In the sequel, it argues about general comparison of the feature of inverse interpreters and inverse compilers.

- Inverse interpreters work every time when inverse computation of P is required, from a given program P, while inverse compilers work once for a pair of P and a number k as a pre-process of inverse computation.

- While most of inverse interpreters can search all solutions, all of inverse compilers require a strategy of the interpretation for their generated programs, which enables interpretation of the programs to search all solutions. If the ordinary interpreter of the generated programs does not have such a strategy unfortunately, then ones need to give (develop and implement) such a strategy to it.

- Since inverse interpreters are algorithms and inverse compilers generate programs for inverse computation, discussing theoretical properties of inverse interpreters, such as proving the correctness, the termination and so on, may be more difficult than doing of inverse compilers. For instance in term rewriting, to prove the termination of inverse computation which is done by a program generated by an inverse compiler, we can use several techniques which have already proposed to prove the termination of term rewriting systems.

- Programs generated by inverse compilers are usable for several purposes while inverse interpreters are not. In other words, programs generated by inverse compilers are more incorporable into other programs than inverse interpreters.

The first work on *program inversion* appears to be [35], suggesting a "generate and test approach" for Turing machines. After then, *inverse functions* are mentioned only twice in the textbook on recursive functions [10].

As a technique for semantic unification, some algorithms for inverse computation of convergent constructor TRSs are proposed and improved in [12, 13, 14], which is classified into inverse interpreters. *Narrowing* [28] which is a method for equational unification, is also classified into inverse interpreters for TRSs. Then, we consider equational unification methods as inverse interpreters.

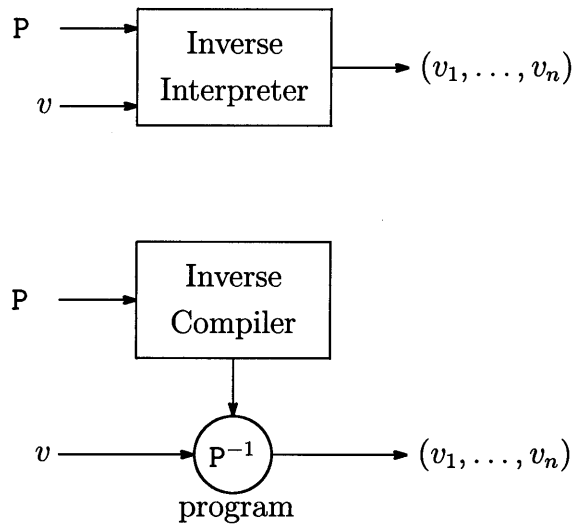

Figure 1.1: Approaches to full inverse computation for $\mathtt{P}(v_1, \ldots, v_n) = v$.

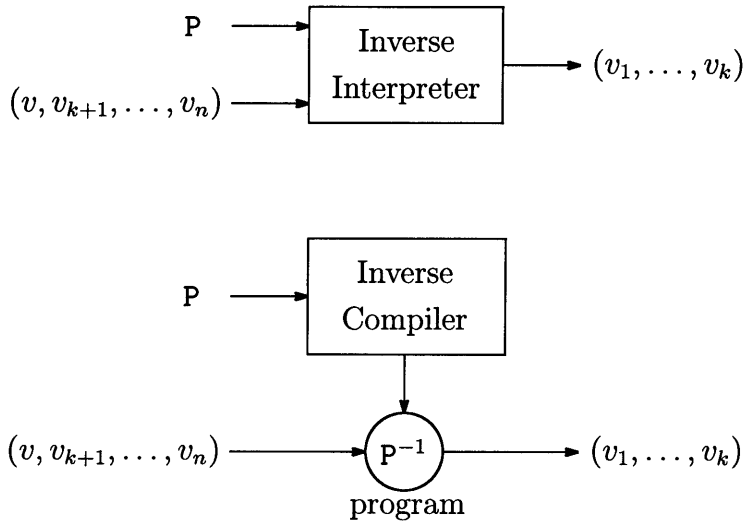

Figure 1.2: Approaches to partial inverse computation for $\mathtt{P}(v_1, \ldots, v_n) = v$.

Universal Resolving Algorithm [1, 3, 2, 19] is an inverse interpreter for first-order functional languages. This algorithm is a fully automated program inversion method, which constructs a process tree from the object program, and solutions are extracted from the leaves of the tree in form of substitution/restriction pairs. The experiments in [20] apply the idea of prototyping programming language tools from robust semantics. An approach to software verification by program inversion is presented in [21].

An inverse compiler for the applicative programming language *Refal* is proposed in [44, 45]. This compiler produces an inverse program written in *Refal-R* which is an extended language of Refal with non-deterministic construct. This method approaches to both full and partial inverse computations. That transformation is fine-looking and visibly skillful. However, the correctness of that transformation is not discussed. Syntactic properties and other properties, such as termination, of the generated programs are not discussed, too.

A method for inverting a given program with respect to a given output is developed in [48], which generates a finite dag grammar that gives a complete description of the input: "Running" the dag grammar produces a (possibly infinite) set of terms that will contain all tuples of terms that result in the given output.

Function inversion is also considered in [23]; The general one-argument function inversion algorithm and its specialization in the Backus's FP language are described.

A method for synthesizing recursive inverses for first-order functions are presented in [24]. An automatic system for synthesizing recursive programs from first-order functional programs is *InvX* [29].

As far as we know, there is no inverse compiler whose input and output are written are the same language, in other words, both of them are executable on the same interpretation.

1.3 Overview of the Thesis

The next chapter gives preliminaries of term rewriting theories, which are need later on.

In Chapter 3, we first propose an inverse compiler for TRSs which define pure treeless functions. We show the correctness of the inverse compiler, that is, a generated program performs as inverse computation of the input. We second

extend the above inverse compiler to that for constructor TRSs. We show the correctness of the extended inverse compiler. We also show the relationship between the syntactic properties of the input and output.

Pure treeless functions which is the first target of our inverse computation, are often treated in program transformations. Their representation power is restricted and their computation power is weak than that of Turing machines. However they are suitable for the first step of program transformation and program analysis. Then, we extend the target to constructor TRSs of which the whole class properly includes all pure treeless functions. The class of constructor TRSs is standard in term rewriting since it is known that their computation power is equivalent to that of Turing machines. Therefore, the extension of the target to constructor TRSs is valuable.

Each of the inverse compilers which we propose in this thesis, consist of two program transformations: a *program inverter* from an input TRS into a conditional TRS (CTRS) which is an inverse program of the input, and a transformation so-called *unraveling* from the generated CTRS into a TRS with extra variables (an EV-TRS, for short). Extra variables are variables appearing only in the right-hand sides of rewrite rules. Although CTRSs are enough for inverse computation in the sense that they correctly define inverses of inputs, we transform them into EV-TRSs. The reason is that

- computation of CTRSs is more complicated than (EV-)TRSs,
- introducing reduction strategies in computation of EV-TRSs is easier than in that of CTRSs,
- the transformation of CTRSs into EV-TRSs is useful to analyze the properties of CTRSs, especially the termination property.

Hence, the transformation of CTRSs into EV-TRSs is valuable.

In Section 3.6.5, we analyze properties of the generated EV-TRSs, such as syntactic properties, termination, confluence and so on. We show that inverse EV-TRSs generated by the inverse compiler of constructor TRSs have no extra variable (that is, they are TRSs) if the input are non-erasing. We also show that our inverse EV-TRSs are right-linear if the input are left-linear, and those are left-linear if the input are right-linear.

In Chapter 4, we give a condition of an input constructor TRS to obtain an innermost terminating TRS (without extra variables) by the second inverse

compiler which is for constructor TRSs. The condition is related with the depth of the both hand side of rewrite rules. Since every redex is the innermost in rewrite sequences representing inverse computation, innermost termination is sufficient as termination of inverse computation.

In Chapter 5, to achieve practical inverse computation, we show that narrowing with substituting a fresh variable for each extra variable can simulate a ground rewrite sequence of an EV-TRS if the EV-TRS is right-linear or the any redex which is reduced in the rewrite sequence is not introduced by means of extra variables. This result solves the problems for practical inverse computation, which are infinitely branching and non-termination of reduction of EV-TRSs. Treating only ground rewrite sequences is sufficient because variables occurring in rewrite sequences work as the same with constants.

It is clear that narrowing of TRSs on ground terms is equivalent to the rewrite relation. Hence, target TRSs of inverse computation also work on narrowing similarly on rewrite relation. This means that target TRSs and their inverse EV-TRSs generated by our inverse compiler work on the same interpretation *narrowing.*

It is well-known that termination is one of the most important properties for computation. In Section 5.2, as a technique to guarantee termination of inverse computation, we propose two termination proof methods for narrowing of EV-TRSs (starting from ground terms). One is based on the relation of function calls of EV-TRSs. The other is based on the dependency pair method which is a useful tool to prove the termination of rewrite relation.

In Section 5.3, we discuss the strategy for narrowing derivations on right-linear EV-TRSs. It is known that for right-linear constructor TRSs, all normal forms of a given terminating term can be obtained by innermost strategy. Then, we can improve the efficiency of the inverse computation of non-erasing constructor TRSs. We show that for linear constructor EV-TRSs, all normal forms with respect to narrowing of a given term from which narrowing is terminating, can be obtained by innermost narrowing. We also show that basic narrowing is sufficient to simulate rewrite sequences of right-linear EV-TRSs. EV-TRSs generated by our inverse compiler are in general not confluent as shown by the fact that inverse images of many-to-one functions are one-to-many. It is known that searching all solutions is very expensive. Therefore, introducing the innermost strategy improves the efficiency of computation of the generated EV-TRSs.

To improve efficiency of inverse computation, we show that innermost nar-

rowing can find all solutions of a given ground term which is terminating on narrowing. We also show that basic narrowing is sufficient for inverse computation of left-linear programs.

In Chapter 6, we tackle partial inverse by extending the inverse compiler of constructor TRSs which is proposed in Chapter 3. We extend the latter inverse compiler, which is for constructor TRSs, to a partial inverse compiler that generates a conditional TRS which performs as partial inverse computation of the input.

In this thesis, we also deal with partial inverse computation, by extending the proposed inverse compiler for partial inverse EV-TRSs. This partial inverse compiler can automatically generates the definitions of subtraction and division from the definition of addition and multiplication, respectively.

In Chapter 7, we show three applications of inverse programs which are generated by our inverse compilers. In the first of them, we give a solution of our motivation problem, that is, that our motivation program *gcd* can be transformed into an equivalent executable program, by incorporating the inverse program of "+" into the program.

In Chapter 8, we conclude this thesis and show our future works.

Through this thesis, it is shown that the program which performs as inverse computation of an input program, can be automatically obtained by the inverse compilers proposed here.

Chapter 2

Preliminaries

This thesis follows standard notations of term rewriting [7, 30, 42].

2.1 Binary Relations and Orderings

Let $\triangleright$ be a binary relation on a set S. We say that $\triangleright$ is

- *transitive* iff for all $a, b, c \in S$, $a \triangleright b$ and $b \triangleright c$ imply $a \triangleright c$,
- *reflexive* iff $a \triangleright a$ for all $a \in S$,
- *irreflexive* iff $\neg(a \triangleright a)$ for all $a \in S$,
- *symmetric* iff for all $a, b \in S$, $a \triangleright b$ implies $b \triangleright a$,
- *asymmetric* iff there are no elements $a, b \in S$ such that $a \triangleleft b$ and $a \triangleright b$, and
- *antisymmetric* iff for all $a, b \in S$, $a \triangleright b$ and $b \triangleright a$ imply $a = b$.

An *equivalence relation* is a reflexive, transitive and symmetric relation.

A *partial ordering*[1] $\succ$ is a binary irreflexive and transitive relation. Note that every partial ordering is asymmetric. A *quasi-ordering* is a binary reflexive and transitive relation $\succsim$. A *reflexive partial ordering* is a quasi-ordering that is also antisymmetric. Note that $a \prec b$ means $b \succ a$, and $a \not\succ b$ means $\neg(a \succ b)$ (and similarly for $\succsim$).

A relation $\triangleright$ on a set S is called *total* (or *linear*) if $a \triangleright b$, $a = b$ or $b \triangleright a$ for all a and b in S.

[1] In this paper, the phrase "partial ordering" will always mean "irreflexive partial ordering".

2.2 Abstract Reduction Systems

An *abstract reduction system* (*ARS*, for short) is a pair $\mathcal{A} = (S, \to)$ where $\to$ is a binary relation on the set S, that is, $\to \subseteq S \times S$. Instead of $(a, b) \in \to$ we write $a \to b$ and we say that a is *reduced* (or *irreducible*) to b. Note that $\to$ is often called the *reduction relation* of $\mathcal{A}$, and we may write $\to_{\mathcal{A}}$ instead of $\to$. A *reduction sequence* of $\mathcal{A}$ (or *$\to$-sequence*) is either a finite sequence $a_0 \to a_1 \to \cdots \to a_n$ or an infinite sequence $a_0 \to a_1 \to \cdots$ of elements in S. A finite sequence $a_0 \to a_1 \to \cdots \to a_n$ may be abbreviated to $a_0 \xrightarrow{n} a_n$, where $\xrightarrow{n}$ is called *n-fold composition* of $\to$. The relation $\xrightarrow{0}$ is the identity on S, that is, $\xrightarrow{0} = \{ (a, a) \mid a \in S \}$. The transitive closure $\xrightarrow{+}$ of $\to$ is defined as $\xrightarrow{+} = \bigcup_{i>0} \xrightarrow{i}$. The reflexive and transitive closure $\xrightarrow{*}$ of $\to$ is defined as $\xrightarrow{*} = \bigcup_{i \geq 0} \xrightarrow{i}$. Let $a, b \in S$ with $a \xrightarrow{*} b$. We say that b is a *normal form* (or *irreducible*) of a with respect to $\to$ if there exists no element $c \in S$ such that $b \to c$. The set of all normal forms of a with respect to $\to$ is denoted by $NF_a^{\to}$, and the set of all normal forms with respect to $\to$ is denoted by $NF^{\to}$, that is, $NF^{\to} = \bigcup_{a \in S} NF_a^{\to}$. We say that a is *strongly normalizing* (or *terminating*) with respect to $\to$, written as $\mathrm{SN}_a^{\to}$, if there is no infinite $\to$-sequence starting from a. We also say that a is *weakly normalizing* with respect to $\to$, written as $\mathrm{WN}_a^{\to}$, if it has at least one normal form with respect to $\to$. $\mathcal{A}$ is said to be *strongly normalizing* (or *terminating*) with respect to $\to$, written as $\mathrm{SN}^{\to}$, if $\mathrm{SN}_a^{\to}$ for all $a \in S$. $\mathcal{A}$ is said to be *weakly normalizing* with respect to $\to$, written as $\mathrm{WN}^{\to}$, if $\mathrm{WN}_a^{\to}$ for all $a \in S$. $\mathcal{A}$ is called *confluent* if for all $a, b_1, b_2 \in S$, $b_1 \xleftarrow{*} a \xrightarrow{*} b_2$ implies $b_1 \xrightarrow{*} c \xleftarrow{*} b_2$ for some $c \in S$. $\mathcal{A}$ is called *convergent* if it is confluent and $\mathrm{SN}^{\to}$.

2.3 Terms

A *signature* Σ is a set of *function symbols*, where each $f \in \Sigma$ is associated with a non-negative integer, the *arity* of f which is represented by $\mathtt{arity}(f)$. Function symbols whose arity is zero are called *constant symbols*. Let $\mathcal{X}$ be a countably infinite set of *variables*. Throught this thesis, we use $\mathcal{X}$ as a countably infinite set of variables. We assume that $\mathcal{G} \cap \mathcal{X} = \emptyset$ for any signature $\mathcal{G}$. We use x, y, z as variables.

Let $\mathcal{F}$ be a signature. The set $\mathcal{T}(\mathcal{F}, \mathcal{X})$ of all *terms* over $\mathcal{F}$ (and $\mathcal{X}$) is inductively defined as follows:

- $\mathcal{X} \subseteq \mathcal{T}(\mathcal{F}, \mathcal{X})$ (that is, every variable is a term),

- for all $f \in \mathcal{F}$ with $\mathtt{arity}(f) = n$, and all $t_1, \ldots, t_n \in \mathcal{T}(\mathcal{F}, \mathcal{X})$, we have $f(t_1, \ldots, t_n) \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ (that is, application of function symbols to terms yields terms).

A term which consists of one constant symbol is called *constant*. We denote the set of *variables occurring in* terms $t_1, \ldots, t_n$ by $\mathcal{V}ar(t_1, \ldots, t_n)$. We use s, t, u as terms, and a, b as constants. A term $t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ is called *ground* iff $\mathcal{V}ar(t) = \emptyset$. The set of all ground terms over $\mathcal{F}$ is denoted by $\mathcal{T}(\mathcal{F}, \emptyset)$ or simply $\mathcal{T}(\mathcal{F})$. *Identity* of terms is denoted by $\equiv$. We say that t is *linear* if no variable occurs twice in t. If f is a unary function symbol, then $f^n(t)$ abbreviates the term $f(f(\cdots f(t)\cdots))$, the *n-fold application* of f to t. For a term t, $\mathcal{S}ym(t)$ denotes the set of all function symbols appearing in t, that is, $\mathcal{S}ym(x) = \emptyset$ and $\mathcal{S}ym(f(t_1, \ldots, t_n)) = \{f\} \cup \bigcup_{i=1}^{n} \mathcal{S}ym(t_i)$.

Let $\mathcal{F}$ be a signature and t be a term over $\mathcal{F}$. The set of all *positions* of the term t is a set $\mathcal{O}(t)$ of strings over the alphabet of positive integers, which is inductively defined as follows:

- If $t \equiv x \in \mathcal{X}$, then $\mathcal{O}(t) = \{\varepsilon\}$, where ε denotes the empty string.
- If $t \equiv f(t_1, \ldots, t_n)$, then $\mathcal{O}(t) = \{\varepsilon\} \cup \bigcup_{i=1}^{n} \{\, ip \mid p \in \mathcal{O}(t_i) \,\}$.

The position ε is called the *root position* of the term t, and the function or variable symbol at this position is called the *root symbol* of t. We write $\mathtt{root}(t)$ to represent the root symbol of t. The *prefix ordering* defined as

$$p \leq q \text{ iff there exists } p' \text{ such that } pp' = q$$

is a partial ordering on positions. We say that the positions p and q are *parallel*, written as $p \parallel q$, iff p and q are incomparable with respect to $\leq$. It is obvious that for parallel positions p and q, there exists a position r and positive integers i, j such that $i \neq j$, $p = rip'$ and $q = rjq'$ for some p', q'. In this case, p is said to be *more left* than q if $i < j$, p is said to be *more right* than q if $i > j$. The position p is called *above* q if $p \leq q$ and p is *strictly above* q if $p < q$ (*below* is defined analogously). For $p \in \mathcal{O}(t)$, the *subterm of t at position p*, denoted by $t|_p$, is defined by induction on the length of p:

- $t|_\varepsilon \equiv t$,
- $f(t_1, \ldots, t_n)|_{iq} \equiv t_i|_q$.

Note that, for $p = iq$, $p \in \mathcal{O}(t)$ implies that t is of the form $t \equiv f(t_1, \ldots, t_n)$ with $i \leq n$. We say that $t|_p$ is a *proper* subterm if $\varepsilon < p$. We write $t \trianglerighteq u$ to represent that a term u is a subterm of t, and write $t \triangleright u$ to represent that u is a proper subterm of t. We call $p \in \mathcal{O}(t)$ a *function-symbol* position if $\mathtt{root}(t|_p)$ is a function symbol, and a *variable* position if $\mathtt{root}(t|_p)$ is a variable symbol. We denote the set of all function-symbol positions of t by $\mathcal{O}_{\mathcal{F}}(t)$, and the set of all variable positions of t by $\mathcal{O}_{\mathcal{X}}(t)$. The *size* of term t denoted by $|t|$ is the capability of $\mathcal{O}(t)$.

Let $\mathcal{F}$ be a signature. Let $\Box \notin \mathcal{F} \cup \mathcal{X}$ be a special constant symbol, called the *hole*. A *context* is a term in $\mathcal{T}(\mathcal{F} \cup \{\Box\}, \mathcal{X})$. The set of all contexts with n occurrences of $\Box$ is denoted by $\mathscr{C}^n(\mathcal{F}, \mathcal{X})$. We abbreviate $\mathscr{C}^1(\mathcal{F}, \mathcal{X})$ to $\mathscr{C}(\mathcal{F}, \mathcal{X})$. Note that $\mathscr{C}^0(\mathcal{F}, \mathcal{X}) = \mathcal{T}(\mathcal{F}, \mathcal{X})$. If $C \in \mathscr{C}^n(\mathcal{F}, \mathcal{X})$ and $t_1, \ldots, t_n$ are terms, then $C[t_1, \ldots, t_n]$ is the result of replacing the occurrences of $\Box$ with $t_1, \ldots, t_n$ from left to right. When we explicitly specify the positions p_i of $\Box$ in C, we write $C[t_1, \ldots, t_n]_{p_1, \ldots, p_n}$.

2.4 Substitutions

Let $\mathcal{F}$ be a signature. A *substitution* is a function σ: $\mathcal{X} \to \mathcal{T}(\mathcal{F}, \mathcal{X})$ such that $\sigma(x) \not\equiv x$ for only finitely many xs. The (finite) set of variables that σ does not map to themselves is called the *domain* of σ: $\mathcal{D}om(\sigma) = \{x \in \mathcal{X} \mid \sigma(x) \not\equiv x\}$. If $\mathcal{D}om(\sigma) = \{x_1, \ldots, x_n\}$, then we may write σ as $\sigma = \{ x_1 \mapsto \sigma(x_1), \ldots, x_n \mapsto \sigma(x_n) \}$. The *range* of σ is $\mathcal{R}an(\sigma) = \{\sigma(x) \mid x \in \mathcal{D}om(\sigma)\}$, and the *variable range* of σ consists of the variables occurring in $\mathcal{R}an(\sigma)$: $\mathcal{VR}an(\sigma) = \bigcup_{x \in \mathcal{D}om(\sigma)} \mathcal{V}ar(\sigma(x))$. We call σ *ground* if $\mathcal{VR}an(\sigma) = \emptyset$. To simplify notations, we may abbreviate $\mathcal{D}om(\sigma_1) \cup \cdots \cup \mathcal{D}om(\sigma_m)$ to $\mathcal{D}om(\sigma_1, \ldots, \sigma_m)$ (and similarly for $\mathcal{R}an$ and $\mathcal{VR}an$).

Any substitution σ can be *extended* to a mapping $\hat{\sigma}$: $\mathcal{T}(\mathcal{F}, \mathcal{X}) \to \mathcal{T}(\mathcal{F}, \mathcal{X})$ as follows: for $x \in \mathcal{X}$, $\hat{\sigma}(x) \equiv \sigma(x)$, and for a non-variable term $t \equiv f(t_1, \ldots, t_n)$ we define $\hat{\sigma}(t) \equiv f(\hat{\sigma}(t_1), \ldots, \hat{\sigma}(t_n))$. We sometimes simply write $x\sigma$ (resp. $t\hat{\sigma}$) instead of $\sigma(x)$ for the application of the substitution σ to the variable x. Let σ, θ be substitutions. If $\mathcal{D}om(\sigma) = \mathcal{D}om(\sigma)$ and $\sigma(x) \equiv \theta(x)$ for all $x \in \mathcal{D}om(\sigma)$, then σ and θ are *equivalent*, written as $\sigma = \theta$.

The *composition* $\sigma\theta$ of two substitutions σ and θ is defined as $x\sigma\theta \equiv \hat{\theta}(\sigma(x))$. Obviously, $\sigma\theta$ is a mapping of $\mathcal{X}$ into $\mathcal{T}(\mathcal{F}, \mathcal{X})$, and since $x\sigma\theta \equiv x$ holds for all $x \in \mathcal{X} \setminus \mathcal{D}om(\sigma, \theta)$, we know that it is again a substitution. In addition, it is easy to see that composition of substitutions is an associative operation. The definition

of composition makes sure that the extension of the composition $\sigma\theta$ is just the composition of the extensions of σ and θ, that is, $\widehat{\sigma\theta} = \hat{\sigma}\hat{\theta}$ (where $\hat{\sigma}\hat{\theta}$ denotes just the usual composition of mappings). To simplify notation, we usually do not distinguish between a substitution σ: $\mathcal{X} \to \mathcal{T}(\mathcal{F}, \mathcal{X})$ and its extension $\hat{\sigma}$: $\mathcal{T}(\mathcal{F}, \mathcal{X}) \to \mathcal{T}(\mathcal{F}, \mathcal{X})$. In the following, σ will be used to denote both. We use σ, θ, δ and η for substitutions.

A term t is called an *instance* of a term s iff there exists a substitution σ such that $\sigma(s) \equiv t$. The terms s and t are called *renamings* (or *variants*) of each other iff they are instances of each other.

Let $T' \subseteq \mathcal{T}(\mathcal{F}, \mathcal{X})$. We call σ a *T'-substitution* if $\mathcal{R}an(\sigma) \subseteq T'$. The set of all T'-substitutions is denoted by $\mathcal{S}ub(T')$. The *restriction* of a substitution σ to a variable set $V \subseteq \mathcal{X}$ is defined as $\sigma|_V = \{\, x \mapsto \sigma(x) \mid x \in \mathcal{D}om(\sigma) \cap V \,\}$.

A substitution σ is *more general* than a substitution σ' if there is a substitution δ such that $\sigma' = \sigma\delta$. In this case we write $\sigma \lesssim \sigma'$. It is known that the relation $\lesssim$ on substitutions is a quasi-ordering. A *unifier* of terms s and t is a substitution σ such that $s\sigma \equiv t\sigma$. If a unifier of s and t exists, then s and t are said to be *unifiable*. A unifier σ of terms s and t is called *most general* if $\sigma \lesssim \sigma'$ for all unifiers σ' of s and t. It is known that a most general unifier of two terms is unique up to renamings if it exists. Hence, we denote the most general unifier of terms s and t by $\mathtt{mgu}(s, t)$ if it exists.

2.5 Abstract Term Rewriting Systems

Let $\to$ be a binary relation on terms over a signature $\mathcal{F}$. The relation $\to$ is called *monotone* (or *closed under contexts*) if for all terms s, t, $s \to t$ implies $C[s] \to C[t]$ for all contexts $C \in \mathscr{C}(\mathcal{F}, \mathcal{X})$. An *abstract term rewriting system* (*ATRS*, for short) over the signature $\mathcal{F}$ is an ARS $\mathcal{A} = (\mathcal{T}(\mathcal{F}, \mathcal{X}), \to)$. A $\to$-sequence $t_0 \to t_1 \to \cdots$ is called *ground* if the term t_0 is ground, and it is called *totally ground* if t_i is ground for all $i \geq 0$. To specify a signature for $NF^{\to}$, we write $NF^{\to}(\mathcal{F}, \mathcal{X})$ instead of $NF^{\to}$. $\mathcal{A}$ is said to be *ground strongly normalizing* (or *ground terminating*) with respect to $\to$, written as $\mathrm{GSN}^{\to}$, if $\mathrm{SN}^{\to}_t$ for every ground term $t \in \mathcal{T}(\mathcal{F})$. The ATRS $\mathcal{A}$ is said to be *ground weakly normalizing* with respect to $\to$, written as $\mathrm{GWN}^{\to}$, if $\mathrm{WN}^{\to}_t$ for every ground term $t \in \mathcal{T}(\mathcal{F})$. $\mathcal{A}$ is called *ground convergent* if $\mathcal{A}$ is confluent on ground terms (that is, the ATRS $(\mathcal{T}(\mathcal{X}), \to)$ is confluent) and $\mathrm{GSN}^{\to}$. The *innermost derivation* $\underset{\text{in}}{\to}$ of the relation $\to$ is the binary relation on terms over $\mathcal{F}$ that is defined as; $C[s] \underset{\text{in}}{\to}$

$C'[t]$ iff $C, C' \in \mathscr{C}(\mathcal{F}, \mathcal{X})$, $C[s] \to C'[t]$, $s \to t$ and every proper subterm u of s is in normal forms with respect to $\to$. The *left-most* innermost derivation $\xrightarrow[\text{lin}]{}$ of $\to$ is defined as,; in the above definition of $\xrightarrow[\text{in}]{}$, the position of s is the most left among the positions q such that q is parallel with p and $(C[s])|_q$ is not in normal forms with respect to $\to$, that is, $\{\, q \mid p \parallel q,\ (\exists s', (C[s])|_q \to s') \,\}$. The *right-most innermost derivation* $\xrightarrow[\text{rin}]{}$ of $\to$ is defined as; in the above definition of $\xrightarrow[\text{in}]{}$, the position of s is the most right among $\{\, q \mid p \parallel q,\ (\exists s', (C[s])|_q \to s') \,\}$. We use $\text{SIN}^{\to}$, $\text{WIN}^{\to}$, $\text{GSIN}^{\to}$ and $\text{GWIN}^{\to}$, called the *innermost normalizing* properties, instead of $\text{SN}^{\xrightarrow[\text{in}]{}}$, $\text{WN}^{\xrightarrow[\text{in}]{}}$, $\text{GSN}^{\xrightarrow[\text{in}]{}}$ and $\text{GWN}^{\xrightarrow[\text{in}]{}}$, respectively.

2.6 Conditional Term Rewriting Systems

An *oriented conditional rewrite rule* over a signature $\mathcal{F}$ is a triple $(l, r, \mathcal{C}ond)$, written as $l \to r \Leftarrow \mathcal{C}ond$, where l is a non-variable term over $\mathcal{F}$ and r is a term over $\mathcal{F}$. We say that l is the *left-hand* side of the rule $l \to r \Leftarrow \mathcal{C}ond$, r is the *right-hand side* of the rule, and that $\mathcal{C}ond$ is the *conditional part* of the rule. The conditional part $\mathcal{C}ond$ is a sequence $\mathcal{C}ond_1 \wedge \cdots \wedge \mathcal{C}ond_n$ of *conditions* $\mathcal{C}ond_i$ in the form of either `true` or $s \twoheadrightarrow t$, called an *oriented condition*, where $n > 0$ and s, t are terms over $\mathcal{F}$. To simplify notation, we may abbreviate sequences $\mathcal{C}ond' \wedge$ `true` or `true` $\wedge\, \mathcal{C}ond'$ in conditional parts to $\mathcal{C}ond'$. The set of all variables which occur in condition $s \twoheadrightarrow t$ is denoted by $\mathcal{V}ar(s \twoheadrightarrow t)$, and the set of all variables in conditional part $\mathcal{C}ond$ is denoted by $\mathcal{V}ar(\mathcal{C}ond)$: $\mathcal{V}ar(\mathcal{C}ond) = \bigcup_{i=1}^{n} \mathcal{V}ar(\mathcal{C}ond_i)$ where $\mathcal{V}ar(\texttt{true}) = \emptyset$. For terms $u_1, \ldots, u_m$, we abbreviate $\mathcal{V}ar(u_1, \ldots, u_m) \cup \bigcup_{i=1}^{n} \mathcal{V}ar(\mathcal{C}ond_i)$ to $\mathcal{V}ar(u_1, \ldots, u_m, \mathcal{C}ond_1, \ldots, \mathcal{C}ond_n)$. Note that a subsequence of conditional parts is also a conditional part. If the conditional part $\mathcal{C}ond$ is in the form `true` $\wedge \cdots \wedge$ `true`, then $l \to r \Leftarrow \mathcal{C}ond$ is called an *unconditional rewrite rule* (or simply a *rewrite rule*), and it may be abbreviated to $l \to r$. To simplify the notation of conditional rewrite rules, we may give the unique label ρ, written as $\rho : l \to r \Leftarrow \mathcal{C}ond$, for each conditional rewrite rule $l \to r \Leftarrow \mathcal{C}ond$ and may write ρ instead of either $l \to r \Leftarrow \mathcal{C}ond$ or $\rho : l \to r \Leftarrow \mathcal{C}ond$.

We say that a $\mathcal{T}(\mathcal{F}, \mathcal{X})$-substitution σ and a binary relation $\to$ on terms over $\mathcal{F}$ *satisfy* the conditional part $\mathcal{C}ond$, written as $\mathcal{C}ond(\sigma, \to)$, if either of the following holds inductively:

- $\mathcal{C}ond$ is in form of `true`,
- $\mathcal{C}ond$ is in form of $s \twoheadrightarrow t$ and $s\sigma \to t\sigma$, or

- $\mathcal{C}ond$ is in form of $\mathcal{C}ond_1 \wedge \cdots \wedge \mathcal{C}ond_n$ $(n > 1)$ and $\mathcal{C}ond_i(\sigma, \to)$ for $1 \leq i \leq n$.

Let R be a finite set of conditional rewrite rules over a signature $\mathcal{F}$. The *n-level rewrite relation* $\xrightarrow[n]{}_R$ associated with R is inductively defined as follows:

- $\xrightarrow[0]{}_R = \emptyset$, and
- $\xrightarrow[i+1]{}_R = \xrightarrow[i]{}_R \cup \{ (C[l\sigma]_p, C[r\sigma]_p) \mid \rho : l \to r \Leftarrow \mathcal{C}ond \in R, C \in \mathscr{C}(\mathcal{F}, \mathcal{X}), \sigma \in \mathcal{S}ub(\mathcal{T}(\mathcal{F}, \mathcal{X})), \mathcal{C}ond(\sigma, \xrightarrow[i]{*}_R) \}$

  where $\xrightarrow[i]{*}_R$ is the reflexive and transitive closure of $\xrightarrow[i]{}_R$.

The *rewrite relation* of R is $\to_R = \bigcup_{n \geq 0} \xrightarrow[n]{}_R$. To specify the position p and the rule ρ for $C[l\sigma] \to_R C[r\sigma]$ in the above definition of $\xrightarrow[n]{}_R$, we may write $\to_R^{[p,\rho]}$ or $\to_R^p$ instead of $\to_R$.

An *oriented conditional term rewriting system* is an ATRS $(\mathcal{T}(\mathcal{F}, \mathcal{X}), \to_R)$ consisting of the set of all terms over a signature $\mathcal{F}$ and the rewrite relation of a finite set R of oriented conditional rewrite rules. Since this thesis treat only oriented conditional term rewriting systems, we call an oriented conditional rewrite rule and an oriented conditional rewriting system simply a *conditional rewrite rule* and a *conditional rewriting system* (*CTRS*, for short), respectively. We abbreviate $(\mathcal{T}(\mathcal{F}, \mathcal{X}), \to_R)$ to R. An instance of the left-hand sides of a conditional rewrite rule in R is called a *redex*.

Let $\rho : l \to r \Leftarrow \mathcal{C}ond$ be a conditional rewrite rule over a signature $\mathcal{F}$. Variables which occur not in the left-hand side l of the rule ρ but in either the right-hand side r or the conditional part $\mathcal{C}ond$ are called *extra variables* of ρ, that is, in $\mathcal{V}ar(r, \mathcal{C}ond) \setminus \mathcal{V}ar(l)$. The set of all extra variables of ρ is denoted by $\mathcal{EV}ar(\rho)$: $\mathcal{EV}ar(\rho) = \mathcal{V}ar(r, \mathcal{C}ond) \setminus \mathcal{V}ar(l)$.

A CTRS R is called a *term rewriting system with extra variables* (*EV-TRS*, for short) if it contains only unconditional rewrite rules. The CTRS R is a *term rewriting system* (*TRS*, for short) if it is an EV-TRS and every rewrite rule $l \to r \in R$ satisfies $\mathcal{V}ar(l) \supseteq \mathcal{V}ar(r)$.

A conditional rewrite rule $l \to r \Leftarrow \mathcal{C}ond$ will be classified according to the distribution of variables among l, r and $\mathcal{C}ond$, as follows:

Type	Requirement
1	$\mathcal{V}ar(r, \mathcal{C}ond) \subseteq \mathcal{V}ar(l)$
2	$\mathcal{V}ar(r) \subseteq \mathcal{V}ar(l)$
3	$\mathcal{V}ar(r) \subseteq \mathcal{V}ar(l, \mathcal{C}ond)$
4	No restrictions

An *n-CTRS* contains only rewrite rules of type n. Thus a 1-CTRS has no extra variables, a 2-CTRS has no extra variables on the right-hand sides of the rules, and a 3-CTRS may contain extra variables of the rules provided that these also occur in the conditional part. It is clear that every i-CTRS is a j-CTRS for $1 \leq i < j \leq 4$. It is also clear that every EV-TRS is a 4-CTRS but not a 3-CTRS if it has at least one extra variable, and every TRS is a 1-CTRS. Hence, the class of all EV-TRSs is included in that of all 4-CTRSs but not in that of all 3-CTRSs.

A CTRS R over a signature $\mathcal{F}$ is called *normal* if each t_i is a normal form of R for every rule $l \to r \Leftarrow s_1 \twoheadrightarrow t_1 \wedge \cdots \wedge s_n \twoheadrightarrow t_n \in R$.

2.7 Syntactic Properties of Conditional Term Rewriting Systems

Let R be a CTRS and $\rho : l \to r \Leftarrow \mathcal{C}ond$ be a conditional rewrite rule. The rule ρ is called *left-linear* if l is linear, and called *right-linear* if r is linear. The CTRS R is called *left-linear* if every rule in R is left-linear, and called *right-linear* if every rule in R is right-linear. The CTRS R is called *linear* if it is left-linear and right-linear. The rule ρ is called *non-erasing* (or *variable-preserving*) if $\mathcal{V}ar(l) \subseteq \mathcal{V}ar(r)$. We say that R is *non-erasing* if every rule in R is non-erasing. The rule ρ is called *collapsing* if r is a variable. R is called *collapsing* if every rule in R is collapsing.

Let s and t be terms over a signature $\mathcal{F}$. We say that s *overlaps* t at a function-symbol position p ($\in \mathcal{O}_{\mathcal{F}}(t)$) if s is unifiable with $t|_p$. Let R be a CTRS over $\mathcal{F}$, $\rho_1 : l_1 \to r_1 \Leftarrow \mathcal{C}ond_1$ and $\rho_2 : l_2 \to r_2 \Leftarrow \mathcal{C}ond_2$ be conditional rewrite rules in R. We say that the rule ρ_1 is *overlapping* at a function-symbol position q ($\in \mathcal{O}_{\mathcal{F}}(l_2)$) of ρ_1 if l_1 overlaps l_2 at the position q. Moreover, ρ_1 and ρ_2 are *root-overlapping* if $q = \varepsilon$, and they are *inside-overlapping* if $\varepsilon < q$. Suppose that $\mathcal{V}ar(l_1) \cap \mathcal{V}ar(l_2) = \emptyset$, $l_2 \equiv C[u]_p$ with $u \notin \mathcal{X}$ ρ_1 overlaps ρ_2 at p. Let $\sigma = \mathtt{mgu}(u, l_1)$. Then, we call $\langle C[r_1\sigma]_p, r_2\sigma \rangle$ a *critical pair* of R with respect to $\to_R$, and an instance of $\langle C[r_1\sigma]_p, r_2\sigma \rangle$ a *critical peak* of R with respect to $\to_R$. We also call critical pair $\langle C[r_1\sigma]_p, r_2\sigma \rangle$ and instances of it *inside* if $\varepsilon < p$. R is said to be *non-overlapping* if every two rules in R are not overlapping, and it is called an *overlay system* if every two rules are not inside-overlapping.

The rules of a CTRS R over a signature $\mathcal{F}$ partition $\mathcal{F}$ into two disjoint sets as follows:

- $\mathcal{D}_R = \{ \mathtt{root}(l) \mid l \to r \Leftarrow \mathcal{C}ond \in R \}$ of *defined symbols*, and
- $C_R = \mathcal{F} \setminus \mathcal{D}_R$ of *constructors*.

A term $t \in \mathcal{T}(\mathcal{C}_R, \mathcal{X})$ is called a *constructor term* (of R). Furthermore, the CTRS R is a *constructor system* if every rule $f(t_1, \ldots, t_n) \to r \Leftarrow \mathcal{C}ond \in R$ satisfies $t_1, \ldots, t_n \in \mathcal{T}(\mathcal{C}_R, \mathcal{X})$. It is obvious that constructor systems are overlay systems. We write $C[\![t_1, \ldots, t_n]\!]$ instead of a term $C[t_1, \ldots, t_n]$ if C is a constructor context (that is, $C \in \mathscr{C}^n(\mathcal{C}_R, \mathcal{X})$) and $\mathtt{root}(t_i)$ is a defined symbol of R for $1 \leq i \leq n$.

Throughout this thesis, we will use the operator $\uplus$ to denote the *disjoint union* of sets. For example, to emphasize that $\mathcal{F}$ is the disjoint union of $\mathcal{D}_R$ and $\mathcal{C}_R$, we will often write $\mathcal{F} = \mathcal{D}_R \uplus \mathcal{C}_R$ instead of $\mathcal{F} = \mathcal{D}_R \cup \mathcal{C}_R$.

Let R be a CTRS over $\mathcal{F}$. A defined symbol f of R is said to be *sufficiently complete* if for all ground constructor terms $t_1, \ldots, t_n \in \mathcal{T}(\mathcal{C}_R)$, there exists a ground constructor term $t \in \mathcal{T}(\mathcal{C}_R)$ such that $f(t_1, \ldots, t_n) \xrightarrow{*}_R t$.

A conditional rewrite rule $\rho : l \to r \Leftarrow s_1 \twoheadrightarrow t_1 \wedge \cdots \wedge s_n \twoheadrightarrow t_n$ is called *deterministic* if it satisfies $\mathcal{V}ar(s_i) \subseteq \mathcal{V}ar(l, t_1, \ldots, t_{i-1})$ for $1 \leq i \leq n$. A 4-CTRS is called *deterministic* if all of its conditional rewrite rules are deterministic[2].

Let R be a CTRS over a signature $\mathcal{F}$ and $F \subseteq \mathcal{D}_R$. $R/_F$ denotes the set $\{ l \to r \Leftarrow \mathcal{C}ond \in R \mid \mathtt{root}(l) \in F \}$. $\mathcal{D}_{R,F}$ denotes the set of defined symbols which is necessary for computing functions in F, that is, $\mathcal{D}_{R,F}$ containing F is the least set that contains $f \in \mathcal{S}ym(r) \cap \mathcal{D}_R$ for every rule $l \to r \Leftarrow \mathcal{C}ond$ with $\mathtt{root}(l) \in \mathcal{D}_{R,F}$. R_F denotes the CTRS $\{ l \to r \Leftarrow \mathcal{C}ond \in R \mid \mathtt{root}(l) \in \mathcal{D}_{R,F} \}$. It is clear that $R = R_{\mathcal{D}_R}$.

2.8 Reduction Orderings

Let $\rhd$ be a binary relation on terms over a signature $\mathcal{F}$. We say that $\rhd$ is *closed under substitutions* if $s \rhd t$ implies $s\sigma \rhd t\sigma$ for all $\mathcal{T}(\mathcal{F}, \mathcal{X})$-substitutions σ.

A partial ordering $>$ is called *well-founded* if the ATRS $(\mathcal{T}(\mathcal{F}, \mathcal{X}), >)$ is strongly normalizing with respect to $>$.

A *rewrite ordering* is a partial ordering on terms which is closed under substitutions and contexts. A *reduction ordering* is a well-founded rewrite ordering.

The *strict part* of a quasi-ordering $\succsim$ is a partial ordering $\succ^{\mathrm{s}}$ defined as,

$$s \succ^{\mathrm{s}} t \text{ iff } s \succsim t \text{ and } t \not\succsim s,$$

[2]This definition is a natural extension of that for 3-CTRSs.

and the *stable-strict part* $\succ^{ss}$ is defined as,

$$s \succ^{ss} t \text{ iff } s\sigma \succ^{s} t\sigma \text{ for all ground substitutions } \sigma.$$

The quasi-ordering $\succsim$ is called *well-founded* if its stable-strict part $\succ^{ss}$ is well-founded. A *quasi-reduction ordering* is a well-founded quasi-ordering $\succsim$ that is closed under contexts and substitutions.

The *finite multiset extension* $\succ_{\mathsf{mul}}$ of a partial ordering $\succ$ on a set S is defined as; $M \succ_{\mathsf{mul}} N$ iff there exist finite multisets X, Y of S such that

- $\emptyset \neq X \subseteq M$,
- $N = (M \setminus X) \uplus Y$, and
- for all $y \in Y$, there exists an $x \in X$ such that $x \succ y$.

Let $\succ_{\mathsf{p}}$ be a partial ordering, called a *precedence* (ordering), on a signature $\mathcal{F}$. The corresponding *recursive path ordering* (*rpo*, for short) $\succ_{\mathsf{rpo}}$ on terms over $\mathcal{F}$ is defined as, $s \equiv f(s_1, \ldots, s_m) \succ_{\mathsf{rpo}} t$ iff

- $s_i \succeq_{\mathsf{rpo}} t$ for some i with $1 \leq i \leq m$, or
- $t \equiv g(t_1, \ldots, t_n)$, $f \succ_{\mathsf{p}} g$, and $s \succeq_{\mathsf{rpo}} t_j$ for $1 \leq i \leq n$, or
- $t \equiv f(t_1, \ldots, t_m)$ and $\{s_1, \ldots, s_m\} \succ_{\mathsf{rpo}}^{\mathsf{mul}} \{t_1, \ldots, t_m\}$.

Note that $\succ_{\mathsf{rpo}}^{\mathsf{mul}}$ denotes the finite multiset extension of $\succ_{\mathsf{rpo}}$.

Chapter 3

Inverse Compilers: Transformations for Inverse Computation

In this chapter, we define what is an *inverse system* (CTRS) of a CTRS and then we propose two inverse compilers of TRSs. The former one is for pure treeless functions which are often used in program transformations. The latter one is an extension of the former to constructor TRSs. Each of the inverse compilers first produces an inverse CTRS of a given TRS, and then transforms the generated CTRS into an EV-TRS which works similarly to the generated CTRS with respect to inverse functions and constructor terms.

This chapter is organized as follows. In Section 3.1, we define inverse CTRSs. Section 3.2 introduces pure treeless functions and the relationship between pure treeless functions and constructor TRSs. In Section 3.3, we intuitively explain an idea for generating inverses. In Section 3.4, we propose an inverse compiler for pure treeless TRSs and show the correctness of the transformation. Section 3.5 explains an idea to extend the inverse compiler for pure treeless TRSs to that for constructor TRSs. In Section 3.6, we propose an inverse compiler of constructor TRSs, which is an simple extension of the first one. We also show the correctness of the latter transformation. In Section 3.7, we improve the inverse compiler for pure treeless functions.

3.1 Inverse Systems

We first give a concrete definition of an *inverse system* of a CTRS, which computes inverses of functions defined by the CTRS.

We prepare special constructors $\{ \mathsf{tp}_0, \mathsf{tp}_1, \dots \}$, called *tuple symbols*, to represent tuples $(t_1, \dots, t_i)$ of i-terms $t_1, \dots, t_i$, encoding as $\mathsf{tp}_i(t_1, \dots, t_i)$. The reason why introducing tuple symbols is that inverses of n-ary functions return n terms, for example, $f^{-1}(t) = (t_1, \dots, t_n)$ for n-ary function f with $f(t_1, \dots, t_n) = t$. The tuple $\mathsf{tp}_1(t)$ of one term t may be abbreviated to the term t.

Introducing tuple symbols, we define inverse CTRSs as follows.

Definition 3.1.1 (Inverse CTRS) *Let R be a CTRS over a signature $\mathcal{F}$, R' be a CTRS over a signature $\mathcal{F}'$, and $(\mathcal{C}_R \setminus \{ \mathsf{tp}_i \mid 0 \leq i \}) \subseteq \mathcal{C}_{R'}$. We say that a defined symbol g of R' is an* inverse *of a defined symbol f of R if all of the following hold:*

- $\mathsf{tp}_n \in \mathcal{C}_{R'}$ *where* $n = \mathtt{arity}(f)$, *and*
- *for all constructor terms $t, t_1, \dots, t_n$ with respect to R, $f(t_1, \dots, t_n) \xrightarrow{*}_R t$ iff $g(t) \xrightarrow{*}_{R'} \mathsf{tp}_n(t_1, \dots, t_n)$.*

The CTRS R' is called an inverse system *of R if there exists an inverse defined symbol of R' for every defined symbol of R. Especially, R' is called an* inverse system *of R with respect to a set $D \subseteq \mathcal{D}_R$ of defined symbols of R if R' defines an inverse of every defined symbol in $\mathcal{D}_{R,D}$.*

Note that the first condition $\mathsf{tp}_n \in \mathcal{C}_{R'}$ above is not necessary if $n = 1$ and $\mathsf{tp}_1(t)$ is abbreviated to t.

Example 3.1.2 Let a signature $\mathcal{F} = \{\mathsf{double}, \mathsf{half}, \mathsf{s}, \mathsf{0}\}$. Consider the following TRSs over $\mathcal{F}$:

$$R_1 = \{ \mathsf{double}(\mathsf{0}) \to \mathsf{0},\ \ \mathsf{double}(\mathsf{s}(x)) \to \mathsf{s}^2(\mathsf{double}(x)) \},$$

$$R_2 = \{ \mathsf{half}(\mathsf{0}) \to \mathsf{0},\ \ \mathsf{half}(\mathsf{s}^2(x)) \to \mathsf{s}(\mathsf{half}(x)) \}.$$

The common constructor terms of R_1 and R_2 are in forms of $\mathsf{s}^n(\mathsf{0})$ for $n \geq 0$. R_1 and R_2 are inverse systems of each other since we have $\mathsf{double}(\mathsf{s}^n(\mathsf{0})) \xrightarrow{*}_{R_1} \mathsf{s}^{2n}(\mathsf{0})$ and $\mathsf{half}(\mathsf{s}^{2n}(\mathsf{0})) \xrightarrow{*}_{R_2} \mathsf{s}^n(\mathsf{0})$. □

3.2 Pure Treeless Functions and Constructor TRSs

In this section, we explain *pure treeless function* and constructor TRSs, both of which are the targets of inverse computation in this thesis. We first give a definition of pure treeless functions and then clarify the relationship between the classes of all pure treeless functions and all constructor TRSs.

We first explain what is a pure treeless function [8, 49]. This thesis follows the definition in [8]. The simplicity of the definition of pure treeless functions makes it easy to understand ideas, definitions and properties of program transformations. Therefore, pure treeless functions are often used as targets in program transformations, such as *deforestation* [49], *fusion transformation* [8] and so on.

We introduce treeless terms to which the right-hand sides of rules of pure treeless functions are restricted.

Definition 3.2.1 (Simple treeless terms) *Let $\mathcal{F}$ be a signature divided into defined symbols and constructors: $\mathcal{F} = \mathcal{D} \uplus \mathcal{C}$. A* simple treeless term *over $\mathcal{F}$ is inductively defined as follows:*

- *Variables are simple treeless terms.*
- *If c is a constructor with $\mathtt{arity}(c) = n$ and $tt_1, \ldots, tt_n$ are simple treeless terms, then the term $c(tt_1, \ldots, tt_n)$ is simple treeless.*
- *If f is a defined symbol with $\mathtt{arity}(f) = n$ and $x_1, \ldots, x_n$ are variables, then the term $f(x_1, \ldots, x_n)$ is simple treeless.*

It is clear that variables, constants and constructor terms are simple treeless terms.

Example 3.2.2 Consider the TRS R_1 in Example 3.1.2 again. The terms 0, $\mathsf{s}(x)$, $\mathsf{s}(\mathsf{double}(0))$ and $\mathsf{s}^2(\mathsf{double}(y))$ are simple treeless. On the other hand, the terms $\mathsf{double}(\mathsf{double}(x))$, $\mathsf{double}(\mathsf{s}(y))$ and $\mathsf{double}(0)$ are not. □

Pure treeless functions determined by TRSs are defined as follows [8].

Definition 3.2.3 (Pure treeless function) *Let R be a TRS over a signature $\mathcal{F}$. A defined symbol f of R is called a* pure treeless function *if $R/_{\{f\}}$ is in either forms of the following:*

(a) $R/_{\{f\}} = \{\ f(x_1, \ldots, x_n) \to tt\ \}$ *where* tt *is a simple treeless term over* $\mathcal{F}$ *and* $x_1, \ldots, x_n$ *are disjoint, or*

(b) $R/_{\{f\}} = \{\ f(c_i(x_{i,1}, \ldots, x_{i,n_1}), y_{i,2}, \ldots, y_{i,m_i}) \to tt_i \mid 1 \leq i \leq k, c_i \in \mathcal{C}_R\ \}$ *where* $c_1, \ldots, c_k$ *are different from each other,* $tt_1, \ldots, tt_k$ *are simple treeless terms over* $\mathcal{F}$ *and* $x_{i,1}, \ldots, x_{i,n_i}, y_{i,2}, \ldots, y_{i,m_i}$ *are disjoint variables for every* i *with* $1 \leq i \leq k$.

The TRS R *is called* pure treeless *(*PT-TRS*, for short) if every defined symbol of* R *is pure treeless. Especially,* R *is called* pure treeless *with respect to a set* $F \subseteq \mathcal{D}_R$ *of defined symbols of* R *if every defined symbol in* F *is pure treeless.*

It is clear that PT-TRSs are non-overlapping left-linear constructor TRSs.

Example 3.2.4 The TRSs R_1 and R_2 in Example 3.1.2 are pure treeless. The following TRS which computes addition of natural numbers encoded by s and $\mathsf{0}$ is also pure treeless:

$$R_3 = \{\ \mathsf{add}(\mathsf{0}, y) \to y,\ \ \mathsf{add}(\mathsf{s}(x), y) \to \mathsf{s}(\mathsf{add}(x, y))\ \}.$$

□

The definition of simple treeless terms (Definition 3.2.1) is a simple extension of that of treeless terms seen in [8], relaxing the linearity of treeless terms. Therefore, the definition of pure treeless functions in this thesis is also a simple extension of the original of pure treeless functions in [8].

A constructor term which contains at least one constructor is called a *pattern*, and it is called a *simple pattern* if it contains just a constructor, that is, in forms of $c(x_1, \ldots, x_k)$. A pattern which is not simple is called a *nested pattern*. It is known that any definition of a function with nested pattern in the first argument can be translated to an equivalent definition with only simple patterns [6].

The description in pure treeless functions is very restricted since it is not allowed them to contain a nest of defined symbols in the right-hand side of rewrite rules. For example, any pure treeless function cannot express the multiplication of natural numbers, based on the usual definition:

$$\begin{aligned} \mathsf{0} \times y &= \mathsf{0}, & \mathsf{0} + y &= y, \\ \mathsf{s}(x) \times y &= x \times y + y, & \mathsf{s}(x) + y &= \mathsf{s}(x + y). \end{aligned}$$

The multiplication of natural numbers is defined as a pure treeless function in the following way:

$$\begin{aligned} R_4 = \{\ & \mathsf{mul_{PT}}(\mathsf{0}, y) \to \mathsf{0}, \\ & \mathsf{mul_{PT}}(\mathsf{s}(x), y) \to \mathsf{addmul}(y, x, y), \\ & \mathsf{addmul}(\mathsf{0}, y, z) \to \mathsf{mul_{PT}}(y, z), \\ & \mathsf{addmul}(\mathsf{s}(x), y, z) \to \mathsf{s}(\mathsf{addmul}(x, y, z))\ \}. \end{aligned}$$

The 3-ary function addmul computes the addition of its first argument and the multiplication of the second and third arguments, that is, $\mathsf{addmul}(v_1, v_2, v_3) = v_1 + v_2 \times v_3$ where v_i is a natural number.

Since the description in pure treeless functions is very restricted, the computability of pure treeless functions is weaker than that of Turing machines, that is, the pure treeless expression cannot define all computable functions [49]. For example, there is no pure treeless definition of the function flatten which computes a list consisting of all leaves of a given binary tree;

$$\begin{aligned} \mathsf{flatten}(\mathsf{nil}) &= \mathsf{nil}, & \mathsf{flatten}(\mathsf{branch}(x, y)) &= \mathsf{cons}(\mathsf{flatten}(x), \mathsf{flatten}(y)), \\ \mathsf{append}(\mathsf{nil}, y) &= y, & \mathsf{append}(\mathsf{cons}(x, xs), ys) &= \mathsf{cons}(x, \mathsf{append}(xs, ys)) \end{aligned}$$

where lists are constructed by cons and nil, and binary trees by branch and nil. It is obvious that the function flatten above can be easily defined by a constructor TRS. It is also clear that PT-TRSs are constructor systems. Therefore, the class of all pure treeless functions (PT-TRSs) is properly included in that of all constructor TRSs.

On the other hand, the computability of constructor TRSs is equivalent to that of Turing machines. It is because they can express all *recursive program schemes* (*RPSs*) [30] and it is known that the computability of RPSs is equal to that of Turing machines.

Here we describe the general form of rewrite rules of PT-TRSs. Let $\mathcal{F}$ be a signature divided into defined symbols and constructors: $\mathcal{F} = \mathcal{D} \uplus \mathcal{C}$. It is clear that every simple treeless term can be represented as $C[\![t_1, \ldots, t_n]\!]$ by a constructor context $C \in \mathscr{C}^n(\mathcal{C}, \mathcal{X})$ and terms $t_1, \ldots, t_n$ with $\mathtt{root}(t_i) \in \mathcal{D}$. Then, the following proposition is easily led from Definition 3.2.3.

Proposition 3.2.5 *Let R be a PT-TRS. Every rewrite rule in R is represented as the following form:*

$$f(p, x_2, \ldots, x_n) \to C[\![f_1(x_{1,1}, \ldots, x_{1,n_1}), \ldots, f_m(x_{m,1}, \ldots, x_{m,n_m})]\!]$$

where p is either a variable or a simple pattern, $f(p, x_2, \ldots, x_n)$ is linear, C is a constructor context of R, f_i is a defined symbol of R and $x_{i,j} \in \mathcal{V}ar(p) \cup \{x_2, \ldots, x_n\}$.

3.3 Idea for Transformation

In this section, we give an intuitive explanation of an idea for the transformations, which we propose in this chapter as inverse compilers for PT-TRSs and constructor TRSs respectively.

Each of the inverse compilers in this chapter consists of two parts: a actual *program inverter* from an input TRS to a CTRS which is an inverse of the input TRS, and an *unraveling* transformation from the generated CTRS to an equivalent EV-TRS. The first transformation is our main transformation as an inverse compiler. The reason why our inverse compilers consist of two parts will be shown in the following explanation.

We first consider a property which an inverse of any function should satisfy in general. As a mathematical property of inverses, the following equation holds for all inverses:

$$f^{-1}(f(x)) = x$$

where f is a function (in other words, a defined symbol) and f^{-1} is an inverse of f. Generalizing the above equation for n-ary functions by decoding an n-term tuple $(x_1, \ldots, x_n)$ as the term $\mathsf{tp}_n(x_1, \ldots, x_n)$, we obtain the following equation:

$$f^{-1}(f(x_1, \ldots, x_n)) = \mathsf{tp}_n(x_1, \ldots, x_n). \tag{3.1}$$

If the above equation is considered as a rewrite rule of f^{-1}, such a system is not useful for practical inverse computation. It is because the argument of f^{-1} is the term $f(x_1, \ldots, x_n)$ whose root symbol is a defined symbol f of the input TRS. Therefore, we try to transform a given TRS using the equation (3.1) which inverses should satisfy.

A basic idea of our transformation is to construct one rule for each rewrite rule $f(\cdots) \to r$ in a given TRS, by transforming it as follows;

Step 1. Apply the inverse function f^{-1} of the given function f to that rewrite rule $f(\cdots) \to r$;

Step 2. Replace the both hand sides of the rule obtained after **Step 1**;

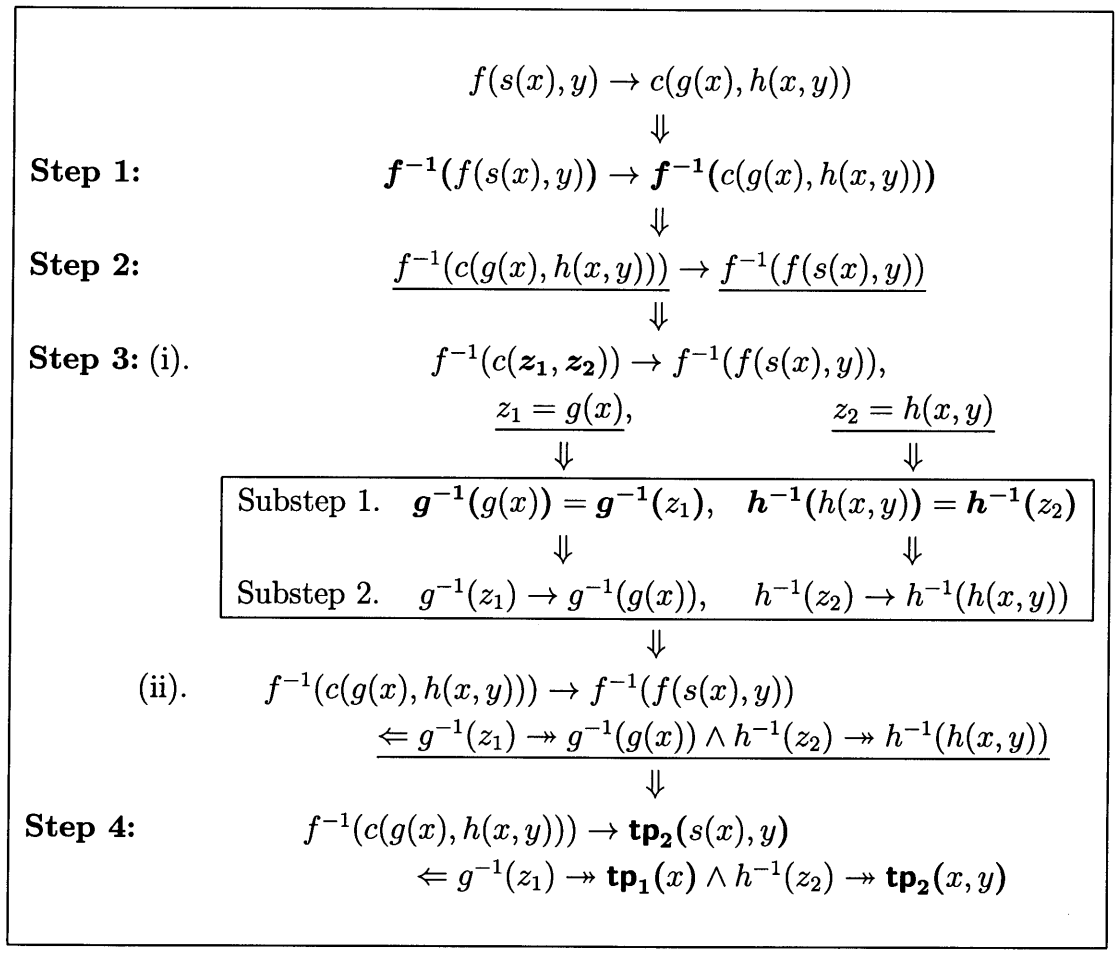

Figure 3.1: Outline of transformation of a rule $f(s(x),y) \to c(g(x),h(x,y))$.

Step 3. Remove all defined symbols of the given TRS R by replacing each term whose root symbol is a defined symbol of R, with a fresh variable, and then adding such replacement in the conditional part; and

Step 4. Simplify the rule by replacing each term $g^{-1}(g(u_1,\ldots,u_m))$ with a term tuple $\mathsf{tp}_m(u_1,\ldots,u_m)$, based on the equation (3.1).

An instance of the above transformation is shown in Figure 3.1. Constructing one rule for each rewrite is a natural approach because each rewrite rule of a given function defines some property of the function.

The above transformation **Step 1–4** of a given rewrite rule is technical and hence we explain the detail in the following paragraphs.

Consider a PT-TRS R and a defined symbol f of R. As shown in Proposi-

tion 3.2.5, every rewrite rule of f can be represented as follows:

$$f(p, x_2, \ldots, x_n) \to C[\![f_1(x_{1,1}, \ldots, x_{1,n_1}), \ldots, f_m(x_{m,1}, \ldots, x_{m,n_m})]\!] \tag{3.2}$$

where f_i is a defined symbol of R.

After **Step 1** is done for the above rule, we obtain the following rule:

$$f^{-1}(f(p, x_2, \ldots, x_n)) \to f^{-1}(C[\![f_1(x_{1,1}, \ldots, x_{1,n_1}), \ldots, f_m(x_{m,1}, \ldots, x_{m,n_m})]\!]). \tag{3.3}$$

In the sense of mathematics, the equation obtained by applying a function to an invariant equation, also holds, that is,

$$s = t \text{ implies } g(s) = g(t) \text{ for every unary function } g.$$

Since a rewrite rule can be considered as an equation with a direction, the semantic of the rule (3.3) is correct (the correctness of the whole transformation will be shown in Subsection 3.4.2).

Next **Step 2** is done. Then we obtain the following rule:

$$f^{-1}(C[\![f_1(x_{1,1}, \ldots, x_{1,n_1}), \ldots, f_m(x_{m,1}, \ldots, x_{m,n_m})]\!]) \to f^{-1}(f(p, x_2, \ldots, x_n)). \tag{3.4}$$

Because the defined symbols $f_1, \ldots, f_m$ of R are still remaining in the left-hand side, the next procedure **Step 3** is necessary.

To eliminate defined symbols $f_1, \ldots, f_m$ of R, as the first substep of the **Step 3**, we replace the terms $f_i(x_{i,1}, \ldots, x_{i,n_i})$ with fresh variables y_i:

$$f^{-1}(C[\![y_1, \ldots, y_m]\!]) \to f^{-1}(f(p, x_2, \ldots, x_n))$$

where $y_i \equiv f_i(x_{i,1}, \ldots, x_{i,n_i})$. Here, the word 'fresh' means that $y_i \notin \mathcal{V}ar(p) \cup \{ x_2, \ldots, x_n \}$ and $y_1, \ldots, y_m$ are disjoint. As the second substep, we add the oriented condition $f_i^{-1}(y_i) \twoheadrightarrow f_i^{-1}(f_i(x_{i,1}, \ldots, x_{i,n_i}))$ which is obtained by applying **Step 1** and **2** to the equation $y_i \equiv f_i(x_{i,1}, \ldots, x_{i,n_i})$, into the conditional part, as follows:

$$\begin{aligned} f^{-1}(C[\![y_1, \ldots, y_m]\!]) \to f^{-1}(f(p, x_2, \ldots, x_n)) \\ \Leftarrow f_1^{-1}(y_1) \twoheadrightarrow f_1^{-1}(f_1(x_{1,1}, \ldots, x_{1,n_1})) \\ \wedge \cdots \wedge f_m^{-1}(y_m) \twoheadrightarrow f_m^{-1}(f_m(x_{m,1}, \ldots, x_{m,n_m})). \end{aligned} \tag{3.5}$$

Finally, **Step 4** is done. In this step, a term such as $g^{-1}(g(t_1, \ldots, t_k))$ in the rule (3.4) is replaced with the k-term tuple $\mathsf{tp}_k(t_1, \ldots, t_k)$ based on the property

(3.1), and hence we obtain the following conditional rewrite rule:

$$\begin{aligned} f^{-1}(C[\![y_1,\ldots,y_m]\!]) \to \mathsf{tp}_n(p,x_2,\ldots,x_n) \\ \Leftarrow f_1^{-1}(y_1) \twoheadrightarrow \mathsf{tp}_{n_1}(x_{1,1},\ldots,x_{1,n_1}) \\ \wedge\cdots\wedge f_m^{-1}(y_m) \twoheadrightarrow \mathsf{tp}_{n_m}(x_{m,1},\ldots,x_{m,n_m}). \end{aligned} \tag{3.6}$$

A CTRS R^{-1} consisting of only rules like (3.6) is a constructor system because the argument $C[\![y_1,\ldots,y_m]\!]$ of f^{-1} in the left-hand side of (3.6) is a constructor term.

On the other hand, applying **Step 3** and **4** to the equation (3.1) is not sufficient to generate useful conditional rewrite rules of inverses. If it is done, then the following rule is obtained:

$$f^{-1}(y) \to \mathsf{tp}_n(x_1,\ldots,x_n) \Leftarrow f^{-1}(y) \twoheadrightarrow \mathsf{tp}_n(x_1,\ldots,x_n).$$

It is clear that the above rule is redundant and usable.

To formalize the above idea of transformation, we focus on **Step 3** because **Step 1**, **2** and **4** are straightforward and can be defined syntactically, and hence they are easily defined after **Step 3** is done. For example, the rule (3.6) is easily constructed from the original rule (3.2) with the substitutive term $C[\![y_1,\ldots,y_m]\!]$ of the term $C[\![f_1(x_{1,1},\ldots,x_{1,n_1}),\ldots,f_m(x_{m,1},\ldots,x_{m,n_m})]\!]$ and the conditional part $f_1^{-1}(y_1) \twoheadrightarrow \mathsf{tp}_{n_1}(x_{1,1},\ldots,x_{1,n_1}) \wedge\cdots\wedge f_m^{-1}(y_m) \twoheadrightarrow \mathsf{tp}_{n_m}(x_{m,1},\ldots,x_{m,n_m})$. Therefore, instead of **Step 3**, we give a procedure which generates the constructor term and the conditional part from the input term similarly to **Step 3**, that is,

$C[y_1,\ldots,y_m]$ and
$f_1^{-1}(y_1) \twoheadrightarrow \mathsf{tp}_{n_1}(x_{1,1},\ldots,x_{1,n_1}) \wedge\cdots\wedge f_m^{-1}(y_m) \twoheadrightarrow \mathsf{tp}_{n_m}(x_{m,1},\ldots,x_{m,n_m})$

from $C[\![f_1(x_{1,1},\ldots,x_{1,n_1}),\ldots,f_m(x_{m,1},\ldots,x_{m,n_m})]\!]$.

In the next section, we will formalize the above idea, and then we will show the correctness of the transformation (the above idea), that is, the above R^{-1} computes inverses of functions defined in the input system R.

3.4 Inverse Compiler for Pure Treeless TRSs

In this section, we give a precise definition of our inverse compiler, following the idea described in Section 3.3.

To represent the inverse function symbols defined by our inverse compiler, we use the symbol #. That is, $f^{\#}$ is the inverse function symbol of the defined

symbol f. This thesis assumes that any TRS R over a given signature $\mathcal{F}$ provides no constructor with the superscription $\#$: $\mathcal{C}_R \cap \{ f^{\#} \mid f \in \mathcal{F} \} = \emptyset$. Suppose that the signature $\mathcal{F}$ divided into the constructor set $\mathcal{C}$ and the defined-symbol set $\mathcal{D}$: $\mathcal{F} = \mathcal{C} \uplus \mathcal{D}$. We define the signature $\mathcal{F}^{\#}$ determined by $\mathcal{F}$ as follows:

$$\mathcal{F}^{\#} = \mathcal{F} \cup \{ f^{\#} \mid f \in \mathcal{D} \} \cup \{ \mathsf{tp}_i \mid \min_{f \in \mathcal{D}}(\mathtt{arity}(f)) \leq i \leq \max_{f \in \mathcal{D}}(\mathtt{arity}(f)) \}.$$

3.4.1 Generation of Inverse CTRSs

As a formalization of **Step 3**, we first define a procedure $\mathscr{T}_{\mathsf{PT}}$ which outputs the pair $\langle\!\langle\, u \,;\, \mathcal{C}ond \,\rangle\!\rangle$ of a term u and a conditional part $\mathcal{C}ond$ from an input simple treeless term r. This procedure $\mathscr{T}_{\mathsf{PT}}$ defines **Step 3** formally.

Definition 3.4.1 ($\mathscr{T}_{\mathsf{PT}}$) *Let $\mathcal{F}$ be a signature divided into the constructor set $\mathcal{C}$ and the defined-symbol set $\mathcal{D}$; $\mathcal{F} = \mathcal{C} \uplus \mathcal{D}$. The procedure $\mathscr{T}_{\mathsf{PT}}$ inputted a simple treeless term is inductively defined as follows:*

(a) $\mathscr{T}_{\mathsf{PT}}(x) = \langle\!\langle\, x \,;\, \mathtt{true} \,\rangle\!\rangle$ *where x is a variable,*

(b) $\mathscr{T}_{\mathsf{PT}}(c(t_1, \ldots, t_n)) = \langle\!\langle\, c(u_1, \ldots, u_n) \,;\, \mathcal{C}ond_1 \wedge \cdots \wedge \mathcal{C}ond_n \,\rangle\!\rangle$
where $c \in \mathcal{C}$ and $\mathscr{T}_{\mathsf{PT}}(t_i) = \langle\!\langle\, u_i \,;\, \mathcal{C}ond_i \,\rangle\!\rangle$, and

(c) $\mathscr{T}_{\mathsf{PT}}(f(x_1, \ldots, x_n)) = \langle\!\langle\, y \,;\, f^{\#}(y) \twoheadrightarrow \mathsf{tp}_n(x_1, \ldots, x_n) \,\rangle\!\rangle$
where $f \in \mathcal{D}$ and y is a fresh variable.

The word 'fresh' in (c) means that in (c), $y \notin \{x_1, \ldots, x_n\}$, and in (b), variables introduced in $\mathscr{T}_{\mathsf{PT}}(t_i) = \langle\!\langle\, u_i \,;\, \mathcal{C}ond_i \,\rangle\!\rangle$ and $\mathscr{T}_{\mathsf{PT}}(t_j) = \langle\!\langle\, u_j \,;\, \mathcal{C}ond_j \,\rangle\!\rangle$ are disjoint, that is, $\mathcal{V}ar(u_i, \mathcal{C}ond_i) \cap \mathcal{V}ar(t_j, u_j, \mathcal{C}ond_j) = \emptyset$ for each i and j with $i \neq j$.

Since the size of input terms is finite, the above procedure $\mathscr{T}_{\mathsf{PT}}$ terminates and always return a pair.

Example 3.4.2 Consider the following PT-TRS again:

$$R_3 = \{ \mathsf{add}(0, y) \to y,\ \mathsf{add}(\mathsf{s}(x), y) \to \mathsf{s}(\mathsf{add}(x, y)) \}.$$

Inputting term $\mathsf{s}(\mathsf{add}(x, y))$ to $\mathscr{T}_{\mathsf{PT}}$, we have the following result:

$$\mathscr{T}_{\mathsf{PT}}(\mathsf{s}(\mathsf{add}(x, y))) = \langle\!\langle\, \mathsf{s}(z) \,;\, \mathsf{add}^{\#}(z) \twoheadrightarrow \mathsf{tp}_2(x, y) \,\rangle\!\rangle.$$

□

The following proposition shows the relationship between a given input term and the output pair of a term and a conditional part of $\mathscr{T}_{\mathsf{PT}}$.

Proposition 3.4.3 *Let $\mathcal{F}$ be a signature divided into the constructor set $\mathcal{C}$ and the defined-symbol set $\mathcal{D}$: $\mathcal{F} = \mathcal{C} \uplus \mathcal{D}$. Let t be a simple treeless term with $t \equiv C[\![f_1(x_{1,1}, \ldots, x_{1,n_1}), \ldots, f_m(x_{m,1}, \ldots, x_{m,n_m})]\!]$ and suppose that $\mathscr{T}_{\mathsf{PT}}(t) = \langle\!\langle\, u \,;\, \mathcal{C}ond \,\rangle\!\rangle$. Then,*

(a) $u \equiv C[y_1, \ldots, y_m]$, and

(b) $\mathcal{C}ond = \bigwedge_{i=1}^{m} f_i^{\#}(y_i) \twoheadrightarrow \mathsf{tp}_{n_i}(x_{i,1}, \ldots, x_{i,n_i})$

for some variables $y_1, \ldots, y_m$ such that $y_i \notin \mathcal{V}ar(C) \cup \bigcup_{i=1}^{m} \{\, x_{i,1}, \ldots, x_{i,n_i} \,\}$. Moreover, if t is linear, then

(c) u is linear,

(d) $\mathsf{tp}_{n_i}(x_{i,1}, \ldots, x_{i,n_i})$ is linear,

(e) $\mathsf{tp}_{n_i}(x_{i,1}, \ldots, x_{i,n_i}) \cap \mathsf{tp}_{n_j}(x_{j,1}, \ldots, x_{j,n_j}) = \emptyset$ for $i \neq j$, and

(f) $\mathcal{V}ar(C) \cap \mathsf{tp}_{n_i}(x_{i,1}, \ldots, x_{i,n_i}) = \emptyset$.

Proof. The claims (a) and (b) follow from the definition of $\mathscr{T}_{\mathsf{PT}}$. The claims (c)–(f) follow from the linearity of t. □

Now, using the above procedure $\mathscr{T}_{\mathsf{PT}}$ (**Step 3**), we define a transformation which formalize **Step 1**, **2** and **4**.

Definition 3.4.4 ($\mathcal{I}nv_{\mathsf{PT}}$) *Let R be a PT-TRS over a signature $\mathcal{F}$. The transformation $\mathcal{I}nv_{\mathsf{PT}}$ of PT-TRSs is defined as follows:*

$$\begin{aligned}\mathcal{I}nv_{\mathsf{PT}}(R) = \{\; & f^{\#}(r') \to \mathsf{tp}_n(p, x_2, \ldots, x_n) \Leftarrow \mathcal{C}ond \\ & \mid\; f(p, x_2, \ldots, x_n) \to r \in R,\ \mathscr{T}_{\mathsf{PT}}(r) = \langle\!\langle\, r' \,;\, \mathcal{C}ond \,\rangle\!\rangle, \\ & (\mathcal{V}ar(r', \mathcal{C}ond) \setminus \mathcal{V}ar(r)) \cap \mathcal{V}ar(p, x_2, \ldots, x_n) = \emptyset \;\}.\end{aligned}$$

It is clear that $\mathcal{I}nv_{\mathsf{PT}}(R)$ above is a finite set of conditional rewrite rules over the signature $\mathcal{F}^{\#}$ and hence a CTRS over $\mathcal{F}^{\#}$. The correctness of $\mathcal{I}nv_{\mathsf{PT}}$, i.e., the fact that $\mathcal{I}nv_{\mathsf{PT}}(R)$ is an inverse system of R, will be proved in Subsection 3.4.2.

Example 3.4.5 The PT-TRS R_3 in Example 3.4.2 is transformed by $\mathcal{I}nv_{\mathsf{PT}}$ to the following CTRS:

$$R_5 = \mathcal{I}nv_{\mathsf{PT}}(R_3) = \{\ \mathsf{add}^{\#}(y) \to \mathsf{tp}_2(0, y),\\ \mathsf{add}^{\#}(\mathsf{s}(z)) \to \mathsf{tp}_2(\mathsf{s}(x), y) \Leftarrow \mathsf{add}^{\#}(z) \twoheadrightarrow \mathsf{tp}_2(x, y)\ \}.$$

□

In the rest of this subsection, we discuss a relationship between function symbols of a PT-TRS R and the generated CTRS $\mathcal{I}nv_{\mathsf{PT}}(R)$ from R, and also syntactic properties of $\mathcal{I}nv_{\mathsf{PT}}(R)$ associated with those of R.

Proposition 3.4.6 *Let R be a PT-TRS over a signature $\mathcal{F}$.*

(a) Every defined symbol of $\mathcal{I}nv_{\mathsf{PT}}(R)$ consists of a defined symbol of R and $\#$, that is, $\mathcal{D}_{\mathcal{I}nv_{\mathsf{PT}}(R)} = \{\ f^{\#} \mid f \in \mathcal{D}_R\ \}$.

(b) Every function symbol in $\mathcal{F}$ is a constructor of $\mathcal{I}nv_{\mathsf{PT}}(R)$, that is, $\mathcal{F} \subseteq \mathcal{C}_{\mathcal{I}nv_{\mathsf{PT}}(R)}$.

(c) Every term over $\mathcal{F}$ is a constructor term of $\mathcal{I}nv_{\mathsf{PT}}(R)$, that is, $\mathcal{T}(\mathcal{F}, \mathcal{X}) \subseteq \mathcal{T}(\mathcal{C}_{\mathcal{I}nv_{\mathsf{PT}}(R)}, \mathcal{X})$.

Proof. From Definition 3.4.4, every rewrite rule $f(p, x_2, \ldots, x_n) \to r \in R$ is transformed to the conditional rewrite rule $f^{\#}(r') \to \mathsf{tp}_n(p, x_2, \ldots, x_n) \Leftarrow \mathcal{C}ond \in \mathcal{I}nv_{\mathsf{PT}}(R)$ where $\mathscr{T}_{\mathsf{PT}}(r) = \langle\!\langle\ r'\ ;\ \mathcal{C}ond\ \rangle\!\rangle$. Then, it follows from the definition of defined symbols that $\mathcal{D}_{\mathcal{I}nv_{\mathsf{PT}}(R)} = \{\ f^{\#} \mid f \in \mathcal{D}_R\ \}$. Hence, the claim (a) holds. It is clear that the claim (b) follows from (a), and the claim (c) follows from (b). □

Proposition 3.4.7 *Let R be a PT-TRS over a signature $\mathcal{F}$ and a rewrite rule $\rho : f(p, x_2, \ldots, x_n) \to C[\![f_1(x_{1,1}, \ldots, x_{1,n_1}), \ldots, f_m(x_{m,1}, \ldots, x_{m,n_m})]\!] \in R$. Then, there exists the following conditional rewrite rule in $\mathcal{I}nv_{\mathsf{PT}}(R)$:*

$$f^{\#}(C[y_1, \ldots, y_m]) \to \mathsf{tp}_n(p, x_2, \ldots, x_n) \Leftarrow \bigwedge_{i=1}^{m} f_i^{\#}(y_i) \twoheadrightarrow \mathsf{tp}_{n_i}(x_{i,1}, \ldots, x_{i,n_i})$$

where $y_i \notin \mathcal{V}ar(p) \cup \{x_2, \ldots, x_n\}$, $\mathsf{tp}_n(p, x_2, \ldots, x_n)$ is linear, C is a constructor term of $\mathcal{I}nv_{\mathsf{PT}}(R)$ and $\mathcal{V}ar(C) \subseteq \mathcal{V}ar(p) \cup \{x_2, \ldots, x_n\}$. Moreover, if the rule ρ is right-linear, then the above conditional rewrite rule satisfies the following two properties:

(a) $\mathcal{V}ar(C[y_1,\ldots,y_m]) \cap \bigcup_{i=1}^{m} \mathcal{V}ar(\mathsf{tp}_{n_i}(x_{i,1},\ldots,x_{i,n_i})) = \emptyset$, *and*

(b) $\mathcal{V}ar(\mathsf{tp}_{n_i}(x_{i,1},\ldots,x_{i,n_i})) \cap \mathcal{V}ar(\mathsf{tp}_{n_j}(x_{j,1},\ldots,x_{j,n_j})) = \emptyset$ *for any* i *and* j *with* $i \neq j$.

Proof. The claims of this proposition clearly hold from Proposition 3.2.5, 3.4.3 and Definition 3.4.4. □

The above Proposition 3.4.7 and the linearity of rewrite rules provide the following proposition.

Proposition 3.4.8 *Let R be a PT-TRS over a signature $\mathcal{F}$.*

- *If R is left-linear, then $\mathcal{I}nv_{\mathsf{PT}}(R)$ is right-linear.*
- *If R is right-linear, then $\mathcal{I}nv_{\mathsf{PT}}(R)$ is left-linear.*

3.4.2 Correctness of Generation

In this subsection, we show that for any PT-TRS R, the CTRS $\mathcal{I}nv_{\mathsf{PT}}(R)$ is an inverse system of R.

Theorem 3.4.9 *Let R be a PT-TRS over a signature $\mathcal{F}$. Let f be a defined symbol of R and $t, t_1, \ldots, t_n$ be constructor terms with respect to R. Then, $f(t_1,\ldots,t_n) \xrightarrow{*}_R t$ iff $f^{\#}(t) \to_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_n(t_1,\ldots,t_n)$.*

Proof. We first prove the *only-if* part by induction on the number k of steps of $f(t_1,\ldots,t_n) \xrightarrow{k}_R t$.

Since t is a constructor term of R, we have $k > 0$. Then, from Proposition 3.2.5, we can assume the following:

$$\begin{aligned} f(t_1,\ldots,t_n) &\equiv f(p,x_2,\ldots,x_n)\sigma \\ &\to_R^{[\varepsilon,\rho]} (C[\![f_1(x_{1,1},\ldots,x_{1,n_1}),\ldots,f_m(x_{m,1},\ldots,x_{m,n_m})]\!])\sigma \xrightarrow{k-1}_R t \end{aligned}$$

where $\rho : f(p,x_2,\ldots,x_n) \to C[\![f_1(x_{1,1},\ldots,x_{1,n_1}),\ldots,f_m(x_{m,1},\ldots,x_{m,n_m})]\!] \in R$ and p is either a variable or a simple pattern. From $t_1,\ldots,t_n \in \mathcal{T}(\mathcal{C}_R,\mathcal{X})$, we can assume without loss of generality that σ is a $\mathcal{T}(\mathcal{C}_R,\mathcal{X})$-substitution. It follows from $C \in \mathscr{C}^m(\mathcal{C}_R,\mathcal{X})$ that $t \equiv (C\sigma)[u_1,\ldots,u_m]$ for some $u_1,\ldots,u_m \in \mathcal{T}(\mathcal{C}_R,\mathcal{X})$. Then, we have the following sequence:

$$f_i(x_{i,1},\ldots,x_{i,n_i})\sigma \xrightarrow{k_i}_R u_i$$

where $k_i \leq k-1$ and $x_{i_j}\sigma \in \mathcal{T}(\mathcal{C}_R, \mathcal{X})$. Then, by the induction hypothesis, the following holds:

$$f_i^{\#}(u_i) \rightarrow_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_{n_i}(x_{i,1}, \ldots, x_{i,n_i})\sigma.$$

On the other hand, we have the following rule in $\mathcal{I}nv_{\mathsf{PT}}(R)$ from Proposition 3.4.7 for the rule ρ:

$$\begin{aligned} f^{\#}(C[y_1, \ldots, y_m]) &\rightarrow \mathsf{tp}_n(p, x_2, \ldots, x_n) \\ &\Leftarrow \textstyle\bigwedge_{i=1}^{m} f_i^{\#}(y_i) \twoheadrightarrow \mathsf{tp}_{n_i}(x_{i,1}, \ldots, x_{i,n_i}) \in \mathcal{I}nv_{\mathsf{PT}}(R). \end{aligned}$$

Let $\theta = \sigma|_{\mathcal{V}ar(p,x_2,\ldots,x_n)} \cup \bigcup_{i=1}^{m}\{y_i \mapsto u_i\}$. Since $y_i \notin \mathcal{V}ar(p, x_2, \ldots, x_n)$ follows from Proposition 3.4.7, θ is a substitution. Then θ and $\xrightarrow{*}_{\mathcal{I}nv_{\mathsf{PT}}(R)}$ satisfy the conditional part of the above conditional rule of $f^{\#}$ and hence

$$f^{\#}(t) \equiv f^{\#}(C[y_1, \ldots, y_m])\theta \rightarrow_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_n(p, x_2, \ldots, x_n)\theta \equiv \mathsf{tp}_n(t_1, \ldots, t_n).$$

Secondly, we prove the *if* part by induction on the level k of the rewrite relation of $f^{\#}(t) \xrightarrow[k]{}_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_n(t_1, \ldots, t_n)$.

We assume that the rewrite step $f^{\#}(t) \xrightarrow[k]{}_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_n(t_1, \ldots, t_n)$ is done by the following rule in $\mathcal{I}nv_{\mathsf{PT}}(R)$:

$$\begin{aligned} f^{\#}(C[y_1, \ldots, y_m]) &\rightarrow \mathsf{tp}_n(p, x_2, \ldots, x_n) \\ &\Leftarrow \textstyle\bigwedge_{i=1}^{m} f_i^{\#}(y_i) \twoheadrightarrow \mathsf{tp}_{n_i}(x_{i,1}, \ldots, x_{i,n_i}) \in \mathcal{I}nv_{\mathsf{PT}}(R). \end{aligned}$$

From the construction of $\mathcal{I}nv_{\mathsf{PT}}(R)$, we also have the following rule in R:

$$f(p, x_1, \ldots, x_n) \rightarrow C[\![f_1(x_{1,1}, \ldots, x_{1,n_1}), \ldots, f_m(x_{m,1}, \ldots, x_{m,n_m})]\!] \in R.$$

Then there exists a substitution θ which satisfies with $\xrightarrow[k-1]{*}_{\mathcal{I}nv_{\mathsf{PT}}(R)}$ the conditional part of the above conditional rewrite rule in $\mathcal{I}nv_{\mathsf{PT}}(R)$, and it satisfies all of the following:

- $y_i\theta \equiv t_i$,
- $(C[y_1, \ldots, y_m])\theta \equiv t$, and
- $p\theta \equiv t_1$, $x_i\theta \equiv t_i$ for $2 \leq i \leq n$.

It clearly follows from the form of rules in $\mathcal{I}nv_{\mathsf{PT}}(R)$ that $f_i^{\#}(y_i)\theta \xrightarrow[k-1]{}_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_{n_i}(x_{i,1}, \ldots, x_{i,n_i})\theta$. It is also clear that $y_i\theta$, $x_{i,j}\theta \in \mathcal{T}(\mathcal{C}_R, \mathcal{X})$. Then, by the induction hypothesis, we have the following:

$$f_i(x_{i,1}, \ldots, x_{i,n_i})\theta \xrightarrow{*}_R y_i\theta_i.$$

Therefore, the following holds:

$$\begin{aligned} f(t_1,\ldots,t_n) &\equiv f(p,x_2,\ldots,x_n)\theta \\ &\to_R (C[\![f_1(x_{1,1},\ldots,x_{1,n_1}),\ldots,f_m(x_{m,1},\ldots,x_{m,n_m})]\!])\theta \\ &\overset{*}{\to}_R (C[y_1,\ldots,y_m])\theta \equiv t. \end{aligned}$$

□

According to Theorem 3.4.9, $\mathcal{I}nv_{\mathsf{PT}}(R)$ is an inverse system of R in the sense of Definition 3.1.1.

Example 3.4.10 Consider the PT-TRS R_3 and the CTRS R_5 in Example 3.4.2 and 3.4.5, respectively. The addition of 1 and 2 is computed by R_3 as the following rewrite sequence:

$$\mathsf{add}(\mathsf{s}(0),\mathsf{s}^2(0)) \overset{*}{\to}_{R_3} \mathsf{s}^3(0).$$

On the other hand, we have a rewrite sequence

$$\mathsf{add}^{\#}(\mathsf{s}^3(0)) \to_{R_5} \mathsf{tp}_2(\mathsf{s}(0),\mathsf{s}^2(0)),$$

which is one of inverse computation of $\mathsf{add}(\mathsf{s}(0),\mathsf{s}^2(0)) \overset{*}{\to}_{R_3} \mathsf{s}^3(0)$. □

Inverse CTRSs generated by $\mathcal{I}nv_{\mathsf{PT}}$ are not always confluent, while every PT-TRS is confluent. For example, we have more three rewrite sequences, $\mathsf{add}^{\#}(\mathsf{s}^3(0)) \to_{R_5} \mathsf{tp}_2(0,\mathsf{s}^3(0))$, $\mathsf{add}^{\#}(\mathsf{s}^3(0)) \to_{R_5} \mathsf{tp}_2(\mathsf{s}^2(0),\mathsf{s}(0))$ and $\mathsf{add}^{\#}(\mathsf{s}^3(0)) \to_{R_5} \mathsf{tp}_2(\mathsf{s}^3(0),0)$ for $\mathsf{add}(\mathsf{s}(0),\mathsf{s}^2(0)) \overset{*}{\to}_{R_3} \mathsf{s}^3(0)$ in Example 3.4.10.

3.4.3 Transformation of CTRSs into EV-TRSs

A CTRS generated by the transformation $\mathcal{I}nv_{\mathsf{PT}}$ can be easily transformed into an unconditional one because the form of conditional rewrite rules in the CTRS is restricted as shown in Proposition 3.4.7. In this subsection, we show a transformation $\mathbb{U}_{\mathsf{PT}}$ of the generated CTRS into an EV-TRS. We also show an interesting property between input PT-TRS R and its inverse EV-TRS $\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))$ generated by $\mathcal{I}nv_{\mathsf{PT}}$ and $\mathbb{U}_{\mathsf{PT}}$, that is, if R is right-linear then $\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))$ is left-linear, and if R is left-linear then $\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))$ is right-linear.

Through this thesis, we assume that variables in $\mathcal{X}$ are ordered by some total ordering $>_{\mathcal{X}}$ that for all $x,y \in \mathcal{X}$, $x \neq y$ implies either $x >_{\mathcal{X}} y$ or $y >_{\mathcal{X}} x$. For a finite subset $X \subseteq \mathcal{X}$, $\mathcal{V}list(X)$ denotes the list of all variables in X which is determined by the ordering $>_{\mathcal{X}}$, that is, $x_1 <_{\mathcal{X}} x_2 <_{\mathcal{X}} \cdots <_{\mathcal{X}} x_n$ implies $\mathcal{V}list(\{x_1,\ldots,x_n\}) = x_1,\ldots,x_n$.

Definition 3.4.11 ($\mathbb{U}_{\mathsf{PT}}$) *Let R be a CTRS over a signature $\mathcal{F}$. Suppose that every conditional rewrite rule $l \to r \Leftarrow \bigwedge_{i=1}^{m} s_i \twoheadrightarrow t_i$ satisfies $\mathcal{V}ar(s_1, \ldots, s_m) \subseteq \mathcal{V}ar(l)$. For each rule $\rho \in R$, we prepare a fresh function symbol u^ρ which is not in $\mathcal{F}$. The transformation $\mathbb{U}_{\mathsf{PT}}$ of a conditional rewrite rule $\rho : l \to r \Leftarrow \bigwedge_{i=1}^{m} s_i \twoheadrightarrow t_i$ is defined as follows:*

$$\mathbb{U}_{\mathsf{PT}}(\rho) = \{\ l \to \mathsf{u}^\rho(\mathcal{V}list(X), s_1, \ldots, s_m),\quad \mathsf{u}^\rho(\mathcal{V}list(X), t_1, \ldots, t_m) \to r\ \}$$

where X is a set of variables appearing in both l and r, that is, $X = \mathcal{V}ar(l) \cap \mathcal{V}ar(r)$. Note that $\mathbb{U}_{\mathsf{PT}}(l \to r) = \{ l \to r \}$. $\mathbb{U}_{\mathsf{PT}}$ can be extended on CTRSs, that is, $\mathbb{U}_{\mathsf{PT}}(R) = \bigcup_{\rho \in R} \mathbb{U}_{\mathsf{PT}}(\rho)$.

$\mathcal{F}_{\mathbb{U}_{\mathsf{PT}}}$ denotes the signature determined by R, $\mathcal{F}$ and $\mathbb{U}_{\mathsf{PT}}$, defined as $\mathcal{F}_{\mathbb{U}_{\mathsf{PT}}} = \mathcal{F} \cup \{ \mathsf{u}^\rho \mid \rho \in R \}$. It is clear that $\mathbb{U}_{\mathsf{PT}}(R)$ in the above definition is an EV-TRS over the signature $\mathcal{F}_{\mathbb{U}_{\mathsf{PT}}}$. It is also clear that defined symbols of R are defined symbols of $\mathbb{U}_{\mathsf{PT}}(R)$ and constructors of R and $\mathcal{I}nv_{\mathsf{PT}}(R)$ are equivalent, that is, $\mathcal{D}_{\mathbb{U}_{\mathsf{PT}}(R)} = \mathcal{D}_R \cup \{ \mathsf{u}^\rho \mid \rho \in R \}$ and $\mathcal{C}_R = \mathcal{C}_{\mathbb{U}_{\mathsf{PT}}(R)}$.

Example 3.4.12 The following TRS is obtained by $\mathbb{U}_{\mathsf{PT}}$ from the CTRS R_5 in Example 3.4.5:

$$\begin{aligned} R_6 &= \mathbb{U}_{\mathsf{PT}}(R_5)\ (= \mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R_3))) \\ &= \{\ \mathsf{add}^{\#}(y) \to \mathsf{tp}_2(0, y), \\ &\qquad \mathsf{add}^{\#}(\mathsf{s}(z)) \to \mathsf{u}_1(\mathsf{add}^{\#}(z)),\quad \mathsf{u}_1(\mathsf{tp}_2(x, y)) \to \mathsf{tp}_2(\mathsf{s}(x), y)\ \}. \end{aligned}$$

□

Next we show that for a right-linear PT-TRS R over a signature $\mathcal{F}$, the EV-TRS $\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))$ is equivalent to the CTRS $\mathcal{I}nv_{\mathsf{PT}}(R)$ with respect to terms $f^{\#}(t)$ where t is a constructor term over $\mathcal{F}^{\#}$.

Theorem 3.4.13 *Let R be a PT-TRS over a signature $\mathcal{F}$. Let f be a defined symbol of R and $t, t_1, \ldots, t_n$ be constructor terms of R. Then, $f^{\#}(t) \to_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_n(t_1, \ldots, t_n)$ iff $f^{\#}(t) \xrightarrow{*}_{\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))} \mathsf{tp}_n(t_1, \ldots, t_n)$.*

Proof. We first prove the *only-if* part by induction on the level k of the rewrite relation of $f^{\#}(t) \xrightarrow[k]{}_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_n(t_1, \ldots, t_n)$.

We assume that $f^{\#}(t) \xrightarrow[k]{}_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_n(t_1, \ldots, t_n)$ is rewritten by a rule $f^{\#}(r') \to \mathsf{tp}_n(p, x_2, \ldots, x_n) \Leftarrow \mathcal{C}ond \in \mathcal{I}nv_{\mathsf{PT}}(R)$ where p is either a variable or a simple pattern. Then there exists a substitution θ such that $r'\theta \equiv t$, $p\theta \equiv t_1$, $x_i\theta$

$\equiv t_i$ and $\mathcal{C}ond(\theta, \xrightarrow[k-1]{*}_{\mathcal{I}nv_{\mathsf{PT}}(R)})$: $f^{\#}(t) \equiv f^{\#}(r')\theta \to_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_n(p, x_2, \ldots, x_n)\theta \equiv \mathsf{tp}_n(t_1, \ldots, t_n)$. Since $t, t_1, \ldots, t_n \in \mathcal{T}(\mathcal{C}_R, \mathcal{X})$, we can assume without loss of generality that θ is a $\mathcal{T}(\mathcal{C}_R, \mathcal{X})$-substitution.

- Consider the case of $\mathcal{C}ond = \mathtt{true}$. Then, we have the rewrite rule $f^{\#}(r') \to \mathsf{tp}_n(p, x_2, \ldots, x_n) \in \mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))$ and hence

$$f^{\#}(t) \equiv f^{\#}(r')\theta \to_{\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))} \mathsf{tp}_n(p, x_2, \ldots, x_n)\theta \equiv \mathsf{tp}_n(t_1, \ldots, t_n).$$

- Consider the remaining case of $\mathcal{C}ond = \bigwedge_{i=1}^{m} f_i^{\#}(y_i) \twoheadrightarrow \mathsf{tp}_{n_i}(x_{i,1}, \ldots, x_{i,n_i})$. In this case, we have the following rules in $\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))$:

$$f^{\#}(r') \to \mathsf{u}^{\rho}(\mathcal{V}list(X), f_1^{\#}(y_1), \ldots, f_m^{\#}(y_m)) \in \mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R)),$$

$$\mathsf{u}^{\rho}(\mathcal{V}list(X), \mathsf{tp}_{n_1}(x_{1,1}, \ldots, x_{1,n_1}), \ldots, \mathsf{tp}_{n_m}(x_{m,1}, \ldots, x_{m,n_m})) \to \mathsf{tp}_n(p, x_2, \ldots, x_n) \in \mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))$$

where $X = \mathcal{V}ar(r') \cap \mathcal{V}ar(p, x_2, \ldots, x_n)$.

It follows from $\mathcal{C}ond(\theta, \xrightarrow[k-1]{*}_{\mathcal{I}nv_{\mathsf{PT}}(R)})$ that we have the rewrite sequence $f_i^{\#}(y_i)\theta \xrightarrow[k-1]{*}_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_{n_i}(x_{i,1}, \ldots, x_{i,n_i})\theta$. Since $\mathcal{I}nv_{\mathsf{PT}}(R)$ is a constructor system and θ is a $\mathcal{T}(\mathcal{C}_R, \mathcal{X})$-substitution, we also have $f_i^{\#}(y_i)\theta \xrightarrow[k-1]{}_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_{n_i}(x_{i,1}, \ldots, x_{i,n_i})\theta$. we can assume without loss of generality that θ is a $\mathcal{T}(\mathcal{C}_R, \mathcal{X})$-substitution. Hence $y_i\theta$, $x_{i,j}\theta \in \mathcal{T}(\mathcal{C}_R, \mathcal{X})$. Then, by the induction hypothesis, we have the following:

$$f_i^{\#}(y_i)\theta \xrightarrow{*}_{\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))} \mathsf{tp}_{n_i}(x_{i,1}, \ldots, x_{i,n_i})\theta.$$

Therefore, the following rewrite sequence exists:

$$\begin{aligned} f^{\#}(t) \equiv f^{\#}(r')\theta &\to_{\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))} \mathsf{u}^{\rho}(\mathcal{V}list(X), f_1^{\#}(y_1), \ldots, f_m^{\#}(y_m))\theta \\ &\xrightarrow{*}_{\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))} \mathsf{u}^{\rho}(\mathcal{V}list(X), \mathsf{tp}_{n_1}(x_{1,1}, \ldots, x_{1,n_1}), \\ &\qquad \ldots, \mathsf{tp}_{n_m}(x_{m,1}, \ldots, x_{m,n_m}))\theta \\ &\to_{\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))} \mathsf{tp}_n(p, x_2, \ldots, x_n)\theta \equiv \mathsf{tp}_n(t_1, \ldots, t_n). \end{aligned}$$

Next we prove the *if* part by induction on the number k of steps of $f^{\#}(t) \xrightarrow{k}_{\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))} \mathsf{tp}_n(t_1, \ldots, t_n)$. Since $k > 0$ holds clearly, we can assume the following:

$$f^{\#}(t) \equiv f^{\#}(r')\theta \to_{\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))} v\theta \xrightarrow{k-1}_{\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))} \mathsf{tp}_n(t_1, \ldots, t_n)$$

where $f^{\#}(r') \to v \in \mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))$.

- Consider the case of $v \equiv \mathsf{tp}_n(p, x_2, \ldots, x_n)$. It follows from $t \in \mathcal{T}(\mathcal{C}_R, \mathcal{X})$ that $p\theta,\ x_i\theta \in \mathcal{T}(\mathcal{C}_R, \mathcal{X})$, and hence $p\theta \equiv t_1$ and $x_i\theta \equiv t_i$. From the construction of $\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))$, $f^{\#}(r') \to v \in \mathcal{I}nv_{\mathsf{PT}}(R)$ holds. Therefore, we have $f^{\#}(t) \to_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_n(t_1, \ldots, t_n)$.

- Consider the remaining case that $\mathtt{root}(v) = \mathsf{u}^{\rho}$. Now we can assume the following:

$$v \equiv \mathsf{u}^{\rho}(\mathcal{V}list(X), f_1^{\#}(y_1), \ldots, f_m^{\#}(y_m)),$$

$$\mathsf{u}^{\rho}(\mathcal{V}list(X), \mathsf{tp}_{n_1}(x_{1,1}, \ldots, x_{1,n_1}), \ldots, \mathsf{tp}_{n_m}(x_{m,1}, \ldots, x_{m,n_m})) \to \mathsf{tp}_n(p, x_2, \ldots, x_n) \in \mathbb{U}_{\mathsf{PT}}(InvPT(R)),$$

$$\rho : f^{\#}(r') \to \mathsf{tp}_n(p, x_2, \ldots, x_n) \Leftarrow \textstyle\bigwedge_{i=1}^{m} f_i^{\#}(y_i) \twoheadrightarrow \mathsf{tp}_{n_i}(x_{i,1}, \ldots, x_{i,n_i}) \in \mathcal{I}nv_{\mathsf{PT}}(R),$$

where $X = \mathcal{V}ar(r') \cap \mathcal{V}ar(p, x_2, \ldots, x_n)$. Since the above rule of u^{ρ} is the unique rule of u^{ρ}, we have the following rewrite sequence:

$$\begin{aligned} f^{\#}(t) \equiv f^{\#}(r')\theta &\to_{\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))} \mathsf{u}^{\rho}(\mathcal{V}list(X), f_1^{\#}(y_1), \ldots, f_m^{\#}(y_m))\theta \\ &\xrightarrow{k-2}_{\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))} \mathsf{u}^{\rho}(\mathcal{V}list(X), \mathsf{tp}_{n_1}(x_{1,1}, \ldots, x_{1,n_1}), \\ &\qquad \ldots, \mathsf{tp}_{n_m}(x_{m,1}, \ldots, x_{m,n_m}))\sigma \\ &\to_{\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))} \mathsf{tp}_n(p, x_2, \ldots, x_n)\sigma \equiv \mathsf{tp}_n(t_1, \ldots, t_n). \end{aligned}$$

Since $t, t_1, \ldots, t_n \in \mathcal{T}(\mathcal{C}_R, \mathcal{X})$, we can assume without loss of generality that θ, σ are $\mathcal{T}(\mathcal{C}_R, \mathcal{X})$-substitution, and hence $y_i\theta \in \mathcal{T}(\mathcal{C}_R, \mathcal{X})$ and $x_{i,j}\sigma \in \mathcal{T}(\mathcal{C}_R, \mathcal{X})$. Then we have $f_i^{\#}(y_i)\theta \xrightarrow{k_i}_{\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))} \mathsf{tp}_{n_i}(x_{i,1}, \ldots, x_{i,n_i})\sigma$. By the induction hypothesis, we obtain the following sequence:

$$f_i^{\#}(y_i)\theta \to_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_{n_i}(x_{i,1}, \ldots, x_{i,n_i})\sigma.$$

Let $\delta = \sigma|_{\mathcal{V}ar(p, x_2, \ldots, x_n)} \cup \bigcup_{i=1}^{m} \{y_i \mapsto y_i\theta\}$. From Proposition 3.4.7, δ is a substitution. Moreover, $f_i^{\#}(y_i)\delta \to_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_{n_i}(x_{i,1}, \ldots, x_{i,n_i})\delta$ holds. Therefore, the rule ρ can be applied to $f^{\#}(t)$ as follows:

$$f^{\#}(t) \equiv f^{\#}(r')\delta \to_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_n(p, x_2, \ldots, x_n)\delta \equiv \mathsf{tp}_n(t_1, \ldots, t_n).$$

□

Example 3.4.14 For the rewrite sequence $\mathsf{add}^{\#}(\mathsf{s}^3(0)) \to_{R_5} \mathsf{tp}_2(\mathsf{s}(0), \mathsf{s}^2(0))$ in Example 3.4.10, we have the following rewrite sequence of R_6:

$$\mathsf{add}^{\#}(\mathsf{s}^3(0)) \to_{R_6} \mathsf{u}_1(\mathsf{add}(\mathsf{s}^2(0))) \to_{R_6} \mathsf{u}_1(\mathsf{tp}_2(0, \mathsf{s}^2(0))) \to_{R_6} \mathsf{tp}_2(\mathsf{s}(0), \mathsf{s}^2(0)).$$

□

Now we show the following interesting relationship between a PT-TRS R and EV-TRS $\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))$.

Proposition 3.4.15 *Let R be a PT-TRS.*

(a) If R is right-linear then $\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))$ is left-linear.

(b) If R is left-linear then $\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))$ is right-linear.

Proof. Consider the following rule

$$\rho_1 : f(p, x_2, \ldots, x_n) \to C[\![f_1(x_{1,1}, \ldots, x_{1,n_1}), \ldots, f_m(x_{m,1}, \ldots, x_{m,n_m})]\!] \in R.$$

From the definitions of $\mathcal{I}nv_{\mathsf{PT}}$ and $\mathbb{U}_{\mathsf{PT}}$, we obtain the following rules:

$$\begin{aligned}
&\rho_2 : f^{\#}(C[y_1, \ldots, y_m]) \to \mathsf{tp}_n(p, x_2, \ldots, x_n) \\
&\qquad\qquad \textstyle\bigwedge_{i=1}^{m} f_i^{\#}(y_i) \twoheadrightarrow \mathsf{tp}_{n_i}(x_{i,1}, \ldots, x_{i,n_i}) \in \mathcal{I}nv_{\mathsf{PT}}(R), \\
&\rho_3 : f^{\#}(C[y_1, \ldots, y_m]) \to \mathsf{u}^{\rho}(\mathcal{V}list(X), f_1^{\#}(y_1), \ldots, f_m^{\#}(y_m)) \in \mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R)), \\
&\rho_4 : \mathsf{u}^{\rho}(\mathcal{V}list(X), \mathsf{tp}_{n_1}(x_{1,1}, \ldots, x_{1,n_1}), \ldots, \mathsf{tp}_{n_m}(x_{m,1}, \ldots, x_{m,n_m})) \\
&\qquad\qquad \to \mathsf{tp}_n(p, x_2, \ldots, x_n) \in\in \mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(R))
\end{aligned}$$

where $X = \mathcal{V}ar(C[y_1, \ldots, y_m]) \cap \mathcal{V}ar(p, x_2, \ldots, x_n)$.

We first show that if ρ_1 is right-linear then ρ_3 and ρ_4 are left-linear. It follows from Proposition 3.4.7 that $C[y_1, \ldots, y_m]$ is linear, $\mathcal{V}ar(C[y_1, \ldots, y_m]) \cap \{x_{i,1}, \ldots, x_{i,n_i}\} = \emptyset$ and $x_{i,j} \not\equiv x_{i',j'}$ for $i \neq j$ or $i' \neq j'$. Then, $f^{\#}(C[y_1, \ldots, y_m])$ and $\mathsf{u}^{\rho}(\mathcal{V}list(X), \mathsf{tp}_{n_1}(x_{1,1}, \ldots, x_{1,n_1}), \ldots, \mathsf{tp}_{n_m}(x_{m,1}, \ldots, x_{m,n_m}))$ are linear, and hence ρ_3, ρ_4 are left-linear.

Next we show that if ρ_1 is left-linear then ρ_3, ρ_4 are right-linear. It is clear that $\mathsf{tp}_n(p, x_2, \ldots, x_n)$ is linear. From the definition of $\mathcal{I}nv_{\mathsf{PT}}$, $y_1, \ldots, y_m$ are disjoint and $y_i \notin \mathcal{V}ar(C, p, x_2, \ldots, x_n)$, and hence $y_i \notin X$. Then, it is clear that $\mathsf{u}^{\rho}(\mathcal{V}list(X), f_1^{\#}(y_1), \ldots, f_m^{\#}(y_m))$ is linear. Therefore, ρ_3, ρ_4 are right-linear. □

3.5 Idea of Extension for Constructor TRSs

In this section, we explain an idea to extend $\mathcal{I}nv_{\mathsf{PT}}$ for constructor TRSs.

One of the largest differences between the syntax of PT-TRSs and constructor TRSs is the form of arguments of defined symbols in the right-hand sides of rewrite rules; For a rewrite rule $l \to C[\![f_1(t_{1,1}, \ldots, t_{1,n_1}), \ldots, f_m(t_{m,1}, \ldots, t_{m,n_m})]\!]$, all $t_{i,j}$s are variables for PT-TRSs, and all $t_{i,j}$s are terms for constructor TRSs.

In **Step 3** and **Step 4** of the basic idea introduced in Section 3.3, we have transformed the right-hand side r of a rewrite rule $f(t_1, \ldots, t_n) \to r$ in an input system into a constructor term r' as the argument of $f^\#$ in the left-hand side of a corresponding rule of $f^\#$, and conditions as the conditional part of the rule of $f^\#$.

If the same transformation is applied to a constructor TRS, defined symbols of the input TRS may still remain in the conditional part after the transformation. For example, consider the rule $f(x, y) \to c(f(g(a)), g(x))$ where f, g are defined symbols and a, c are constructors. To apply **Step 1–4** to this rule, we obtain the following rule:

$$f^\#(c(x_1, x_2)) \to \mathsf{tp}_2(x, y) \Leftarrow f^\#(x_2) \twoheadrightarrow \mathsf{tp}_1(g(a)) \wedge g^\#(x) \twoheadrightarrow \mathsf{tp}_1(x).$$

The defined symbol g still remains in the condition $f^\#(x_2) \twoheadrightarrow \mathsf{tp}_1(g(a))$.

To eliminate g, we again replace the term $g(a)$ with a fresh variable z. Then, the condition $f^\#(x_2) \twoheadrightarrow \mathsf{tp}_1(g(a))$ is transformed to $f^\#(x_2) \twoheadrightarrow \mathsf{tp}_1(z)$ and the new condition $g^\#(z) \twoheadrightarrow \mathsf{tp}_1(a)$ is generated. Adding the new condition into the conditional part, we can obtain the following conditional rewrite rule:

$$\begin{aligned} f^\#(c(x_1, x_2)) &\to \mathsf{tp}_2(x, y) \\ &\Leftarrow f^\#(x_2) \twoheadrightarrow \mathsf{tp}_1(z) \wedge g^\#(x) \twoheadrightarrow \mathsf{tp}_1(x) \wedge g^\#(z) \twoheadrightarrow \mathsf{tp}_1(a). \end{aligned}$$

Finally, there is no defined symbol f, g in the transformed rule.

These transformation is considered as the inductive application of **Step 3** and **Step 4**. Since **Step 3** and **Step 4** are implemented as the transformation $\mathscr{T}_{\mathsf{PT}}$, we can easily construct such an extended compiler by inductively defining $\mathscr{T}_{\mathsf{PT}}$, especially the case (c) in Definition 3.4.1; (c) $\mathscr{T}_{\mathsf{PT}}(f(t_1, \ldots, t_n)) = \langle\!| y ; f^\#(y) \twoheadrightarrow \mathsf{tp}_n(u_1, \ldots, u_n) \wedge \mathit{Cond}_1 \wedge \mathit{Cond}_n |\!\rangle$ where y is a fresh variable and $\mathscr{T}_{\mathsf{PT}}(t_i) = \langle\!| u_i ; \mathit{Cond}_i |\!\rangle$.

3.6 Inverse Compiler for Constructor TRSs

In this section, we actually extend the definition of $\mathcal{I}nv_{\mathsf{PT}}$ to that for constructor TRSs. In Subsection 3.6.1 we present an inverse compiler for constructor TRSs, and we show the correctness of it in Subsection 3.6.3. In Subsection 3.6.2, we show the relationship between the syntactic properties of a target constructor TRS and the inverse CTRS generated by the inverse compiler proposed in Subsection 3.6.1. In Subsection 3.6.4 we transform the generated CTRS into

an equivalent EV-TRS. We discuss in Subsection 3.6.5 the relationship between syntactic properties of an input constructor TRS and its inverse EV-TRS.

3.6.1 Generation of Inverse CTRSs

We extend the procedure $\mathscr{T}_{\mathsf{PT}}$ by inductively defining the case (c) in Definition 3.4.1.

Definition 3.6.1 ($\mathscr{T}_{\mathsf{full}}$) *Let $\mathcal{F}$ be a signature divided into the constructor set $\mathcal{C}$ and the defined-symbol set $\mathcal{D}$: $\mathcal{F} = \mathcal{C} \uplus \mathcal{D}$. The procedure $\mathscr{T}_{\mathsf{full}}$ inputted a term is inductively defined as follows:*

(a) $\mathscr{T}_{\mathsf{full}}(x) = \langle\!|\, x \,;\, \mathtt{true} \,|\!\rangle$ *where x is a variable,*

(b) $\mathscr{T}_{\mathsf{full}}(c(t_1, \ldots, t_n)) = \langle\!|\, c(u_1, \ldots, u_n) \,;\, \mathcal{C}ond_1 \wedge \cdots \wedge \mathcal{C}ond_n \,|\!\rangle$
where $c \in \mathcal{C}$ and $\mathscr{T}_{\mathsf{full}}(t_i) = \langle\!|\, u_i \,;\, \mathcal{C}ond_i \,|\!\rangle$, and

(c) $\mathscr{T}_{\mathsf{full}}(f(t_1, \ldots, t_n)) = \langle\!|\, y \,;\, f^{\#}(y) \twoheadrightarrow \mathsf{tp}_n(u_1, \ldots, u_n) \wedge \mathcal{C}ond_1 \wedge \mathcal{C}ond_n \,|\!\rangle$
where $f \in \mathcal{D}$, y is a fresh variable and $\mathscr{T}_{\mathsf{full}}(t_i) = \langle\!|\, u_i \,;\, \mathcal{C}ond_i \,|\!\rangle$.

The word 'fresh' in (c) means that in (c), $y \notin \mathcal{V}ar(t_i, u_i, \mathcal{C}ond_i)$, and in each of (b) and (c), variables introduced in $\mathscr{T}_{\mathsf{full}}(t_i) = \langle\!|\, u_i \,;\, \mathcal{C}ond_i \,|\!\rangle$ and $\mathscr{T}_{\mathsf{full}}(t_j) = \langle\!|\, u_j \,;\, \mathcal{C}ond_j \,|\!\rangle$ are disjoint, that is, $\mathcal{V}ar(u_i, \mathcal{C}ond_i) \cap \mathcal{V}ar(t_j, u_j, \mathcal{C}ond_j) = \emptyset$ for each i and j with $i \neq j$.

For any input term, the above procedure $\mathscr{T}_{\mathsf{full}}$ terminates and returns a pair of a term and a condition since the input term is finite. The above definition of $\mathscr{T}_{\mathsf{full}}$ leads the following proposition.

Proposition 3.6.2 *Let $\mathcal{F}$ be a signature divided into the constructor set $\mathcal{C}$ and the defined-symbol set $\mathcal{D}$: $\mathcal{F} = \mathcal{C} \uplus \mathcal{D}$. Then, $\mathscr{T}_{\mathsf{full}}(t) = \langle\!|\, u \,;\, \mathcal{C}ond \,|\!\rangle$ implies the following:*

(a) if $t \equiv C[\![g_1(t_{1,1}, \ldots, t_{1,n_1}), \ldots, g_m(t_{m,1}, \ldots, t_{m,n_m})]\!]$, then

- *$u \equiv C[y_1, \ldots, y_m]$, and*
- *$\mathcal{C}ond = \bigwedge_{i=1}^{m} \left(g_i^{\#}(y_i) \twoheadrightarrow \mathsf{tp}_{n_i}(u_{i,1}, \ldots, u_{i,n_i}) \wedge \bigwedge_{j=1}^{n_i} \mathcal{C}ond_{i,j} \right)$*

for some variables $y_1, \ldots, y_m$ such that $y_i \notin \mathcal{V}ar(t_{i,j}, u_{i,j}, \mathcal{C}ond_{i,j})$, where $\mathscr{T}_{\mathsf{full}}(t_{i,j}) = \langle\!|\, u_{i,j} \,;\, \mathcal{C}ond_{i,j} \,|\!\rangle$ and $\mathcal{V}ar(u_{i,j}, \mathcal{C}ond_{i,j}) \cap \mathcal{V}ar(t_{i',j'}, u_{i',j'}, \mathcal{C}ond_{i',j'}) = \emptyset$ for each i and j such that $i \neq j$ or $i' \neq j'$,

(b) u is a constructor term, and

(c) $Cond$ is in the forms of

$$\mathtt{true} \wedge f_1^{\#}(y_1) \twoheadrightarrow \mathsf{tp}_{n_1}(x_{1,1}, \ldots, x_{1,n_1}) \wedge \cdots \wedge f_k^{\#}(y_k) \twoheadrightarrow \mathsf{tp}_{n_k}(x_{k,1}, \ldots, x_{k,n_k})$$

where f_i is an n_i-ary defined symbol with respect to R.

Proof. The claim (a) follows from the definition of $\mathscr{T}_{\mathsf{full}}$. The claims (b) and (c) follow from the claim (a). □

Example 3.6.3 Consider the following constructor TRS over the signature { s, 0, add, mul }, which computes addition and multiplication of natural numbers:

$$\begin{aligned} R_7 = \{\ & \mathsf{add}(0, y) \to y, \quad \mathsf{add}(\mathsf{s}(x), y) \to \mathsf{s}(\mathsf{add}(x, y)), \\ & \mathsf{mul}(0, y) \to 0, \quad \mathsf{mul}(x, 0) \to 0, \\ & \mathsf{mul}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{s}(\mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y)) \quad \}. \end{aligned}$$

$\mathscr{T}_{\mathsf{full}}$ generates, from the right-hand side of the fifth rule above, the following pair of the term and the condition:

$$\begin{aligned} &\mathscr{T}_{\mathsf{full}}\,(\mathsf{s}(\mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y))) \\ &\qquad = \langle\!|\, \mathsf{s}(z)\ ;\ \mathsf{add}^{\#}(z) \twoheadrightarrow \mathsf{tp}_2(w, y) \wedge \mathsf{mul}^{\#}(w) \twoheadrightarrow \mathsf{tp}_2(x, \mathsf{s}(y))\ |\!\rangle. \end{aligned}$$

□

Using the procedure $\mathscr{T}_{\mathsf{full}}$, we extend $\mathcal{I}nv_{\mathsf{PT}}$ to the transformation $\mathcal{I}nv_{\mathsf{full}}$ for constructor TRSs as follows.

Definition 3.6.4 ($\mathcal{I}nv_{\mathsf{full}}$) *Let R be a constructor TRS over a signature $\mathcal{F}$. For a rewrite rule $\rho : f(w_1, \ldots, w_n) \to r \in R$, its corresponding conditional rewrite rule $\mathcal{I}nv\mathcal{R}ule_{\mathsf{full}}(\rho)$ of $f^{\#}$ is defined as follows:*

$$\mathcal{I}nv\mathcal{R}ule_{\mathsf{full}}\,(f(w_1, \ldots, w_n) \to r) = f^{\#}(r') \to \mathsf{tp}_n(w_1, \ldots, w_n) \Leftarrow \mathcal{C}ond$$

where $\mathscr{T}_{\mathsf{full}}(r) = \langle\!|\, r'\ ;\ \mathcal{C}ond\ |\!\rangle$ and $(\mathcal{V}ar(r', \mathcal{C}ond) \setminus \mathcal{V}ar(r)) \cap \mathcal{V}ar(w_1, \ldots, w_n) = \emptyset$.

The transformation $\mathcal{I}nv_{\mathsf{full}}$ of a constructor TRS is defined as follows:

$$\begin{aligned} \mathcal{I}nv_{\mathsf{full}}(R) = \{\, & \mathcal{I}nv\mathcal{R}ule_{\mathsf{full}}(l \to r) \mid l \to r \in R \,\} \\ & \cup \{\, f^{\#}(f(x_1, \ldots, x_n)) \to \mathsf{tp}_n(x_1, \ldots, x_n) \mid f \in \mathcal{D}_R \,\}. \end{aligned}$$

The rewrite rule $f^{\#}(f(x_1, \ldots, x_n)) \to \mathsf{tp}_n(x_1, \ldots, x_n)$ in the above definition is called the inverse-property rule *of the defined symbol f.*

It is clear that $\mathcal{I}nv\mathcal{R}ule_{\mathsf{full}}(\rho)$ is a conditional rewrite rule over the signature $\mathcal{F}^{\#}$ and hence $\mathcal{I}nv_{\mathsf{full}}(R)$ is a CTRS over $\mathcal{F}^{\#}$. In Subsection 3.6.3, we show the correctness of $\mathcal{I}nv_{\mathsf{full}}$, that is, that $\mathcal{I}nv_{\mathsf{full}}(R)$ is an inverse CTRS of R. We also show the necessity of the inverse-property rules in the last of Subsection 3.6.3.

Example 3.6.5 The following CTRS is obtained by $\mathcal{I}nv_{\mathsf{full}}$ from R_7 in Example 3.6.3:

$$
\begin{aligned}
R_8 &= \mathcal{I}nv_{\mathsf{full}}(R_7)\\
&= \{\ \mathsf{add}^{\#}(y) \to \mathsf{tp}_2(0,y),\\
&\quad\ \mathsf{add}^{\#}(\mathsf{s}(z)) \to \mathsf{tp}_2(\mathsf{s}(x),y) \Leftarrow \mathsf{add}^{\#}(z) \twoheadrightarrow \mathsf{tp}_2(x,y),\\
&\quad\ \mathsf{add}^{\#}(\mathsf{add}(x,y)) \to \mathsf{tp}_2(x,y),\\
&\quad\ \mathsf{mul}^{\#}(0) \to \mathsf{tp}_2(0,y),\\
&\quad\ \mathsf{mul}^{\#}(0) \to \mathsf{tp}_2(x,0),\\
&\quad\ \mathsf{mul}^{\#}(\mathsf{s}(z)) \to \mathsf{tp}_2(\mathsf{s}(x),\mathsf{s}(y))\\
&\qquad\qquad \Leftarrow \mathsf{add}^{\#}(z) \twoheadrightarrow \mathsf{tp}_2(w,y) \wedge \mathsf{mul}^{\#}(w) \twoheadrightarrow \mathsf{tp}_2(x,\mathsf{s}(y)),\\
&\quad\ \mathsf{mul}^{\#}(\mathsf{mul}(x,y)) \to \mathsf{tp}_2(x,y) \qquad\qquad \}.
\end{aligned}
$$

□

The following proposition shows relationships between the signatures of an input R and the CTRS $\mathcal{I}nv_{\mathsf{full}}(R)$, and between constructor terms of R and $\mathcal{I}nv_{\mathsf{full}}(R)$.

Proposition 3.6.6 *Let R be a constructor TRS over a signature $\mathcal{F}$.*

(a) Every defined symbol of $\mathcal{I}nv_{\mathsf{full}}(R)$ consists of a defined symbol of R and the superscription $\#$, that is, $\mathcal{D}_{\mathcal{I}nv_{\mathsf{full}}(R)} = \{\ f^{\#} \mid f \in \mathcal{D}_R\ \}$.

(b) Every function symbol in $\mathcal{F}$ is a constructor of $\mathcal{I}nv_{\mathsf{full}}(R)$, that is, $\mathcal{F} \subseteq \mathcal{C}_{\mathcal{I}nv_{\mathsf{full}}(R)}$.

(c) Every term over $\mathcal{F}$ is a constructor term of $\mathcal{I}nv_{\mathsf{full}}(R)$, that is, $\mathcal{T}(\mathcal{F},\mathcal{X}) \subseteq \mathcal{T}(\mathcal{C}_{\mathcal{I}nv_{\mathsf{full}}(R)},\mathcal{X})$.

Proof. From Definition 3.6.4, every rewrite rule $f(w_1,\ldots,w_n) \to r \in R$ is transformed to a conditional rewrite rule $f^{\#}(r') \to \mathsf{tp}_n(w_1,\ldots,w_n) \Leftarrow \mathit{Cond} \in \mathcal{I}nv_{\mathsf{full}}(R)$ where $\mathscr{T}_{\mathsf{full}}(r) = \langle\!|\ r'\ ;\ \mathit{Cond}\ |\!\rangle$. Then, it follows from the definition of defined symbols that $\mathcal{D}_{\mathcal{I}nv_{\mathsf{full}}(R)} = \{\ f^{\#} \mid f \in \mathcal{D}_R\ \}$. Hence, the claim (a) holds. It is clear that the claim (b) follows from (a), and the claim (c) follows from (b). □

3.6.2 Syntactic Properties of Inverse CTRSs

In this subsection, we analyze syntactic properties of CTRSs generated by $\mathcal{I}nv_{\mathsf{full}}$, some of which are preserved by $\mathcal{I}nv_{\mathsf{full}}$ from those of input constructor TRSs. The discussion here is the first half of the analysis on the properties of EV-TRSs generated by our inverse compiler, which we appeal in this thesis.

We first define some syntactic properties of CTRSs. Let R be a deterministic 4-CTRS over a signature $\mathcal{F}$ and a conditional rewrite rule $\rho : l \to r \Leftarrow \mathcal{C}ond$ over $\mathcal{F}$ where $\mathcal{C}ond = s_1 \twoheadrightarrow t_1 \wedge \cdots \wedge s_n \twoheadrightarrow t_n$. The rule ρ is called a *strict constructor rule* if $t_1, \ldots, t_n$ and all proper subterms of the left-hand side l are constructor terms of R, that is, $\{t_1, \ldots, t_n\} \cup \{\, u \mid u \lhd l \,\} \subseteq \mathcal{T}(\mathcal{C}_R, \mathcal{X})$. R is called a *strict constructor system* if every rule in R is a strict constructor rule. Note that strict constructor CTRSs are constructor CTRSs. The rule ρ is called *strictly non-erasing* if all variables in the left-hand side l appear in either the right-hand side r or the conditional part $\mathcal{C}ond$, and all variables in t_i appear in r and the later conditions $s_{i+1} \twoheadrightarrow t_{i+1} \wedge \cdots \wedge s_n \twoheadrightarrow t_n$, that is,

- $\mathcal{V}ar(l) \subseteq \mathcal{V}ar(r, \mathcal{C}ond)$, and
- $\mathcal{V}ar(t_i) \subseteq \mathcal{V}ar(r) \cup \bigcup_{j=i+1}^{n} \mathcal{V}ar(s_j, t_j)$ for $1 \leq i \leq n$.

R is called *strictly non-erasing* if every rule in R is strictly non-erasing. The rule ρ is *strictly left-linear* if l and $t_1, \ldots, t_n$ are linear and any variable in t_i does not appear in l or $t_1, \ldots, t_{i-1}$, that is, $\mathcal{V}ar(t_i) \cap \mathcal{V}ar(l, t_1, \ldots, t_{i-1}) = \emptyset$ for $1 \leq i \leq n$. The rule ρ is called *strictly right-linear* if r and $s_1, \ldots, s_n$ are linear and any variable in s_i does not appear in r, t_i or $s_{i+1}, t_{i+1}, \ldots, s_n, t_n$, that is, $\mathcal{V}ar(s_i) \cap (\mathcal{V}ar(r, t_i) \cup \bigcup_{j=i+1}^{n} \mathcal{V}ar(s_j, t_j))$ for $1 \leq i \leq n$. R is called *strictly left-linear* if every rule in R is strictly left-linear, and called *strictly right-linear* if every rule in R is strictly right-linear.

Let $\mathcal{C}ond$ be a conditional part in form of $\mathtt{true} \wedge s_1 \twoheadrightarrow v_1 \wedge \cdots \wedge s_m \twoheadrightarrow v_m$. For a substitution θ, $\theta(\mathcal{C}ond)$ denotes $\mathtt{true} \wedge s_1\theta \twoheadrightarrow v_1\theta \wedge \cdots \wedge s_m\theta \twoheadrightarrow v_m\theta$.

As a preparation for the analysis, we show the properties on variable appearance in $\mathscr{T}_{\mathsf{full}}(t) = \langle\!\langle\, u \,;\, \mathcal{C}ond \,\rangle\!\rangle$.

Proposition 3.6.7 *Let $\mathcal{F}$ be a signature divided into the constructor set $\mathcal{C}$ and the defined-symbol set $\mathcal{D}$: $\mathcal{F} = \mathcal{C} \uplus \mathcal{D}$. Let t be a term over $\mathcal{F}$ and $\mathscr{T}_{\mathsf{full}}(t) = \langle\!\langle\, u \,;\, \mathcal{C}ond \,\rangle\!\rangle$. Then, all of the following hold:*

(a) each variable in either t, u or $\mathcal{C}ond$ is appearing in the others: $\mathcal{V}ar(t) \subseteq \mathcal{V}ar(u, \mathcal{C}ond)$, $\mathcal{V}ar(u) \subseteq \mathcal{V}ar(t, \mathcal{C}ond)$ and $\mathcal{V}ar(\mathcal{C}ond) \subseteq \mathcal{V}ar(t, u)$,

(b) $\mathcal{V}ar(u, \mathcal{C}ond) \setminus \mathcal{V}ar(t) = \mathcal{V}ar(\mathcal{C}ond) \setminus \mathcal{V}ar(t) = \mathcal{V}ar(u) \setminus \mathcal{V}ar(t)$, *and*

In addition, suppose that $t \equiv g(t_1, \ldots, t_n)$, $g \in \mathcal{F}$ *and* $\mathscr{T}_{\mathsf{full}}(t_i) = \langle\!\langle\, u_i \,;\, \mathcal{C}ond_i \,\rangle\!\rangle$ *such that for* i *and* j *with* $i \neq j$, $(\mathcal{V}ar(u_i, \mathcal{C}ond_i) \setminus \mathcal{V}ar(t_i)) \cap \mathcal{V}ar(t_j, u_j, \mathcal{C}ond_j) = \emptyset$. *Then,*

(c) $\mathcal{V}ar(u_i, \mathcal{C}ond_i) \setminus \mathcal{V}ar(t_i) = \mathcal{V}ar(u_i, \mathcal{C}ond_i) \setminus \mathcal{V}ar(t)$.

Proof. These claims clearly follow from the definition of $\mathscr{T}_{\mathsf{full}}$. □

Next, for $\mathscr{T}_{\mathsf{full}}(t) = \langle\!\langle\, u \,;\, \mathcal{C}ond \,\rangle\!\rangle$, we clarify the relationship between t, u and $\mathcal{C}ond$.

Lemma 3.6.8 *Let* $\mathcal{F}$ *be a signature divided into the constructor set* $\mathcal{C}$ *and the defined-symbol set* $\mathcal{D}$: $\mathcal{F} = \mathcal{C} \uplus \mathcal{D}$. *Suppose that*

- t *is a term over* $\mathcal{F}$,
- $\mathscr{T}_{\mathsf{full}}(t) = \langle\!\langle\, u \,;\, \mathcal{C}ond \,\rangle\!\rangle$,
- $\mathcal{C}ond = \mathtt{true} \wedge \mathcal{C}ond_1 \wedge \cdots \mathcal{C}ond_m$, *and*
- $\mathcal{C}ond_i = s_i \twoheadrightarrow v_i$.

Then, all of the following hold:

(a) $\mathcal{V}ar(s_i) \subseteq \mathcal{V}ar(u, v_1, \ldots, v_{i-1})$,

(b) u *is a constructor term over* $\mathcal{C}$, *and* $v_1, \ldots, v_m$ *are constructor terms over* $\mathcal{C} \cup \{\, \mathtt{tp}_i \mid 0 \leq i \leq \mathtt{max}_{f \in \mathcal{D}}(\mathtt{arity}(f)) \,\}$,

(c) $\mathcal{V}ar(v_i) \subseteq \mathcal{V}ar(t, \mathcal{C}ond_{i+1}, \ldots, \mathcal{C}ond_m)$,

(d) $s_1, \ldots, s_m$ *are linear,*

(e) $\mathcal{V}ar(s_i) \cap \mathcal{V}ar(t, v_i, \mathcal{C}ond_{i+1}, \ldots, \mathcal{C}ond_m) = \emptyset$, *and*

(f) if t *is linear, then* u *and* $v_1, \ldots, v_m$ *are also linear and for every* i, $\mathcal{V}ar(v_i) \cap \mathcal{V}ar(u, v_1, \ldots, v_{i-1}) = \emptyset$.

Proof. (Sketch.) The claim (b) obviously holds from Proposition 3.6.2. The other claims can be easily proved by induction on the structure of term t, by using Proposition 3.6.7. □

CTRSs generated by $\mathcal{I}nv_{\mathsf{full}}$ have the following syntactic properties.

Proposition 3.6.9 *Let R be a constructor TRS over a signature $\mathcal{F}$. Then,*

(a) $\mathcal{I}nv_{\mathsf{full}}(R)$ is a strictly non-erasing and strict constructor 4-CTRS over the signature $\mathcal{F}^{\#}$.

Moreover, all of the following hold:

(b) if R is non-erasing, then $\mathcal{I}nv_{\mathsf{full}}(R)$ is 3-CTRS,

(c) if R is left-linear, then $\mathcal{I}nv_{\mathsf{full}}(R)$ is strictly right-linear, and

(d) if R is right-linear, then $\mathcal{I}nv_{\mathsf{full}}(R)$ is strictly left-linear.

Proof. (Sketch.) To prove this lemma, it is sufficient to show that all of the following hold for a rewrite rule $\rho : l \to r \in R$:

(1) $\mathcal{I}nv\mathcal{R}ule_{\mathsf{full}}(\rho)$ is deterministic,

(2) $\mathcal{I}nv\mathcal{R}ule_{\mathsf{full}}(\rho)$ is a strictly non-erasing and strict constructor system,

(3) if ρ is non-erasing, then every extra variable in the right-hand side of $\mathcal{I}nv\mathcal{R}ule_{\mathsf{full}}(\rho)$ is always appearing in the conditional part of $\mathcal{I}nv\mathcal{R}ule_{\mathsf{full}}(\rho)$,

(4) if ρ is left-linear, then $\mathcal{I}nv\mathcal{R}ule_{\mathsf{full}}(\rho)$ is strictly right-linear, and

(5) if ρ is right-linear, then $\mathcal{I}nv\mathcal{R}ule_{\mathsf{full}}(\rho)$ is strictly left-linear.

The above claims can be easily proved by using Lemma 3.6.8. □

3.6.3 Correctness of Generation

In this subsection, we prove the correctness of the transformation $\mathcal{I}nv_{\mathsf{full}}$, that is, that for a convergent constructor TRS R, $\mathcal{I}nv_{\mathsf{full}}(R)$ is an inverse CTRS of R.

We first show that $\mathscr{T}_{\mathsf{full}}(t) = \langle\!\langle\, u \,;\, \mathcal{C}ond \,\rangle\!\rangle$ is stable for $\mathcal{T}(\mathcal{C}, \mathcal{X})$-substitutions.

Proposition 3.6.10 *Let $\mathcal{F}$ be a signature divided into the constructor set $\mathcal{C}$ and the defined-symbol set $\mathcal{D}$: $\mathcal{F} = \mathcal{C} \uplus \mathcal{D}$. Let t be a term over $\mathcal{F}$ and $\mathscr{T}_{\mathsf{full}}(t) = \langle\!\langle\, u \,;\, \mathcal{C}ond \,\rangle\!\rangle$. Then, $\mathscr{T}_{\mathsf{full}}(t\theta) = \langle\!\langle\, u\theta \,;\, \theta(\mathcal{C}ond) \,\rangle\!\rangle$ for each $\mathcal{T}(\mathcal{C}, \mathcal{X})$-substitution θ such that $(\mathcal{D}om(\theta) \cup \mathcal{VR}an(\theta)) \cap (\mathcal{V}ar(u, \mathcal{C}ond) \setminus \mathcal{V}ar(t)) = \emptyset$.*

Proof. The procedure $\mathscr{T}_{\mathsf{full}}$ is defined inductively on the structure of terms and $\mathscr{T}_{\mathsf{full}}$ does not transform any constructor term, that is, $\mathscr{T}_{\mathsf{full}}(s) = \langle\!\langle\, s \,;\, \mathtt{true} \,\rangle\!\rangle$ for every constructor term s, (see Definition 3.6.1). On the other hand, variables in the domain and range of θ satisfies the condition on variables during the procedure $\mathscr{T}_{\mathsf{full}}$. Therefore, the proposition holds. □

Next, we prepare the following lemma on rewrite sequences of constructor CTRSs.

Lemma 3.6.11 *Let R be a constructor CTRS over a signature $\mathcal{F}$, s be a term over $\mathcal{F}$ and t be a normal form with respect to R. In addition, let δ be a $\mathcal{T}(\mathcal{C}_R, \mathcal{X})$-substitution, σ be an $NF^{\rightarrow_R}(\mathcal{F}, \mathcal{X})$-substitution such that $\mathcal{V}ar(s) \cap \mathcal{D}om(\sigma) = \emptyset$ and for every $x \in \mathcal{D}om(\sigma)$, $\mathtt{root}(x\sigma) \in \mathcal{D}_R$. Then, $s\delta\sigma \xrightarrow{*}_R t$ implies $s\delta \xrightarrow{*}_R u$ and $t \equiv u\sigma$ for some term u (similarly for $\xrightarrow[\mathrm{in}]{}_R$).*

Proof. This lemma follows from the syntactic properties and rewrite relations on constructor systems. □

Next we show that every substitution can be decomposed into two subsitutitons one of which substitute constructor terms, the other of which substitute terms headed by defined symbols.

Proposition 3.6.12 *Let $\mathcal{F}$ be a signature divided into the constructor set $\mathcal{C}$ and the defined-symbol set $\mathcal{D}$: $\mathcal{F} = \mathcal{C} \uplus \mathcal{D}$. Every substitution θ can be decomposed into the substitutions δ and σ such that*

- $\theta = \delta\sigma$,
- $\mathcal{D}om(\theta) = \mathcal{D}om(\delta)$,
- $(\mathcal{D}om(\theta) \cup \mathcal{VR}an(\theta)) \cap \mathcal{D}om(\sigma) = \emptyset$, *and*
- *for all $x \in \mathcal{D}om(\sigma)$, $\mathtt{root}(x\sigma) \in \mathcal{D}$.*

Proof. It is clear that every term s over $\mathcal{F}$ can be divided into a constructor term t over $\mathcal{F}$ and a $\mathcal{T}(\mathcal{F}, \mathcal{X})$-substitution σ' such that $\mathcal{VR}an(\sigma') \cap \mathcal{V}ar(s) = \emptyset$ and $\mathtt{root}(u) \in \mathcal{D}$ for every $u \in \mathcal{R}an(\sigma')$. This fact implies the claim of this proposition. □

The following proposition holds between substitutions and conditional parts.

Proposition 3.6.13 *Let R be a CTRS, $l \to r \Leftarrow Cond \in R$. Let σ, θ be substitutions. Then, the both of the following hold (similarly for $\xrightarrow{n}_R$ and $\xrightarrow{*}_R$):*

(a) $Cond(\sigma, \to_R)$ implies $Cond(\sigma\theta, \to_R)$, and

(b) $(\sigma(Cond))(\theta, \to_R)$ iff $Cond(\sigma\theta, \to_R)$.

Proof. Thiese are obvious because rewrite relation is closed under substitutions. □

In the following lemma, we show a condition that for a conditional rewrite rule $l \to r \Leftarrow Cond \in R$, $Cond(\sigma, \xrightarrow{*}_R)$ implies $Cond(\sigma, \to_R)$.

Lemma 3.6.14 *Let R be a strictly non-erasing and strict constructor CTRS over a signature $\mathcal{F}$. Suppose that every rule $l \to r \Leftarrow s_1 \twoheadrightarrow t_1 \wedge \cdots \wedge s_m \twoheadrightarrow t_m \in R$ satisfies all of the followings:*

(1) r and $t_1, \ldots, t_m$ are non-variable constructor terms of R, and

(2) $\mathtt{root}(s_i) \in \mathcal{D}_R$ and every proper subterm of s_i is a constructor term of R.

Let $l \to r \Leftarrow Cond \in R$, u be a non-variable constructor term of R, θ be a substitution and σ be an $NF^{\to_R}(\mathcal{F}, \mathcal{X})$-substitution. Then, all of the following hold:

(a) $l\sigma \xrightarrow{k}_R u\sigma$ and $Cond(\sigma, \xrightarrow{*}_R)$ implies $k = 1$,

(b) if $Cond(\theta, \xrightarrow{*}_R)$ holds, then there exists a $\mathcal{T}(\mathcal{C}_R, \mathcal{X})$-substitution δ such that $Cond(\delta, \to_R)$ and $\delta \lesssim \theta$,

(c) $Cond(\theta, \xrightarrow{*}_R)$ implies $Cond(\theta, \to_R)$, and

(d) if $l\sigma \to_R r\sigma$, then there exists an $NF^{\to_R}(\mathcal{F}, \mathcal{X})$-substitution σ' such that $Cond(\sigma', \to_R)$ and $\sigma|_{Var(l,r)} = \sigma'|_{Var(l,r)}$.

Proof. Since R is a constructor system and satisfies (1) and (2), the claim (a) holds obviously.

We prove the claim (b). Let $Cond = s_1 \twoheadrightarrow t_1 \wedge \cdots \wedge s_m \twoheadrightarrow t_m$ and suppose that $Cond(\theta, \xrightarrow{*}_R)$. Then, we have $s_i\theta \xrightarrow{*}_R t_i\theta$. Now, from Proposition 3.6.12, there exist a $\mathcal{T}(\mathcal{C}_R, \mathcal{X})$-substitution δ and a substitution θ' such that

- $\theta = \delta\theta'$,

- $\mathcal{D}om(\theta) = \mathcal{D}om(\delta)$,
- $(\mathcal{D}om(\theta) \cup \mathcal{VR}an(\theta)) \cap \mathcal{D}om(\theta') = \emptyset$, and
- for all $x \in \mathcal{D}om(\theta')$, $\mathtt{root}(x\theta') \in \mathcal{D}_R$.

We can assume that $\mathcal{V}ar(s_i) \cap \mathcal{D}om(\theta') = \emptyset$ without loss of generality. Then, it follows from a natural extension of Lemma 3.6.11 that $s_i\delta \xrightarrow{*}_R t_i\delta$. Here we can prove that $s_i\delta \to_R t_i\delta$ from the claim (a), and hence $\mathcal{C}ond(\delta, \to_R)$. It follows from $\theta = \delta\theta'$ that $\delta \lesssim \theta$.

The claim (c) follows from the claim (b) and Proposition 3.6.13 (a).

Now we prove the claim (d). Suppose that $l\sigma \to_R r\sigma$. Then, there exists a substitution η such that $l\sigma \equiv l\eta \to_R r\eta \equiv r\sigma$ and $\mathcal{C}ond(\eta, \xrightarrow{*}_R)$. Now there exists a $\mathcal{T}(\mathcal{C}_R, \mathcal{X})$-substitution η' such that $\mathcal{C}ond(\eta', \to_R)$ and $\eta' \lesssim \eta$.

On the other hand, let η'' be an $NF^{\to_R}(\mathcal{F}, \mathcal{X})$-substitution such that $\eta|_{\mathcal{V}ar(l,r)} = (\eta'|_{\mathcal{V}ar(l,r)})\eta''$. Moreover, let $\sigma' = \eta'\eta''$. Then, it is clear that $\mathcal{C}ond(\sigma', \to_R)$ and $\sigma|_{\mathcal{V}ar(l,r)} = \sigma'|_{\mathcal{V}ar(l,r)}$. □

To prove the correctness of $\mathcal{I}nv_{\mathsf{full}}$, we prepare two technical lemmas. We first explain the meanings of them by using an example.

Let R be a constructor TRS. Let $t \equiv f(g(a), h(b))$, f, g, h be defined symbols of R, a, b are constructor terms of R and c be a normal form with respect to $\to_R$. Then, applying $\mathscr{T}_{\mathsf{full}}$ to t, we obtain the following:

$$\mathscr{T}_{\mathsf{full}}(t) = \langle\!\langle\, x\,;\, f^{\#}(x) \twoheadrightarrow \mathsf{tp}_2(y, z) \wedge g^{\#}(y) \twoheadrightarrow a \wedge h^{\#}(z) \twoheadrightarrow b \,\rangle\!\rangle.$$

The above transformation can be considered that $\mathscr{T}_{\mathsf{full}}$ decompose the term t into the following simultaneous equations:

$$g(a) = y, \quad h(b) = z, \quad f(y, z) = x. \tag{3.7}$$

In addition, the conditional part generated by $\mathscr{T}_{\mathsf{full}}$ represents the following simultaneous equations for the inverses $f^{\#}$, $g^{\#}$ and $h^{\#}$:

$$f^{\#}(x) = (y, z), \quad g^{\#}(y) = a, \quad h^{\#}(z) = b. \tag{3.8}$$

Now assume that $t \xrightarrow[\mathrm{in}]{*}_R c$. Since this rewriting is done by innermost reduction, there exist normal forms c_1 and c_2 with respect to $\to_R$ such that $f(g(a), h(b)) \xrightarrow[\mathrm{in}]{*}_R f(c_1, c_2) \xrightarrow[\mathrm{in}]{*}_R c$. This means that the following equations hold:

$$g(a) = c_1, \quad h(b) = c_2, \quad f(c_1, c_2) = c. \tag{3.9}$$

Then, the substitution $\sigma = \{x \mapsto c, y \mapsto c_1, z \mapsto c_2\}$ can be considered as a solution of the equation (3.7). Now, the fact that $\mathcal{C}ond$ is satisfied by σ and $\rightarrow_{\mathcal{I}nv_{\mathsf{full}}(R)}$, means that the equations (3.8) are satisfied by σ, and hence the computations of $f^{\#}$, $g^{\#}$ and $h^{\#}$ are correct. This is the meaning of Lemma 3.6.15. Lemma 3.6.16 means the converse, that is, the satisfaction of the equations (3.8) implies the standard computation $g(a) \xrightarrow[\text{in}]{*}_R c_1$, $h(b) \xrightarrow[\text{in}]{*}_R c_2$ and $f(c_1, c_2) \xrightarrow[\text{in}]{*}_R c$. This means that the equations (3.9) hold.

Lemma 3.6.15 *Let R be a constructor TRS over a signature $\mathcal{F}$. Let t be a term over $\mathcal{F}$, s be a normal form with respect to $\rightarrow_R$ and $\mathscr{T}_{\mathsf{full}}(t) = \langle\!\langle\, u \,;\, \mathcal{C}ond \,\rangle\!\rangle$. If $t \xrightarrow[\text{in}]{*}_R s$, then $\mathcal{C}ond(\sigma, \rightarrow_{\mathcal{I}nv_{\mathsf{full}}(R)})$ and $s \equiv u\sigma$ for some $NF^{\rightarrow_R}(\mathcal{F}, \mathcal{X})$-substitution σ with $\mathcal{D}om(\sigma) \subseteq \mathcal{V}ar(\mathcal{C}ond) \setminus \mathcal{V}ar(t)$.*

Proof. We prove this lemma by induction on the lexicographic combination of the number k of steps of $t \xrightarrow[\text{in}]{k}_R s$, and the structure of the term t. Since the case that $k = 0$ or t is a variable is trivial, we consider the other cases.

- Consider the case of $t \equiv c(t_1, \ldots, t_n)$ with $c \in \mathcal{C}_R$. From the definition of $\mathscr{T}_{\mathsf{full}}$, we can assume that $u \equiv c(u_1, \ldots, u_n)$, $\mathcal{C}ond = \mathtt{true} \wedge \mathcal{C}ond_1 \wedge \cdots \wedge \mathcal{C}ond_n$ and $\mathscr{T}_{\mathsf{full}}(t_i) = \langle\!\langle\, u_i \,;\, \mathcal{C}ond_i \,\rangle\!\rangle$, and

$$(\mathcal{V}ar(u_i, \mathcal{C}ond_i) \setminus \mathcal{V}ar(t_i)) \cap \mathcal{V}ar(t_j, u_j, \mathcal{C}ond_j) = \emptyset \text{ for } i \neq j. \tag{3.10}$$

  Since the case of $n = 0$ is obvious, we consider the case of $n > 0$.

  Since c is a constructor of R, there exist terms s_i and numbers k_i such that $s \equiv c(s_1, \ldots, s_n)$, $t_i \xrightarrow[\text{in}]{k_i}_R s_i$ and $k_i \leq k$. Now t_i is a proper subterm of t. Then, by the induction hypothesis, there exists an $NF^{\rightarrow_R}(\mathcal{F}, \mathcal{X})$-substitution σ_i such that $\mathcal{C}ond_i(\sigma_i, \rightarrow_{\mathcal{I}nv_{\mathsf{full}}(R)})$,

$$s_i \equiv u_i\sigma_i, \text{ and} \tag{3.11}$$

$$\mathcal{D}om(\sigma_i) \subseteq \mathcal{V}ar(\mathcal{C}ond_i) \setminus \mathcal{V}ar(t_i). \tag{3.12}$$

  Let $\sigma = \bigcup_{i=1}^{n} \sigma_i$. It follows from (3.10) and (3.12) that $\mathcal{D}om(\sigma_i) \cap \mathcal{D}om(\sigma_j) = \emptyset$ for $i \neq j$. Then, σ is a substitution. Here it is clear that σ is an $NF^{\rightarrow_R}(\mathcal{F}, \mathcal{X})$-substitution and $\mathcal{D}om(\sigma) \subseteq \mathcal{V}ar(\mathcal{C}ond) \setminus \mathcal{V}ar(t)$.

  On the other hand, from Proposition 3.6.7 (c), we have $\mathcal{V}ar(\mathcal{C}ond_i) \setminus \mathcal{V}ar(t_i) \subseteq \mathcal{V}ar(\mathcal{C}ond) \setminus \mathcal{V}ar(t)$. Moreover, (3.10) and (3.12) hold now.

Then, we have $u_i\sigma \equiv s_i$. It is clear that $\mathcal{C}ond_i(\sigma, \to_{\mathcal{I}nv_{\mathsf{full}}(R)})$. Therefore, $\mathcal{C}ond(\sigma, \to_{\mathcal{I}nv_{\mathsf{full}}(R)})$ and

$$u\sigma \equiv c(u_1\sigma, \ldots, u_n\sigma) \equiv c(s_1, \ldots, s_n) \equiv s.$$

- Consider the remaining case, that is, $t \equiv f(t_1, \ldots, t_n)$ with $f \in \mathcal{D}_R$. From the definition of $\mathscr{T}_{\mathsf{full}}$, we can assume that $u \equiv y \in \mathcal{X}$, $\mathcal{C}ond = f^{\#}(y) \twoheadrightarrow \mathsf{tp}_n(u_1, \ldots, u_n) \wedge \mathcal{C}ond_1 \wedge \cdots \wedge \mathcal{C}ond_n$, $\mathscr{T}_{\mathsf{full}}(t_i) = \langle\!\langle\, u_i \,;\, \mathcal{C}ond_i \,\rangle\!\rangle$,

$$y \notin \mathcal{V}ar(t_i, u_i, \mathcal{C}ond_i), \text{ and} \tag{3.13}$$

$$(\mathcal{V}ar(u_i, \mathcal{C}ond_i) \setminus \mathcal{V}ar(t_i)) \cap \mathcal{V}ar(t_j, u_j, \mathcal{C}ond_j) = \emptyset \text{ for } i \neq j. \tag{3.14}$$

From the sequence $f(t_1, \ldots, t_n) \xrightarrow[\text{in}]{k}_R s$, there exist normal forms s_i with respect to $\to_R$ and numbers k', k'' such that $k' + k'' = k$ and

$$f(t_1, \ldots, t_n) \xrightarrow[\text{in}]{k'}_R f(s_1, \ldots, s_n) \xrightarrow[\text{in}]{k''}_R s.$$

Then, there exist numbers k_i such that $k_i \leq k'$ and $t_i \xrightarrow[\text{in}]{k_i}_R s_i$. Here, by the induction hypothesis, there exists an $NF^{\to_R}(\mathcal{F}, \mathcal{X})$-substitution σ_i such that $\mathcal{C}ond_i(\sigma_i, \to_{\mathcal{I}nv_{\mathsf{full}}(R)})$, $s_i \equiv u_i\sigma_i$, and

$$\mathcal{D}om(\sigma_i) \subseteq \mathcal{V}ar(\mathcal{C}ond_i) \setminus \mathcal{V}ar(t_i). \tag{3.15}$$

Since $s_1, \ldots, s_n$ are normal forms with respect to $\to_R$, we consider the cases in which $f(s_1, \ldots, s_n)$ is a redex of R, or not.

  - Consider the subcase that $f(s_1, \ldots, s_n)$ is a redex of R. Then, there exist a rewrite rule $\rho : f(w_1, \ldots, w_n) \to r \in R$ and an $NF^{\to_R}(\mathcal{F}, X)$-substitution θ such that

$$\mathcal{D}om(\theta) \subseteq \mathcal{V}ar(w_1, \ldots, w_n), \text{ and} \tag{3.16}$$

$$f(s_1, \ldots, s_n) \equiv f(w_1, \ldots, w_n)\theta \to_R^{[\varepsilon,\rho]} r\theta \xrightarrow[\text{in}]{k''-1}_R s.$$

    Let $\mathscr{T}_{\mathsf{full}}(r) = \langle\!\langle\, r' \,;\, \mathcal{C}ond' \,\rangle\!\rangle$. Then, we have a conditional rewrite rule $f^{\#}(r') \to \mathsf{tp}_n(w_1, \ldots, w_n) \Leftarrow \mathcal{C}ond' \in \mathcal{I}nv_{\mathsf{full}}(R)$ which satisfies $(\mathcal{V}ar(r', \mathcal{C}ond') \setminus \mathcal{V}ar(r)) \cap \mathcal{V}ar(w_1, \ldots, w_n) = \emptyset$.

    From Proposition 3.6.12, the $NF^{\to_R}(\mathcal{F}, \mathcal{X})$-substitution θ is decomposed into substitutions δ and σ' such that $\theta = \delta\sigma'$,

$$\mathcal{D}om(\theta) = \mathcal{D}om(\delta), \tag{3.17}$$

$(\mathcal{D}om(\theta) \cup \mathcal{VR}an(\theta)) \cap \mathcal{D}om(\sigma') = \emptyset$, and $\mathtt{root}(x\sigma') \in \mathcal{D}_R$ for all $x \in \mathcal{D}om(\sigma')$. We can assume without loss of generality that variables in $\mathcal{V}ar(r', \mathcal{C}ond') \setminus \mathcal{V}ar(r)$ are not used in the others. Then, we have

$$(\mathcal{D}om(\delta) \cup \mathcal{VR}an(\delta)) \cap (\mathcal{V}ar(r', \mathcal{C}ond') \setminus \mathcal{V}ar(r)) = \emptyset. \tag{3.18}$$

Then, from Proposition 3.6.10, we have

$$\mathscr{T}_{\mathsf{full}}(r\delta) = \langle\!\langle\, r'\delta \,;\, \delta(\mathcal{C}ond') \,\rangle\!\rangle.$$

On the other hand, from Lemma 3.6.11 for $r\theta \equiv r\delta\sigma' \xrightarrow[\text{in}]{k''-1}_R s$, there exists a normal form s' with respect to R such that $s \equiv s'\sigma'$ and $r\delta \xrightarrow[\text{in}]{k''-1}_R s'$. Here, by the induction hypothesis, there is an $NF^{\rightarrow_R}(\mathcal{F}, \mathcal{X})$-substitution σ'' such that $s' \equiv (r'\delta)\sigma''$,

$$(\delta(\mathcal{C}ond'))\,(\sigma'', \rightarrow_{\mathcal{I}nv_{\mathsf{full}}(R)}), \text{ and} \tag{3.19}$$

$$\mathcal{D}om(\sigma'') \subseteq \mathcal{V}ar\,(\delta(\mathcal{C}ond')) \setminus \mathcal{V}ar(r\delta). \tag{3.20}$$

It follows from (3.18) that $\mathcal{V}ar\,(\delta(\mathcal{C}ond')) \setminus \mathcal{V}ar(r\delta) = \emptyset$, and hence

$$\mathcal{D}om(\sigma'') \subseteq \mathcal{V}ar(r', \mathcal{C}ond') \setminus \mathcal{V}ar(r). \tag{3.21}$$

On the other hand, it follows from (3.19) and Proposition 3.6.13 that $\mathcal{C}ond'(\delta\sigma''\sigma', \rightarrow_{\mathcal{I}nv_{\mathsf{full}}(R)})$. Then, we have

$$f^{\#}(r')\delta\sigma''\sigma' \rightarrow_{\mathcal{I}nv_{\mathsf{full}}(R)} \mathsf{tp}_n(w_1, \ldots, w_n)\delta\sigma''\sigma'.$$

It follows from (3.16)–(3.18) and (3.21) that $\mathcal{V}ar(w_i) \cap \mathcal{D}om(\sigma'') = \emptyset$. It also follows from (3.18) and (3.21) that $\mathcal{VR}an(\delta|_{\mathcal{V}ar(w_i)}) \cap \mathcal{D}om(\sigma'') = \emptyset$. Then, we have $w_i\delta\sigma'' \equiv w_i\delta$, and hence

$$w_i\delta\sigma''\sigma' \equiv w_i\delta\sigma' \equiv w_i\theta.$$

Let $\sigma = \{y \mapsto s\} \cup \bigcup_{i=1}^{n} \sigma_i$. It follows from (3.13)–(3.15) that $y \notin \mathcal{D}om(\sigma_i)$ and $\mathcal{D}om(\sigma_i) \cap \mathcal{D}om(\sigma_j) = \emptyset$ for $i \neq j$. Hence σ is a substitution. It is clear that $\mathcal{D}om(\sigma) \subseteq \mathcal{V}ar(\mathcal{C}ond) \setminus \mathcal{V}ar(t)$, $u\sigma \equiv s$, $u_i\sigma \equiv s_i$ and $\mathcal{C}ond_i(\sigma, \rightarrow_{\mathcal{I}nv_{\mathsf{full}}(R)})$. It is also clear that σ is an $NF^{\rightarrow_R}(\mathcal{F}, \mathcal{X})$-substitution.

Now we show that the first condition $f^{\#}(y) \twoheadrightarrow \mathsf{tp}_n(u_1, \ldots, u_n)$ is satisfied by σ and $\rightarrow_{\mathcal{I}nv_{\mathsf{full}}(R)}$ as follows:

$$\begin{aligned} f^{\#}(y)\sigma \equiv f^{\#}(s) \equiv f^{\#}(r')\delta\sigma''\sigma' &\rightarrow_{\mathcal{I}nv_{\mathsf{full}}(R)} \mathsf{tp}_n(w_1, \ldots, w_n)\delta\sigma''\sigma' \\ &\equiv \mathsf{tp}_n(s_1, \ldots, s_n) \equiv \mathsf{tp}_n(u_1, \ldots, u_n)\sigma. \end{aligned}$$

Therefore, $\mathcal{C}ond(\sigma, \rightarrow_{\mathcal{I}nv_{\mathsf{full}}(R)})$ holds.

– Consider the remaining subcase that $f(s_1, \ldots, s_n)$ is not a redex of R, that is, a normal form with respect to $\rightarrow_R$. Now we have $s \equiv f(s_1, \ldots, s_n)$. Let $\sigma = \{y \mapsto s\} \cup \bigcup_{i=1}^{n} \sigma_i$. It follows from (3.13)–(3.15) that $y \notin \mathcal{D}om(\sigma_i)$ and $\mathcal{D}om(\sigma_i) \cap \mathcal{D}om(\sigma_j) = \emptyset$ for $i \neq j$. Hence σ is a substitution. It is clear that $\mathcal{D}om(\sigma) \subseteq \mathcal{V}ar(\mathcal{C}ond) \setminus \mathcal{V}ar(t)$, $u\sigma \equiv s$, $u_i\sigma \equiv s_i$ and $\mathcal{C}ond_i(\sigma, \rightarrow_{\mathcal{I}nv_{\mathsf{full}}(R)})$. It is also clear that σ is an $NF^{\rightarrow_R}(\mathcal{F}, \mathcal{X})$-substitution. On the other hand, we have the inverse-property rule $f^{\#}(f(x_1, \ldots, x_n)) \rightarrow \mathsf{tp}_n(x_1, \ldots, x_n) \in \mathcal{I}nv_{\mathsf{full}}(R)$, and hence

$$f^{\#}(y)\sigma \equiv f^{\#}(s) \equiv f^{\#}(f(s_1, \ldots, s_n)) \rightarrow_{\mathcal{I}nv_{\mathsf{full}}(R)} \mathsf{tp}_n(s_1, \ldots, s_n) \equiv \mathsf{tp}_n(u_1, \ldots, u_n)\sigma.$$

Therefore, $\mathcal{C}ond(\sigma, \rightarrow_{\mathcal{I}nv_{\mathsf{full}}(R)})$ holds.

□

Lemma 3.6.16 *Let R be a constructor TRS over a signature $\mathcal{F}$. Let t be a term over $\mathcal{F}$, $\mathscr{T}_{\mathsf{full}}(t) = \langle\!\langle\, u \,;\, \mathcal{C}ond \,\rangle\!\rangle$ and σ be a substitution with $\mathcal{R}an(\sigma|_{\mathcal{V}ar(t,u)}) \subseteq NF^{\rightarrow_R}(\mathcal{F}, \mathcal{X})$. Then, $\mathcal{C}ond(\sigma, \rightarrow_{\mathcal{I}nv_{\mathsf{full}}(R)})$ implies $t\sigma \xrightarrow[\mathrm{in}]{*}_R u\sigma$ $(\in NF^{\rightarrow_R}(\mathcal{F}, \mathcal{X}))$.*

Proof. We prove this lemma by induction on the lexicographic combination of the level k of rewrite relation $\mathcal{C}ond(\sigma, \xrightarrow[k]{}_{\mathcal{I}nv_{\mathsf{full}}(R)})$, and the structure of the term t. Since the case that t is a variable is trivial, we consider the other cases.

- Consider the case of $t \equiv c(t_1, \ldots, t_n)$ with $c \in \mathcal{C}_R$. From the definition of $\mathscr{T}_{\mathsf{full}}$, we can assume the followings:

  – $u \equiv c(u_1, \ldots, u_n)$, $\mathcal{C}ond = \mathtt{true} \wedge \mathcal{C}ond_1 \wedge \cdots \wedge \mathcal{C}ond_n$ and $\mathscr{T}_{\mathsf{full}}(t_i) = \langle\!\langle\, u_i \,;\, \mathcal{C}ond_i \,\rangle\!\rangle$, and

  – $(\mathcal{V}ar(u_i, \mathcal{C}ond_i) \setminus \mathcal{V}ar(t_i)) \cap \mathcal{V}ar(t_j, u_j, \mathcal{C}ond_j) = \emptyset$ for $i \neq j$.

  Since the case of $n = 0$ is obvious, we consider the case of $n > 0$. It follows from $\mathcal{C}ond(\sigma, \xrightarrow[k]{}_{\mathcal{I}nv_{\mathsf{full}}(R)})$ that $\mathcal{C}ond_i(\sigma, \xrightarrow[k]{}_{\mathcal{I}nv_{\mathsf{full}}(R)})$. Then, by the induction hypothesis for the proper subterm t_i, we have

  $$t_i\sigma \xrightarrow[\mathrm{in}]{*}_R u_i\sigma.$$

  Therefore, the following holds:

  $$t\sigma \equiv c(t_1, \ldots, t_n)\sigma \xrightarrow[\mathrm{in}]{*}_R c(u_1, \ldots, u_n)\sigma \equiv u\sigma.$$

- Consider the remaining case, that is, the case of $t \equiv f(t_1, \ldots, t_n)$ with $f \in \mathcal{D}_R$. From the definition of $\mathscr{T}_{\mathsf{full}}$, we can assume the followings:
  - $u \equiv y \in \mathcal{X}$, $\mathcal{C}ond = f^{\#}(y) \twoheadrightarrow \mathsf{tp}_n(u_1, \ldots, u_n) \wedge \mathcal{C}ond_1 \wedge \cdots \wedge \mathcal{C}ond_n$ and $\mathscr{T}_{\mathsf{full}}(t_i) = \langle\!\langle\, u_i \,;\, \mathcal{C}ond_i \,\rangle\!\rangle$,
  - $y \notin \mathcal{V}ar(t_i, u_i, \mathcal{C}ond_i)$, and
  - $(\mathcal{V}ar(u_i, \mathcal{C}ond_i) \setminus \mathcal{V}ar(t_i)) \cap \mathcal{V}ar(t_j, u_j, \mathcal{C}ond_j) = \emptyset$ for $i \neq j$.

  It follows from $\mathcal{C}ond(\sigma, \xrightarrow[k]{}_{\mathcal{I}nv_{\mathsf{full}}(R)})$ that $\mathcal{C}ond_i(\sigma, \xrightarrow[k]{}_{\mathcal{I}nv_{\mathsf{full}}(R)})$. Then, by the induction hypothesis for the proper subterm t_i, we have

  $$t_i\sigma \xrightarrow[\text{in}]{*}_R u_i\sigma.$$

  Here we consider which rule is applied to the sequence $f^{\#}(y)\sigma \xrightarrow[k]{}_{\mathcal{I}nv_{\mathsf{full}}(R)} \mathsf{tp}_n(u_1, \ldots, u_n)\sigma$.
  - Consider the subcase that $f^{\#}(r') \to \mathsf{tp}_n(w_1, \ldots, w_n) \Leftarrow \mathcal{C}ond' \in \mathcal{I}nv_{\mathsf{full}}(R)$ is applied. Then, there exists a substitution θ such that

    $$f^{\#}(y)\sigma \equiv f^{\#}(r')\theta \xrightarrow[k]{}_{\mathcal{I}nv_{\mathsf{full}}(R)} \mathsf{tp}_n(w_1, \ldots, w_n)\theta \equiv \mathsf{tp}_n(u_1, \ldots, u_n)\sigma$$

    and $\mathcal{C}ond'(\theta, \xrightarrow[k-1]{}_{\mathcal{I}nv_{\mathsf{full}}(R)})$. On the other hand, we have the rule $f(w_1, \ldots, w_n) \to r \in R$ and $\mathscr{T}_{\mathsf{full}}(r) = \langle\!\langle\, r' \,;\, \mathcal{C}ond' \,\rangle\!\rangle$ holds.
    Since R is a TRS, we have $\mathcal{V}ar(r) \subseteq \mathcal{V}ar(w_1, \ldots, w_n)$. It follows from $\mathcal{VR}an(\sigma|_{\mathcal{V}ar(t,u)}) \subseteq NF^{\to_R}(\mathcal{F}, \mathcal{X})$ that $y\sigma \in NF^{\to_R}(\mathcal{F}, \mathcal{X})$, and hence $y\sigma \equiv r'\theta \in NF^{\to_R}(\mathcal{F}, \mathcal{X})$. We also have $u_i\sigma \equiv w_i\theta \in NF^{\to_R}(\mathcal{F}, \mathcal{X})$. Then, we have $\mathcal{VR}an(\theta|_{\mathcal{V}ar(r,r')}) \subseteq NF^{\to_R}(\mathcal{F}, \mathcal{X})$. From Proposition 3.6.14 (d), there exists a $\mathcal{T}(\mathcal{C}_{\mathcal{I}nv_{\mathsf{full}}(R)}, \mathcal{X})$-substitution such that $\delta \lesssim \theta$ and $\mathcal{C}ond'(\delta, \xrightarrow[k-1]{}_{\mathcal{I}nv_{\mathsf{full}}(R)})$. Then, from Proposition 3.6.13 (a), we have $\mathcal{C}ond'(\theta, \xrightarrow[k-1]{}_{\mathcal{I}nv_{\mathsf{full}}(R)})$. Here, by the induction hypothesis, we have the following:

    $$r\theta \xrightarrow[\text{in}]{*}_R r'\theta.$$

    Therefore, the following sequence holds:

    $$\begin{aligned} t\sigma &\equiv f(t_1, \ldots, t_n)\sigma \xrightarrow[\text{in}]{*}_R f(u_1, \ldots, u_n)\sigma \\ &\equiv f(w_1, \ldots, w_n)\theta \xrightarrow[\text{in}]{*}_R r'\theta \equiv y\sigma \equiv u\sigma. \end{aligned}$$

  - Consider the remaining subcase that the rule $f^{\#}(f(x_1, \ldots, x_n)) \to \mathsf{tp}_n(x_1, \ldots, x_n) \in \mathcal{I}nv_{\mathsf{full}}(R)$ is applied. Now the following holds:

    $$y\sigma \equiv f(u_1, \ldots, u_n)\sigma \in NF^{\to_R}(\mathcal{F}, \mathcal{X}).$$

Therefore, we have

$$t\sigma \equiv f(t_1, \ldots, t_n) \xrightarrow[\text{in}]{*}_R f(u_1, \ldots, u_n)\sigma \equiv y\sigma \equiv u\sigma.$$

□

Theorem 3.6.17 *Let R be a constructor TRS over a signature $\mathcal{F}$. Let f be a defined symbol of R and $t, t_1, \ldots, t_n$ are normal forms with respect to $\to_R$. Then, $f(t_1, \ldots, t_n) \xrightarrow[\text{in}]{*}_R t$ iff $f^{\#}(t) \to_{\mathcal{I}nv_{\mathsf{full}}(R)} \mathsf{tp}_n(t_1, \ldots, t_n)$.*

Proof. From the definition of $\mathscr{T}_{\mathsf{full}}$, we assume that all of the following:

(a) $\mathscr{T}_{\mathsf{full}}(f(t_1, \ldots, t_n)) = \langle\!\langle y; \mathcal{C}ond \rangle\!\rangle$, $\mathcal{C}ond = f^{\#}(y) \twoheadrightarrow \mathsf{tp}_n(u_1, \ldots, u_n) \wedge \mathcal{C}ond_1 \wedge \cdots \wedge \mathcal{C}ond_n$, $\mathscr{T}_{\mathsf{full}}(t_i) = \langle\!\langle\, u_i \,;\, \mathcal{C}ond_i \,\rangle\!\rangle$,

(b) y is a fresh variable, that is, $y \notin \bigcup_{i=1}^{n} \mathcal{V}ar(t_i, u_i, \mathcal{C}ond_i)$, and

(c) $(\mathcal{V}ar(u_i, \mathcal{C}ond_i) \setminus \mathcal{V}ar(t_i)) \cap \mathcal{V}ar(t_j, u_j, \mathcal{C}ond_j) = \emptyset$.

We first prove the *only-if* part. Suppose that $f(t_1, \ldots, t_n) \xrightarrow[\text{in}]{*}_R t$. Then, from Lemma 3.6.15, there exists an $NF^{\to_R}(\mathcal{F}, \mathcal{X})$-substitution σ such that $t \equiv y\sigma$, $\mathcal{C}ond(\sigma, \to_{\mathcal{I}nv_{\mathsf{full}}(R)})$ and $\mathcal{D}om(\sigma) \subseteq \mathcal{V}ar(\mathcal{C}ond) \setminus \mathcal{V}ar(t)$. Since $\mathcal{C}ond_i(\sigma, \to_{\mathcal{I}nv_{\mathsf{full}}(R)})$ holds and σ is an $NF^{\to_R}(\mathcal{F}, X)$-substitution, we have $t_i\sigma \xrightarrow[\text{in}]{*}_R u_i\sigma$. It follows from $\mathcal{D}om(\sigma) \subseteq \mathcal{V}ar(\mathcal{C}ond) \setminus \mathcal{V}ar(t)$ that $\mathcal{V}ar(t_i) \cap \mathcal{D}om(\sigma) = \emptyset$, and hence $t_i\sigma \equiv t_i$. Since σ is an $NF^{\to_R}(\mathcal{F}, \mathcal{X})$-substitution, we have $t_i\sigma \equiv t_i \equiv u_i\sigma$.

On the other hand, it follows from $\mathcal{C}ond(\sigma, \to_{\mathcal{I}nv_{\mathsf{full}}(R)})$ that $f^{\#}(y)\sigma \to_{\mathcal{I}nv_{\mathsf{full}}(R)} \mathsf{tp}_n(u_1, \ldots, u_n)\sigma$. Therefore, we have the following:

$$f^{\#}(t) \equiv f^{\#}(y)\sigma \to_{\mathcal{I}nv_{\mathsf{full}}(R)} \mathsf{tp}_n(u_1, \ldots, u_n)\sigma \equiv \mathsf{tp}_n(t_1, \ldots, t_n).$$

Next we prove the *if* part. Suppose that $f^{\#}(t) \to_{\mathcal{I}nv_{\mathsf{full}}(R)} \mathsf{tp}_n(t_1, \ldots, t_n)$. Since t_i is a normal form with respect to $\to_R$, $t_i \xrightarrow[\text{in}]{*}_R t_i$ holds. Then, from Lemma 3.6.15, there exists an $NF^{\to_R}(\mathcal{F}, \mathcal{X})$-substitution σ_i such that $t_i \equiv u_i\sigma_i$, $\mathcal{C}ond(\sigma_i, \to_{\mathcal{I}nv_{\mathsf{full}}(R)})$ and $\mathcal{D}om(\sigma_i) \subseteq \mathcal{V}ar(\mathcal{C}ond_i) \setminus \mathcal{V}ar(t_i)$.

Let $\sigma = \{y \mapsto t\} \cup \bigcup_{i=1}^{n} \sigma_i$. It follows from the assumptions (b) and (c) that y and variables in $\mathcal{D}om(\sigma_i)$ are disjoint, and hence σ is a substitution. It is clear that $\mathcal{C}ond_i(\sigma, \to_{\mathcal{I}nv_{\mathsf{full}}(R)})$ and $f^{\#}(y)\sigma \to_{\mathcal{I}nv_{\mathsf{full}}(R)} \mathsf{tp}_n(u_1, \ldots, u_n)\sigma$. Hence $\mathcal{C}ond(\sigma, \to_{\mathcal{I}nv_{\mathsf{full}}(R)})$ holds.

On the other hand, it follows from the assumption (b), $y \notin \mathcal{V}ar(\mathcal{C}ond)$ and $\mathcal{D}om(\sigma_i) \subseteq \mathcal{V}ar(\mathcal{C}ond_i) \setminus \mathcal{V}ar(t_i)$ that $\mathcal{D}om(\sigma) \subseteq \mathcal{V}ar(\mathcal{C}ond) \setminus \mathcal{V}ar(f(t_1, \ldots, t_n))$,

and hence $\sigma|_{\mathcal{V}ar(f(t_1,\ldots,t_n))} = \emptyset$. Therefore, from Lemma 3.6.16, we have the following sequence:

$$f(t_1,\ldots,t_n) \equiv f(u_1,\ldots,u_n)\sigma \xrightarrow[\text{in}]{*}_R y\sigma \equiv t.$$

□

Convergent TRSs have the following nice property on $\xrightarrow[\text{in}]{}_R$.

Proposition 3.6.18 *Let R be a convergent TRS. Let s be a term and t be in normal forms with respect to $\to_R$. Then, $s \xrightarrow{*}_R t$ iff $s \xrightarrow[\text{in}]{*}_R t$.*

The above proposition holds obviously from the convergence of R. Consequently, we obtain the following corollary.

Corollary 3.6.19 *Let R be a convergent constructor TRS over a signature $\mathcal{F}$. Let f be a defined symbol of R and $t, t_1, \ldots, t_n$ are normal forms of R. Then, $f(t_1,\ldots,t_n) \xrightarrow{*}_R t$ iff $f^{\#}(t) \to_{\mathcal{I}nv_{\mathsf{full}}(R)} \mathsf{tp}_n(t_1,\ldots,t_n)$.*

From the above corollary and $\mathcal{T}(\mathcal{C}_R, \mathcal{X}) \subseteq NF^{\to_R}(\mathcal{F}, \mathcal{X})$, it is shown that the CTRS $\mathcal{I}nv_{\mathsf{full}}(R)$ is an inverse system of the convergent constructor TRS R in the sense of Definition 3.1.1.

Remark that $\mathcal{I}nv_{\mathsf{full}}(R)$ is not always confluent even if R is confluent. The reason is that inverses of many-to-one functions are one-to-many images. For example, R_7 is confluent but R_8 $(= \mathcal{I}nv_{\mathsf{full}}(R_7))$ is not.

Now we explain the reason why we add the inverse-property rules in the definition of $\mathcal{I}nv_{\mathsf{full}}(R)$. The inverse-property rule $f^{\#}(f(x_1,\ldots,x_n)) \to \mathsf{tp}_n(x_1,\ldots,x_n)$ comes from the essential property (3.1) which inverses should satisfy. This rule is necessary while it is not if the defined symbol f is sufficiently complete. This is not necessary in the case of PT-TRSs, too. In contrast, this is necessary for inverse computation containing partial functions. Consider the following TRS over the signature $\{\,\mathsf{s}, \mathsf{0}, \mathsf{half}, \mathsf{add}, \mathsf{mul}, \mathsf{mh}\,\}$:

$$\begin{aligned} R_9 = \{\ & \mathsf{half}(\mathsf{0}) \to \mathsf{0}, && \mathsf{half}(\mathsf{s}^2(x)) \to \mathsf{s}(\mathsf{half}(x)), \\ & \mathsf{add}(\mathsf{0}, y) \to y, && \mathsf{add}(\mathsf{s}(x), y) \to \mathsf{s}(\mathsf{add}(x, y)), \\ & \mathsf{mul}(\mathsf{0}, y) \to \mathsf{0}, && \mathsf{mul}(x, \mathsf{0}) \to \mathsf{0}, \\ & \mathsf{mul}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{s}(\mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y)), \\ & \mathsf{mh}(x) \to \mathsf{mul}(x, \mathsf{half}(\mathsf{s}(x))) && \}. \end{aligned}$$

The defined symbol half is not sufficiently complete since there is no ground constructor term t such that $\mathsf{half}(\mathsf{s}(\mathsf{0})) \xrightarrow{*}_{R_9} t$. An inverse CTRS of R_9 obtained

by $\mathcal{I}nv_{\mathsf{full}}$ is as follows:

$$
\begin{aligned}
R_{10} &= \mathcal{I}nv_{\mathsf{full}}(R_9) \\
&= \{\ \mathsf{half}^{\#}(\mathsf{0}) \to \mathsf{tp}_1(\mathsf{0}), \\
&\quad\ \mathsf{half}^{\#}(\mathsf{s}(y) \to \mathsf{tp}_1(\mathsf{s}^2(x)) \Leftarrow \mathsf{half}^{\#}(y) \twoheadrightarrow \mathsf{tp}_(x), \\
&\quad\ \mathsf{half}^{\#}(\mathsf{half}(x)) \to \mathsf{tp}_1(x), \\
&\quad\ \mathsf{add}^{\#}(y) \to \mathsf{tp}_2(\mathsf{0}, y), \\
&\quad\ \mathsf{add}^{\#}(\mathsf{s}(z)) \to \mathsf{tp}_2(\mathsf{s}(x), y) \Leftarrow \mathsf{add}^{\#}(z) \twoheadrightarrow \mathsf{tp}_2(x, y), \\
&\quad\ \mathsf{add}^{\#}(\mathsf{add}(x, y)) \to \mathsf{tp}_2(x, y), \\
&\quad\ \mathsf{mul}^{\#}(\mathsf{0}) \to \mathsf{tp}_2(\mathsf{0}, y), \\
&\quad\ \mathsf{mul}^{\#}(\mathsf{0}) \to \mathsf{tp}_2(x, \mathsf{0}), \\
&\quad\ \mathsf{mul}^{\#}(\mathsf{s}(z)) \to \mathsf{tp}_2(\mathsf{s}(x), \mathsf{s}(y)) \\
&\qquad\qquad \Leftarrow \mathsf{add}^{\#}(z) \twoheadrightarrow \mathsf{tp}_2(w, y) \wedge \mathsf{mul}^{\#}(w) \twoheadrightarrow \mathsf{tp}_2(x, \mathsf{s}(y)), \\
&\quad\ \mathsf{mul}^{\#}(\mathsf{mul}(x, y)) \to \mathsf{tp}_2(x, y), \\
&\quad\ \mathsf{mh}^{\#}(y) \to \mathsf{tp}_1(x) \Leftarrow \mathsf{mul}^{\#}(y) \twoheadrightarrow \mathsf{tp}_2(x, z) \wedge \mathsf{half}^{\#}(z) \twoheadrightarrow \mathsf{tp}_1(\mathsf{s}(x)), \\
&\quad\ \mathsf{mh}^{\#}(\mathsf{mh}(x)) \to \mathsf{tp}_1(x) \qquad\qquad \}.
\end{aligned}
$$

Since we have the rewrite sequence $\mathsf{mh}(\mathsf{0}) \xrightarrow{*}_{R_9} \mathsf{0}$, the rewrite sequence $\mathsf{mh}^{\#}(\mathsf{0}) \xrightarrow{*}_{R_{10}} \mathsf{0}$ must hold. To apply the rule $\mathsf{mh}^{\#}(y) \to \mathsf{tp}_1(x) \Leftarrow \mathsf{half}^{\#}(y) \twoheadrightarrow \mathsf{tp}_1(x)$ to the term $\mathsf{mh}^{\#}(\mathsf{0})$ and to rewrite $\mathsf{mh}^{\#}(\mathsf{0})$ to $\mathsf{0}$, the conditional part must be satisfied by $\xrightarrow{*}_{R_{10}}$ and a substitution which contains $\{x \mapsto \mathsf{0}, y \mapsto \mathsf{0}\}$. Since $\mathsf{mul}^{\#}(\mathsf{0})$ can be rewritten to $\mathsf{tp}_2(\mathsf{0}, t)$ for any term t, $\mathsf{mul}^{\#}(\mathsf{0}) \to_{R_{10}} \mathsf{tp}_2(\mathsf{0}, t)$ holds for any term t. On the other hand, a term u and a rule which enable to rewrite $\mathsf{half}^{\#}(u)$ to $\mathsf{s}(\mathsf{0})$ is only $\mathsf{half}(\mathsf{s}(\mathsf{0}))$ and $\mathsf{half}^{\#}(\mathsf{half}(x)) \to \mathsf{tp}_1(x)$. Therefore, the inverse-property rule $\mathsf{half}^{\#}(\mathsf{half}(x)) \to \mathsf{tp}_1(x)$ is necessary.

3.6.4 Transformation of Inverse CTRSs into EV-TRSs

In this subsection, we give a transformation of the CTRS $\mathcal{I}nv_{\mathsf{full}}(R)$ into an EV-TRS whose rewriting on terms $f^{\#}(t)$ is equivalent to that of the CTRS. Since $\mathcal{I}nv_{\mathsf{full}}(R)$ is deterministic 4-CTRS, we extend the transformation method of deterministic 3-CTRSs to TRSs, which is proposed by E. Ohlebusch [41], into that of 4-CTRSs.

Definition 3.6.20 ($\mathbb{U}$) *Let R be a deterministic 4-CTRS over a signature $\mathcal{F}$. For every conditional rewrite rule $\rho : l \to r \Leftarrow s_1 \twoheadrightarrow t_1 \wedge \cdots \wedge s_n \twoheadrightarrow t_n \in R$, we prepare n fresh function symbols $\mathsf{u}^{\rho}_1, \ldots, \mathsf{u}^{\rho}_n$ all of which do not appear in $\mathcal{F}$.*

Then, the set $\mathbb{U}(\rho)$ *of rewrite rules determined by* ρ *is defined as follows:*

$$\begin{array}{rcl}
\mathbb{U}(\rho) = \{ \quad l & \to & \mathsf{u}_1^\rho(s_1, \mathcal{V}list(X_1)), \\
\mathsf{u}_1^\rho(t_1, \mathcal{V}list(X_1)) & \to & \mathsf{u}_2^\rho(s_2, \mathcal{V}list(X_2)), \\
& \vdots & \\
\mathsf{u}_i^\rho(t_i, \mathcal{V}list(X_i)) & \to & \mathsf{u}_{i+1}^\rho(s_{i+1}, \mathcal{V}list(X_{i+1})), \\
& \vdots & \\
\mathsf{u}_n^\rho(t_n, \mathcal{V}list(X_n)) & \to & r \qquad\qquad \}
\end{array}$$

where X_i *is the set of variables that appear in either* l *or* $t_1, \ldots, t_{i-1}$*, and also in either* r*,* t_i *or* $s_j \twoheadrightarrow t_j$ *for* $i < j \leq n$*, that is,*

$$X_i = \mathcal{V}ar(l, t_1, \ldots, t_{i-1}) \cap \mathcal{V}ar(r, t_i, s_{i+1}, t_{i+1}, \ldots, s_n, t_n).$$

Note that $\mathbb{U}(l \to r) = \{ l \to r \}$*.* $\mathbb{U}$ *is naturally extended over deterministic 4-CTRSs, defined as* $\mathbb{U}(R) = \bigcup_{\rho \in R} \mathbb{U}(\rho)$*.*

$\mathcal{F}_\mathbb{U}$ denotes the signature determined by $\mathcal{F}$, $\mathbb{U}$ and R, that is, $\mathcal{F}_\mathbb{U} = \mathcal{F} \cup \{ \mathsf{u}_i^\rho \mid \rho : l \to r \Leftarrow s_1 \twoheadrightarrow t_1 \wedge \cdots \wedge s_n \twoheadrightarrow t_n \in R, 1 \leq i \leq n \}$. From the above definition, it is obvious that $\mathbb{U}(R)$ is an EV-TRS over the signature $\mathcal{F}_\mathbb{U}$, $\mathcal{D}_{\mathbb{U}(R)} = \mathcal{D}_R \cup \{ \mathsf{u}_i^\rho \mid \rho : l \to r \Leftarrow s_1 \twoheadrightarrow t_1 \wedge \cdots \wedge s_n \twoheadrightarrow t_n \in R, 1 \leq i \leq n \}$ and $\mathcal{C}_{\mathbb{U}(R)} = \mathcal{C}_R$.

Example 3.6.21 From the CTRS R_8 in Example 3.6.5, we obtain the following EV-TRS by $\mathbb{U}$:

$$\begin{array}{rl}
R_{11} = \mathbb{U}(R_8) (= \mathcal{I}nv_{\mathsf{full}}(R_7)) & \\
= \{ & \mathsf{add}^\#(y) \to \mathsf{tp}_2(0, y), \\
& \mathsf{add}^\#(\mathsf{s}(z)) \to \mathsf{u}_1(\mathsf{add}^\#(z)), \\
& \mathsf{u}_1(\mathsf{tp}_2(x, y)) \to \mathsf{tp}_2(\mathsf{s}(x), y), \\
& \mathsf{add}^\#(\mathsf{add}(x, y)) \to \mathsf{tp}_2(x, y), \\
& \mathsf{mul}^\#(0) \to \mathsf{tp}_2(0, y), \\
& \mathsf{mul}^\#(0) \to \mathsf{tp}_2(x, 0), \\
& \mathsf{mul}^\#(\mathsf{s}(z)) \to \mathsf{u}_2(\mathsf{add}^\#(z)), \\
& \mathsf{u}_2(\mathsf{tp}_2(w, y)) \to \mathsf{u}_3(\mathsf{mul}^\#(w), y), \\
& \mathsf{u}_3(\mathsf{tp}_2(x, \mathsf{s}(y)), y) \to \mathsf{tp}_2(\mathsf{s}(x), \mathsf{s}(y)), \\
& \mathsf{mul}^\#(\mathsf{mul}(x, y)) \to \mathsf{tp}_2(x, y) \qquad \}.
\end{array}$$

□

Next we analyze the syntactic properties of $\mathbb{U}(R)$ preserved by $\mathbb{U}$ from those of R. This is the second half of the analysis on the syntactic properties of inverse systems generated by our transformation $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(\cdots))$.

Proposition 3.6.22 *Let R be a deterministic 4-CTRS.*

(a) R is a deterministic 3-CTRS iff $\mathbb{U}(R)$ is a TRS.

(b) R is strictly non-erasing iff $\mathbb{U}(R)$ is non-erasing.

(c) R is a strict constructor system iff $\mathbb{U}(R)$ is a constructor system.

(d) R is strictly left-linear iff $\mathbb{U}(R)$ is right-linear.

(e) R is strictly right-linear iff $\mathbb{U}(R)$ is left-linear.

Proof. Since the claim (b)–(e) hold obviously from the definition of the properties, we only prove the claim (a). We first prove the *only-if* part of (a). Consider the rules of $\mathbb{U}(\rho)$ in Definition 3.6.20. It is clear that $X_{i+1} \subseteq X_i$. It follows from the definition of the deterministic rule that $\mathcal{V}ar(s_{i+1}) \subseteq X_i \cup \mathcal{V}ar(t_i)$. Then, $\mathcal{V}ar\left(\mathsf{u}^{\rho}_{i+1}(s_{i+1}, \mathcal{V}list(X_{i+1}))\right) \subseteq \mathcal{V}ar\left(\mathsf{u}^{\rho}_{i}(t_i, \mathcal{V}list(X_i))\right)$ holds. Since R is a 3-CTRS, $\mathcal{V}ar(r) \subseteq X_n \cup \mathcal{V}ar(t_n)$ also holds, and hence $\mathcal{V}ar(r) \subseteq \mathcal{V}ar\left(\mathsf{u}^{\rho}_{n}(t_n, \mathcal{V}list(X_n))\right)$. Therefore, $\mathbb{U}(R)$ is a TRS.

Next, we prove the *if* part of (a). Suppose that $\mathbb{U}(\rho)$ in Definition 3.6.20 is a TRS. Then, we have $\mathcal{V}ar(r) \subseteq \mathcal{V}ar(t_n) \cup X_n$, $\mathcal{V}ar(s_{i+1}) \cup X_{i+1} \subseteq \mathcal{V}ar(t_i) \cup X_i$, and $\mathcal{V}ar(s_1) \cup X_1 \subseteq \mathcal{V}ar(l)$. These conditions on variable appearance in the rule satisfy the condition of deterministic rules. Then, ρ is a deterministic and type 3. Therefore, R is a deterministic 3-CTRS. □

In the sequel, we discuss the relationship between rewriting of CTRS R and EV-TRS $\mathbb{U}(R)$. Let R be a deterministic 4-CTRS over a signature $\mathcal{F}$. The transformation $\mathbb{U}$ is said to be *sound* for R if for every terms s and t over $\mathcal{F}$, $s \xrightarrow{*}_R t$ implies $s \xrightarrow{*}_{\mathbb{U}(R)} t$. Conversely, $\mathbb{U}$ is said to be *complete* for R if for every terms s and t over $\mathcal{F}$, $s \xrightarrow{*}_{\mathbb{U}(R)} t$ implies $s \xrightarrow{*}_R t$. From the definition of $\mathbb{U}(\rho)$, it is trivial that $\mathbb{U}$ is sound for every deterministic 4-CTRS. However, $\mathbb{U}$ is not complete for all deterministic 4-CTRSs. The counterexample of the transformations in [34, 42] is also a counterexample of the transformation $\mathbb{U}$ in this thesis.

Here we give a condition of rewrite sequences, for which $\mathbb{U}$ is complete and which CTRSs generated by $\mathcal{I}nv_{\mathsf{full}}(R)$ satisfy.

Theorem 3.6.23 *Let R be a strictly non-erasing and strict constructor 4-CTRS over a signature $\mathcal{F}$. Suppose that every conditional rewrite rule $\rho : l' \to r' \Leftarrow s'_1 \twoheadrightarrow t'_1 \wedge \cdots \wedge s'_m \twoheadrightarrow t'_m \in R$ satisfies both of the followings:*

1. *r' and $t'_1, \ldots, t'_m$ are non-variable constructor terms of R, and*
2. *for $1 \leq i \leq n$, the root symbol of s'_i is defined symbol of R and every proper subterm of s'_i is a constructor term of R, that is, $\mathtt{root}(s'_i) \in \mathcal{D}_R$ and $\{ u \mid u \lhd s'_i \} \subseteq \mathcal{T}(\mathcal{C}_R, \mathcal{X})$.*

Let f be a defined symbol of R, t and $t_1, \ldots, t_n$ are normal forms with respect to $\to_R$ and the root symbol of t is a constructor of R. Then, $f(t_1, \ldots, t_n) \xrightarrow{*}_{\mathbb{U}(R)} t$ implies $f(t_1, \ldots, t_n) \to_R t$.

Proof. We prove this claim by induction on the number k of steps of $f(t_1, \ldots, t_n) \xrightarrow{k}_{\mathbb{U}(R)} t$. Since the root symbol of t is a constructor, we have $k > 0$.

Now all of the following hold:

- t and $t_1, \ldots, t_n$ do not contain any symbol of u^{ρ}_j,
- the rule of u^{ρ}_j is unique, and
- $\mathbb{U}(R)$ is a non-erasing constructor EV-TRS from Proposition 3.6.22 (b) and (c).

From the above, there exist rules like $\mathbb{U}(\rho)$ in Definition 3.6.20 and the rewrite sequence $f(t_1, \ldots, t_n) \xrightarrow{k}_{\mathbb{U}(R)} t$ can be decomposed as follows:

$$\begin{aligned} f(t_1, \ldots, t_n) \equiv l\sigma \equiv l\sigma_1 &\to_{\mathbb{U}(R)} \mathsf{u}^{\rho}_1(s_1, \mathcal{V}list(X_1))\sigma_1 \\ &\xrightarrow{k_1}_{\mathbb{U}(R)} \mathsf{u}^{\rho}_1(t_1, \mathcal{V}list(X_1))\sigma_2 \to_{\mathbb{U}(R)} \mathsf{u}^{\rho}_2(s_2, \mathcal{V}list(X_2))\sigma_2 \\ &\cdots \\ &\xrightarrow{k_n}_{\mathbb{U}(R)} \mathsf{u}^{\rho}_n(t_n, \mathcal{V}list(X_n))\sigma_{n+1} \to_{\mathbb{U}(R)} r\sigma_{n+1} \equiv r\sigma \equiv t \end{aligned}$$

where $k_i < k$. Now, from the definition of $\mathbb{U}(\rho)$, we have the rule $l \to r \Leftarrow \mathcal{C}ond \in R$ where $\mathcal{C}ond = s_1 \twoheadrightarrow t_1 \wedge \cdots \wedge s_n \twoheadrightarrow t_n$. Since R is a non-erasing constructor system and σ is an $NF^{\to_R}(\mathcal{F}, \mathcal{X})$-substitution, σ_i is also an $NF^{\to_R}(\mathcal{F}, \mathcal{X})$-substitution.

It follows from $\mathtt{root}(s_i) \in \mathcal{D}_R$ and $\mathtt{root}(t_i) \in \mathcal{C}_R$ that $\mathtt{root}(s_i\sigma_i) \in \mathcal{D}_R$ and $\mathtt{root}(t_i\sigma_{i+1}) \in \mathcal{C}_R$. Since σ_i is an $NF^{\to_R}(\mathcal{F}, \mathcal{X})$-substitution, every proper subterm of either $s_i\sigma_i$ or $t_i\sigma_{i+1}$ is a normal form with respect to $\to_R$. By the induction hypothesis for $s_i\sigma_i \xrightarrow{k_i}_{\mathbb{U}(R)} t_i\sigma_{i+1}$, we have $s_i\sigma_i \to_R t_i\sigma_{i+1}$.

Since $\mathbb{U}(R)$ is non-erasing and $\mathcal{V}list(X_i)\sigma_i = \mathcal{V}list(X_i)\sigma_{i+1}$, we can assume that $\sigma_i = \sigma_{i+1}$. Hence we can also assume that $\sigma_i = \sigma$. Then, we have $s_i\sigma \equiv s_i\sigma_i \to_R t_i\sigma_{i+1} \equiv t_i\sigma$. This implies $\mathcal{C}ond(\sigma, \to_R)$. Therefore, $f(t_1, \ldots, t_n) \equiv l\sigma \to_R r\sigma \equiv t$. □

Since $\mathcal{I}nv_{\mathsf{full}}(R)$ satisfies the conditions in the above theorem, we can obtain the following result.

Corollary 3.6.24 *Let R be a constructor TRS over a signature $\mathcal{F}$. Let f be a defined symbol of R and $t, t_1, \ldots, t_n$ are normal forms of R. Then, $f^{\#}(t) \xrightarrow{*}_{\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R))} \mathsf{tp}_n(t_1, \ldots, t_n)$ implies $f^{\#}(t) \to_{\mathcal{I}nv_{\mathsf{full}}(R)} \mathsf{tp}_n(t_1, \ldots, t_n)$.*

The above corollary means that $\mathbb{U}$ is complete for inverse computation which is done by using $\mathcal{I}nv_{\mathsf{full}}(R)$.

The reason why the completeness does not hold in general, seems that proper subterms $\mathsf{u}^{\rho}_j(\cdots)$ in redexes are sometimes erased by erasing rule before the evaluation of the conditional part of ρ is finished. Hence, we have the following conjectures.

Conjecture 3.6.25 *$\mathbb{U}$ is complete for strictly non-erasing 4-CTRSs.*

Conjecture 3.6.26 *$\mathbb{U}$ is complete for strictly left-linear 4-CTRSs.*

Conjecture 3.6.26 is an extension of the theorem in [34] that the transformation for normal CTRSs which is proposed in [34], is complete for left-linear systems. The reason why we guess Conjecture 3.6.26 is that we consider $\mathbb{U}$ as a simple extension of the transformation in [34].

3.6.5 Syntactic Properties of Inverse EV-TRSs

Finally, from Proposition 3.6.9 and 3.6.22, we conclude properties of inverse EV-TRSs generated by the decomposition $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(\cdots))$ of our transformations.

Theorem 3.6.27 *Let R be a constructor TRS over a signature $\mathcal{F}$. Then,*

(a) $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R))$ is a non-erasing constructor EV-TRS over the signature $\mathcal{F}^{\#}_{\mathbb{U}}$.

Moreover, $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R))$ has the following properties:

(b) If R is non-erasing, then $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R))$ is a TRS.

(c) If R is left-linear, then $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R))$ is right-linear.

(d) If R is right-linear, then $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R))$ is left-linear.

It is shown that for a right-linear constructor TRS, all normal forms of a given terminating term can be obtained by innermost strategy [47] (see Section 5.3). This result is valuable to improve efficiency of inverse computation by our generated inverse TRSs. On the other hand, the left-linearity is assumed in many functional languages. Therefore, the property (b) and (c) in the above theorem are said to be useful and they will play important roles later.

3.7 Extension

In Subsection 3.6.1, we have shown that the inverse-property rules of the form $f^{\#}(f(x_1,\ldots,x_n)) \to \mathsf{tp}_n(x_1,\ldots,x_n)$ are necessary for inverse computation of constructor TRSs. Here we add them to the inverse compiler $\mathcal{I}nv_{\mathsf{PT}}$ for PT-TRSs as follows:

$$\begin{aligned}\mathcal{I}nv_{\mathsf{PT}}(R) = \{\ & f^{\#}(r') \to \mathsf{tp}_n(p,x_2,\ldots,x_n) \Leftarrow \mathcal{C}ond \\ & \quad \mid\ f(p,x_2,\ldots,x_n) \to r \in R,\ \mathscr{T}_{\mathsf{PT}}(r) = \langle\!\langle\, r'\ ;\ \mathcal{C}ond\,\rangle\!\rangle, \\ & \qquad (\mathcal{V}ar(r',\mathcal{C}ond) \setminus \mathcal{V}ar(r)) \cap \mathcal{V}ar(p,x_2,\ldots,x_n) = \emptyset\ \ \} \\ & \cup \{\ f^{\#}(f(x_1,\ldots,x_n)) \to \mathsf{tp}_n(x_1,\ldots,x_n) \mid f \in \mathcal{D}_R\ \}.\end{aligned}$$

According to the above extension, Theorem 3.4.9 which guarantees the correctness of $\mathcal{I}nv_{\mathsf{PT}}$ is extended to a stronger result as follows:

Theorem 3.7.1 *Let R be a PT-TRS over a signature $\mathcal{F}$. Let f be a defined symbol of R and $t, t_1, \ldots, t_n$ be normal forms with respect to $\to_R$. Then, $f(t_1,\ldots,t_n) \xrightarrow{*}_R t$ iff $f^{\#}(t) \to_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_n(t_1,\ldots,t_n)$.*

Proof. We first prove the *only-if* part by induction on the number k of steps of $f(t_1,\ldots,t_n) \xrightarrow{k}_R t$. The difference from the proof of Theorem 3.4.9 is the case of $k = 0$. Here we only show the case of $k = 0$ because the rest is similar to the proof of Theorem 3.4.9. In this case, $f(t_1,\ldots,t_n)$ is a normal form with respect to $\to_R$. Since we have the rule $f^{\#}(f(x_1,\ldots,x_n)) \to \mathsf{tp}_n(x_1,\ldots,x_n)$, the following holds:

$$f^{\#}(t) \equiv f^{\#}(f(t_1,\ldots,t_n)) \to_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_n(t_1,\ldots,t_n).$$

Next we prove the *if* part by induction on the level of the rewrite relation of $f^{\#}(t) \xrightarrow[k]{}_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_n(t_1,\ldots,t_n)$. The difference from the proof of Theorem 3.4.9 is which form of rules are applied to $f^{\#}(t) \xrightarrow[k]{}_{\mathcal{I}nv_{\mathsf{PT}}(R)} \mathsf{tp}_n(t_1,\ldots,t_n)$. Here we only show the case of $f^{\#}(f(x_1,\ldots,x_n)) \to \mathsf{tp}_n(x_1,\ldots,x_n)$ because the rest is

also similar to the proof of Theorem 3.4.9. In this case, $f(t_1, \ldots, t_n)$ is again a normal form of R. Therefore, the claim holds obviously. □

Although we extend $\mathcal{I}nv_{\mathsf{PT}}$, $\mathcal{I}nv_{\mathsf{PT}}(R)$ is still an inverse system of R in the sense of Definition 3.1.1.

Chapter 4

Termination of Inverse TRSs

Termination is an important property for computation. In this chapter, we focus on the termination of inverse TRSs which has no extra variable. We show a condition of an input constructor TRS that an innermost-terminating inverse TRS is generated from the input TRS. The condition is associated with the depth of the both-hand sides of each rewrite rule of input TRSs;

- the root symbol of the right-hand side is not a defined symbol, and
- the depth of the left-hand side is less than the number of function symbols which appear on the shortest path from the root symbol to either a variable or a defined symbol.

As we have shown already, a condition that generated systems are TRSs, is the non-erasingness of input TRSs. Since these conditions are related with the syntax of input TRSs only, before the transformation, we can easily know from the syntax of the input whether the output inverse TRS is innermost terminating or not.

4.1 Condition for Innermost Termination

We first give notations on the depth of terms. Let $\mathcal{F}$ be a signature divided into the constructor set $\mathcal{C}$ and the defined-symbol set $\mathcal{D}$; $\mathcal{F} = \mathcal{C} \uplus \mathcal{D}$. Let $\mathcal{F}_{\mathsf{tp}} = \mathcal{F} \cup \{ \mathsf{tp}_i \mid 0 \leq i \leq \mathtt{max}_{f \in \mathcal{F}}(\mathtt{arity}(f)) \}$. The *depth* of term t over $\mathcal{F}_{\mathsf{tp}}$ is denoted by $depth(t)$ and defined as follows:

(1) $depth(x) = 1$ if x is a variable,

(2) $depth(f(t_1,\ldots,t_n)) = 1 + \mathtt{max}_{1\leq i\leq n}(depth(t_i))$, and

(3) $depth(\mathtt{tp}_n(t_1,\ldots,t_n)) = \mathtt{max}_{1\leq i\leq n}(depth(t_i))$.

The number of function symbols in term t, which appear on the shortest path from the root symbol of t to either a variables or a defined symbol, is denoted by $\mathcal{C}depth(t)$ defined as follows:

(a) $\mathcal{C}depth(x) = 1$ if x is a variable,

(b) $\mathcal{C}depth(c(t_1,\ldots,t_n)) = 1 + \mathtt{min}_{1\leq i\leq n}(\mathcal{C}depth(t_i))$ if c is a constructor,

(c) $\mathcal{C}depth(f(t_1,\ldots,t_n)) = 1$ if f is a defined symbol, and

(d) $\mathcal{C}depth(\mathtt{tp}_n(t_1,\ldots,t_n)) = \mathtt{min}_{1\leq i\leq n}(\mathcal{C}depth(t_i))$.

For example, $\mathcal{C}depth\,(c(f(x), c(s(s(y)), b)) = 2$ where f is a defined symbol and b, c, s are constructors.

In the above definitions of *depth* and $\mathcal{C}depth$, we ignore the number of tuple symbols $\mathtt{tp}_i$. Ignoring tuple symbols is essential because they are used as special constructors to represent tuples of terms. The other reason is that any term t is considered as an abbreviation of $\mathtt{tp}_1(t)$.

Definition 4.1.1 (Constructor-increasing) *Let R be a CTRS over a signature $\mathcal{F}$. R is called* constructor-increasing *if every conditional rewrite rule $l \to r \Leftarrow \mathcal{C}ond \in R$ satisfies both of the following:*

(1) the root symbol of r is not a defined symbol, that is, $\mathtt{root}(r) \notin \mathcal{D}_R$, and

(2) $depth(l) \leq \mathcal{C}depth(r) + 1$.

The notion of constructor-increasing is a little relaxed condition that the inverse interpreter *Inversion Algorithm* [14] for ground-convergent constructor TRSs terminates. In the following, we say that a TRS is *NECI* if it is non-erasing and constructor-increasing.

The constructor-increasingness is an important condition to generate terminating inverse TRSs.

Theorem 4.1.2 *Let R be an NECI constructor TRS. Then, the inverse TRS $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R))$ is strongly normalizing with respect to $\xrightarrow[\mathrm{in}]{}_{\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R))}$.*

We will prove this theorem after the following example.

Example 4.1.3 Consider the following TRS and its inverse TRS:

$$R_3 = \{\, \mathsf{add}(0, y) \to y,\ \ \mathsf{add}(\mathsf{s}(x), y) \to \mathsf{s}(\mathsf{add}(x, y)) \,\},$$

$$\begin{aligned} R_6 = \{\ & \mathsf{add}^{\#}(y) \to \mathsf{tp}_2(0, y), \\ & \mathsf{add}^{\#}(\mathsf{s}(z)) \to \mathsf{u}_1(\mathsf{add}^{\#}(z)),\quad \mathsf{u}_1(\mathsf{tp}_2(x, y)) \to \mathsf{tp}_2(\mathsf{s}(x), y)\ \ \}. \end{aligned}$$

The former TRS R_3 is clearly constructor-increasing. Then, it is guaranteed by Theorem 4.1.2 that SIN^6 holds. In fact, it can be easily proved by recursive path ordering that R_6 is terminating with respect to $\to_{R_6}$. That is, R_6 is innermost terminating with respect to $\to_{R_6}$. □

To prove Theorem 4.1.2, we introduce the notion of *quasi-reductivity* and the result associated with the termination of quasi-reductive 3-CTRSs.

We first define quasi-reductivity 3-CTRSs. Let $\succ$ be a well-founded partial ordering which is closed under contexts. Then, the ordering $\succ_{\mathsf{st}}$ is defined as $\succ_{\mathsf{st}} = (\succ \cup \rhd)^+$. It is known that $\succ_{\mathsf{st}}$ is well-founded.

Definition 4.1.4 (Quasi-reductivity [41]) *A deterministic 3-CTRS R over a signature $\mathcal{F}$ is called* quasi-reductive *if there is an extension $\mathcal{F}'$ of the signature $\mathcal{F}$ (so $\mathcal{F} \subseteq \mathcal{F}'$) and a reduction ordering $\succ$ on terms $\mathcal{T}(\mathcal{F}', \mathcal{X})$ which for every rule $l \to r \Leftarrow s_1 \twoheadrightarrow t_1 \wedge \cdots \wedge s_n \twoheadrightarrow t_n \in R$, every $\mathcal{T}(\mathcal{F}', \mathcal{X})$-substitution σ and every i with $0 \le i < n$ satisfies the followings:*

(a) if $s_j\sigma \succeq t_j\sigma$ for $1 \le j \le i$, then $l\sigma \succ_{\mathsf{st}} s_{i+1}\sigma$, and

(b) if $s_j\sigma \succeq t_j\sigma$ for $1 \le j \le n$, then $l\sigma \succ r\sigma$.

Quasi-reductive deterministic 3-CTRSs are introduced in [15] without mentioning that the original signature can be extended.

Quasi-reductive deterministic 3-CTRSs are transformed into innermost terminating TRSs.

Theorem 4.1.5 ([41]) *If R is a quasi-reductive deterministic 3-CTRS, then $\mathbb{U}(R)$ is innermost terminating with respect to $\to_{\mathbb{U}(R)}$ ($SIN^{\to_{\mathbb{U}(R)}}$).*

The original of the above theorem in [41] is for the transformation proposed in [41]. However, the above theorem holds for our transformation $\mathbb{U}$ since $\mathbb{U}$ is a simple extension of that transformation.

According to Theorem 4.1.5, we may show that the generated inverse TRS $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R))$ is quasi-reductive. To show it, we prepare the following proposition and lemmas.

Proposition 4.1.6 *Let R be an NECI constructor TRS. Let $l \to r \in R$, and σ be a substitution. Then, $depth(l\sigma) \leq depth(r\sigma) + 1$.*

Proof. Let t be a term and p be a position of term t. Since positions are non-negative integer sequences, we denote the length of p by $\mathcal{L}en(t)$. Since R is non-erasing, $\mathcal{V}ar(l) = \mathcal{V}ar(r)$ holds. Then, it follows clearly from the definition of constructor-increasingness that every variable $x \in \mathcal{V}ar(l)$ satisfies $\mathcal{L}en(p_1) \leq \mathcal{L}en(p_2) + 1$ for each positions $p_1 \in \mathcal{O}_{\mathcal{X}}(l)$ and $p_2 \in \mathcal{O}_{\mathcal{X}}(r)$ of x such that $l|_{p_1} + 1 \equiv x$ and $r|_{p_2} \equiv x$. Therefore, $depth(l\sigma) \leq depth(r\sigma) + 1$. □

Lemma 4.1.7 *Let R be an NECI constructor TRS over a signature $\mathcal{F}$. Let s, t be terms over $\mathcal{F}$. Then, $s \xrightarrow{*}_R t$ implies $depth(s) \leq depth(t) + 1$.*

Proof. We prove this lemma by induction on the lexicographic combination of the number k of steps of $s \xrightarrow[\text{in}]{k}_R t$, and the structure of the term s. Since the case of $k = 0$ is trivial, we consider the other cases.

Consider the case that $s \equiv f(s_1, \ldots, s_n) \xrightarrow{k}{}^{\varepsilon <}_R f(t_1, \ldots, t_n) \equiv t$. Now we have $s_i \xrightarrow{k_i}_R t_i$ for some $k_i \leq k$. Then, by the induction hypothesis, we have $depth(s_i) \leq depth(t_i) + 1$. Hence, it follows from the definition of *depth* that $depth(f(t_1, \ldots, t_n)) \leq depth(f(t_1, \ldots, t_n))+1$. Therefore, $depth(s) \leq depth(t)+1$.

Consider the remaining case that $s \equiv f(s_1, \ldots, s_n) \xrightarrow{k'}{}^{\varepsilon <}_R f(t_1, \ldots, t_n) \equiv f(u_1, \ldots, u_n)\theta \to^{\varepsilon}_R r\theta \xrightarrow{k''}_R t$, where $f(u_1, \ldots, u_n) \to r \in R$ and $k' + k'' \leq k + 1$. By the induction hypothesis, we have $depth(s_i) \leq depth(t_i) + 1$. It follows from Proposition 4.1.6 that $depth(f(t_1, \ldots, t_n)) \leq depth(r\theta) + 1$. By induction hypothesis, we also have $depth(r\theta) \leq depth(t) + 1$. Hence, we have $depth(s_i) \leq depth(t) + 1$. Therefore, $depth(s) = depth(f(s_1, \ldots, s_n)) \leq depth(t) + 1$. □

Lemma 4.1.8 *Let R be an NECI constructor TRS over a signature $\mathcal{F}$. Let f be a defined symbol of R and $t, t_1, \ldots, t_n$ be terms over $\mathcal{F}$. Then, $f^{\#}(t) \to_{\mathcal{I}nv_{\mathsf{full}}(R)} \mathsf{tp}_n(t_1, \ldots, t_n)$ implies $depth(f^{\#}(t)) > depth(\mathsf{tp}_n(t_1, \ldots, t_n))$.*

Proof. It can be proved by Lemma 3.6.11 that there exist constructor terms $u, u_1, \ldots, u_n$ of R and a substitution σ such that $u\sigma \equiv t$, $u_i\sigma \equiv t_i$ and $f^{\#}(u) \to_{\mathcal{I}nv_{\mathsf{full}}(R)} \mathsf{tp}_n(u_1, \ldots, u_n)$. Then, from Theorem 3.6.17, we have $f(u_1, \ldots, u_n) \xrightarrow[\text{in}]{*}_R u$. Since rewrite relation is closed under substitutions, we have $f(t_1, \ldots, t_n) \xrightarrow[\text{in}]{*}_R t$. Here, from Lemma 4.1.7, we have $depth(t_i) \leq depth(t)$. Therefore, from the definition of *depth*, $depth(f^{\#}(t)) > depth(\mathsf{tp}_n(t_1, \ldots, t_n))$ holds. □

Lemma 4.1.9 *Let R be an NECI constructor TRS over a signature $\mathcal{F}$. Let $\mathscr{T}_{\mathsf{full}}(t) = \langle\!\langle u; Cond \rangle\!\rangle$, $Cond = s_1 \twoheadrightarrow t_1 \wedge \cdots \wedge s_m \twoheadrightarrow t_m$, and σ be a substitution. Then, $Cond(\sigma, \to_{\mathcal{I}nv_{\mathsf{full}}(R)})$ implies $depth(u\sigma) + 1 \geq depth(s_i\sigma)$ for $1 \leq i \leq m$. Especially, $depth(u\sigma) + 1 > depth(s_i\sigma)$ if $\mathtt{root}(t) \in \mathcal{C}_R$.*

Proof. We prove this lemma by induction on the structure of term t. In the case that t is a variable, it is trivial. We consider the other cases.

- Consider the case of $t \equiv c(t_1, \ldots, t_n)$ with $c \in \mathcal{C}_R$. From the definition of $\mathscr{T}_{\mathsf{full}}$, we can assume the followings:

  - $u \equiv c(u_1, \ldots, u_n)$, $Cond = \mathtt{true} \wedge Cond_1 \wedge \cdots \wedge Cond_n$ and $\mathscr{T}_{\mathsf{full}}(t_i) = \langle\!\langle u_i ; Cond_i \rangle\!\rangle$.

  Suppose that $Cond_i = \bigwedge_{j=1}^{n_i} s_{i,j} \twoheadrightarrow t_{i,j}$. It follows from $Cond(\sigma, \to_{\mathcal{I}nv_{\mathsf{full}}(R)})$ that $Cond_i(\sigma, \to_{\mathcal{I}nv_{\mathsf{full}}(R)})$. Then, by the induction hypothesis, $depth(u\sigma)+1 \geq depth(s_{i,j}\sigma)$ for $1 \leq j \leq n_i$. Therefore, the following holds:

  $$depth(c(u_1, \ldots, u_n)) + 1 > depth(u_i\sigma) + 1 \geq depth(s_{i,j}\sigma).$$

- Consider the remaining case, that is, $t \equiv f(t_1, \ldots, t_n)$ with $f \in \mathcal{D}_R$. From the definition of $\mathscr{T}_{\mathsf{full}}$, we can assume that $u \equiv y \in \mathcal{X}$, $Cond = f^{\#}(y) \twoheadrightarrow \mathsf{tp}_n(u_1, \ldots, u_n) \wedge Cond_1 \wedge \cdots \wedge Cond_n$ and $\mathscr{T}_{\mathsf{full}}(t_i) = \langle\!\langle u_i ; Cond_i \rangle\!\rangle$.

  Suppose that $Cond_i = \bigwedge_{j=1}^{n_i} s_{i,j} \twoheadrightarrow t_{i,j}$. It follows from $Cond(\sigma, \to_{\mathcal{I}nv_{\mathsf{full}}(R)})$ that $Cond_i(\sigma, \to_{\mathcal{I}nv_{\mathsf{full}}(R)})$. Let g be a defined symbol of R. Then, by the induction hypothesis, $depth(u_i\sigma) + 1 \geq depth(s_{i,j}\sigma)$ for $1 \leq j \leq n_i$.

  On the other hand, it follows from $Cond(\sigma, \to_{\mathcal{I}nv_{\mathsf{full}}(R)})$ that

  $$f^{\#}(y)\sigma \to_{\mathcal{I}nv_{\mathsf{full}}(R)} \mathsf{tp}_n(u_1, \ldots, u_n)\sigma.$$

  Then, from Lemma 4.1.8, the following holds:

  $$depth(f^{\#}(y\sigma)) > depth(\mathsf{tp}_n(u_1, \ldots, u_n)\sigma).$$

  It follows from the above inequality that $depth(f^{\#}(y\sigma)) = depth(y\sigma)+1$ and $depth(f^{\#}(y\sigma)) \geq depth(\mathsf{tp}_n(u_1, \ldots, u_n)\sigma)+1$. Then, we have $depth(y\sigma)+1 \geq depth(u_i\sigma) + 1$. Therefore, the following holds:

  $$depth(y\sigma) + 1 \geq depth(s_{i,j}\sigma).$$

□

Here, we define an ordering for quasi-reductivity, which is determined by a deterministic 3-TRS and the depth of terms.

Definition 4.1.10 *Let R be a deterministic 3-CTRS. The n-level relation $\hookrightarrow_{n^\prime R}$ on terms over $\mathcal{T}(\mathcal{F},\mathcal{X})$ is inductively defined as*

- $C[s] \hookrightarrow_{0^\prime R} C[t]$ *iff* $s \to_R t$, *and*
- $C[s] \hookrightarrow_{n+1^\prime R} C[t]$ *iff*
  - $s \hookrightarrow_{n^\prime R} t$, *or*
  - *there exist a rule* $\rho : l \to r \Leftarrow Cond \in R$, *a substitution* σ, *a term* $s^\prime$ *over* $\mathcal{F}_{\mathbb{U}}$, *and a term* $s^{\prime\prime}$ *over* $\mathcal{F}$, *such that* $s \equiv l\sigma\ (\to_R^{\varepsilon <} \cup \to_{\mathbb{U}(\rho)}^{\varepsilon})^*\ s^\prime \trianglerighteq s^{\prime\prime}$, $Cond(\sigma, \to_{*^\prime R})$ *and* $s^{\prime\prime} \hookrightarrow_{n^\prime R}^* t$.

The relation $\hookrightarrow_R$ on terms over $\mathcal{F}$ is defined as $\hookrightarrow_R = \bigcup_{i\geq 0} \hookrightarrow_{i^\prime R}$.

The orderings $\succsim_R$ and $\succ_R$ on terms over $\mathcal{T}(\mathcal{F},\mathcal{X})$ are defined as

- $s \succsim_R t$ *iff* $s \hookrightarrow_R^* t$ *and* $depth(s) \geq depth(t)$, *and*
- $s \succ_R t$ *iff* $s \hookrightarrow_R^* t$ *and* $depth(s) > depth(t)$, *or,* $s \hookrightarrow_R^+ t$ *and* $depth(s) \geq depth(t)$.

It is clear that the above ordering $\succ_R$ is a partial ordering which is closed under contexts.

In the following proposition, we show that for a given NECI constructor TRS R, the ordering $\succ_{\mathcal{I}nv_{\mathsf{full}}(R)}$ is a reduction ordering.

Proposition 4.1.11 *Let R be an NECI constructor TRS. Then, the ordering $\succ_{\mathcal{I}nv_{\mathsf{full}}(R)}$ is a reduction ordering.*

Proof. It follows from Lemma 4.1.8 that $\succ_{\mathcal{I}nv_{\mathsf{full}}(R)}$ is closed under substitutions. It also follows from Lemma 4.1.8 that $s \to_{\mathcal{I}nv_{\mathsf{full}}(R)}^{\varepsilon} t$ implies $depth(s) > depth(t)$. Hence, $\succ_{\mathcal{I}nv_{\mathsf{full}}(R)}$ is well-founded. Therefore, $\succ_{\mathcal{I}nv_{\mathsf{full}}(R)}$ is a reduction ordering. □

Proof of Theorem 4.1.2

From the definition of the constructor-increasing, the rules in $\mathcal{I}nv_{\mathsf{full}}(R)$ from a non-erasing constructor TRS R are in form of either of the followings:

(1) $f^{\#}(x) \to \mathsf{tp}_n(p, \ldots, c_n(x, \ldots, x)) \in \mathcal{I}nv_{\mathsf{full}}(R)$ where p is either x or in form of $c(x, \ldots, x)$ for some $c \in \mathcal{C}_R$,

(2) $f^{\#}(C[y_1, \ldots, y_m]) \to \mathsf{tp}_n(w_1, \ldots, w_n) \Leftarrow (\bigwedge_{i=1}^{m} f_i^{\#}(y_i) \twoheadrightarrow t_i) \wedge (\bigwedge_{i=m+1}^{k} f_i^{\#}(z_i) \twoheadrightarrow t_i) \in \mathcal{I}nv_{\mathsf{full}}(R)$ where $\mathcal{C}depth(C[y_1, \ldots, y_m]) > 0$, or

(3) $f^{\#}(f(x_1, \ldots, x_n)) \to \mathsf{tp}_n(x_1, \ldots, x_n) \in \mathcal{I}nv_{\mathsf{full}}(R)$.

We show that in each case above the rule satisfies the condition of quasi-reductivity. The cases (1) and (3) are trivial. We consider the case (2).

Firstly, we show that the condition of Definition 4.1.4 (b) is satisfied by the rule (2). It is clear that if $f_i^{\#}(y_i)\sigma \gtrsim_{\mathcal{I}nv_{\mathsf{full}}(R)} t_i\sigma$ and $f_j^{\#}(z_j)\sigma \gtrsim_{\mathcal{I}nv_{\mathsf{full}}(R)} t_j\sigma$, then we have $\mathcal{C}ond(\sigma, \xrightarrow{*}_{\mathcal{I}nv_{\mathsf{full}}(R)})$ and hence $f^{\#}(C[y_1, \ldots, y_m])\sigma \to_{\mathcal{I}nv_{\mathsf{full}}(R)} \mathsf{tp}_n(w_1, \ldots, w_n)\sigma$. From Lemma 4.1.9, we obtain $f^{\#}(C[y_1, \ldots, y_m])\sigma \succ_{\mathcal{I}nv_{\mathsf{full}}(R)} \mathsf{tp}_n(w_1, \ldots, w_n)\sigma$.

Next, we show that the condition of Definition 4.1.4 (a) is satisfied by the rule (2), that is, that if $s_j\sigma \gtrsim t_j\sigma$ for $1 \leq j \leq i$, then $l\sigma \; (\succ_{\mathcal{I}nv_{\mathsf{full}}(R)})_{\mathsf{st}} \; s_{i+1}\sigma$.

Consider the case of $1 \leq i \leq m$. It is obvious that $depth(f^{\#}(C[y_1, \ldots, y_m])\sigma) > depth(f_i^{\#}(y_i)\sigma)$ for any substitution σ, and hence $f^{\#}(C[y_1, \ldots, y_m])\sigma \succ_{\mathcal{I}nv_{\mathsf{full}}(R)} f_i^{\#}(y_i)\sigma$. Therefore, $f^{\#}(C[y_1, \ldots, y_m])\sigma \; (\succ_{\mathcal{I}nv_{\mathsf{full}}(R)})_{\mathsf{st}} \; f_i^{\#}(y_i)\sigma$

Consider the remaining case, that is, $m + 1 \leq i \leq k$. We prove this case by induction on i. Suppose that $f_j^{\#}(y_j)\sigma \gtrsim_{\mathcal{I}nv_{\mathsf{full}}(R)} t_j$ and $f_j^{\#}(z_j)\sigma \gtrsim_{\mathcal{I}nv_{\mathsf{full}}(R)} t_j$. Since $\mathcal{I}nv_{\mathsf{full}}(R)$ is deterministic, there exists unique j in $1 \leq j < i$ such that z_i appears in t_j.

- Consider the case of $1 \leq j \leq m$. It follows from $f_j^{\#}(y_j)\sigma \gtrsim_{\mathcal{I}nv_{\mathsf{full}}(R)} t_j\sigma$ and Lemma 4.1.8 that $depth(f_j^{\#}(y_j)\sigma) > depth(f_i^{\#}(z_i)\sigma)$ holds. Then, we have $depth(f^{\#}(C[y_1, \ldots, y_m])\sigma) > depth(f_i^{\#}(z_i)\sigma)$. Hence, $f^{\#}(C[y_1, \ldots, y_m])\sigma \succ_{\mathcal{I}nv_{\mathsf{full}}(R)} f_i^{\#}(z_i)\sigma$.

- Consider the remaining case that $m < j < i$. By the induction hypothesis, $f^{\#}(C[y_1, \ldots, y_m])\sigma \succ_{\mathcal{I}nv_{\mathsf{full}}(R)} f_j^{\#}(z_j)\sigma$ holds. Hence, we obtain $depth(f^{\#}(C[y_1, \ldots, y_m])\sigma) > depth(f_j^{\#}(z_j)\sigma)$. Now, it follows from $f_j^{\#}(z_j)\sigma \gtrsim_{\mathcal{I}nv_{\mathsf{full}}(R)} t_j\sigma$ and Lemma 4.1.8 that $depth(f_j^{\#}(z_j)\sigma) > depth(f_i^{\#}(z_i)\sigma)$. Then, we have $depth(f^{\#}(C[y_1, \ldots, y_m])\sigma) > depth(f_i^{\#}(z_i)\sigma)$, and hence $f^{\#}(C[y_1, \ldots, y_m])\sigma \succ_{\mathcal{I}nv_{\mathsf{full}}(R)} f_i^{\#}(z_i)\sigma$.

It is clear that $\succ \subseteq \succ_{\mathsf{st}}$. Therefore, $f^{\#}(C[y_1, \ldots, y_m])\sigma \; (\succ_{\mathcal{I}nv_{\mathsf{full}}(R)})_{\mathsf{st}} \; f_i^{\#}(z_i)\sigma$ □

4.2 Discussion

In the rewrite sequences on the generated TRS $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R))$ starting from $f^{\#}(t)$ where t is a constructor term, every redex is the innermost because of the syntactic properties of $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R))$. Therefore, innermost termination is sufficient for inverse computation by the inverse TRS $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R))$ obtained from the NECI constructor TRS R.

On the other hand, we know the following result which is useful for the inverse computation of left-linear constructor TRSs.

Theorem 4.2.1 ([46]) *Let R be a right-linear overlay TRS. Then, $SIN^{\to_R}$ implies $SN^{\to_R}$.*

According to Theorem 3.6.27, 4.1.2 and 4.2.1, we obtain the following corollary.

Corollary 4.2.2 *If a constructor TRS R is NECI and left-linear, then TRS $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R))$ is strongly normalizing with respect to $\to_{\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R))}$.*

Consider the following TRS and its inverse TRS:

$$R_2 = \{\, \mathsf{half}(0) \to 0,\ \mathsf{half}(\mathsf{s}^2(x)) \to \mathsf{s}(\mathsf{half}(x)) \,\},$$

$$\begin{aligned} R_{12} = \{\ & \mathsf{half}^{\#}(0) \to 0, && \mathsf{half}^{\#}(\mathsf{half}(x)) \to x, \\ & \mathsf{half}^{\#}(\mathsf{s}(y)) \to \mathsf{u}_6(\mathsf{half}^{\#}(y)), && \mathsf{u}_6(\mathsf{s}(x)) \to \mathsf{s}^2(x) \quad \}. \end{aligned}$$

Although R_2 is not constructor-increasing, R_{12} is terminating with respect to $\to_{R_{12}}$. Hence, the condition for the innermost termination of inverse TRSs — constructor-increasing — which have shown shown in Theorem 4.1.2, may be relaxed or improved.

Chapter 5

Computation of Inverse TRSs with Extra Variables

The inverse compilers $\mathbb{U}_{\mathsf{PT}}(\mathcal{I}nv_{\mathsf{PT}}(\cdots))$ and $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(\cdots))$ proposed in Chapter 3, generate EV-TRSs in general. However, rewrite relation of EV-TRSs is infinitely branching and non-terminating, essentially. This fact gives rise to necessity of a simulation method of EV-TRSs for inverse computation using our generated programs.

In this chapter, we show that narrowing [28] can simulate ground rewrite sequences of EV-TRSs by substituting a fresh variable for each extra variable. Focusing on ground rewrite sequences does not lose generality because every rewrite sequence has a corresponding ground rewrite sequence obtained by replacing each variable with a fresh constant introduced in its signature. Since narrowing is finitely branching and narrowing starting ground terms is sometimes terminating to the similar extent as the termination of rewrite relation of TRSs, the serious problems of EV-TRSs — infinitely branching and non-termination — can be solved. To guarantee the termination of inverse computation by EV-TRSs generated by our inverse compilers, we propose two termination proof techniques of narrowing, especially starting from ground terms. To improve efficiency of narrowing derivation, we show that for a linear constructor EV-TRS, every normal form (with respect to narrowing) of a given linear term which is terminating with respect to innermost narrowing, can be obtained by innermost narrowing. We also show that for right-linear EV-TRSs basic narrowing [28] is sufficient to simulate rewrite relation of EV-TRSs.

Narrowing (of TRSs) have been proposed as a method to solve equational unification [28], and there are many results related with narrowing. However,

as far as we know, there is no result on simulation of rewrite sequences using narrowing.

There are some studies on narrowing of CTRSs with extra variables [22, 36, 37]. The targets of their results are 3-CTRSs, in which every extra variable must appear in conditional parts. On the other hand, EV-TRSs belong to 4-CTRSs which are CTRSs with no restrictions, but not to 3-CTRSs. In addition, any inverse CTRS which is an intermediate output of our inverse compilers, is not 3-CTRS but 4-CTRS. Hence, the studies of narrowing on EV-TRSs are valuable.

In studies on normalizing reduction strategies [26, 27, 38, 43], several kinds of EV-TRSs as approximations of TRSs are used.

5.1 Narrowing-Based Simulation of Inverse EV-TRSs

In this section, we show that narrowing which is restricted to substitute a fresh variable for each extra variable can simulate a ground rewrite sequence of EV-TRS if the EV-TRS is right-linear or the rewrite sequence is EV-safe.

In Subsection 5.1.1, we define narrowing of EV-TRSs, which is a natural extension of that of TRSs. Subsection 5.1.2 explains an idea to simulate rewrite sequences of EV-TRSs by narrowing. We first treat general EV-TRSs, and then we discuss the case of inverse EV-TRSs which are generated by the transformation $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(\cdots))$.

In Subsection 5.1.3, we prove the soundness of narrowing sequences for ground rewrite sequences. Conversely, we prove in Subsection 5.1.4 the completeness of narrowing sequences. Since the completeness does not hold for all ground rewrite sequences of any EV-TRS, we only show the completeness for right-linear EV-TRSs, and for EV-safe rewrite sequences (defined later). It is also shown that such a restricted completeness is sufficient for the inverse computation using the inverse EV-TRSs which we generate.

In Subsection 5.1.5, we show the inverse computation by narrowing of the generated inverse EV-TRSs.

5.1.1 Narrowing on EV-TRSs

In this subsection, we define narrowing on EV-TRSs by extending narrowing [28] on TRSs naturally.

Definition 5.1.1 (Narrowing) *Let R be an EV-TRS over a signature $\mathcal{F}$. A term s is said to be* narrowable *into a term t with a substitution σ, a position p of s and a rewrite rule $\rho \in R$, written as $s \;{}_{\sigma|_{\mathcal{V}ar(s|_p)}}\!\!\leadsto_R^{[p,\rho]} t$, if all of the following hold:*

(a) p is a function-symbol position of s, that is, $p \in \mathcal{O}_\mathcal{F}(s)$,

(b) $\sigma = \mathtt{mgu}(s, C[l])$ and $t \equiv (C[r]_p)\sigma$ for some context $C \in \mathscr{C}(\mathcal{F}, \mathcal{X})$ with $s \equiv C[s|_p]_p$, and

(c) $\mathcal{VR}an(\sigma|_{\mathcal{EV}ar(\rho)}) \cap \mathcal{V}ar(C\sigma) = \emptyset$

where $l \to r$ is a renaming of the rule ρ without shared variables for s, that is, $\mathcal{V}ar(l, r) \cap \mathcal{V}ar(s) = \emptyset$. The relation $\leadsto_R$ is called narrowing *of R.*

Note that each extra variable introduces a fresh variable which does not occur in $s\sigma|_{\mathcal{V}ar(s|_p)}$ or $\mathcal{VR}an(\sigma|_{\mathcal{V}ar(l)})$, since the definition of the most general unifiers guarantees that $x\sigma \in \mathcal{X} \setminus \mathcal{VR}an(\sigma)$ for all $x \in \mathcal{EV}ar(\rho)$ and $x \not\equiv y$ implies $x\sigma \not\equiv y\sigma$ for all $x, y \in \mathcal{EV}ar(\rho)$.

The first condition $\sigma = \mathtt{mgu}(s, C[l])$ in the case (b) is different from that in the common definition of narrowing on TRSs ($\sigma = \mathtt{mgu}(s|_p, l)$). This extended condition guarantees $\mathcal{VR}an(\sigma) \cap (\mathcal{V}ar(s) \setminus \mathcal{D}om(\sigma)) = \emptyset$. Since such a condition is assumed implicitly in the common definition of narrowing behind the definition of the most general unifiers, the condition (b) is equivalent to the common definition.

The condition (c) is extra from the common definition. However, this condition is essential because introduced variables should be fresh in the notion of narrowing.

We sometimes write $s \;{}_\delta\!\overset{*}{\leadsto}_R t$ (resp. $s \;{}_\delta\!\overset{n}{\leadsto}_R t$) instead of $\overset{*}{\leadsto}_R$ (resp. $\overset{n}{\leadsto}_R$) if there exists a narrowing sequence $s \equiv t_0 \;{}_{\delta_0}\!\!\leadsto_R t_1 \;{}_{\delta_1}\!\!\leadsto_R \cdots {}_{\delta_{k-1}}\!\!\leadsto_R t_k \equiv t$ ($n = k$) and $\delta = \delta_0 \cdots \delta_{k-1}$ (if $k = 0$ then $\delta = \emptyset$). In this case, we assume that all variables introduced by δ_i are fresh, that is, $\mathcal{VR}an(\delta_i) \cap \mathcal{V}ar(t_0, \ldots, t_{i-1}) = \emptyset$. From the definition of $\leadsto_R$, it is clear that $\leadsto_R$ is finitely branching for every EV-TRS R, and that $\to_R = \leadsto_R$ on ground terms for every TRS R.

Example 5.1.2 Let the following EV-TRS over $\{\, \mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{f}, \mathsf{g}, \mathsf{s}, \mathsf{0} \,\}$:

$$R_{13} = \{\ \mathsf{f}(\mathsf{s}(x), \mathsf{0}) \to x, \quad \mathsf{g}(x) \to \mathsf{c}(\mathsf{f}(y, x), \mathsf{f}(\mathsf{s}(x), y)), \quad \mathsf{a} \to \mathsf{b}\ \}.$$

$$
\begin{array}{llllllll}
 & & \vdots & \vdots & \iddots & & & \\
 & & \nearrow_{R_{14}} & \mathsf{h}(0,\mathsf{a}) & \to_{R_{14}} & \cdots & & \\
\text{(i)} & \mathsf{g}(0) & \to_{R_{14}} & \mathsf{h}(0,0) & \to_{R_{14}} & \mathsf{f}(0,0) & \to_{R_{14}} & \mathsf{s}(0) \\
 & & \searrow_{R_{14}} & \mathsf{h}(0,\mathsf{g}(\mathsf{b})) & \to_{R_{14}} & \cdots & & \\
 & & \vdots & \vdots & \ddots & & &
\end{array}
$$

$$
\text{(ii)} \quad \mathsf{g}(0) \quad \rightsquigarrow_{R_{14}} \quad \mathsf{h}(0,z) \quad \rightsquigarrow_{R_{14}} \quad \mathsf{f}(z,z) \quad \{z \mapsto 0\} \rightsquigarrow_{R_{14}} \quad \mathsf{s}(0)
$$

Figure 5.1: (i) Rewrite, and (ii) narrowing, sequences starting from $\mathsf{g}(0)$.

The following sequences are narrowing sequences starting from the term $\mathsf{g}(0)$:

$$
\mathsf{g}(0) \rightsquigarrow_{R_{13}} \mathsf{c}(\mathsf{f}(y,0),\mathsf{f}(\mathsf{s}(0),y)) \; \{y \mapsto 0\} \rightsquigarrow_{R_{13}} \mathsf{c}(\mathsf{f}(0,0),0)
$$
$$
\{y \mapsto \mathsf{s}(z)\} \rightsquigarrow_{R_{13}} \mathsf{c}(z,\mathsf{f}(\mathsf{s}(0),\mathsf{s}(z))).
$$

On the other hand, $\mathsf{g}(0)$ cannot be narrowable to $\mathsf{c}(\mathsf{f}(z,z),\mathsf{f}(\mathsf{s}(z),z))$. □

To avoid confusion and redundancy, this thesis does not admit $\mathsf{g}(z)$ to be narrowable into $\mathsf{c}(\mathsf{f}(y,x),\mathsf{f}(\mathsf{s}(x),y))$, to $\mathsf{f}(\mathsf{s}(x),0)$ to be narrowable into y, and so on.

5.1.2 Idea for Computation

Here we intuitively explain an idea to simulate rewrite sequences of EV-TRSs by narrowing.

Consider the following EV-TRS over $\{\mathsf{a},\mathsf{b},\mathsf{f},\mathsf{g},\mathsf{h},\mathsf{s},0\}$:

$$
R_{14} = \{\ \mathsf{f}(x,0) \to \mathsf{s}(x),\quad \mathsf{g}(x) \to \mathsf{h}(x,y),\quad \mathsf{h}(0,x) \to \mathsf{f}(x,x),\quad \mathsf{a} \to \mathsf{b}\ \}.
$$

Since arbitrary terms can be substituted for extra variables in rewrite relation, the term $\mathsf{g}(0)$ is rewritten by R_{14} to any term of form $\mathsf{h}(0,t)$, as shown in Figure 5.1 (i), where t represents an arbitrary term. By using a fresh variable z as a representation of all ts and by using narrowing instead of rewrite relation, those rewrite sequences in Figure 5.1 (i) are represented by a single narrowing sequence in Figure 5.1 (ii).

In these way, narrowing can simulate rewrite sequences of EV-TRSs.

5.1.3 Soundness

In this subsection, we show the soundness of narrowing sequences for ground rewrite sequences.

Lemma 5.1.3 *Let R be an EV-TRS. Let s, t be terms and δ be a substitution. Then, $s \overset{*}{\underset{\delta}{\rightsquigarrow}}_R t$ implies $s\delta \overset{*}{\rightarrow}_R t$.*

Proof. We prove this claim by induction on the number k of steps of $s \overset{k}{\underset{\delta}{\rightsquigarrow}}_R t$. Since the case of $k = 0$ is trivial, we assume the following:

$$s \overset{\delta'}{\underset{k-1}{\rightsquigarrow}}_R C[u]_p \overset{\delta''}{\underset{R}{\rightsquigarrow}}^{[p,\rho]} (C[r]_p)\sigma \equiv t$$

where $\rho : l \rightarrow r \in R$, $\mathcal{V}ar(l,r) \cap \mathcal{V}ar(C[u])$, $\sigma = \mathtt{mgu}(C[u], C[l])$ and $\delta'' = \sigma|_{\mathcal{V}ar(u)}$. It follows from $\sigma = \mathtt{mgu}(C[u], C[l])$ and $\delta'' = \sigma|_{\mathcal{V}ar(u)}$ that $u\delta'' \equiv u\sigma \equiv l\sigma$. It also follows from $\mathcal{V}ar(l,r) \cap \mathcal{V}ar(C[u])$ and $\sigma = \mathtt{mgu}(C[u], C[l])$ that $(\mathcal{V}ar(C) \setminus \mathcal{V}ar(u)) \cap \mathcal{D}om(\sigma) = \emptyset$, and hence $C\delta'' \equiv C\sigma$. By the induction hypothesis, we have $s\delta' \overset{*}{\rightarrow}_R C[u]$. Let $\delta = \delta'\delta''$. Then, we have the following sequence:

$$s\delta \equiv s\delta'\delta'' \overset{*}{\rightarrow}_R (C[u])\delta'' \equiv C\delta''[u\delta''] \equiv C\delta''[l\sigma]_p \rightarrow_R^{[p,\rho]} C\delta''[r\sigma]_p \equiv (C[r]_p)\sigma \equiv t.$$

□

Theorem 5.1.4 *Let R be an EV-TRS. Let s be a ground term and t be a term. Then, $s \overset{*}{\rightsquigarrow}_R t$ implies $s \overset{*}{\rightarrow}_R t\sigma$ for all substitutions σ.*

Proof. It follows from Lemma 5.1.3 that $s \overset{*}{\rightarrow}_R t$. Since s is ground and rewrite relation is closed under substitutions, we have $s \overset{*}{\rightarrow}_R t\sigma$ for any substitution σ.

□

The above theorem is similar to the claim shown in the proof of the soundness of narrowing [28].

5.1.4 Completeness

In this subsection, we show the completeness of narrowing sequences for ground rewrite sequences of right-linear EV-TRSs. We also show the completeness for EV-safe rewrite sequences.

We first give a counterexample for full completeness.

Example 5.1.5 Consider the following EV-TRS over $\{ \mathsf{a}, \mathsf{b}, \mathsf{f}, \mathsf{g}, \mathsf{h}, \mathsf{s}, \mathsf{0} \}$ again:

$$R_{14} = \{ \ \mathsf{f}(x, \mathsf{0}) \to \mathsf{s}(x), \quad \mathsf{g}(x) \to \mathsf{h}(x, y), \quad \mathsf{h}(\mathsf{0}, x) \to \mathsf{f}(x, x), \quad \mathsf{a} \to \mathsf{b} \ \}.$$

Then, we have the following rewrite sequence:

$$\mathsf{g}(\mathsf{0}) \to_{R_{14}} \mathsf{h}(\mathsf{0}, \mathsf{a}) \to_{R_{14}} \mathsf{f}(\mathsf{a}, \mathsf{a}) \to_{R_{14}} \mathsf{f}(\mathsf{a}, \mathsf{b}).$$

Unfortunately, the above sequence cannot be simulated by narrowing because the initial term $\mathsf{g}(\mathsf{0})$ cannot be narrowable (even by many steps) to any term which has an instance $\mathsf{f}(\mathsf{a}, \mathsf{b})$ (see Figure 5.1 (ii)). □

The reason why narrowing cannot simulate the rewrite sequence in the above example seems that the term a in $\mathsf{h}(\mathsf{0}, \mathsf{a})$, which is introduced by the extra variable y of the second rule of R_{14}, is copied by the third rule in the second step $\mathsf{h}(\mathsf{0}, \mathsf{a})$ $\to_{R_{14}}$ $\mathsf{f}(\mathsf{a}, \mathsf{a})$ and then one of a is rewritten by the fourth rule. To avoid such a reduction, any term introduced by means of extra variables must not copied or must not be rewritten. Therefore, either conditions that a given EV-TRS is right-linear or that any redex introduced by means of extra variables is not rewritten in a given rewrite sequence, is necessary for the completeness.

Here we prepare the proposition associated with the form of a term t with $t\theta \equiv C[u]$.

Proposition 5.1.6 *Let t be a term over a signature $\mathcal{F}$, θ be a substitution, C be a one-hole context and u be a term over $\mathcal{F}$. If $t\theta \equiv C[u]_p$, then t can be represented as either of the following:*

(1) $t \equiv C'[x]_{p'}$, $x\theta \equiv C''[u]_q$, $p = p'q$ and $(C'\theta)[C''[x\theta]_q]_{p'} \equiv C[u]_p$ for some one-hole contexts C', C'' and some variable x in t, or

(2) $t \equiv C'[v]_p$, $v\theta \equiv u$ and $C'\theta \equiv C$ for some one-hole context C' and some term v.

Proof. This proposition is trivial from Figure 5.2. □

5.1.4.1 Completeness for Right-Linear Systems

We first show the completeness for right-linear EV-TRSs.

As the first step, we analyze the case of one-step reduction.

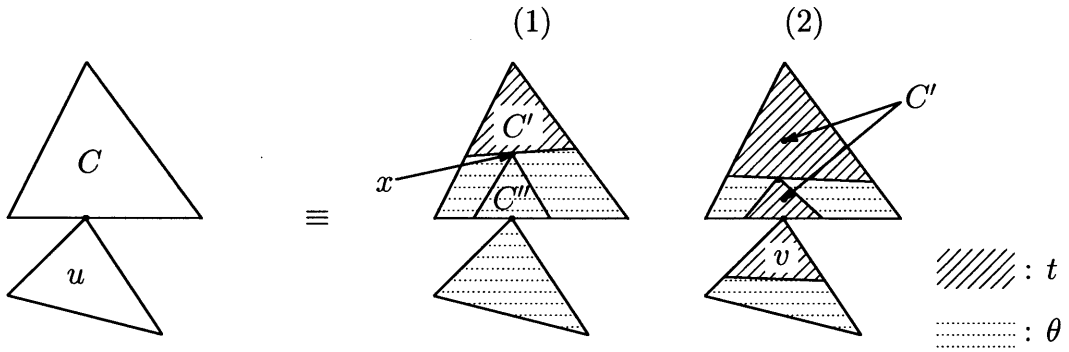

Figure 5.2: Forms of t when $t\theta \equiv C[u]_p$.

Lemma 5.1.7 *Let R be an EV-TRS and suppose that $s\theta \to_R^{[p,\rho]} t$. If $s|_p$ is not a variable, then there exist a term u and a substitution σ such that $s \rightsquigarrow_R^{[p,\rho]} u$ and $u\sigma \equiv t$.*

Proof. Let $\rho : l \to r \in R$. Then, there exists a one-hole context C and a substitution θ' such that $s\theta \equiv C[l\theta']_p$ and $t \equiv C[r\theta']_p$. From Proposition 5.1.6 and the non-variable term $s|_p$, there exist a one-hole context C' and a term v such that $s \equiv C'[v]_p$, $v\theta \equiv l\theta'$ and $C'\theta \equiv C$. Now we can assume that $\mathcal{V}ar(l,r) \cap \mathcal{V}ar(s) = \emptyset$ and $\mathcal{D}om(\theta) \cap \mathcal{D}om(\theta') = \emptyset$ without loss of generality, and hence $\mathcal{V}ar(v) \cap \mathcal{V}ar(l) = \emptyset$.

Since $v\theta \equiv l\theta'$ and $\mathcal{V}ar(v) \cap \mathcal{V}ar(l) = \emptyset$, there exists the most general unifier of v and l. Then, we can suppose that $\delta = \mathtt{mgu}(C'[v], C'[l])$, and hence

$$s \equiv C'[v]_p \, {}_{\delta|_{\mathcal{V}ar(v)}}\!\!\rightsquigarrow_R^{[p,\rho]} (C'[r]_p)\delta.$$

It follows from $\mathcal{D}om(\theta) \cap \mathcal{D}om(\theta') = \emptyset$ that $\theta \cup \theta'$ is a substitution and it is a unifier of v and l. Since δ is the most general unifier of $C'[v]$ and $C'[l]$, $\delta \lesssim (\theta \cup \theta')|_{\mathcal{V}ar(v,l)}$ holds. Therefore, it follows from $C[r\sigma]_p \equiv (C'[r])((\theta \cup \theta')|_{\mathcal{V}ar(v,l)})$ that there exists a substitution σ' such that $((C'[r]_p)\delta)\,\sigma' \equiv C[r\sigma]$. □

The following propositions are associated with the linearity of terms in one-step narrowing.

Proposition 5.1.8 *Let $\mathcal{F}$ be a signature, s be a linear term over $\mathcal{F}$ and t be a term over $\mathcal{F}$ such that $\mathcal{V}ar(s) \cap \mathcal{V}ar(t) = \emptyset$. If s and t are unifiable, then every unifier σ of s and t satisfies that*

(a) $\mathcal{V}ar(x\sigma) \cap \mathcal{V}ar(y\sigma) = \emptyset$ for each $x, y \in \mathcal{D}om(\sigma|_{\mathcal{V}ar(t)})$ such that $x \not\equiv y$, and

(b) every term $u \in \mathcal{R}an(\sigma|_{\mathcal{V}ar(t)})$ *is linear.*

Proof. We first extend the definition of unifiers for sets of pairs of terms. Let S be a finite set of pairs of terms. We say that a substitution θ is a unifier of S if $s_i\theta \equiv t_i\theta$ for every pair $(s_i, t_i) \in S$. A unifier θ' of S is said to be more general if $\theta' \lesssim \theta$ for all unifiers θ of S. The size $|S|$ of S is defined as $|S| = \sum_{(s_i,t_i)\in S}(|s_i| + |t_i|)$.

It is clear that a substitution δ is a (most general) unifier of a singleton set $\{ (f(s_1, \ldots, s_n), f(t_1, \ldots, t_n)) \}$ iff δ is a (most general) unifier of a set $\{ (s_i, t_i) \mid 1 \leq i \leq n \}$. Hence, the following claim can be proved by induction on the size of $|S|$:

> Let S be a finite set $\{ (s_i, t_i) \mid 1 \leq i \leq n \}$ of pairs of terms such that
>
> (1) s_i is linear for every pair $(s_i, t_i) \in S$, and
>
> (2) $\mathcal{V}ar(s_i) \cap \mathcal{V}ar(s_j) = \emptyset$ for each $(s_i, t_i), (s_j, t_j) \in S$ such that $i \neq j$.
>
> Suppose that there exists the most general unifier σ of S. Then, σ satisfies that
>
> (i) $\mathcal{V}ar(x\sigma) \cap \mathcal{V}ar(y\sigma) = \emptyset$ for each $x, y \in \mathcal{D}om(\sigma|_{\mathcal{V}ar(t_1,\ldots,t_n)})$ such that $x \not\equiv y$, and
>
> (ii) every term $u \in \mathcal{R}an(\sigma|_{\mathcal{V}ar(t_1,\ldots,t_n)})$ is linear.

Therefore, this proposition holds. □

Proposition 5.1.9 *Let R be a right-linear EV-TRS. Let s be a linear term. If $s \rightsquigarrow_R t$, then t is linear.*

Proof. Suppose that $s \rightsquigarrow_R^{[p,\rho]} t$, $\rho : l \to r \in R$, $s \equiv C[u]_p$, and $\sigma = \mathtt{mgu}(s, C[l]_p)$ such that $\mathcal{V}ar(s) \cap \mathcal{V}ar(l, r) = \emptyset$. Then, we have $t \equiv (C[r]_p)\sigma$ from the definition of $\rightsquigarrow_R$. We can assume without loss of generality that $\mathcal{D}om(\sigma) \subseteq \mathcal{V}ar(u, l, r)$. It follows from the linearity of s that $\mathcal{V}ar(C) \cap \mathcal{V}ar(u) = \emptyset$. Then, we have $\mathcal{V}ar(C) \cap \mathcal{D}om(\sigma) = \emptyset$, and hence $C\sigma \equiv C$. This implies that $t \equiv C[r\sigma]_p$. Since u is linear, σ satisfies all of the followings from Proposition 5.1.8:

- $\mathcal{V}ar(x\sigma) \cap \mathcal{V}ar(y\sigma) = \emptyset$ for each $x, y \in \mathcal{D}om(\sigma|_{\mathcal{V}ar(l)})$ such that $x \not\equiv y$, and
- every term $t \in \mathcal{R}an(\sigma|_{\mathcal{V}ar(l)})$ is linear.

It follows from the definition of rewrite rules that $\mathcal{V}ar(r) \subseteq \mathcal{V}ar(l) \cup \mathcal{EV}ar(\rho)$. It also follows from the definition of narrowing that $\mathcal{R}an(\sigma|_{\mathcal{EV}ar(\rho)})$ is a set of fresh variables which are not appearing in $\mathcal{VR}an(\sigma|_{\mathcal{V}ar(l)})$. Then, the followings holds:

- $\mathcal{V}ar(x\sigma) \cap \mathcal{V}ar(y\sigma) = \emptyset$ for each $x, y \in \mathcal{D}om(\sigma|_{\mathcal{V}ar(r)})$ such that $x \not\equiv y$, and
- every term $u' \in \mathcal{R}an(\sigma|_{\mathcal{V}ar(r)})$ is linear.

Since r is linear, $r\sigma$ is also linear. It follows from the definition of narrowing that $\mathcal{VR}an(\sigma) \cap \mathcal{V}ar(C) = \emptyset$. Then, we have $\mathcal{V}ar(C) \cap \mathcal{V}ar(r\sigma) = \emptyset$. Therefore, t $(\equiv C[r\sigma]_p)$ is linear. □

Now we show the completeness for right-linear EV-TRSs.

Theorem 5.1.10 *Let R be a right-linear EV-TRS. Let s, t be ground terms. Then, $s \xrightarrow{*}_R t$ implies $s \overset{*}{\rightsquigarrow}_R u$ and $t \equiv u\theta$ for some linear term u and some substitution σ.*

Proof. We prove this claim by induction on the number k of steps of $s \xrightarrow{k}_R t$.

Since the case of $k = 0$ is trivial, we assume the following:

$$s \xrightarrow{k-1}_R C[l\sigma]_p \to_R^{[p,\rho]} C[r\sigma]_p \equiv t$$

where $\rho : l \to r \in R$. By the induction hypothesis, there exist a linear term u and a substitution θ such that $s \overset{*}{\rightsquigarrow}_R u$ and $C[l\sigma]_p \equiv u\sigma$.

From Proposition 5.1.6, u and $C[l\sigma]$ satisfy either of the following cases:

(1) $u \equiv C'[x]_{p'}$, $x\theta \equiv C''[l\sigma]_q$, $p = p'q$ and $(C'\theta)[C''[x\theta]_q]_{p'} \equiv C[u]_p$ for some one-hole contexts C', C'' and some variable x in u, or

(2) $u \equiv C'[v]_p$, $v\theta \equiv l\sigma$ and $C'\theta \equiv C$ for some one-hole context C' and some term v.

In the sequel, we consider the both cases above.

- Consider the first case (1). It follows from the linearity of u that $x \notin \mathcal{V}ar(C')$. Let $\theta' = \{x \mapsto C''[r\sigma]\} \cup \theta|_{\mathcal{V}ar(C')}$. Then, θ' is clearly a substitution and $u\theta' \equiv C[r\sigma]$. Therefore, $s \overset{*}{\rightsquigarrow}_R u$ and $u\theta' \equiv t$.
- Consider the second case (2). Since v is not a variable, from Lemma 5.1.7, there exist a term u' and a substitution θ' such that $u \equiv C'[v]_p \rightsquigarrow_R^{[p,\rho]} (C'[r]_p)\delta \equiv u'$ and $u'\theta' \equiv t$. It follows from Proposition 5.1.9 that u' is linear. Therefore, $s \overset{*}{\rightsquigarrow}_R u'$, u' is linear and $u'\theta' \equiv t$.

□

5.1.4.2 EV-safe Reduction Sequences

We here introduce EV-safe rewrite sequences of EV-TRSs.

Intuitively speaking, a rewrite sequence is *EV-safe* if any redex which is rewritten in the sequence is not introduced by means of extra variables. One of the typical instances of EV-safe rewrite sequences are rewrite sequences constructed by substituting normal forms for extra variables. In the sequel, we give a precise definition of EV-safe rewrite sequences of EV-TRSs.

Let t be a term over a signature $\mathcal{F}$ and x be a variable. The set of all positions of x in t is denoted by $\mathcal{O}_x(t)$. Let P_0, P_1 be sets of positions of t: $P_0, P_1 \subseteq \mathcal{O}(t)$. We write $P_0 \leq P_1$ if for all position $q \in P_1$ there exists a position $p \in P_0$ such that $p \leq q$. The set $P_0\backslash_p$ is defined as $P_0\backslash_p = \{\, q \mid pq \in P_0 \,\}$. The minimal set $\lceil P_0 \rceil$ of P_0 is defined as follows:

$$\lceil P_0 \rceil = \{\, p \in P_0 \mid \neg(\exists q \in P_0, q < p) \,\}.$$

For example, $\lceil \{11, 1, 2\} \rceil = \{1, 2\}$. P_0 is said to be *minimal* if $P_0 = \lceil P_0 \rceil$. We define the minimal set of union, and intersection, of P_0 and P_1 as follows:

$$P_0 \sqcup P_1 = \lceil P_0 \cup P_1 \rceil,$$

$$P_0 \sqcap P_1 = \bigcup_{i=0,1} \{\, p \in \lceil P_i \rceil \mid (\exists q \in P_{1-i}, q \leq p) \,\}.$$

For example, $\{11, 22\} \sqcup \{112, 2, 31\} = \{11, 2, 31\}$, $\{11, 22\} \sqcap \{112, 2, 31\} = \{112, 2, 31\}$.

We give a notion of the transition of positions at one-step rewriting, adding the positions of extra variables to the initial set of positions.

Definition 5.1.11 *Let a rewrite rule $\rho : l \to r$, P be a minimal set of positions, Q be a set of positions and p be a position. We write $P \Rrightarrow^{[p,\rho]} Q$ if there is no position q with $q \leq p$, and $Q = \lceil Q' \rceil$ for some position set Q' defined as follows:*

$$Q' = (P \setminus \{q \mid p \leq q\}) \cup \left(\bigsqcup_{x \in \mathcal{EV}ar(\rho)} \{\, pq \mid q \in \mathcal{O}_x(r) \,\} \right) \cup \left(\bigsqcup_{x \in \mathcal{V}ar(l)} \left\{ pqw \mid q \in \mathcal{O}_x(r), w \in \left(\textstyle\prod_{q' \in \mathcal{O}_x(l)} P\backslash_{pq'} \right) \right\} \right).$$

It is clear that if $P \subseteq \mathcal{O}(C[l\sigma]_p)$ for some context C and some substitution σ then $Q \subseteq \mathcal{O}(C[r\sigma]_p)$.

The notion of the transition above is similar to that of descendants that follows redex positions [27]. This notion is also similar to *basic positions* using in the definition of basic narrowing (see Subsection 5.3.2). The notation $P \Rrightarrow^{[p,\rho]} Q$ shows the transition in the one-step rewriting at the position p by the rule ρ. For example, consider $P = \{12, 2\} \subseteq \mathcal{O}(\mathsf{f}(\mathsf{h}(0, \mathsf{s}(y)), 0))$ and $\rho : \mathsf{h}(0, x) \to \mathsf{f}(x, x)$. Then we have $P \Rrightarrow^{[1,\rho]} \{11, 12, 2\}$ as shown in the following sequence:

$$\mathsf{f}(\mathsf{h}(0, \underline{\mathsf{s}(y)}), \underline{0}) \to^{[1,\rho]}_{R_{14}} \mathsf{f}(\mathsf{f}(\underline{\mathsf{s}(y)}, \underline{\mathsf{s}(y)}), \underline{0}).$$

Next we define EV-safe rewrite sequences.

Definition 5.1.12 (EV-safety) *Let R be an EV-TRS, t_0 be a term, $\rho_i : l_i \to r_i \in R$ and $P_0 \subseteq \mathcal{O}(t_0)$ be a minimal set. A rewrite sequence $t_0 \to^{[p_0,\rho_0]}_R t_1 \to^{[p_1,\rho_1]}_R \cdots$ is called* EV-safe *with respect to P_0 if $P_0 \Rrightarrow^{[p_0,\rho_0]} P_1 \Rrightarrow^{[p_1,\rho_1]} \cdots$ holds for some $P_1, P_2, \ldots$. In this case, we write $P_0 : t_0 \to^{[p_0,\rho_0]}_R P_1 : t_1 \to^{[p_1,\rho_1]}_R \cdots$, and positions in P_i are called* EV-safe positions *of t_i. Especially, the above rewrite sequence is simply called* EV-safe *if $P_0 = \emptyset$. We say that $s \overset{*}{\to}_R t$ has an EV-safe path with respect to a set P of positions in s if there exists an EV-safe rewrite sequence $s \to_R \cdots \to_R t$ with respect to P.*

Example 5.1.13 Consider the following EV-TRS over $\{\, \mathsf{a}, \mathsf{b}, \mathsf{f}, \mathsf{g}, \mathsf{h}, \mathsf{s}, 0 \,\}$ again:

$$R_{14} = \{\; \mathsf{f}(x, 0) \to \mathsf{s}(x), \quad \mathsf{g}(x) \to \mathsf{h}(x, y), \quad \mathsf{h}(0, x) \to \mathsf{f}(x, x), \quad \mathsf{a} \to \mathsf{b} \;\}.$$

Then, $\mathsf{g}(0) \overset{*}{\to}_{R_{14}} \mathsf{s}(0)$ has an EV-safe path with respect to $\{1\}$ because the following EV-safe rewrite sequence exists:

$$\{1\} : \mathsf{g}(0) \to_{R_{14}} \{1, 2\} : \mathsf{h}(0, 0) \to_{R_{14}} \{1, 2\} : \mathsf{f}(0, 0) \to_{R_{14}} \{1\} : \mathsf{s}(0).$$

On the other hand, the rewrite sequence $\mathsf{g}(0) \to_{R_{14}} \mathsf{h}(0, \mathsf{a}) \to_{R_{14}} \mathsf{h}(0, \mathsf{b})$ is not EV-safe because the term a introduced by an extra variable is rewritten. □

5.1.4.3 Completeness for EV-Safe Rewrite Sequences

We here show the completeness of narrowing sequences for EV-safe rewrite sequences.

Firstly, we clarify the transition of EV-safe positions in the one-step narrowing.

Lemma 5.1.14 *Let R be an EV-TRS and suppose that $\rho : l \to r \in R$, $s \rightsquigarrow^{[p,\rho]}_R t$ and a set $P \leq \mathcal{O}_{\mathcal{X}}(s)$ is minimal. Moreover, suppose that p is not a variable position and $P \Rrightarrow^{[p,\rho]} Q$. Then, $Q \leq \mathcal{O}_{\mathcal{X}}(t)$.*

Proof. (Sketch.) We first decompose Q as follows:

- $Q = Q_1 \cup Q_2 \cup Q_3$,
- $Q_1 = P \setminus \{q \mid p \leq q\}$,
- $Q_2 = \bigsqcup_{x \in \mathcal{EV}ar(\rho)} \{\, pq \mid q \in \mathcal{O}_x(r) \,\}$,
- $Q_3 = \bigsqcup_{x \in \mathcal{V}ar(l)} \left\{ pqw \mid q \in \mathcal{O}_x(r), w \in \left(\prod_{q' \in \mathcal{O}_x(l)} P \backslash pq' \right) \right\}$.

Assume that $s \equiv C[u] \rightsquigarrow_R^{[p,\rho]} (C[r])_p)\sigma \equiv t$. Then, $\mathcal{O}_\mathcal{X}(t)$ is decomposed as follows:

- $\mathcal{O}_\mathcal{X}(t) = O_1 \cup O_2 \cup O_3$,
- $O_1 = \mathcal{O}_\mathcal{X}(C\sigma)$,
- $O_2 = \bigcup_{x \in \mathcal{EV}ar(\rho)} \{\, pq \mid r|_q \equiv x \,\}$,
- $O_3 = \bigcup_{x \in \mathcal{V}ar(r) \setminus \mathcal{EV}ar(\rho)} \left\{ pqw \mid q \in \mathcal{O}_x(r), w \in \left(\bigcup_{y \in \mathcal{V}ar(x\sigma)} \mathcal{O}_y(x\sigma) \right) \right\}$.

Here $Q_i \leq O_i$ can be easily proved. Therefore, $Q \leq \mathcal{O}_\mathcal{X}(t)$ holds. □

Next, we give the following lemma which shows a more general case of the completeness for EV-safe rewrite sequences.

Lemma 5.1.15 *Let R be an EV-TRS over. Let s, t be terms and σ be a substitution. If $s\sigma \xrightarrow{*}_R t$ has an EV-safe path with respect to $\mathcal{O}_\mathcal{X}(s)$ then $s \overset{*}{\rightsquigarrow}_R u$ and $t \equiv u\theta$ for some term u and some substitution θ.*

Proof. We prove this claim by induction on the number k of steps of $s\sigma \xrightarrow{k}_R t$. Since the case of $k = 0$ is trivial, we consider the case of $k > 0$. Suppose that $s\sigma \xrightarrow{*}_R t$ has an EV-safe path with respect to $\mathcal{O}_\mathcal{X}(s)$. Then, there exists an EV-safe rewrite sequence with respect to $\mathcal{O}_\mathcal{X}(s)$. Now we assume the following EV-safe sequence with respect to $\mathcal{O}_\mathcal{X}(s)$:

$$s\sigma \equiv C[l\delta]_p \rightarrow_R^{[p,\rho]} C[r\delta]_p \xrightarrow{k-1}_R t$$

where $\rho : l \to r \in R$. Since the above sequence is EV-safe, p is not a variable position. Let $P = \mathcal{O}_\mathcal{X}(s)$. Then, there is an EV-safe position set Q such that $P \Rightarrow^{[p,\rho]} Q$ since p is not a variable position. Since the above sequence is EV-safe, $C[r\delta]_p \xrightarrow{k-1}_R t$ is also EV-safe with respect to Q.

On the other hand, from Lemma 5.1.7 and 5.1.14, there exist a term u and a substitution σ' such that $s \rightsquigarrow_R^{[p,\rho]} u$, $u\sigma' \equiv C[r\delta]_p$, and $Q \leq \mathcal{O}_{\mathcal{X}}(u)$.

Since $C[r\delta]_p \xrightarrow{k-1}_R t$ is also EV-safe with respect to Q, $C[r\delta]_p \xrightarrow{k-1}_R t$ has an EV-safe path with respect to Q. It follows from $Q \leq \mathcal{O}_{\mathcal{X}}(u)$ that $C[r\delta]_p \xrightarrow{k-1}_R t$ has an EV-safe path with respect to $\mathcal{O}_{\mathcal{X}}(u)$. Now we have $u\sigma \xrightarrow{k-1}_R t$ and this has an EV-safe path with respect to $\mathcal{O}_{\mathcal{X}}(u)$. Then, by the induction hypothesis, there exists a term u' and a substitution θ such that $u \overset{*}{\rightsquigarrow}_R u'$ and $t \equiv u'\theta$. Therefore, we have $s \overset{*}{\rightsquigarrow}_R u'$ and $u'\theta \equiv t$. □

At last, we show the completeness for ground EV-safe rewrite sequences.

Theorem 5.1.16 *Let R be an EV-TRS. Let s be a ground term and t be a term. If $s \overset{*}{\rightarrow}_R t$ has an EV-safe path then $s \overset{*}{\rightsquigarrow}_R u$ and $t \equiv u\theta$ for some term u and some substitution θ.*

Proof. A direct consequence of Lemma 5.1.15. □

The above theorem is a more general variant of the completeness lemma of narrowing [28].

5.1.5 Narrowing of Inverse EV-TRSs

In this subsection, we discuss the narrowing of inverse EV-TRSs generated by our inverse compiler $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(\cdots))$. We first show that every rewrite sequence considered as inverse computation has an EV-safe path. Next we give a theorem that narrowing can find general solutions of inverse computation. Finally, we show a example of rewrite sequences which means inverse computation.

Before showing that every inverse computation sequence has an EV-safe path, we give the following theorem for general EV-TRSs:

Theorem 5.1.17 *Let R be a strictly non-erasing and strict constructor 4-CTRS over a signature $\mathcal{F}$. Suppose that every conditional rewrite rule $l \to r \Leftarrow s_1 \twoheadrightarrow t_1 \wedge \cdots \wedge s_n \twoheadrightarrow t_n \in R$ satisfies both of the followings:*

(a) r and $t_1, \ldots, t_n$ are non-variable constructor terms of R, and

(b) for $1 \leq i \leq n$, the root symbol of s_i is a defined symbol of R and every proper subterm of s_i is a constructor term of R.

Let $l \to r \Leftarrow s_1 \twoheadrightarrow t_1 \wedge \cdots \wedge s_n \twoheadrightarrow t_n$ be a conditional rewrite rule in R and σ be an $NF^{\to_R}(\mathcal{F}, \mathcal{X})$-substitution. If $l\sigma \to_R r\sigma$, then $l\sigma \xrightarrow{*}_{\mathbb{U}(R)} r\sigma$ holds and it has an EV-safe path.

Proof. We prove this claim by induction on the level k of the rewrite relation of $l\sigma \xrightarrow[k]{}_R r\sigma$.

From Lemma 3.6.14 (d), there exists an $NF^{\to_R}(\mathcal{F}, \mathcal{X})$-substitution σ' such that $\mathit{Cond}(\sigma', \xrightarrow[k-1]{}_R)$.

Here let $\mathit{Cond} = s_1 \twoheadrightarrow t_1 \wedge \cdots \wedge s_n \twoheadrightarrow t_n$. Then, $\mathbb{U}(\rho)$ is defined as the same with that in Definition 3.6.20.

On the other hand, it follows from $\mathit{Cond}(\sigma', \xrightarrow[k-1]{}_R)$ that $s_i\sigma' \xrightarrow[k-1]{}_R t_i\sigma'$. Then, there are a rule $l_i \to r_i \Leftarrow \mathit{Cond}_i \in R$ and a substitution θ_i such that $s_i\sigma' \equiv l_i\theta_i \xrightarrow[k-1]{}_R r_i\theta_i \equiv t_i\sigma'$ and $\mathit{Cond}_i(\theta_i, \xrightarrow[k-2]{*}_R)$. Since σ is an $NF^{\to_R}(\mathcal{F}, \mathcal{X})$-substitution, from Lemma 3.6.14 (d), we can assume without loss of generality that θ_i is also an $NF^{\to_R}(\mathcal{F}, \mathcal{X})$-substitution. Then, by the induction hypothesis, $l_i\theta_i \xrightarrow{*}_{\mathbb{U}(R)} r_i\theta_i$ has an EV-safe path.

Since σ' is an $NF^{\to_R}(\mathcal{F}, \mathcal{X})$-substitution and $s_i\sigma \equiv l_i\theta_i \xrightarrow{*}_{\mathbb{U}(R)} r_i\theta_i \equiv t_i\sigma$, we have the following rewrite sequence of $\mathbb{U}(R)$:

$$
\begin{aligned}
l\sigma \equiv l\sigma' \ &\to_{\mathbb{U}(R)} \mathsf{u}_1^\rho(s_1, \mathcal{V}list(X_1))\sigma' \xrightarrow{*}_{\mathbb{U}(R)} \mathsf{u}_1^\rho(t_1, \mathcal{V}list(X_1))\sigma' \\
&\to_{\mathbb{U}(R)} \mathsf{u}_2^\rho(s_2, \mathcal{V}list(X_2))\sigma' \xrightarrow{*}_{\mathbb{U}(R)} \mathsf{u}_2^\rho(t_2, \mathcal{V}list(X_2))\sigma' \\
&\cdots \\
&\to_{\mathbb{U}(R)} \mathsf{u}_n^\rho(s_n, \mathcal{V}list(X_n))\sigma' \xrightarrow{*}_{\mathbb{U}(R)} \mathsf{u}_n^\rho(t_n, \mathcal{V}list(X_n))\sigma' \\
&\to_{\mathbb{U}(R)} r\sigma' \equiv r\sigma.
\end{aligned}
$$

Since $s_i\sigma \xrightarrow{*}_{\mathbb{U}(R)} t_i\sigma$ has an EV-safe path, the above sequence can be considered to be EV-safe. Therefore, $l\sigma \xrightarrow{*}_{\mathbb{U}(R)} r\sigma$ has an EV-safe path. □

From Proposition 3.6.9 (a), it is known that the generated CTRS $\mathcal{I}nv_{\mathsf{full}}(R)$ is a strictly non-erasing and strict constructor 4-CTRS. From the definitions of $\mathscr{T}_{\mathsf{full}}$ and $\mathcal{I}nv\mathcal{R}ule_{\mathsf{full}}$, it is clear that the rule $\mathcal{I}nv\mathcal{R}ule_{\mathsf{full}}(\rho)$ satisfies the conditions (a) and (b) in Theorem 5.1.17. Therefore, the following theorem holds obviously:

Theorem 5.1.18 *Let R be a constructor TRS over a signature $\mathcal{F}$. Let f be a defined symbol of R and $t, t_1, \ldots, t_n$ be normal forms with respect to $\to_R$. If $f^\#(t) \to_{\mathcal{I}nv_{\mathsf{full}}(R)} \mathsf{tp}_n(t_1, \ldots, t_n)$, then $f^\#(t) \xrightarrow{*}_{\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R))} \mathsf{tp}_n(t_1, \ldots, t_n)$ holds and it has an EV-safe path.*

Now we obtain the following theorem, by concluding from the correctness of $\mathcal{I}nv_{\mathsf{full}}$, the soundness and completeness of $\mathbb{U}$ for inverse computation, the

soundness and completeness of narrowing for EV-safe rewrite sequences, and EV-safety of inverse computation sequences:

Theorem 5.1.19 *Let R be a constructor TRS over a signature $\mathcal{F}$. Let f be a defined symbol of R and t be a ground normal form of R.*

(a) For all normal forms $t_1, \ldots, t_n$ with respect to $\to_R$, $f(t_1, \ldots, t_n) \xrightarrow[\text{in}]{*}_R t$ implies $f^{\#}(t) \overset{*}{\leadsto}_{\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R))} \mathsf{tp}_n(u_1, \ldots, u_n)$ and $t_i \equiv u_i\theta$ for some terms $u_1, \ldots, u_n$ and some substitution θ.

(b) For all normal forms $u_1, \ldots, u_n$ with respect to $\to_R$, $f^{\#}(t) \overset{*}{\leadsto}_{\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R))} \mathsf{tp}_n(u_1, \ldots, u_n)$ implies $f(u_1\theta, \ldots, u_n\theta) \xrightarrow[\text{in}]{*}_R t$ for all substitution θ.

Proof. The first claim (a) is a direct consequence of Theorem 3.6.17, 5.1.17 and 5.1.16. From Theorem 5.1.4, 3.6.23 and 3.6.17, $f(u_1, \ldots, u_n) \xrightarrow[\text{in}]{*}_R t$ holds. Since t is ground and the rewrite relation $\to_R$ is closed under substitutions, it is clear that $f(u_1, \ldots, u_n)\theta \xrightarrow[\text{in}]{*}_R t$ for all substitution θ. □

Similarly to Corollary 3.6.19, the following corollary can be obtained from the above theorem:

Corollary 5.1.20 *Let R be a convergent constructor TRS over a signature $\mathcal{F}$. Let f be a defined symbol of R and t be a ground normal form of R.*

(a) For all normal forms $t_1, \ldots, t_n$ with respect to $\to_R$, $f(t_1, \ldots, t_n) \xrightarrow{*}_R t$ implies $f^{\#}(t) \xrightarrow{*}_{\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R))} \mathsf{tp}_n(u_1, \ldots, u_n)$ and $t_i \equiv u_i\theta$ for some terms $u_1, \ldots, u_n$ and some substitution θ.

(b) For all normal forms $u_1, \ldots, u_n$ with respect to $\to_R$, $f^{\#}(t) \xrightarrow{*}_{\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R))} \mathsf{tp}_n(u_1, \ldots, u_n)$ implies $f(u_1\theta, \ldots, u_n\theta) \xrightarrow{*}_R t$ for all substitution θ.

In the sequel, we show inverse computation by narrowing of the generated inverse EV-TRSs.

Example 5.1.21 Consider the following constructor TRS R_7 over the signature $\{\, \mathsf{s}, \mathsf{0}, \mathsf{add}, \mathsf{mul} \,\}$ and its inverse EV-TRS R_{11} again:

$$\begin{aligned} R_7 = \{\ & \mathsf{add}(\mathsf{0}, y) \to y, \quad \mathsf{add}(\mathsf{s}(x), y) \to \mathsf{s}(\mathsf{add}(x, y)), \\ & \mathsf{mul}(\mathsf{0}, y) \to \mathsf{0}, \quad \mathsf{mul}(x, \mathsf{0}) \to \mathsf{0}, \\ & \mathsf{mul}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{s}(\mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y)) \quad \}, \end{aligned}$$

$$
\begin{aligned}
R_{11} &= \mathbb{U}(R_8)(= \mathcal{I}nv_{\mathsf{full}}(R_7)) \\
&= \{\ \mathsf{add}^{\#}(y) \to \mathsf{tp}_2(0, y), \\
&\quad\ \mathsf{add}^{\#}(\mathsf{s}(z)) \to \mathsf{u}_1(\mathsf{add}^{\#}(z)), \\
&\quad\ \mathsf{u}_1(\mathsf{tp}_2(x, y)) \to \mathsf{tp}_2(\mathsf{s}(x), y), \\
&\quad\ \mathsf{add}^{\#}(\mathsf{add}(x, y)) \to \mathsf{tp}_2(x, y), \\
&\quad\ \mathsf{mul}^{\#}(0) \to \mathsf{tp}_2(0, y), \\
&\quad\ \mathsf{mul}^{\#}(0) \to \mathsf{tp}_2(x, 0), \\
&\quad\ \mathsf{mul}^{\#}(\mathsf{s}(z)) \to \mathsf{u}_2(\mathsf{add}^{\#}(z)), \\
&\quad\ \mathsf{u}_2(\mathsf{tp}_2(w, y)) \to \mathsf{u}_3(\mathsf{mul}^{\#}(w), y), \\
&\quad\ \mathsf{u}_3(\mathsf{tp}_2(x, \mathsf{s}(y)), y) \to \mathsf{tp}_2(\mathsf{s}(x), \mathsf{s}(y)), \\
&\quad\ \mathsf{mul}^{\#}(\mathsf{mul}(x, y)) \to \mathsf{tp}_2(x, y) \qquad\qquad \}.
\end{aligned}
$$

First, we consider the following equation:

$$\mathsf{mul}(x, y) =^{?} 0. \tag{5.1}$$

To solve the above equation, we compute $\mathsf{mul}^{\#}(0)$ by narrowing. The narrowing sequences starting from $\mathsf{mul}^{\#}(0)$ are shown in Figure 5.3. Thus we obtain two

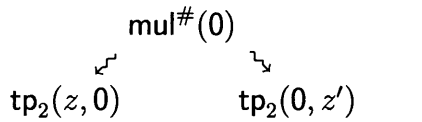

Figure 5.3: The narrowing sequences starting from $\mathsf{mul}^{\#}(0)$.

normal forms $\mathsf{tp}_2(z, 0)$ and $\mathsf{tp}_2(0, z')$ (with respect to $\rightsquigarrow_{R_{11}}$). These normal forms and an arbitrary substitution θ — $(z\theta, 0)$ and $(0, z'\theta)$ — are solutions of the equation (5.1), that is, $\mathsf{mul}(z\theta, 0) \xrightarrow{*}_{R_{11}} 0$ and $\mathsf{mul}(0, z'\theta) \to_{R_{11}} 0$.

Next we consider the following equation:

$$\mathsf{mul}(x, y) =^{?} \mathsf{s}^4(0). \tag{5.2}$$

The narrowing sequences starting from $\mathsf{mul}^{\#}(\mathsf{s}^4(0))$ are shown in Figure 5.4. Thus we obtain sixteen normal forms (with respect to $\rightsquigarrow_{R_{11}}$), and three of them, $\mathsf{tp}_2(\mathsf{s}(0), \mathsf{s}^4(0))$, $\mathsf{tp}_2(\mathsf{s}^2(0), \mathsf{s}^2(0))$ and $\mathsf{tp}_2(\mathsf{s}^4(0), \mathsf{s}(0))$ are considered as solutions of the equation (5.2). The tree representation of narrowing sequences in Figure 5.4 shows that there is no infinite narrowing sequence starting from $\mathsf{mul}^{\#}(\mathsf{s}^4(0))$. The following sequence shows the details of the narrowing sequence from $\mathsf{mul}^{\#}(\mathsf{s}^4(0))$

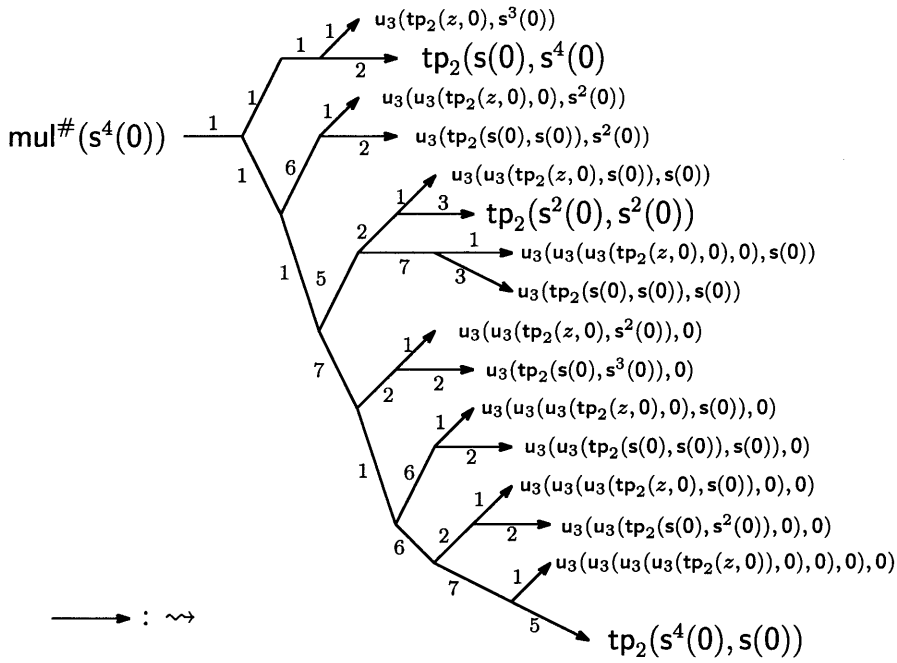

Figure 5.4: The narrowing sequences starting from $\mathsf{mul}^{\#}(\mathsf{s}^4(0))$.

to $\mathsf{tp}_2(\mathsf{s}^2(0), \mathsf{s}^2(0))$:

$$
\begin{array}{lll}
\mathsf{mul}^{\#}(\mathsf{s}^4(0)) & \leadsto_{R_{11}} \mathsf{u}_2(\mathsf{add}^{\#}(\mathsf{s}^3(0))) & \leadsto_{R_{11}} \mathsf{u}_2(\mathsf{u}_1(\mathsf{add}^{\#}(\mathsf{s}^2(0)))) \\
& \leadsto_{R_{11}} \mathsf{u}_2(\mathsf{u}_1(\mathsf{u}_1(\mathsf{add}^{\#}(\mathsf{s}(0))))) & \leadsto_{R_{11}} \mathsf{u}_2(\mathsf{u}_1(\mathsf{u}_1(\mathsf{tp}_2(0, \mathsf{s}(0))))) \\
& \overset{4}{\leadsto}_{R_{11}} \mathsf{u}_3(\mathsf{u}_2(\mathsf{add}^{\#}(\mathsf{s}(0))), \mathsf{s}(0)) & \leadsto_{R_{11}} \mathsf{u}_3(\mathsf{u}_2(\mathsf{tp}_2(0, \mathsf{s}(0))), \mathsf{s}(0)) \\
& \leadsto_{R_{11}} \mathsf{u}_3(\mathsf{u}_3(\mathsf{mul}^{\#}(0), \mathsf{s}(0)), \mathsf{s}(0)) & \leadsto_{R_{11}} \mathsf{u}_3(\mathsf{u}_3(\mathsf{tp}_2(0, z), \mathsf{s}(0)), \mathsf{s}(0)) \\
\{z \mapsto \mathsf{s}^2(0)\} \leadsto_{R_{11}} & \mathsf{u}_3(\mathsf{tp}_2(\mathsf{s}(0), \mathsf{s}^2(0)), \mathsf{s}(0)) & \leadsto_{R_{11}} \mathsf{tp}_2(\mathsf{s}^2(0), \mathsf{s}^2(0)).
\end{array}
$$

The thirteen normal forms whose root symbols are u_3, means that those paths are failure of the inverse computation for $\mathsf{mul}(x, y) \to_{R_7} \mathsf{s}^4(0)$. □

Theorem 5.1.19 (b) and Corollary 5.1.20 (b) show that every instance of n-term tuple $(u_1, \ldots, u_n)$, that is, $(u_1\theta, \ldots, u_n\theta)$, is a solution of inverse computation for $f(x_1, \ldots, x_n) \overset{*}{\to}_R t$. These results also show that the n-term tuple $(u_1, \ldots, u_n)$ is one of general solutions of inverse computation for $f(x_1, \ldots, x_n) \overset{*}{\to}_R t$. For example, the normal forms $\mathsf{tp}_2(0, z)$ and $\mathsf{tp}_2(z, 0)$ (with respect to $\leadsto_{R_{11}}$) in Example 5.1.21 are the unique general solutions (up to renaming) of inverse computation for $\mathsf{mul}(x, y) \to_{R_7} 0$. In fact, solutions of $x \times y = 0$ are commutative pairs of 0 and an arbitrary natural number, and $\mathsf{tp}_2(z, 0)$ and $\mathsf{tp}_2(0, z)$ can represent all of such solutions.

5.2 Termination of Inverse EV-TRSs

In Section 5.1, we have shown that inverse EV-TRSs generated by our transformation are worked by narrowing starting ground terms. In this section, to give a termination proof technique of our inverse EV-TRSs, we propose two termination proof methods of narrowing, especially starting from ground terms. One is based on the notion of *active chain* [11], the other is based on the dependency pair method which was proposed by T. Arts and J. Giesl to prove the termination of TRSs [4, 5].

In both of the theorems which we will show in this section, we induce the termination of narrowing of EV-TRSs to the termination of others, such as narrowing of another (EV-)TRS, rewrite relation of TRSs and so on. To prove these theorems, we construct an infinite narrowing sequence from another infinite sequence. Such a proof technique is usable under the *top reduced almost terminating* (*TRAT*, for short) property (defined in Subsection 5.2.1). We first show that narrowing of constructor EV-TRSs and that of right-linear EV-TRSs on linear terms have TRAT property. Then, we propose two termination proof methods of narrowing. Especially, we focus on the termination of narrowing starting from ground terms, that is, $\mathrm{GSN}^{\rightsquigarrow_R}$, since we use narrowing to simulate ground rewrite sequences of EV-TRSs.

In the sequel, we give two propositions associated with the basic properties on the termination of narrowing.

Proposition 5.2.1 *For every TRS R, $SN^{\rightarrow_R}$ iff $GSN^{\rightsquigarrow_R}$.*

Proof. It is clear that $\mathrm{SN}^{\rightarrow_R}$ iff $\mathrm{GSN}^{\rightarrow_R}$. Since $\rightarrow_R = \rightsquigarrow_R$ on ground terms for all TRSs, every ground narrowing sequence of R is a ground rewrite sequence of R and the converse also holds. Then, if there exists an infinite ground rewrite (or narrowing) sequence of R, then it is also infinite ground narrowing (rewrite, respectively) sequence of R. Therefore, this proposition holds. □

Proposition 5.2.2 *For every EV-TRS R, $SN^{\rightsquigarrow_R}$ implies $GSN^{\rightsquigarrow_R}$.*

Proof. From the definition of $\mathrm{SN}^{\rightsquigarrow_R}$ and $\mathrm{GSN}^{\rightsquigarrow_R}$, this proposition holds obviously. □

The converse of Proposition 5.2.2 above does not hold in general. For example, consider the TRS R_1 in Example 3.1.2 again:

$$R_1 = \{\, \mathsf{double}(\mathsf{0}) \to \mathsf{0},\ \mathsf{double}(\mathsf{s}(x)) \to \mathsf{s}^2(\mathsf{double}(x)) \,\}.$$

We have $\mathrm{SN}^{\rightarrow_{R_1}}$ obviously, and hence $\mathrm{GSN}^{\rightarrow_{R_1}}$. However, $\mathrm{SN}^{\rightarrow_{R_1}}$ does not hold because infinite narrowing sequences can be easily constructed starting from $\mathsf{double}(x)$. Thus, $\mathrm{SN}^{\rightarrow_R}$ does not hold for most of EV-TRSs (even TRSs). Therefore, $\mathrm{SN}^{\rightarrow_R}$ proof methods are not as powerful as $\mathrm{GSN}^{\rightarrow_R}$ proof methods.

Some termination criteria for narrowing and E-narrowing have shown in [9]. The results in [9] treats TRSs in which the left-hand side of each rewrite rule is flat[1]. The condition of flatness is too restrictive for our inverse EV-TRSs.

When the generated inverse EV-TRS is a TRS, we can use termination proof tools of TRSs, such as *Tsukuba Termination Tool* [25], *AProVE* [16, 17, 18] and so on. Note that these tools are not applicable to EV-TRSs which are not TRSs.

5.2.1 Top Reduced Almost Terminating Property

In this subsection, we introduce the *top reduced almost terminating* property, which should be satisfied to apply the termination proof methods in this section.

Let R be an EV-TRS and $\rightarrow$ be a reduction based on either $\rightarrow_R$ or $\rightsquigarrow_R$. An infinite sequence $t_0 \rightarrow t_1 \rightarrow \cdots$ is said to be *almost terminating* if for every proper subterm u of the initial term t_0 is strongly normalizing with respect to $\rightarrow$ ($\mathrm{SN}^{\rightarrow}_u$). An almost terminating sequence of $\rightarrow$ is called *top reduced* if it contains at least one $\rightarrow^{\varepsilon}$. We call it a *TRAT sequence* for short. We say that the reduction $\rightarrow$ has *top reduced almost terminating* (*TRAT*, for short) property if for every term t with $\neg\mathrm{SN}^{\rightarrow}_t$ there exists a top reduced almost terminating sequence of $\rightarrow$ starting from a subterm of t.

For every monotone reduction, the following holds obviously.

Proposition 5.2.3 *Let R be an EV-TRS and $\rightarrow$ be a relation such that $\rightarrow \subseteq \rightarrow_R$ or $\rightarrow \subseteq \rightsquigarrow_R$. If $\rightarrow$ is monotone, then $\rightarrow$ has TRAT property.*

It is known that for every EV-TRS R, $\rightarrow_R$ is monotone. However, $\rightsquigarrow_R$ is not always monotone. For example, the following EV-TRS over $\{\, \mathsf{0}, \mathsf{c}, \mathsf{d} \,\}$ does not have TRAT property:

$$R_{15} = \{\ \mathsf{d}(\mathsf{d}(x)) \rightarrow x\ \}.$$

Because the following almost terminating sequence is not top reduced:

$$\begin{aligned} \mathsf{c}(\mathsf{d}(x_0), x_0) \ &\rightsquigarrow_{\{x_0 \mapsto \mathsf{d}(x_1)\}R_{15}}\ \mathsf{c}(x_1, \mathsf{d}(x_1)) \\ &\rightsquigarrow_{\{x_1 \mapsto \mathsf{d}(x_2)\}R_{15}}\ \mathsf{c}(x_2, \mathsf{d}(x_2)) \\ &\rightsquigarrow_{\{x_2 \mapsto \mathsf{d}(x_2)\}R_{15}}\ \cdots. \end{aligned}$$

[1]Term $f(t_1, \ldots, t_n)$ is *flat* if t_i is either a variable or a ground term.

Therefore $\rightsquigarrow_R$ does not have TRAT property in general.

As we have shown already, one-step narrowing of right-linear EV-TRSs preserves the linearity of a target term (see Proposition 5.1.9). Moreover, $\rightsquigarrow_R$ on linear terms has nice properties for a right-linear EV-TRS R.

Proposition 5.2.4 *Let R be a right-linear EV-TRS.*

(a) If $s \overset{*}{\rightsquigarrow}_R t$ for a linear term s, then t is linear.

(b) The narrowing $\rightsquigarrow_R$ on linear terms is monotone.

Proof. The first claim (a) follows from Proposition 5.1.9. The second claim (b) follows from the claim (a). □

Let R be a CTRS over a signature $\mathcal{F}$, $\rightarrow$ be a relation either $\rightarrow_R$ or $\rightsquigarrow_R$ and $\blacktriangleleft$ be a prefix ordering either $<$ or $\leq$. Suppose that $s \rightarrow^q t$. Then, we write $s \rightarrow^{p\blacktriangleleft} t$ for any position p with $p \blacktriangleleft q$. A substitution θ is said to be $SN^{\rightarrow}$, written as $SN^{\rightarrow}_{\theta}$, if for all $x \in \mathcal{D}om(\theta)$, $\mathrm{SN}^{\rightarrow}_{x\theta}$ holds.

We also have a nice property on constructor EV-TRSs.

Theorem 5.2.5 *For every constructor EV-TRS R, $\rightsquigarrow_R$ has TRAT property.*

Proof. For a constructor EV-TRS R, the followings hold obviously:

(i) Suppose that $\mathrm{SN}^{\rightsquigarrow R}_t$ and $t \ {}_{\delta}\!\rightsquigarrow_R u$. Then, $\mathrm{SN}^{\rightsquigarrow R}_{\delta}$.

(ii) If $\mathrm{SN}^{\rightsquigarrow R}_t$, then $\mathrm{SN}^{\rightsquigarrow R}_{t\theta}$ for all substitution θ satisfying $\mathrm{SN}^{\rightsquigarrow R}_{\theta}$.

Assuming that there exists an almost terminating sequence $t \equiv t_0 \rightsquigarrow^{\varepsilon<}_R t_1 \rightsquigarrow^{\varepsilon<}_R \cdots$ which is not top reduced, we show a contradiction that there exists a proper subterm u of t that $\neg\mathrm{SN}^{\rightsquigarrow R}_u$. Let $t_i \equiv f(t_{i,1}, \ldots, t_{i,n})$ without loss of generality. From the infinite sequence, for every i, there exists j such that $t_{i,j} \ {}_{\delta_i}\!\rightsquigarrow_R t_{i+1,j}$, $\mathrm{SN}^{\rightsquigarrow R}_{\delta_i}$, and for every j' with $j' \neq j$, $t_{i,j'}\delta_i \equiv t_{i+1,j'}$. Then, we have infinite number of narrowing derivations below k for at least one position k with $1 \leq k \leq n$. Every step from $t_{i,k}$ to $t_{i+1,k}$ consists of either $t_{i,k} \ {}_{\delta_i}\!\rightsquigarrow_R t_{i+1,k}$ or $t_{i,k}\delta_i \equiv t_{i+1,}$. Since narrowing derivations below k is infinite, it follows from the contraposition of (ii) that $\neg\mathrm{SN}^{\rightsquigarrow R}_{t_{i,k}}$, and hence $\neg\mathrm{SN}^{\rightsquigarrow R}_{t_{0,k}}$. Therefore, there is a proper subterm $t_{0,k}$ of t that $\neg\mathrm{SN}^{\rightsquigarrow R}_{t_{0,k}}$. □

Proposition 5.2.4 and Theorem 5.2.5 above are very important because inverse EV-TRSs generated by our transformation $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(\cdots))$ are constructor systems.

5.2.2 Termination Proof Based on Function Calls

We first give a termination proof theorem which is based on the notion of active chains [11].

As the first step, we construct from an EV-TRS a TRS by extracting defined symbols in rewrite rule of the input.

Definition 5.2.6 *Let R be an EV-TRS over a signature $\mathcal{F}$. The TRS $R^\top$ constructed as follows:*

$$R^\top = \{\, \mathtt{root}(l) \to \mathtt{root}(t) \mid l \to r \in R, t \trianglelefteq r, \mathtt{root}(t) \in \mathcal{D}_R \,\}.$$

The above TRS $R^\top$ represent dependency of function (defined-symbol) calls in rewrite rules of R.

For EV-TRSs with TRAT property, we can prove $\mathrm{SN}^{\rightsquigarrow_R}$ using $\mathrm{SN}^{\rightarrow_{R^\top}}$.

Theorem 5.2.7 *Let R be an EV-TRS and suppose that $\rightsquigarrow_R$ has TRAT property. Then, $SN^{\rightarrow_{R^\top}}$ implies $SN^{\rightsquigarrow_R}$.*

Proof. We prove this claim by constructing an infinite rewrite sequence of $R^\top$ from an infinite narrowing sequence of R.

Suppose that $\mathrm{SN}^{\rightsquigarrow_R}$ does not hold. Then, there exists a term from which an infinite narrowing sequence exists. Since $\rightsquigarrow_R$ has TRAT property now, there exists a TRAT term t_0. Here we assume the following infinite narrowing sequence starting from t_0:

$$t_0 \overset{*}{\rightsquigarrow}{}^{\varepsilon<}_R t'_0 \overset{\varepsilon}{\rightsquigarrow}_R t''_0 \rightsquigarrow_R \cdots.$$

Moreover, we assume the following:

$$t'_0 \overset{[\varepsilon,\rho_0]}{\rightsquigarrow}_R r_0\sigma_0 \equiv t''_0$$

where $\rho_0 : l_0 \to r_0 \in R$ and $\sigma_0 = \mathtt{mgu}(t'_0, l_0)$. Since $\mathrm{SN}^{\rightsquigarrow_R}_{t_0}$ does not hold, $\mathrm{SN}^{\rightsquigarrow_R}_{t''_0}$ does not hold, too. Then, t''_0 has a subterm t_1 that is TRAT.

On the other hand, σ_0 is an $NF^{\rightsquigarrow_R}(\mathcal{F}, \mathcal{X})$-substitution since every proper subterm of t'_0 is $\mathrm{SN}^{\rightsquigarrow_R}$. Then, there exists a non-variable subterm u_1 of r_0 such that $u_1\sigma_0 \equiv t_1$. It is clear that the root symbol of u_1 is a defined symbol of R. It is also clear that $\mathtt{root}(t_0) = \mathtt{root}(t'_0) = \mathtt{root}(l_0)$. Now we can represent as $r_0 \equiv C_0[u_1]$. Then, the rule $\mathtt{root}(l_0) \to \mathtt{root}(u_1) \in R^\top$ exists, and hence $\mathtt{root}(l_0) \to_{R^\top} \mathtt{root}(u_1)$.

By repeating the above argument for the term $t_1, \cdots$, we obtain the following infinite rewrite sequence of $R^\top$:

$$\mathtt{root}(t_0) \to_{R^\top} \mathtt{root}(t_1) \to_{R^\top} \mathtt{root}(t_2) \to_{R^\top} \cdots.$$

□

It is trivial that $\mathrm{SN}^{\to}_{R^\top}$ if R has no recursive call of defined symbols in its rules. Therefore, the above theorem shows that if an EV-TRS R without recursive calls of defined symbols is either a constructor system or right-linear, narrowing of R starting from ground terms is always terminating.

Example 5.2.8 Consider the following EV-TRS over $\{\mathsf{a}, \mathsf{b}, \mathsf{f}, \mathsf{g}, \mathsf{h}, \mathsf{s}, \mathsf{0}\}$ again:

$$R_{14} = \{\ \mathsf{f}(x, \mathsf{0}) \to \mathsf{s}(x), \quad \mathsf{g}(x) \to \mathsf{h}(x, y), \quad \mathsf{h}(\mathsf{0}, x) \to \mathsf{f}(x, x), \quad \mathsf{a} \to \mathsf{b}\ \}.$$

From the defined symbols $\mathsf{a}, \mathsf{f}, \mathsf{g}, \mathsf{h}$, the following TRS is obtained from Definition 5.2.6:

$$R_{14}^\top = \{\ \mathsf{g} \to \mathsf{h}, \quad \mathsf{h} \to \mathsf{f}\ \}.$$

R_{14} is a constructor system, and hence $\to_{R_{14}}$ has TRAT property. Since $\mathrm{SN}^{R_{14}^\top}$ holds obviously, $\mathrm{SN}^{\rightsquigarrow_{R_{14}}}$ holds from Theorem 5.2.7. □

Example 5.2.9 Consider the following EV-TRS over $\{\ \mathsf{c}, \mathsf{e}, \mathsf{double}, \mathsf{s}, \mathsf{0}\ \}$:

$$R_{16} = \{\ \mathsf{double}(\mathsf{0}) \to \mathsf{0},\ \ \mathsf{double}(\mathsf{s}(x)) \to \mathsf{s}^2(\mathsf{double}(x)),\ \ \mathsf{e} \to \mathsf{double}(\mathsf{c}(x, x))\ \}.$$

From the defined symbols double, the following TRS is obtained by the method of Definition 5.2.6:

$$R_{16}^\top = \{\ \mathsf{double} \to \mathsf{double}, \quad \mathsf{e} \to \mathsf{double}\ \}.$$

Since $\mathrm{SN}^{\to}_{R_{16}^\top}$ does not hold obviously, and hence $\mathrm{SN}^{\rightsquigarrow_{R_{16}}}$ cannot be proved by Theorem 5.2.7. In fact, $\mathrm{SN}^{\rightsquigarrow_{R_{16}}}$ does not hold because of the infinite narrowing sequence $\mathsf{double}(x_0) \ {}_{\{x_0 \mapsto \mathsf{s}(x_1)\}}\!\rightsquigarrow_{R_{16}} \mathsf{s}^2(\mathsf{double}(x_1)) \ {}_{\{x_1 \mapsto \mathsf{s}(x_2)\}}\!\rightsquigarrow_{R_{16}} \cdots$. □

5.2.3 Termination Proof Using Dependency Pairs

The termination proof method proposed in Subsection 5.2.2 (Theorem 5.2.7) is very weak because many practical EV-TRSs with $\mathrm{GSN}^{\rightsquigarrow}$ have recursive calls of defined symbols. In this subsection, we propose a more practical termination proof technique by extending the dependency pair method [4, 5]. However, Theorem 5.2.7 is still valuable since there exist some cases in which it is usable but the method proposed in this subsection is not.

5.2.3.1 Dependency Pairs and Chains

Let s and t be terms. We denote the pair of terms s and t by $\langle s,t\rangle$. We call variables not in s but in t $(\mathcal{V}ar(t) \setminus \mathcal{V}ar(s))$ *extra variables* of $\langle s,t\rangle$, too. The set of all extra variables of $\langle s,t\rangle$ is denoted by $\mathcal{EV}ar(\langle s,t\rangle)$. We write $\mathcal{V}ar(\langle s,t\rangle)$ to represent the set of all variables appearing in $\langle s,t\rangle$.

The definition of dependency pairs of EV-TRSs is the same with that of TRSs [4, 5]. To illustrate it, we prepare a new function symbol $f^\sharp$ marked with $\sharp$, called the *marked function symbol* of f, for every defined symbol f in a given signature $\mathcal{F}$ divided into the constructor set $\mathcal{C}$ and the defined-symbol set $\mathcal{D}$: $\mathcal{F} = \mathcal{C} \uplus \mathcal{D}$. This thesis assume that a given signature does not include any function symbol marked with $\sharp$, that is, $f^\sharp \notin \mathcal{F}$ for every function symbol $f \in \mathcal{F}$. We denote the set of marked function symbol of $\mathcal{D}$ by $\mathcal{D}^\sharp$; $\mathcal{D}^\sharp = \{\, f^\sharp \mid f \in \mathcal{D} \,\}$. We define $\mathcal{F}^\sharp$ as $\mathcal{F}^\sharp = \mathcal{F} \cup \mathcal{D}^\sharp$.

Definition 5.2.10 (Dependency pair [4, 5]) *Let R be an EV-TRS over a signature $\mathcal{F}$. The pair $\langle f^\sharp(s_1,\ldots,s_m), g^\sharp(t_1,\ldots,t_n)\rangle$ is called a* dependency pair *of R if there are a rewrite rule $f(s_1,\ldots,s_m) \to r \in R$ and a subterm $g(t_1,\ldots,t_n)$ of r with a defined symbol g; $g(t_1,\ldots,t_n) \trianglelefteq r$ and $g \in \mathcal{D}_R$. The set of all dependency pairs of R is denoted by $\mathcal{DP}_R$.*

Example 5.2.11 Consider the following EV-TRS R_{14} over $\{\mathsf{a}, \mathsf{b}, \mathsf{f}, \mathsf{g}, \mathsf{h}, \mathsf{s}, \mathsf{0}\}$ again:

$$R_{14} = \{\ \mathsf{f}(x,\mathsf{0}) \to \mathsf{s}(x),\quad \mathsf{g}(x) \to \mathsf{h}(x,y),\quad \mathsf{h}(\mathsf{0},x) \to \mathsf{f}(x,x),\quad \mathsf{a} \to \mathsf{b}\ \}.$$

The following dependency pairs are obtained from R_{14}:

$$\mathcal{DP}_{R_{14}} = \{\ \langle \mathsf{g}^\sharp(x), \mathsf{h}^\sharp(x,y)\rangle,\quad \langle \mathsf{h}^\sharp(\mathsf{0},x), \mathsf{f}^\sharp(x,x)\rangle\ \}.$$

□

Example 5.2.12 Consider the following EV-TRS R_{16} over $\{\mathsf{c}, \mathsf{e}, \mathsf{double}, \mathsf{s}, \mathsf{0}\}$ again:

$$R_{16} = \{\, \mathsf{double}(\mathsf{0}) \to \mathsf{0},\ \mathsf{double}(\mathsf{s}(x)) \to \mathsf{s}^2(\mathsf{double}(x)),\ \mathsf{e} \to \mathsf{double}(\mathsf{c}(x,x)) \,\}.$$

The following dependency pairs are obtained from R_{16}:

$$\mathcal{DP}_{R_{16}} = \{\ \langle \mathsf{double}^\sharp(\mathsf{s}(x)), \mathsf{double}^\sharp(x)\rangle \quad \langle \mathsf{e}^\sharp, \mathsf{double}^\sharp(\mathsf{c}(x,x))\rangle\ \}.$$

□

$$\begin{array}{ccccccccc} & & \langle\, s_1,\ t_1 \,\rangle & & \langle\, s_2,\ t_2 \,\rangle & & \langle\, s_3,\ t_3 \,\rangle & & \cdots \\ & \sigma_1 = \mathtt{mgu}(s_1', s_1) & \vdots\ \vdots & \sigma_2 = \mathtt{mgu}(s_2', s_2) & \vdots\ \vdots & \sigma_3 = \mathtt{mgu}(s_3', s_3) & \vdots\ \vdots & & \\ \big(T(\mathcal{F}\cup\mathcal{G}) \ni \exists s_0 \overset{*}{\rightsquigarrow}_R\big)\ s_1' & t_1\sigma_1 & \overset{*}{\rightsquigarrow}_R & s_2' & t_2\sigma_2 & \overset{*}{\rightsquigarrow}_R & s_3' & t_3\sigma_3 & \overset{*}{\rightsquigarrow}_R \cdots \end{array}$$

Figure 5.5: A (ground) $\langle\!\langle \rightsquigarrow_R, S \rangle\!\rangle$-chain.

Next we define R-chains [4, 5] and extend them to those which are constructed by narrowing.

Definition 5.2.13 (Chain) *Let R be an EV-TRS over a signature $\mathcal{F}$ and S be a set of pairs of terms over $\mathcal{F}$ and a signature $\mathcal{G}$.*

(a) A sequence $\langle s_1, t_1\rangle\ \langle s_2, t_2\rangle \cdots$ of pairs in S is called a $\langle\!\langle \rightarrow_R, S \rangle\!\rangle$-chain *if there exists a substitution σ_i for $i > 0$ such that $t_i\sigma_i \overset{*}{\rightarrow}_R s_{i+1}\sigma_{i+1}$.*

(b) The sequence $\langle s_1, t_1\rangle\ \langle s_2, t_2\rangle \cdots$ is called a $\langle\!\langle \rightsquigarrow_R, S \rangle\!\rangle$-chain *if there exist a term s_i' and the most general unifier $\sigma_i = \mathtt{mgu}(s_i', s_i)$ for $i > 0$ such that $t_i\sigma_i \overset{*}{\rightarrow}_R s_{i+1}'$*[2]*. Moreover, it is said to be* ground *if there exists a ground term $s_0 \in \mathcal{T}(\mathcal{F}\cup\mathcal{G})$ such that $s_0 \overset{*}{\rightsquigarrow}_R s_1'$ (see Figure 5.5). In this case, we write $s_0\ \langle s_1, t_1\rangle\ \langle s_2, t_2\rangle \cdots$.*

Note that a $\langle\!\langle \rightarrow_R, \mathcal{DP}_R \rangle\!\rangle$-chain is simply called an R-chain *[4, 5].*

Example 5.2.14 Consider the following EV-TRS R_{16} over $\{\mathsf{c}, \mathsf{e}, \mathsf{double}, \mathsf{s}, \mathsf{0}\}$ and its dependency pairs again:

$$R_{16} = \{\ \mathsf{double(0)} \to \mathsf{0},\ \ \mathsf{double(s}(x)) \to \mathsf{s}^2(\mathsf{double}(x)),\ \ \mathsf{e} \to \mathsf{double(c}(x, x))\ \},$$

$$\mathcal{DP}_{R_{16}} = \{\ \langle \mathsf{double}^\sharp(\mathsf{s}(x)), \mathsf{double}^\sharp(x)\rangle \quad \langle \mathsf{e}^\sharp, \mathsf{double}^\sharp(\mathsf{c}(x, x))\rangle\ \}.$$

The following infinite sequence of dependency pairs is a $\langle\!\langle \rightsquigarrow_{R_{16}}, \mathcal{DP}_{R_{16}} \rangle\!\rangle$-chain:

$$\langle \mathsf{double}^\sharp(\mathsf{s}(x_1)), \mathsf{double}^\sharp(x_1)\rangle \langle \mathsf{double}^\sharp(\mathsf{s}(x_2)), \mathsf{double}^\sharp(x_2)\rangle \cdots.$$

The $\langle\!\langle \rightsquigarrow_{R_{16}}, \mathcal{DP}_{R_{16}} \rangle\!\rangle$-chain

$$\langle \mathsf{double}^\sharp(\mathsf{s}(y_1)), \mathsf{double}^\sharp(y_1)\rangle\ \langle \mathsf{double}^\sharp(\mathsf{s}(y_2)), \mathsf{double}^\sharp(y_2)\rangle$$

[2]To avoid variable confliction between pairs and the most general unifiers, this thesis assume that all pairs $\langle s_i, t_i\rangle$s with $i > 0$ are renamed to satisfy $\mathcal{V}ar(s_i, t_i) \cap \mathcal{V}ar(s_{i+1}, t_{i+1}) = \emptyset$.

is ground because there is a ground term $s_0 \equiv \mathsf{double}^\natural(\mathsf{s}^2(0))$ such that $s_0 \xrightarrow{*}_{R_{16}} \mathsf{double}^\natural(\mathsf{s}(y_1))\{y_1 \mapsto \mathsf{s}(0)\}$ and $\mathsf{double}^\natural(y_1)\{y_1 \mapsto \mathsf{s}(0)\} \xrightarrow{*}_{R_{16}} \mathsf{double}^\natural(\mathsf{s}(y_2))\{y_2 \mapsto 0\}$. □

The finiteness of R-chains are usable to prove the termination proof of the TRS R.

Theorem 5.2.15 ([4, 5]) *For every TRS R, $SN^{\rightsquigarrow_R}$ iff there is no infinite R-chain.*

The above theorem is extended to that of narrowing as follows:

Theorem 5.2.16 *Let R be an EV-TRS and suppose that $\rightsquigarrow_R$ has TRAT property.*

(a) $SN^{\rightsquigarrow_R}$ holds iff there is no infinite $\langle\langle \rightsquigarrow_R, \mathcal{DP}_R \rangle\rangle$-chain.

(b) $GSN^{\rightsquigarrow_R}$ holds iff there is no infinite ground $\langle\langle \rightsquigarrow_R, \mathcal{DP}_R \rangle\rangle$-chain.

Proof. We prove only the claim (b) since the proof of (a) is simpler than that of (b). To simplify notation, we abbreviate a term sequence $t_{i,1}, \ldots, t_{i,n_i}$ to $\vec{t_i}$.

To show the *if*-part of (b), we construct an infinite ground $\langle\langle \rightsquigarrow_R, \mathcal{DP}_R \rangle\rangle$-chain from an infinite ground narrowing sequence. Since $\rightsquigarrow_R$ has TRAT property, we can assume the infinite sequence is a top reduced almost terminating narrowing sequence and starts from ground term $s_0 \equiv f_1(\vec{u_0})$. Then, we have

$$f_1(\vec{u_0}) \overset{*}{\rightsquigarrow}{}^{\epsilon<}_R f_1(\vec{v_1}) \equiv s_1' \rightsquigarrow^{[\epsilon,\rho_1]}_R r_1\sigma_1 \rightsquigarrow_R \cdots$$

where $\rho_1 : f_1(\vec{w_1})(\equiv l_1) \to r_1 \in R$ and $\sigma_1 = \mathtt{mgu}(s_1', l_1)$. Since $\mathrm{SN}^{\rightsquigarrow_R}_{v_{1,i}}$ holds, $\mathrm{SN}^{\rightsquigarrow_R}_{x\sigma_1}$ holds for any $x \in \mathcal{D}om(\sigma_1)$. Hence, there is a subterm $t_1 \equiv f_2(\vec{u_1})$ of r_1 such that there exists a top reduced almost terminating sequence starting from $t_1\sigma_1$ since $\rightsquigarrow_R$ has TRAT property. Then, as similar as the case of s_0, we have

$$t_1\sigma_1 \equiv f_2(\vec{u_1}\sigma_1) \overset{*}{\rightsquigarrow}{}^{\epsilon<}_R f_2(\vec{v_2}) \equiv s_2' \rightsquigarrow^{[\epsilon,\rho_2]}_R r_2\sigma_2 \rightsquigarrow_R \cdots$$

where $\rho_2 : f_2(\vec{w_2})(\equiv l_2) \to r_2 \in R$ and $\sigma_2 = \mathtt{mgu}(s_2', l_2)$. Since $\mathrm{SN}^{\rightsquigarrow_R}_{v_{2,i}}$ holds, $\mathrm{SN}^{\rightsquigarrow_R}_{x\sigma_2}$ also holds for any $x \in \mathcal{D}om(\sigma_2)$. Hence, there is a subterm $t_2 \equiv f_3(\vec{u_2})$ of r_2 such that there exists a top reduced almost terminating sequence starting from $t_2\sigma_2$. Here, $\langle f_1^\natural(\vec{w_1}), f_2^\natural(\vec{u_2})\rangle$, $\langle f_2^\natural(\vec{w_2}), f_3^\natural(\vec{u_3})\rangle \in \mathcal{DP}_R$ follow from ρ_1 and ρ_2. Since $u_{0,i} \overset{*}{\rightsquigarrow}_R v_{1,i}$, $\sigma_1 = \mathtt{mgu}(s_1', l_1)$, $u_{1,i} \overset{*}{\rightsquigarrow}_R v_{2,i}$ and $\sigma_2 = \mathtt{mgu}(s_2', l_2)$, we have a ground chain $f_1^\natural(\vec{u_0})\ \langle f_1^\natural(\vec{w_1}), f_2^\natural(\vec{u_2})\rangle\ \langle f_2^\natural(\vec{w_2}), f_3^\natural(\vec{u_3})\rangle$.

By repeating the above argument, we obtain an infinite ground chain

$$f_1^\natural(\vec{u_0})\ \langle f_1^\natural(\vec{w_1}), f_2^\natural(\vec{u_2})\rangle\ \langle f_2^\natural(\vec{w_2}), f_3^\natural(\vec{u_3})\rangle\ \cdots.$$

Next, we prove *only-if*-part of the claim (b) by constructing an infinite ground narrowing-sequence from an infinite ground $\langle\!\langle \rightsquigarrow_R, \mathcal{DP}_R\rangle\!\rangle$-chain

$$f_1^\natural(\vec{u_0})\ \langle f_1^\natural(\vec{w_1}), f_2^\natural(\vec{u_2})\rangle\ \langle f_2^\natural(\vec{w_2}), f_3^\natural(\vec{u_3})\rangle\ \cdots.$$

From the definition of $\langle\!\langle \rightsquigarrow_R, \mathcal{DP}_R\rangle\!\rangle$-chain, there are a term $f_i^\natural(\vec{v_i})$ and the most general unifier $\sigma_i = \mathtt{mgu}(f_i^\natural(\vec{v_i}), f_i^\natural(\vec{w_i}))$ such that $f_i^\natural(\vec{u_i})\sigma_{i-1} \overset{*}{\rightsquigarrow}_R f_i^\natural(\vec{v_i})$, where $f_1^\natural(\vec{u_0})\sigma_0 \equiv f_1^\natural(\vec{u_0})$. From the construction of dependency pairs, we have $\rho_i : f_i(\vec{w_i}) \to C_i[f_{i+1}(\vec{u_{i+1}})] \in R$. Hence, we can easily construct an infinite ground narrowing-sequence

$$\begin{aligned} f_1(\vec{u_0}) \overset{*}{\rightsquigarrow}_R f_1(\vec{v_1}) \ &\rightsquigarrow_R^{[\varepsilon,\rho_1]} C_1\sigma_1[f_2(\vec{u_1})]_{p_1} \overset{*}{\rightsquigarrow}_R^{p_1<} C_1\sigma_1[f_2(\vec{v_2})]_{p_1} \\ &\rightsquigarrow_R^{[p_1,\rho_2]} (C_1\sigma_1[C_2[f_3(\vec{u_2})]_{p_2}]_{p_1})\,\sigma_2 \rightsquigarrow_R \cdots. \end{aligned}$$

□

5.2.3.2 Eliminating All Extra Variables

In the case of TRSs, the termination proof can be done by finding a reduction ordering to ensure no infinite chain. To find such an ordering, *argument filtering functions* [5, 32, 33] are known to be useful. In the sequel, we write $[i_1, \ldots, i_m]$ to represent a list of natural numbers.

Definition 5.2.17 (Argument filtering [32, 33]) *Let $\mathcal{G}$ be a signature. An* argument filtering *is a function π such that for any $f \in \mathcal{G}$, $\pi(f)$ is defined as either of the followings:*

(1) $\pi(f) = i$, *or*

(2) $\pi(f) = [i_1, \ldots, i_m]$

where $n = \mathtt{arity}(f)$, $1 \le i \le n$, $1 \le m \le n$ *and* $1 \le i_1 < \cdots < i_m \le n$. *Note that we assume* $\pi(f) = [1, \ldots, n]$ *if* $\pi(f)$ *is not defined explicitly.*

The argument filtering π is naturally extended over terms as follows:

(a) $\pi(x) = x$ *if* $x \in \mathcal{X}$,

(b) $\pi\left(f(t_1, \ldots, t_n)\right) = \pi(t_i)$ *if* $\pi(f) = i$, *and*

(c) $\pi(f(t_1,\ldots,t_n)) = f(\pi(t_{i_1}),\ldots,\pi(t_{i_m}))$ *if* $\pi(f) = [i_1,\ldots,i_m]$.

Moreover, π *is extended over a set* S *of term-pairs as follows:*

$$\pi(S) = \{\ \langle \pi(s), \pi(t)\rangle \mid \langle s,t\rangle \in S\ \}.$$

Note that every EV-TRS can be considered as a set of term-pairs. This thesis assume that for every defined symbol f, $\pi(f)$ and $\pi(f^\natural)$ are not integers but in form of natural-number lists. We call such an argument filtering *simple.*

The ordering determined by a quasi-ordering and an argument filtering is defined as follows:

Definition 5.2.18 ([32, 33]) *Let* $\mathcal{F}$ *be a signature,* $\succsim$ *be a quasi-ordering* $\succsim$ *on terms over* $\mathcal{F}$ *and* π *be an argument filtering for* $\mathcal{F}$. *The ordering* $\succsim_\pi$ *is defined as follows:*

(a) $s \succsim_\pi t$ *iff* $\pi(s) \succ \pi(t)$ *or* $\pi(s) \equiv \pi(t)$, *and*

(b) $s \succ_\pi t$ *iff there is a context* $C \in \mathscr{C}(\mathcal{F},\mathcal{X})$ *such that* $\pi(s) \succ C[\pi(t)]$ *or* $\pi(s) \equiv C[\pi(t)]_p$ *with* $\varepsilon < p$.

It is known that the above ordering $\succsim_\pi$ is a quasi-reduction ordering [5, 32, 33, 42].

To guarantee that no infinite R-chain exists for TRS R, the following theorem is usable:

Theorem 5.2.19 ([4, 5]) *Let* R *be a TRS. There exists no infinite* R*-chain iff there are an argument filtering* π *and a quasi-ordering* $\succsim$ *on term over* $\mathcal{F}^\natural$ *such that*

- $l \succsim_\pi r$ *for every rewrite rule* $l \to r \in R$, *and*
- $s \succ_\pi t$ *for every dependency pair* $\langle s,t\rangle \in \mathcal{DP}_R$.

On the other hand, finding such an ordering above on EV-TRSs is very difficult if having at least one extra variable. To find it easier, we use argument filterings again to eliminate all extra variables.

Definition 5.2.20 (Eliminate All Extra Variables) *Let* S *be a set of term-pairs over a signature* $\mathcal{G}$, *and* π *be an argument filtering for* $\mathcal{G}$. *We say that* π eliminates all extra variables of S *if* $\mathcal{V}ar(s) \supseteq \mathcal{V}ar(t)$ *for every pair* $\langle s,t\rangle \in S$.

Example 5.2.21 Consider the following EV-TRS R_{14} over $\{\mathsf{a}, \mathsf{b}, \mathsf{f}, \mathsf{g}, \mathsf{h}, \mathsf{s}, \mathsf{0}\}$ and its dependency pairs again:

$$R_{14} = \{\ \mathsf{f}(x,\mathsf{0}) \to \mathsf{s}(x),\quad \mathsf{g}(x) \to \mathsf{h}(x,y),\quad \mathsf{h}(\mathsf{0},x) \to \mathsf{f}(x,x),\quad \mathsf{a} \to \mathsf{b}\ \},$$

$$\mathcal{DP}_{R_{14}} = \{\ \langle \mathsf{g}^\sharp(x), \mathsf{h}^\sharp(x,y)\rangle,\quad \langle \mathsf{h}^\sharp(\mathsf{0},x), \mathsf{f}^\sharp(x,x)\rangle\ \}.$$

Let π_{14} be a simple argument filtering defined as follows:

$$\pi_{14}(\mathsf{f}, \mathsf{f}^\sharp, \mathsf{s}) = [\,],\quad \pi_{14}(\mathsf{h}, \mathsf{h}^\sharp) = [1].$$

Applying π_{14} to R_{14} and $\mathcal{DP}_{R_{14}}$, we obtain the following:

$$\pi_{14}(R_{14}) = \{\ \mathsf{f} \to \mathsf{s},\quad \mathsf{g}(x) \to \mathsf{h}(x),\quad \mathsf{h}(\mathsf{0}) \to \mathsf{f},\quad \mathsf{a} \to \mathsf{b}\ \},$$

$$\pi_{14}(\mathcal{DP}_{R_{14}}) = \{\ \langle \mathsf{g}^\sharp(x), \mathsf{h}^\sharp(x)\rangle,\quad \langle \mathsf{h}^\sharp(\mathsf{0}), \mathsf{f}^\sharp\rangle\ \}.$$

There exists no extra variable in $\pi_{14}(R_{14})$ and $\pi_{14}(\mathcal{DP}_{R_{14}})$. Thus, π_{14} eliminates all extra variables of R_{14} and $\mathcal{DP}_{R_{14}}$. □

In the sequel, we discuss properties satisfied by an argument filtering eliminating all extra variables of S.

Proposition 5.2.22 *Let R be an EV-TRS, S be a set of term-pairs and π be a simple argument filtering which eliminates all extra variables of R and S.*

(a) A ground $\langle\!\langle \rightsquigarrow_{\pi(R)}, \pi(S)\rangle\!\rangle$-chain is a $\langle\!\langle \to_{\pi(R)}, \pi(S)\rangle\!\rangle$-chain.

(b) Let a $\langle\!\langle \rightsquigarrow_{\pi(R)}, \pi(S)\rangle\!\rangle$-chain $\langle s_1, t_1\rangle\ \langle s_2, t_2\rangle\ \cdots$. If there exists some $i > 0$ such that $\pi(t_i)$ is ground, then the sequence $\langle s_{i+1}, t_{i+1}\rangle\ \langle s_{i+2}, t_{i+2}\rangle\ \cdots$ is a $\langle\!\langle \to_{\pi(R)}, \pi(S)\rangle\!\rangle$-chain.

Proof. Since $\pi(R)$ has no extra variable, $\pi(R)$ is a TRS and hence $\rightsquigarrow_{\pi(R)} = \to_{\pi(R)}$ on ground terms. Since $\pi(\mathcal{DP}_R)$ also has no extra variable, no variable appears in the ground $\langle\!\langle \rightsquigarrow_{\pi(R)}, \pi(S)\rangle\!\rangle$-chain. Then, it is a $\langle\!\langle \to_{\pi(R)}, \pi(S)\rangle\!\rangle$-chain. Hence, the claim (a) holds.

Suppose that there exists some $i > 0$ such that $\pi(t_i)$ is ground. Then, it is clear that $\langle s_{i+1}, t_{i+1}\rangle\ \langle s_{i+2}, t_{i+2}\rangle\ \cdots$ is a ground $\langle\!\langle \rightsquigarrow_{\pi(R)}, \pi(S)\rangle\!\rangle$-chain. It follows from the claim (a) that $\langle s_{i+1}, t_{i+1}\rangle\ \langle s_{i+2}, t_{i+2}\rangle\ \cdots$ is a $\langle\!\langle \to_{\pi(R)}, \pi(S)\rangle\!\rangle$-chain. Therefore, the claim (b) holds. □

Let π be a simple argument filtering and θ be a substitution. We define the substitution θ_π as $\theta_\pi = \{ x \mapsto \pi(x\sigma) \mid x \in \mathcal{D}om(\theta) \}$. It is clear that $\pi(t\theta) \equiv \pi(t)\theta_\pi$ for any term t.

The following lemma shows the relationship between $s \overset{*}{\rightsquigarrow}_R t$ and $\pi(s) \overset{*}{\rightsquigarrow}_{\pi(R)} \pi(t)$.

Lemma 5.2.23 *Let R be an EV-TRS and π be a simple argument filtering which eliminates all extra variables of R. Let s, t be terms such that $\pi(s)$ is ground. Then, $s \overset{*}{\rightsquigarrow}_R t$ implies $\pi(s) \overset{*}{\rightsquigarrow}_{\pi(R)} \pi(t)$.*

Proof. We prove by induction on n of $s \overset{n}{\rightsquigarrow}_R t$.

Since the case of $n = 0$ is trivial, we assume that $s \ {}_\delta\rightsquigarrow^{[p,\rho]}_R \ u \ {}_{\delta'} \overset{n-1}{\rightsquigarrow}_R t$ and $\pi(s)$ is ground, where $\rho : l \to r \in R$. Then, there are a context C, a term s' and the most general unifier $\sigma = \mathtt{mgu}(s, C[l]_p)$ such that $s \equiv C[s']_p$ and $u \equiv (C[r]_p)\sigma$. Since $\pi(s)$ is ground, $\pi(C)$ is also ground.

- Consider the case that $\Box$ in C is eliminated by π. Now we have $\pi(s) \equiv \pi(C[s']_p) \equiv \pi(C)$ and $\pi(u) \equiv \pi((C[r]_p)\sigma) \equiv \pi(C\sigma) \equiv \pi(C)\sigma_\pi \equiv \pi(C)$. By the induction hypothesis, we have $\pi(u) \overset{*}{\rightsquigarrow}_{\pi(R)} \pi(t)$. Therefore, $\pi(s) \equiv \pi(u) \overset{*}{\rightsquigarrow}_{\pi(R)} \pi(t)$.

- Consider the remaining case. Since $\pi(C)$ is ground and $\pi(C)$ is a context, we have $\pi(s) \equiv \pi(C[s']_p) \equiv (\pi(C)[\pi(s')]_q$ and $\pi(u) \equiv \pi((C[r]_p)\sigma) \equiv (\pi(C\sigma))[\pi(r\sigma)]_q \equiv \pi(C)\sigma_\pi[\pi(r)\sigma_\pi]_q \equiv \pi(C)[\pi(r)\sigma_\pi]_q$.

  On the other hand, $\pi(s')\sigma_\pi \equiv \pi(l)\sigma_\pi$ follows from $s'\sigma \equiv l\sigma$, $\pi(s'\sigma) \equiv \pi(s')\sigma_\pi$ and $\pi(l\sigma) \equiv \pi(l)\sigma_\pi$. Since $\pi(s')$ is ground, we have $\pi(s')\sigma_\pi \equiv \pi(s') \equiv \pi(l)\sigma_\pi$. We also have $\pi(l) \to \pi(r) \in \pi(R)$.

  It follows from the assumption that $\mathcal{V}ar(\pi(l)) \supseteq \mathcal{V}ar(\pi(r))$ for every $l \to r \in R$, and hence $\pi(R)$ is a TRS. Since $\pi(s')$ is ground, σ_π is the most general unifier of $\pi(s')$ and $\pi(l)$: $\sigma_\pi = \mathtt{mgu}(\pi(s'), \pi(l))$. Hence, $\pi(s') \equiv \pi(l)\sigma_\pi \rightsquigarrow_{\pi(R)} \pi(r)\sigma_\pi$ and $\pi(r)\sigma_\pi$ is ground.

  Since $\pi(u) \equiv \pi(C)[\pi(r)\sigma_\pi]$ is also ground, we have $\pi(u) \overset{*}{\rightsquigarrow}_{\pi(R)} \pi(t)$ by the induction hypothesis. Therefore, we have the following sequence:

$$\pi(s) \equiv \pi(C)[\pi(l)\sigma_\pi]_q \rightsquigarrow_{\pi(R)} \pi(C)[\pi(r)\sigma_\pi]_q \equiv \pi(u) \overset{*}{\rightsquigarrow}_{\pi(R)} \pi(t).$$

□

Note that $\pi(s) \overset{*}{\rightsquigarrow}_{\pi(R)} \pi(t)$ iff $\pi(s) \overset{*}{\to}_{\pi(R)} \pi(t)$, because $\pi(R)$ is a TRS and $\pi(s)$ is ground.

$$\begin{array}{ccccccc} & \langle\, \pi(s_1),\ \pi(t_1)\,\rangle & & \langle\, \pi(s_2),\ \pi(t_2)\,\rangle & & \cdots \\ {\scriptstyle (\sigma_1)_\pi = \atop \mathtt{mgu}(\pi(s_1'),\pi(s_1))} & \vdots\ \ \vdots & {\scriptstyle (\sigma_2)_\pi = \atop \mathtt{mgu}(\pi(s_2'),\pi(s_2))} & \vdots\ \ \vdots & & \\ \Big(\pi(s_0) \overset{*}{\rightsquigarrow}_{\pi(R)} \Big)\ \pi(s_1') & \pi(t_1)(\sigma_1)_\pi \overset{*}{\rightarrow}_{\pi(R)} & \pi(s_2') & \pi(t_2)(\sigma_2)_\pi \overset{*}{\rightarrow}_{\pi(R)} & & \cdots \end{array}$$

Figure 5.6: A $\langle\!\langle \rightarrow_{\pi(R)}, \pi(\mathcal{DP}_R) \rangle\!\rangle$-chain.

The fact that there exists no infinite $\langle\!\langle \rightsquigarrow_{\pi(R)}, \pi(S) \rangle\!\rangle$-chain, implies the termination of $\rightsquigarrow_R$ as follows:

Theorem 5.2.24 *Let R be an EV-TRS and π be a simple argument filtering which eliminates all extra variables of R and $\mathcal{DP}_R$.*

(a) If there is no infinite $\langle\!\langle \rightarrow_{\pi(R)}, \pi(\mathcal{DP}_R) \rangle\!\rangle$-chain, then there is no infinite ground $\langle\!\langle \rightsquigarrow_R, \mathcal{DP}_R \rangle\!\rangle$-chain.

(b) If there is no infinite $\langle\!\langle \rightarrow_{\pi(R)}, \pi(\mathcal{DP}_R) \rangle\!\rangle$-chain and for every dependency pair $\langle s, t \rangle \in \mathcal{DP}_R$, $\pi(t)$ is ground, then there is no infinite $\langle\!\langle \rightsquigarrow_R, \mathcal{DP}_R \rangle\!\rangle$-chain.

Proof. (Sketch.) The first claim (a) can be easily proved by constructing an infinite $\langle\!\langle \rightarrow_{\pi(R)}, \pi(\mathcal{DP}_R) \rangle\!\rangle$-chain from an infinite ground $\langle\!\langle \rightsquigarrow_R, \mathcal{DP}_R \rangle\!\rangle$-chain, using Lemma 5.2.23 and Proposition 5.2.22 (a) (an infinite ground $\langle\!\langle \rightsquigarrow_R, \mathcal{DP}_R \rangle\!\rangle$-chain in Figure 5.5 implies an infinite $\langle\!\langle \rightarrow_{\pi(R)}, \pi(\mathcal{DP}_R) \rangle\!\rangle$-chain in Figure 5.6).

The second claim (b) can be proved by using the claim (a) of this theorem and Proposition 5.2.22 (b). □

When there is no extra variable in $\pi(R)$ and $\pi(\mathcal{DP}_R)$, we can use the termination proof methods for TRSs, which have already been proposed in [4, 5], [7], [33], [42] and so on, to guarantee that no infinite $\langle\!\langle \rightsquigarrow_{\pi(R)}, \pi(S) \rangle\!\rangle$-chain exists.

5.2.3.3 Termination Proof Theorem

Finally, we conclude from Theorem 5.2.16, 5.2.19 and 5.2.24 as follows:

Theorem 5.2.25 *Let R be an EV-TRS and suppose that $\rightsquigarrow_R$ has TRAT property.*

(1) *GSN$^{\rightsquigarrow_R}$ holds if there exist a simple argument filtering π, which eliminates all extra variables of R and $\mathcal{DP}_R$, and a quasi-ordering $\succsim$ on term over $\mathcal{F}^\sharp$ such that*

(1-a) $l \succsim_\pi r$ for every rewrite rule $l \to r \in R$, and

(1-b) $s \succ_\pi t$ for every dependency pair $\langle s, t\rangle \in \mathcal{DP}_R$.

(2) *SN$^{\rightsquigarrow_R}$ holds if there exist a simple argument filtering π, which eliminates all extra variables of R and $\mathcal{DP}_R$, and a quasi-ordering $\succsim$ on term over $\mathcal{F}^\sharp$ such that the above conditions (1-a), (1-b) and the following condition are satisfied:*

(2-c) for every dependency pair $\langle s, t\rangle \in \mathcal{DP}_R$, $\pi(t)$ is ground.

Remark that Theorem 5.2.25 is also usable to prove SN$^{\rightsquigarrow_R}$ for a TRS R.

Example 5.2.26 Consider the following EV-TRS R_{14} over $\{\mathsf{a}, \mathsf{b}, \mathsf{f}, \mathsf{g}, \mathsf{h}, \mathsf{s}, 0\}$, its dependency pairs and the following argument filtering π_{14} again:

$$R_{14} = \{\ \mathsf{f}(x, 0) \to \mathsf{s}(x),\quad \mathsf{g}(x) \to \mathsf{h}(x, y),\quad \mathsf{h}(0, x) \to \mathsf{f}(x, x),\quad \mathsf{a} \to \mathsf{b}\ \},$$

$$\mathcal{DP}_{R_{14}} = \{\ \langle \mathsf{g}^\sharp(x), \mathsf{h}^\sharp(x, y)\rangle,\quad \langle \mathsf{h}^\sharp(0, x), \mathsf{f}^\sharp(x, x)\rangle\ \},$$

$$\pi_{14}(\mathsf{f}, \mathsf{f}^\sharp, \mathsf{s}) = [\,],\quad \pi_{14}(\mathsf{h}, \mathsf{h}^\sharp) = [1].$$

According to π_{14} which eliminates all extra variables of R_{14} and $\mathcal{DP}_{R_{14}}$, we have the following:

$$\pi_{14}(R_{14}) = \{\ \mathsf{f} \to \mathsf{s},\quad \mathsf{g}(x) \to \mathsf{h}(x),\quad \mathsf{h}(0) \to \mathsf{f},\quad \mathsf{a} \to \mathsf{b}\ \},$$

$$\pi_{14}(\mathcal{DP}_{R_{14}}) = \{\ \langle \mathsf{g}^\sharp(x), \mathsf{h}^\sharp(x)\rangle,\quad \langle \mathsf{h}^\sharp(0), \mathsf{f}^\sharp\rangle\ \}.$$

Give $\succsim^{14}$ be a recursive path ordering determined by the following precedence $\geq$ on $\{\mathsf{a}, \mathsf{b}, \mathsf{f}, \mathsf{g}, \mathsf{h}, \mathsf{s}, 0\}$:

$$\mathsf{a} \geq \mathsf{b},\quad \mathsf{f} = \mathsf{g} = \mathsf{h},\quad \mathsf{g}^\sharp > \mathsf{h}^\sharp > \mathsf{f}^\sharp.$$

Then we have the following inequalities:

$$\mathsf{f} \succsim^{14}_{\pi_{14}} \mathsf{s},\quad \mathsf{g}(x) \succsim^{14}_{\pi_{14}} \mathsf{h}(x),\quad \mathsf{h}(0) \succsim^{14}_{\pi_{14}} \mathsf{f},\quad \mathsf{a} \succsim^{14}_{\pi_{14}} \mathsf{b},$$

$$\mathsf{g}^\sharp(x) \succ^{14}_{\pi_{14}} \mathsf{h}^\sharp(x),\quad \mathsf{h}^\sharp(0) \succ^{14}_{\pi_{14}} \mathsf{f}^\sharp.$$

Therefore, GSN$^{\rightsquigarrow_{R_{14}}}$ holds.

Let the following argument filtering:

$$\pi'_{14}(\mathsf{f}, \mathsf{f}^\sharp, \mathsf{h}^\sharp, \mathsf{s}) = [\,], \quad \pi'_{14}(\mathsf{h}) = [1].$$

Then, we have the following:

$$\pi'_{14}(R_{14}) = \{\ \mathsf{f} \to \mathsf{s}, \quad \mathsf{g}(x) \to \mathsf{h}(x), \quad \mathsf{h}(\mathsf{0}) \to \mathsf{f}, \quad \mathsf{a} \to \mathsf{b}\ \},$$

$$\pi'_{14}(\mathcal{DP}_{R_{14}}) = \{\ \langle \mathsf{g}^\sharp(x), \mathsf{h}^\sharp \rangle, \quad \langle \mathsf{h}^\sharp, \mathsf{f}^\sharp \rangle\ \}.$$

Now, for every dependency pair $\langle s, t \rangle$ of R_{14}, $\pi'_{14}(t)$ is ground. Moreover, using the recursive path ordering $\succsim^{14}$ again, the following inequalities hold:

$$\mathsf{f} \succsim^{14}_{\pi'_{14}} \mathsf{s}, \quad \mathsf{g}(x) \succsim^{14}_{\pi'_{14}} \mathsf{h}(x), \quad \mathsf{h}(\mathsf{0}) \succsim^{14}_{\pi'_{14}} \mathsf{f}, \quad \mathsf{a} \succsim^{14}_{\pi'_{14}} \mathsf{b},$$

$$\mathsf{g}^\sharp(x) \succ^{14}_{\pi'_{14}} \mathsf{h}^\sharp, \quad \mathsf{h}^\sharp \succ^{14}_{\pi'_{14}} \mathsf{f}^\sharp.$$

Therefore, $\mathrm{SN}^{\rightsquigarrow_{R_{14}}}$ holds. □

Remark that we can easily modify Theorem 5.2.25 (1-b) and (2-c) stronger ones by using dependency graphs [4, 5] as follows:

(1-b) for every cycle $\mathcal{P}$ of dependency pairs of R,

  (i) for every dependency pair $\langle s, t \rangle \in \mathcal{P}$, $s \succsim_\pi t$, and

  (ii) for at least one dependency pair $\langle s_0, t_0 \rangle \in \mathcal{P}$, $s_0 \succ_\pi t_0$,

(2-c) for every cycle $\mathcal{P}$ of dependency pairs of R, for at least one dependency pair $\langle s_0, t_0 \rangle \in \mathcal{P}$, $\pi(t_0)$ is ground.

Such an extended theorem is more usable and powerful than Theorem 5.2.25.

The following corollary is a little weaker but easier to use the Theorem 5.2.25.

Corollary 5.2.27 *Let R be an EV-TRS, let $\rightsquigarrow_R$ have TRAT property and π be a simple argument filtering which eliminates all extra variables of R and $\mathcal{DP}_R$ and which satisfies $\pi(\mathcal{DP}_R) = \mathcal{DP}_{\pi(R)}$ up to renaming. Then, $SN^{\rightarrow_{\pi(R)}}$ implies $GSN^{\rightsquigarrow_R}$. Moreover, if $\pi(t)$ is ground for every dependency pair $\langle s, t \rangle \in \mathcal{DP}_R$, then $SN^{\rightsquigarrow_R}$ holds.*

Example 5.2.28 Consider the following EV-TRS R_{16} over $\{\mathsf{c}, \mathsf{e}, \mathsf{double}, \mathsf{s}, \mathsf{0}\}$ and its dependency pairs again:

$$R_{16} = \{\ \mathsf{double}(\mathsf{0}) \to \mathsf{0},\ \mathsf{double}(\mathsf{s}(x)) \to \mathsf{s}^2(\mathsf{double}(x)),\ \mathsf{e} \to \mathsf{double}(\mathsf{c}(x, x))\ \},$$

$$\mathcal{DP}_{R_{16}} = \{\ \langle \mathsf{double}^\sharp(\mathsf{s}(x)), \mathsf{double}^\sharp(x)\rangle \quad \langle \mathsf{e}^\sharp, \mathsf{double}^\sharp(\mathsf{c}(x,x))\rangle\ \}.$$

Given the argument filtering $\pi_{16}(\mathsf{c}) = [\,]$, we have the followings:

$$\pi_{16}(R_{16}) = \{\ \mathsf{double}(0) \to 0,\ \ \mathsf{double}(\mathsf{s}(x)) \to \mathsf{s}^2(\mathsf{double}(x)),\ \ \mathsf{e} \to \mathsf{double}(\mathsf{c})\ \},$$

$$\pi_{16}(\mathcal{DP}_{R_{16}}) = \{\ \langle \mathsf{double}^\sharp(\mathsf{s}(x)), \mathsf{double}^\sharp(x)\rangle \quad \langle \mathsf{e}^\sharp, \mathsf{double}^\sharp(\mathsf{c})\rangle\ \}.$$

Now $\pi_{16}(\mathcal{DP}_{R_{16}}) = \mathcal{DP}_{\pi_{16}(R_{16})}$ holds. On the other hand, $\mathrm{SN}^{\rightarrow}_{\pi_{16}(R_{16})}$ is easily shown. Therefore, $\mathrm{GSN}^{\rightsquigarrow}_{R_{16}}$ holds. □

According to Proposition 5.2.4 (b) and Theorem 5.2.5, each of the following is possible:

- to prove $\mathrm{GSN}^{\rightsquigarrow}_{R}$, and
- to prove that $\mathrm{SN}^{\rightsquigarrow}_{t}{}^{R}$ for all linear terms t,

for a right-linear EV-TRS R, and

- to prove $\mathrm{GSN}^{\rightsquigarrow}_{R}$, and
- to prove $\mathrm{SN}^{\rightsquigarrow}_{R}$

for a constructor EV-TRS R.

5.2.4 Discussion

In this subsection, we discuss the termination of narrowing of the example inverse EV-TRSs.

Consider the following constructor TRS R_7 over the signature $\{\mathsf{s}, 0, \mathsf{add}, \mathsf{mul}\}$ and its inverse EV-TRS R_{11} again:

$$\begin{aligned} R_7 = \{\ & \mathsf{add}(0,y) \to y, \quad \mathsf{add}(\mathsf{s}(x),y) \to \mathsf{s}(\mathsf{add}(x,y)), \\ & \mathsf{mul}(0,y) \to 0, \quad \mathsf{mul}(x,0) \to 0, \\ & \mathsf{mul}(\mathsf{s}(x),\mathsf{s}(y)) \to \mathsf{s}(\mathsf{add}(\mathsf{mul}(x,\mathsf{s}(y)),y)) \qquad \}, \end{aligned}$$

$$
\begin{aligned}
R_{11} &= \mathbb{U}(R_8)(= \mathcal{I}nv_{\mathsf{full}}(R_7)) \\
&= \{\ \mathsf{add}^{\#}(y) \to \mathsf{tp}_2(0, y), \\
&\quad\ \mathsf{add}^{\#}(\mathsf{s}(z)) \to \mathsf{u}_1(\mathsf{add}^{\#}(z)), \\
&\quad\ \mathsf{u}_1(\mathsf{tp}_2(x, y)) \to \mathsf{tp}_2(\mathsf{s}(x), y), \\
&\quad\ \mathsf{add}^{\#}(\mathsf{add}(x, y)) \to \mathsf{tp}_2(x, y), \\
&\quad\ \mathsf{mul}^{\#}(0) \to \mathsf{tp}_2(0, y), \\
&\quad\ \mathsf{mul}^{\#}(0) \to \mathsf{tp}_2(x, 0), \\
&\quad\ \mathsf{mul}^{\#}(\mathsf{s}(z)) \to \mathsf{u}_2(\mathsf{add}^{\#}(z)) \\
&\quad\ \mathsf{u}_2(\mathsf{tp}_2(w, y)) \to \mathsf{u}_3(\mathsf{mul}^{\#}(w), y) \\
&\quad\ \mathsf{u}_3(\mathsf{tp}_2(x, \mathsf{s}(y)), y) \to \mathsf{tp}_2(\mathsf{s}(x), \mathsf{s}(y)) \\
&\quad\ \mathsf{mul}^{\#}(\mathsf{mul}(x, y)) \to \mathsf{tp}_2(x, y) \qquad \}.
\end{aligned}
$$

In Example 5.1.21, we have shown that the tree representation of narrowing sequences starting from $\mathsf{mul}^{\#}(\mathsf{s}^4(0))$ is finite up to renaming, that is, there is no infinite narrowing sequences starting from $\mathsf{mul}^{\#}(\mathsf{s}^4(0))$. This fact gives a conjecture, which we will prove later, that $\mathsf{mul}^{\#}(\mathsf{s}^n(0))$ $(n \geq 0)$ is strongly normalizing with respect to $\rightsquigarrow_{R_{11}}$.

Unfortunately, R_{11} is not ground strongly normalizing with respect to $\rightsquigarrow_{R_{11}}$, that is, $\mathrm{GSN}^{\rightsquigarrow_{R_{11}}}$ does not hold. In fact, there is an infinite narrowing sequence as follows:

$$
\begin{aligned}
\mathsf{u}_2(\mathsf{mul}^{\#}(0)) &\rightsquigarrow_{R_{11}} \mathsf{u}_2(\mathsf{tp}_2(x_0, 0)) \\
&\rightsquigarrow_{R_{11}} \mathsf{u}_3(\mathsf{mul}^{\#}(x_0), 0) \\
&\overset{\{x_0 \mapsto \mathsf{s}(x_1)\}}{\rightsquigarrow}_{R_{11}} \mathsf{u}_3(\mathsf{u}_2(\mathsf{add}^{\#}(x_1)), 0) \\
&\overset{\{x_1 \mapsto \mathsf{s}(x_2)\}}{\rightsquigarrow}_{R_{11}} \mathsf{u}_3(\mathsf{u}_2(\mathsf{u}_1(\mathsf{add}^{\#}(x_2))), 0) \\
&\overset{\{x_2 \mapsto \mathsf{s}(x_3)\}}{\rightsquigarrow}_{R_{11}} \mathsf{u}_3(\mathsf{u}_2(\mathsf{u}_1(\mathsf{u}_1(\mathsf{add}^{\#}(x_3)))), 0) \\
&\overset{\{x_3 \mapsto \mathsf{s}(x_4)\}}{\rightsquigarrow}_{R_{11}} \mathsf{u}_3(\mathsf{u}_2(\mathsf{u}_1(\mathsf{u}_1(\mathsf{u}_1(\mathsf{add}^{\#}(x_4))))), 0) \\
&\overset{\{x_3 \mapsto \mathsf{s}(x_4)\}}{\rightsquigarrow}_{R_{11}} \cdots.
\end{aligned}
$$

However, the following claim which makes the inverse computation by R_{11} practical, can be easily proved by induction on the natural number k of the term $\mathsf{s}^k(0)$.

Claim 5.2.29 $\mathsf{add}^{\#}(\mathsf{s}^m(0))$ *and* $\mathsf{mul}^{\#}(\mathsf{s}^n(0))$ *(where* $m, n \geq 0$*) are strongly normalizing with respect to* $\rightsquigarrow_{R_{11}}$*, that is,* $SN^{\rightsquigarrow_{R_{11}}}_{\mathsf{add}^{\#}(\mathsf{s}^m(0))}$ *and* $SN^{\rightsquigarrow_{R_{11}}}_{\mathsf{mul}^{\#}(\mathsf{s}^n(0))}$ *hold.*

According to the above claim, the inverse computation of R_7 is considered to be terminating.

Arbitral reduction systems [26, 31] can be formalized as EV-TRSs whose right-hand sides are extra variables. They introduced an Ω-reduction system to sim-

ulate the reduction sequence, which is a special case of narrowing extended in this paper. Although they are terminating, Theorem 5.2.25 does not work to show their termination. The reason is that argument filtering method in this thesis cannot eliminate all extra variables of collapsing rules. To overcome this problem is one of future works.

5.3 Computation Strategy of Inverse EV-TRSs

It is well-known that all solution search is expensive. In addition, the inverse EV-TRSs generated by our inverse compilers are not confluent in general. However, we have also shown that our inverse EV-TRSs obtained from left-linear constructor TRSs are right-linear. Hence, if they are TRSs, we use the innermost strategy to improve efficiency of all solution search by rewrite relation because of the following result;

> For any right-linear overlay TRS R, all normal forms (with respect to $\rightarrow_R$) of a given terminating term with respect to $\rightarrow_R$, can be obtained by the innermost strategy [47].

Since a lot of practical constructor TRSs are erasing and then inverse systems generated by our transformation from them are not TRSs, we would like to extend the above result to narrowing of EV-TRSs. However, the above result does not hold for narrowing on EV-TRSs. Then, we try to find a condition in order that the above result holds for narrowing.

In this section, we show that for a linear constructor EV-TRS, every normal form (with respect to narrowing) of a given linear term which is terminating with respect to innermost narrowing, can be obtained by innermost narrowing. We also show that for right-linear EV-TRSs *basic narrowing* [28] is sufficient to simulate rewrite relation of EV-TRSs. That is, for any right-linear EV-TRS, every ground rewrite sequence can be simulated by a basic narrowing sequence. Through these works, we improve the efficiency of narrowing derivation of right-linear EV-TRSs.

Some results in our previous papers [39, 40] are wrong since the results there lack a condition that EV-TRSs are left-linear. The result of this section is their revised version.

5.3.1 Innermost Narrowing of Linear Constructor EV-TRSs

The main result in [47] is that for a right-linear terminating overlay TRS R, innermost rewrite relation can find all normal forms (with respect to $\to$) of a given term t, that is, $t \xrightarrow{*}_R u \in NF^{\to_R}(\mathcal{F}, \mathcal{X})$ implies $t \xrightarrow[\text{in}]{*}_R u$. This subsection shows that such a claim also holds for narrowing of linear constructor EV-TRSs which are terminating with respect to innermost narrowing. Remark that these results do not require confluence.

The reason why we restrict overlay systems to constructor systems, is that overlay systems do not guarantee the intention of overlay to be satisfied in the narrowing; Inside critical peak with respect to narrowing arises even for overlay systems. It is shown by the following counterexample for overlay systems.

Example 5.3.1 Consider the following right-linear overlay EV-TRS R_{17} with $\mathrm{SN}^{\rightsquigarrow_{R_{17}}}$ on linear terms:

$$R_{17} = \{\, \mathsf{f}(\mathsf{g}(\mathsf{a})) \to \mathsf{b},\ \ \mathsf{g}(\mathsf{b}) \to \mathsf{a} \,\}.$$

There are two normal forms b and $\mathsf{f}(\mathsf{a})$ of $\mathsf{f}(\mathsf{g}(x))$ with respect to $\rightsquigarrow_{R_{17}}$. However, $\mathsf{f}(\mathsf{g}(x))$ cannot be narrowable to b by innermost strategy. □

The situation in the above example is caused from the fact that there exists an inside critical peak $\langle \mathsf{b}, \mathsf{f}(\mathsf{a}) \rangle$ with respect to $\rightsquigarrow_{R_{17}}$, which is constructed from $\mathsf{f}(\mathsf{g}(x))$ by the rules $\mathsf{f}(\mathsf{g}(\mathsf{a})) \to \mathsf{b}$ and $\mathsf{g}(\mathsf{b}) \to \mathsf{a}$ (see Figure 5.7). On the other

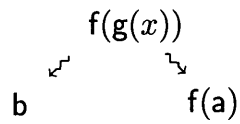

Figure 5.7: A critical peak of narrowing on R_{17}, which is constructed from $\mathsf{f}(\mathsf{g}(x))$.

hand, constructor systems does not raise such inside critical peaks with respect to narrowing, that is, they satisfy the intention of non-inside-overlapping. Therefore, we focus on constructor systems.

Next we show a counterexample for non-left-linear constructor EV-TRSs which is right-linear.

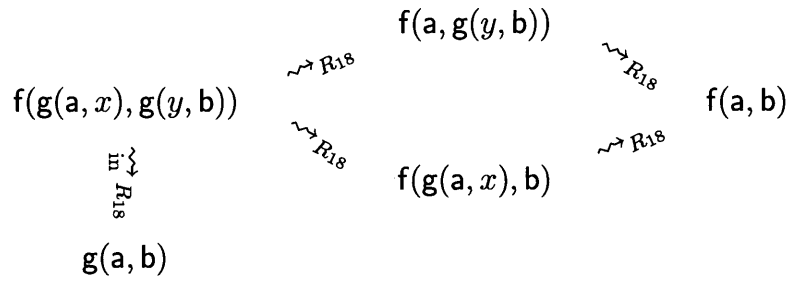

Figure 5.8: Narrowing sequences starting from $\mathsf{f}(\mathsf{g}(\mathsf{a}, x), \mathsf{g}(y, \mathsf{b}))$.

Example 5.3.2 Consider the following constructor TRS which is right-linear but not left-linear:

$$R_{18} = \{\ \mathsf{f}(x, x) \to x, \quad \mathsf{g}(\mathsf{a}, \mathsf{a}) \to \mathsf{a}, \quad \mathsf{g}(\mathsf{b}, \mathsf{b}) \to \mathsf{b}\ \}.$$

Normal forms of the term $\mathsf{f}(\mathsf{g}(\mathsf{a}, x), \mathsf{g}(y, \mathsf{b}))$ with respect to $\to_{R_{18}}$ are $\mathsf{g}(\mathsf{a}, \mathsf{b})$ and $\mathsf{f}(\mathsf{a}, \mathsf{b})$. Term $\mathsf{g}(\mathsf{a}, \mathsf{b})$ cannot be obtained by innermost narrowing (see Figure 5.8). □

We first recall the following property of narrowing, as shown in Proposition 5.2.4;

> Narrowing of any right-linear EV-TRS R which is restricted to a relation on linear terms, is monotone.

This fact makes it easy to treat variables in narrowing. Concretely, since the narrowing relation $s \equiv C[s|_p]_p \rightsquigarrow_R^{[p,\rho]} (C[r]_p)\sigma \equiv t$ where $\rho : l \to r \in R$ and $\sigma = \mathtt{mgu}(s, C[l]_p)$, implies $t \equiv C[r\sigma]_p$ and $\mathcal{V}ar(C) \cap \mathcal{V}ar(r\sigma) = \emptyset$, we do not need to care about the influence of unifier σ on context C.

The following proposition shows the composition and decomposition of narrowing sequences, which are closed under contexts.

Proposition 5.3.3 *Let R be an EV-TRS over a signature $\mathcal{F}$. Let $s_1, \ldots, s_n$ be terms such that $\mathcal{V}ar(s_i) \cap \mathcal{V}ar(s_j) = \emptyset$ for $i \neq j$.*

(a) If $s_i \overset{*}{\rightsquigarrow}_R t_i$ and $\mathcal{V}ar(t_i) \cap \mathcal{V}ar(t_j) = \emptyset$ for $i \neq j$, then $f(s_1, \ldots, s_n) \overset{*}{\rightsquigarrow}_R f(t_1, \ldots, t_n)$ for any n-ary function symbol f in $\mathcal{F}$.

(b) For every n-ary function symbol $f \in \mathcal{F}$, $f(s_1, \ldots, s_n) \overset{*}{\rightsquigarrow}_R^{\varepsilon<} f(t_1, \ldots, t_n)$ implies that $s_i \overset{*}{\rightsquigarrow}_R t_i$ for every i, and $\mathcal{V}ar(t_i) \cap \mathcal{V}ar(t_j) = \emptyset$ for $i \neq j$.

Proof. In the both cases, It follows from the condition $\mathcal{V}ar(s_i) \cap \mathcal{V}ar(s_j) = \emptyset$ and the definition of $\rightsquigarrow_R$ that variables appearing in $s_i \overset{*}{\rightsquigarrow}_R t_i$ and $s_j \overset{*}{\rightsquigarrow}_R t_j$ ($i \neq j$) are disjoint. Then, it also follows from the definition of $\rightsquigarrow_R$ that narrowing sequences $s_i \overset{*}{\rightsquigarrow}_R t_i$ and $s_j \overset{*}{\rightsquigarrow}_R t_j$ are independent of each other where $i \neq j$. Therefore, it is clear that this proposition holds. □

Next, we discuss a property on innermost narrowing sequences.

Lemma 5.3.4 *Let R be an EV-TRS. Then, $s \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R t \in NF^{\rightsquigarrow_R}(\mathcal{F}, \mathcal{X})$ implies $s \overset{*}{\underset{\text{in}}{\rightsquigarrow}}{}^{\varepsilon <}_R u \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R t$ for some term u of which every proper subterm is in normal forms with respect to $\rightsquigarrow_R$.*

Proof. If $s \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R t$ does not contain $\overset{\varepsilon}{\underset{\text{in}}{\rightsquigarrow}}_R$, then we take t as u. Otherwise, we can assume that $s \overset{*}{\underset{\text{in}}{\rightsquigarrow}}{}^{\varepsilon <}_R u \overset{\varepsilon}{\underset{\text{in}}{\rightsquigarrow}}_R u' \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R t$, and hence the claim holds. □

Let $\mathcal{F}$ be a signature and σ be a substitution. We say that σ is *linear* if it satisfies both of the followings:

- $x\sigma$ is linear for all $x \in \mathcal{D}om(\sigma)$, and
- for all $x, y \in \mathcal{D}om(\sigma)$, $x \not\equiv y$ implies $\mathcal{V}ar(x\sigma) \cap \mathcal{V}ar(y\sigma) = \emptyset$.

Let $\rightarrow$ be a relation on terms over $\mathcal{F}$. Then, we say that a substitution σ' is *σ-normalized with respect to $\rightarrow$* if $\mathcal{D}om(\sigma) = \mathcal{D}om(\sigma')$ and $x\sigma \overset{*}{\rightarrow} x\sigma' \in NF^{\rightarrow}(\mathcal{F}, \mathcal{X})$ for all $x \in \mathcal{D}om(\sigma)$ (similarly for $\underset{\text{in}}{\rightarrow}$). Suppose that $\mathrm{WN}^{\rightarrow}_{x\sigma}$ for all $x \in \mathcal{D}om(\sigma)$. Then, it is clear that there exists at least one σ-normalized substitution σ' with respect to $\rightarrow$. Moreover, we can assume without loss of generality that σ' is linear if σ is linear.

Lemma 5.3.5 *Let R be a right-linear EV-TRS. Let s be a linear term and σ be a linear substitution such that $\mathcal{D}om(\sigma) \subseteq \mathcal{V}ar(s)$ and $SIN^{\rightsquigarrow_R}_{x\sigma}$ for all $x \in \mathcal{D}om(\sigma)$. Then, $s\sigma \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R t \in NF^{\rightsquigarrow_R}(\mathcal{F}, \mathcal{X})$ implies $s\sigma \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R s\sigma_\perp \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R t$ for some linear σ-normalized substitution $\sigma_\perp$ with respect to $\rightsquigarrow_R$.*

Proof. We prove by induction on the structure of term s. Since the case that s is a variable is trivial, we assume $s \equiv f(s_1, \ldots, s_n)$. It follows from Lemma 5.3.4 that $s\sigma \equiv f(s_1\sigma, \ldots, s_n\sigma) \overset{*}{\underset{\text{in}}{\rightsquigarrow}}{}^{\varepsilon <}_R f(u_1, \ldots, u_n) \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R t$ and $u_i \in NF^{\rightsquigarrow_R}(\mathcal{F}, \mathcal{X})$ for $1 \leq i \leq n$. From Proposition 5.2.4 (a), $f(u_1, \ldots, u_n)$ is linear and hence we have $\mathcal{V}ar(u_i) \cap \mathcal{V}ar(u_j) = \emptyset$ for each i and j with $i \neq j$. From Proposition 5.3.3 (b), we have $s_i\sigma \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R u_i$ for $1 \leq i \leq n$. Then, by the induction hypothesis, there

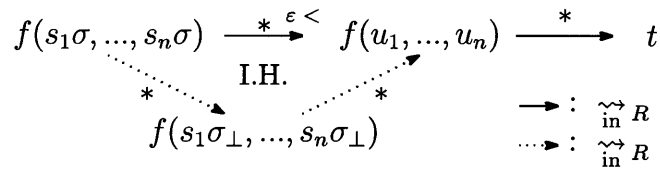

Figure 5.9: Proof sketch of Lemma 5.3.5.

exists a linear $\sigma|_{\mathcal{V}ar(s_i)}$-normalized substitution σ_i with respect to $\underset{\text{in}}{\rightsquigarrow} R$ such that $s_i\sigma \overset{*}{\underset{\text{in}}{\rightsquigarrow}} R\ s_i\sigma_i \overset{*}{\underset{\text{in}}{\rightsquigarrow}} R\ u_i$.

Now we can assume without loss of generality that for each i and j with $i \neq j$, $\mathcal{VR}an(\sigma_i) \cap \mathcal{VR}an(\sigma_j) = \emptyset$. Then, it follows from Proposition 5.3.3 (a) that $f(s_1\sigma_1, \ldots, s_n\sigma_n) \overset{*}{\underset{\text{in}}{\rightsquigarrow}}{}^{\varepsilon <}_{R} f(u_1, \ldots, u_n)$. From the linearity of s, $\sigma_\perp = \bigcup_{i=1}^{n} \sigma_i|_{\mathcal{V}ar(s_i)}$ is a substitution. Moreover, since all σ_is are linear $\sigma|_{\mathcal{V}ar(s_i)}$-normalized substitutions with respect to $\underset{\text{in}}{\rightsquigarrow} R$ and their variables are disjoint, $\sigma_\perp$ is also a linear $\sigma|_{\mathcal{V}ar(s)}$-normalized substitution with respect to $\underset{\text{in}}{\rightsquigarrow} R$. Therefore, we have

$$f(s_1\sigma, \ldots, s_n\sigma) \overset{*}{\underset{\text{in}}{\rightsquigarrow}} R\ f(s_1\sigma_\perp, \ldots, s_n\sigma_\perp) \overset{*}{\underset{\text{in}}{\rightsquigarrow}} R\ f(u_1, \ldots, u_n) \overset{*}{\underset{\text{in}}{\rightsquigarrow}} R\ t$$

and $\sigma_\perp$ is a linear $\sigma|_{\mathcal{V}ar(s)}$-normalized substitution with respect to $\underset{\text{in}}{\rightsquigarrow} R$ (see Figure 5.9). □

To represent the common part of two terms which are unifiable, we define the operator $\diamond$ as follows:

- $x \diamond t = x$ where $x \in \mathcal{X}$,
- $t \diamond x = x$ where $x \in \mathcal{X}$ and t is not a variable, and
- $f(s_1, \ldots, s_n) \diamond f(t_1, \ldots, t_n) = f(s_1 \diamond t_1, \ldots, s_n \diamond t_n)$.

Let θ be a substitution, we denote the set of all linear θ-normalized substitution with respect to $\underset{\text{in}}{\rightsquigarrow} R$ by $Norm(\theta)$.

The following proposition shows the relationship between linear terms and their most general unifier in innermost narrowing.

Proposition 5.3.6 *Let R be a right-linear constructor EV-TRS, and s, t be linear terms such that $\mathcal{V}ar(s) \cap \mathcal{V}ar(t) = \emptyset$. Suppose that $\sigma = \mathtt{mgu}(s, t)$. Let θ be a linear substitution such that $\sigma|_{\mathcal{V}ar(s)} = \theta|_{\mathcal{V}ar(s)}$ and $\mathcal{VR}an(\theta|_{\mathcal{V}ar(t)}) \cap \mathcal{VR}an(\sigma|_{\mathcal{V}ar(s)}) = \emptyset$. Then, θ is a most general unifier of $(s \diamond t)\sigma|_{\mathcal{V}ar(t)}$ and t, that is, $\theta = \mathtt{mgu}((s \diamond t)\sigma|_{\mathcal{V}ar(t)}, t)$.*

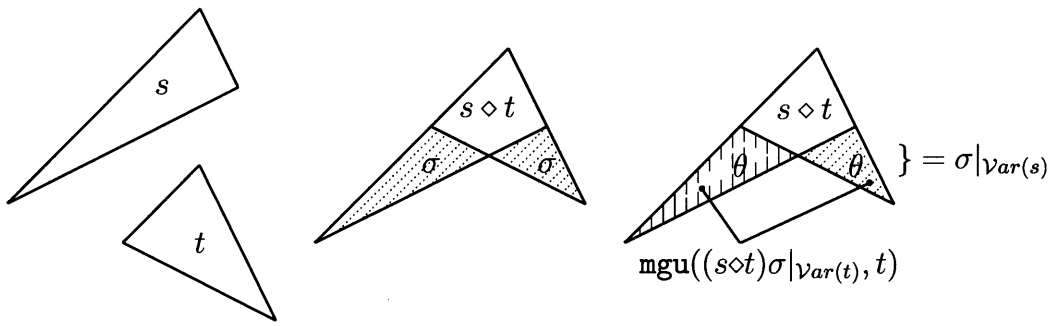

Figure 5.10: Relationship between linear terms s, t and $\sigma = \mathtt{mgu}(s,t)$.

Proof. This proposition holds clearly (see Figure 5.10). □

Next we give a lemma related with one-step narrowing. The lack of the condition of the left-linearity of R (in Lemma 5.3.7) causes the wrong results in [39, 40]. The following lemma is a revision of them.

Lemma 5.3.7 *Let R be a linear constructor EV-TRS. Let s be a linear term and suppose that $WIN^{\overset{\sim}{\to}R}_{s'}$ for every subterm s' of s, and $\rho : l \to r \in R$. Suppose that $s \overset{[\varepsilon,\rho]}{\underset{R}{\leadsto}} r\sigma$ where $\sigma = \mathtt{mgu}(s,l)$, and $\sigma_\perp$ is a linear $\sigma|_{\mathcal{V}ar(r)}$-normalized substitution with respect to $\underset{\mathrm{in}}{\leadsto}_R$. Then, there exists a linear term t such that $s \overset{*}{\underset{\mathrm{in}}{\leadsto}}_R t \overset{[\varepsilon,\rho]}{\underset{\mathrm{in}}{\leadsto}}_R r\sigma_\perp$.*

Proof. From the definition of $\leadsto_R$, we can assume without loss of generality that $\mathcal{V}ar(s) \cap \mathcal{V}ar(l,r) = \emptyset$. It clearly follows from the linearity of s and l that $s \diamond l$ is also linear and $s \equiv (s \diamond l)\sigma|_{\mathcal{V}ar(l)}$ without loss of generality. Since s and l are linear and $\sigma = \mathtt{mgu}(s,l)$, every term in $\mathcal{VR}an(\sigma|_{\mathcal{V}ar(l)})$ is either a variable or a renaming of a subterm of s and σ is linear. Since $\mathrm{WIN}^{\overset{\sim}{\to}R}_{s'}$ for every subterm s' of s, $\mathrm{WIN}^{\overset{\sim}{\to}R}_{u}$ for every term $u \in \mathcal{VR}an(\sigma|_{\mathcal{V}ar(l)})$. Then, for every $x \in \mathcal{D}om(\sigma|_{\mathcal{V}ar(l)})$, there exists a normal form u' of $x\sigma$ with respect to $\leadsto_R$ such that $x\sigma \overset{*}{\underset{\mathrm{in}}{\leadsto}}_R u' \in NF^{\leadsto_R}(\mathcal{F},\mathcal{X})$. Hence, the following relation holds:

$$Norm(\sigma|_{\mathcal{V}ar(r)}) = \{\, \sigma'_\perp|_{\mathcal{V}ar(r)} \mid \sigma'_\perp \in Norm(\sigma|_{\mathcal{V}ar(l,r)}) \,\}.$$

Then, there exists a linear $\sigma|_{\mathcal{V}ar(l,r)}$-normalized substitution $\sigma'_\perp$ with respect to $\underset{\mathrm{in}}{\leadsto}_R$ such that $\sigma_\perp = \sigma'_\perp|_{\mathcal{V}ar(r)}$. Since σ and $\sigma_\perp$ are linear and $\sigma_\perp$ is a $\sigma|_{\mathcal{V}ar(l,r)}$-normalized substitution, we can assume without loss of generality that $\mathcal{VR}an(\sigma_\perp|_{\mathcal{V}ar(l)}) \cap \mathcal{VR}an(\sigma|_{\mathcal{V}ar(s)}) = \emptyset$. Then, it follows form Proposition 5.3.6 that $\sigma'_\perp$ is a most general unifier of $(s \diamond l)\sigma_\perp|_{\mathcal{V}ar(l)}$ and l. Therefore, $s \equiv (s \diamond l)\sigma|_{\mathcal{V}ar(l)} \overset{*}{\underset{\mathrm{in}}{\leadsto}}_R (s \diamond l)\sigma'_\perp|_{\mathcal{V}ar(l)} \overset{[\varepsilon,\rho]}{\underset{R}{\leadsto}} r\sigma'_\perp \equiv r\sigma_\perp$. □

The following proposition shows the relationship between $\mathrm{WIN}^{\rightsquigarrow_R}$ property of subterms.

Proposition 5.3.8 *Let R be a right-linear EV-TRS and t be a linear term. If $WIN_t^{\rightsquigarrow_R}$, then $WIN_u^{\rightsquigarrow_R}$ for every subterm u of t.*

Proof. Suppose that there exists an infinite sequence $u \underset{\text{in}}{\rightsquigarrow}_R u_1 \underset{\text{in}}{\rightsquigarrow}_R u_2 \underset{\text{in}}{\rightsquigarrow}_R \cdots$ which is starting from a subterm u of t. Let $t \equiv C[u]$. Since $\underset{\text{in}}{\rightsquigarrow}_R$ on linear terms is monotone, we have the following infinite sequence:

$$t \equiv C[u] \underset{\text{in}}{\rightsquigarrow}_R C[u_1] \underset{\text{in}}{\rightsquigarrow}_R C[u_2] \underset{\text{in}}{\rightsquigarrow}_R \cdots.$$

Therefore, $\mathrm{WIN}^{\rightsquigarrow_R}$ does not hold. □

Now we show the following theorem on innermost narrowing, which is a revision of those in [39, 40].

Theorem 5.3.9 *Let R be a linear constructor EV-TRS. Let s be a linear term and suppose that $WIN_{s'}^{\rightsquigarrow_R}$ for every term s' with $s \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R s'$. Then, $s \overset{*}{\rightsquigarrow}_R t \in NF^{\rightsquigarrow_R}(\mathcal{F}, \mathcal{X})$ implies $s \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R t$.*

Proof. We prove this theorem by induction on n of $s \overset{n}{\rightsquigarrow}_R t$. Since the case of $n = 0$ is trivial, we assume that $s \equiv C[u]_p \rightsquigarrow_R^{[p,\rho]} (C[r]_p)\sigma \overset{n-1}{\rightsquigarrow}_R t$, where $\rho : l \to r \in R$ and $\sigma = \mathtt{mgu}(s, C[l]_p)$.

It follows from the linearity of s and Proposition 5.2.4 (a) that $(C[r]_p)\sigma \equiv C[r\sigma]_p$ and $C[r\sigma]_p$ is linear. Since $\mathrm{WIN}_{s'}^{\rightsquigarrow_R}$ for every term s' with $s \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R s'$, it also holds that $\mathrm{WIN}_{s''}^{\rightsquigarrow_R}$ for every term s'' with $C[r\sigma]_p \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R s''$. Then, by the induction hypothesis, we have $C[r\sigma]_p \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R t$. From Lemma 5.3.5, there exists a linear $\sigma|_{\mathcal{V}ar(r)}$-normalized substitution $\sigma_\perp$ with respect to $\underset{\text{in}}{\rightsquigarrow}_R$ such that $C[r\sigma]_p \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R C[r\sigma_\perp]_p \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R t$.

On the other hand, it follows from Proposition 5.3.8 that $\mathrm{WIN}_{u'}^{\rightsquigarrow_R}$ for every subterm u' of u. Then, it follows from Lemma 5.3.7 that there exists a linear term v such that $u \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R^{\epsilon<} v \underset{\text{in}}{\rightsquigarrow}_R^{[\epsilon,\rho]} r\sigma_\perp$. Since $\rightsquigarrow_R$ on linear terms is monotone, we have $C[u]_p \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R^{p<} C[v]_p \underset{\text{in}}{\rightsquigarrow}_R^{[p,\rho]} C[r\sigma_\perp]_p$ without loss of generality. Therefore, $s \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R C[r\sigma_\perp]_p \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R t$ (see Figure 5.11). □

It is clear that for every term s, $\mathrm{SIN}_s^{\rightsquigarrow_R}$ implies $\mathrm{WIN}_{s'}^{\rightsquigarrow_R}$ for all terms s' with $s \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R s'$. Then, we obtain the following corollary from Theorem 5.3.9.

Corollary 5.3.10 *Let R be a linear constructor EV-TRS. Let s be a linear term such that $SIN_s^{\rightsquigarrow_R}$. Then, $s \overset{*}{\rightsquigarrow}_R t \in NF^{\rightsquigarrow_R}(\mathcal{F}, \mathcal{X})$ implies $s \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R t$.*

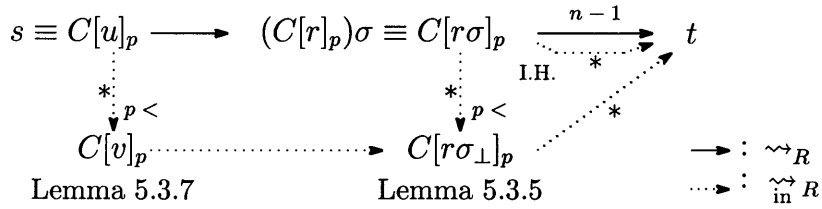

Figure 5.11: Proof sketch of Theorem 5.3.9.

It is clear that $\text{GSIN}^{\rightsquigarrow_R}$ implies $\text{GSIN}_s^{\rightsquigarrow_R}$ for every ground term s. Then, we obtain the following corollary from Corollary 5.3.10.

Corollary 5.3.11 *Let R be a linear constructor EV-TRS with $GSIN^{\rightsquigarrow_R}$. If $s \overset{*}{\rightsquigarrow}_R t \in NF^{\rightsquigarrow_R}(\mathcal{F}, \mathcal{X})$ for ground term s then $s \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R t$.*

While no method to prove $\text{WN}^{\rightarrow_R}$, $\text{WIN}^{\rightarrow_R}$, $\text{WN}^{\rightsquigarrow_R}$ or $\text{WIN}^{\rightsquigarrow_R}$ is known yet, $\text{GSN}^{\rightsquigarrow_R}$ for a right-linear EV-TRS R can be proved by the termination proof method of narrowing, such as the methods proposed in Section 5.2, the method in [9] and so on. Then, Corollary 5.3.10 and 5.3.11 are more practical than Theorem 5.3.9.

From the definition of monotonicity, the following theorem holds obviously.

Theorem 5.3.12 *Let $\rightarrow$ be a monotone relation on terms, and an ATRS $\mathcal{A} = (\mathcal{T}(\mathcal{F}, \mathcal{X}), \rightarrow)$. Then, $s \overset{*}{\underset{\text{in}}{\rightarrow}} t \in NF^{\rightarrow}(\mathcal{F}, \mathcal{X})$ implies $s \overset{*}{\underset{\text{lin}}{\rightarrow}} t$ and $s \overset{*}{\underset{\text{rin}}{\rightarrow}} t$.*

Hence, Theorem 5.3.9, Corollary 5.3.10 and 5.3.11 hold for both $\overset{\rightsquigarrow}{\underset{\text{lin}}{}}_R$ and $\overset{\rightsquigarrow}{\underset{\text{rin}}{}}_R$ instead of $\overset{\rightsquigarrow}{\underset{\text{in}}{}}_R$.

In the sequel, we discuss the relationship between normal forms with respect to $\rightarrow_R$ and $\rightsquigarrow_R$. For constructor normal forms, the following holds.

Theorem 5.3.13 *Let R be a linear constructor EV-TRS. Let s be a linear term with $SIN_s^{\rightsquigarrow_R}$ and σ be a substitution. If $s\sigma \overset{*}{\rightarrow}_R t$ and t is a constructor term then there exist a linear constructor term u and a substitution θ such that $s \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R u \in NF^{\rightsquigarrow_R}(\mathcal{F}, \mathcal{X})$ and $t \equiv u\theta$.*

Proof. Since we have $s\sigma \overset{*}{\rightarrow}_R t$, from Theorem 5.1.4, there exist a term u and a substitution θ such that $s \overset{*}{\rightsquigarrow}_R u$ and $t \equiv u\theta$. It follows from the linearity of s that u is linear. Since t is a constructor term, u is also a constructor term and hence $u \in NF^{\rightsquigarrow_R}(\mathcal{F}, \mathcal{X})$. Therefore, $s \overset{*}{\underset{\text{in}}{\rightsquigarrow}}_R u$ from Theorem 5.3.9. □

The above theorem does not hold in the case that t is not a constructor term. For example, consider the following linear constructor EV-TRS:

$$R_{19} = \{\, \mathsf{f}(x,y) \to x, \quad \mathsf{g}(\mathsf{s}(x)) \to \mathsf{h}(0), \quad \mathsf{a} \to \mathsf{f}(\mathsf{g}(y),0) \,\}.$$

Then we have the following rewrite sequences starting from a:

$$\begin{array}{l} \mathsf{a} \to_{R_{19}} \mathsf{f}(\mathsf{g}(\mathsf{s}(0)),0) \to_{R_{19}} \mathsf{f}(\mathsf{h}(0),0) \to_{R_{19}} \mathsf{h}(0) \\ \qquad \searrow_{R_{19}} \quad \mathsf{f}(\mathsf{g}(0),0) \to_{R_{19}} \mathsf{g}(0). \end{array}$$

On the other hand, narrowing sequences starting from a is as follows:

$$\begin{array}{l} \mathsf{a} \overset{\leadsto}{}_{\mathrm{in}} R_{19} \; \mathsf{f}(\mathsf{g}(y),0) \; \{y \mapsto \mathsf{s}(z)\} \overset{\leadsto}{}_{\mathrm{in}} R_{19} \; \mathsf{f}(\mathsf{h}(0),0) \overset{\leadsto}{}_{\mathrm{in}} R_{19} \; \mathsf{h}(0) \\ \qquad\qquad\qquad\qquad \searrow_{R_{19}} \; \mathsf{g}(y). \end{array}$$

There exists no normal form of a with respect to $\overset{\leadsto}{}_{\mathrm{in}} R_{19}$ which has $\mathsf{g}(0)$ as an instance. In the following subsection, we show that basic narrowing, which is less efficient than innermost narrowing, but more than ordinary narrowing, solves such a problem, that is, that for all reachable terms t from a term $s\sigma$ where s is linear, there exists a term u and a substitution θ such that $t \equiv u\theta$ and s is narrowable to u by basic narrowing.

5.3.2 Basic Narrowing of Right-Linear EV-TRSs

In this subsection, we show that for right-linear EV-TRSs, all terms which are reachable by rewrite sequences from a given term can be obtained by basic narrowing. We also show that for linear EV-TRSs, basic narrowing is equivalent to narrowing.

The basic narrowing of EV-TRSs is defined similarly to that of TRSs [28, 36].

Definition 5.3.14 *Let R be an EV-TRS, $\to$ be a relation either $\to_R$ or $\leadsto_R$ and $\rho_i : l_i \to r_i \in R$. Let $t_1 \to^{[p_1,\rho_1]} t_2 \to^{[p_2,\rho_2]} \cdots \to^{[p_{n-1},\rho_{n-1}]} t_n$ be a $\to$-sequence. We inductively define sets of positions $B_1, \ldots, B_n$ as follows:*

- *$B_1 \subseteq \mathcal{O}_{\mathcal{F}}(t_1)$ which is closed under prefix [3], and*
- *$B_{i+1} = (B_i \setminus \{\, q \in B_i \mid p_i \leq q \,\}) \cup \{\, p_i q \mid q \in \mathcal{O}_{\mathcal{F}}(r_i) \,\}$ for $1 \leq i < n$.*

For $1 \leq i \leq n$, positions in B_i are referred to as basic positions *of t_i and positions in $\mathcal{O}(t_i) \setminus B_i$ are called* non-basic *of t_i. We say that the above $\to$-sequence is* based *on B_1 if $p_i \in B_i$ for $1 \leq i < n$. Especially, it is simply called* basic *if $B_1 = \mathcal{O}_{\mathcal{F}}(t_1)$.*

[3] That is, if $p < q$ and $q \in B_1$ then $p \in B_1$.

Note that $B_2, \ldots, B_n$ above are clearly closed under prefix. This thesis assumes that sets of basic positions are closed under prefix. We use $\underset{\mathrm{b}}{\rightarrow}_R$ and $\underset{\mathrm{b}}{\rightsquigarrow}_R$ to represent basic rewrite and narrowing sequences, respectively.

The following proposition which holds on TRSs also holds on EV-TRSs since basic rewrite sequences are clearly EV-safe and then EV-safe rewrite sequences are simulated by narrowing (see Theorem 5.1.16).

Proposition 5.3.15 ([28]) *Let R be an EV-TRS. Then, $s\sigma \xrightarrow{*}_R t$ based on $\mathcal{O}_{\mathcal{F}}(s)$ implies $s \underset{\mathrm{b}}{\overset{*}{\rightsquigarrow}}_R u$ and $t \equiv u\theta$ for some term u and substitution θ.*

The following proposition which holds for EV-TRSs, means that innermost derivations $\underset{\mathrm{in}}{\rightarrow}_R$ and $\underset{\mathrm{in}}{\rightsquigarrow}_R$ are included in basic reductions $\underset{\mathrm{b}}{\rightarrow}_R$ and $\underset{\mathrm{b}}{\rightsquigarrow}_R$, respectively.

Proposition 5.3.16 ([28]) *Let R be an EV-TRS and $\rightarrow$ be a relation either $\rightarrow_R$ or $\rightsquigarrow_R$. Let σ be an $NF^{\rightarrow}(\mathcal{F}, \mathcal{X})$-substitution. Then, $t_0\sigma \underset{\mathrm{in}}{\rightarrow} t_1 \underset{\mathrm{in}}{\rightarrow} \cdots$ implies $t_0\sigma \rightarrow t_1 \rightarrow \cdots$ based on $\mathcal{O}_{\mathcal{F}}(t_0)$.*

In the following propositions which also holds obviously, we show the composition of basic reduction sequences, and the relationships between a a basic reduction sequence and a substitution, and between a basic rewrite sequence and a context, respectively.

Proposition 5.3.17 *Let R be an EV-TRS. Let s and t be terms and θ be an substitution, and suppose that $s \xrightarrow{*}_R t$ is a rewrite sequence based on $B_s \subseteq \mathcal{O}_{\mathcal{F}}(s)$. Then, there exists a sequence $s\theta \xrightarrow{*}_R t\theta$ based on B_s.*

Proposition 5.3.18 *Let R be an EV-TRS and suppose that $\rightarrow$ be a relation either $\rightarrow_R$ or $\rightsquigarrow_R$. Let s and t be terms, $s \xrightarrow{*} t$ be a $\rightarrow$-sequence based on $B_s \subseteq \mathcal{O}_{\mathcal{F}}(s)$, $t \equiv l\sigma \rightarrow^{[\epsilon,\rho]} r\sigma \equiv u$ be a $\rightarrow$-sequence and $u \xrightarrow{*} v$ be a $\rightarrow$-sequence based on $B_r \subseteq \mathcal{O}_{\mathcal{F}}(r)$, where $\rho : l \rightarrow r \in R$, $t \equiv l\sigma$ if $\rightarrow = \rightarrow_R$, and $\sigma = \mathtt{mgu}(t, l)$ if $\rightarrow = \rightsquigarrow_R$. Then, all of the following hold.*

(a) There exists a $\rightarrow$-sequence $s \xrightarrow{*} u \xrightarrow{*} v$ based on B_s.

(b) There exists a basic $\rightarrow$-sequence $t \underset{\mathrm{b}}{\overset{*}{\rightarrow}} v$.

Proposition 5.3.19 *Let R be an EV-TRS and suppose that $\rightarrow$ be a relation either $\rightarrow_R$ or $\rightsquigarrow_R$. Let f be a function symbol, let $s_i \xrightarrow{*} t_i$ be sequences based on $B_i \subseteq \mathcal{O}_{\mathcal{F}}(s_i)$ for $1 \leq i \leq n$, and suppose that $\rightarrow$ is monotone. Then,*

(a) *there exists a* $\to$*-sequence* $f(s_1,\ldots,s_n) \xrightarrow{*} f(t_1,\ldots,t_n)$ *based on the set* $\{\varepsilon\} \cup \bigcup_{i=1}^{n}\{\, ip \mid p \in B_i \,\}$.

Let s *and* t *be terms,* $s \xrightarrow{*} t$ *be a* $\to$*-sequence based on* $B_s \subseteq \mathcal{O}_{\mathcal{F}}(s)$*, and* $t \equiv l\sigma \to^{[\varepsilon,\rho]} r\sigma \equiv u$ *be a* $\to$*-sequence, where* $\rho : l \to r \in R$*,* $t \equiv l\sigma$ *if* $\to = \to_R$*, and* $\sigma = \mathbf{mgu}(t,l)$ *if* $\to = \rightsquigarrow_R$.

(b) *Let* C *be a context such that* $\mathcal{V}ar(C) \cap \mathcal{V}ar(s,t,u) = \emptyset$*, and suppose that* $C[u]_p \xrightarrow{*} v$ *be a* $\to$*-sequence based on* $\mathcal{O}_{\mathcal{F}}(C[r]_p)$*. Then, there exists a* $\to$*-sequence* $C[s]_p \xrightarrow{*} v$ *(that is,* $C[s]_p \xrightarrow{*} C[t]_p \to^{\varepsilon} C[r\sigma]_p \xrightarrow{*} v$*) based on* $\mathcal{O}_{\mathcal{F}}(C) \cup \{\, pq \mid q \in B_s \,\}$.

We prepare a technical lemma to prove Proposition 5.3.21.

Lemma 5.3.20 *Let* R *be an EV-TRS. Let* s *be a linear term. Then,* $s\sigma \xrightarrow[\mathrm{b}]{*}_R t$ *implies a substitution* σ' *such that there exists a rewrite sequence* $s\sigma' \xrightarrow{*}_R t$ *based on* $\mathcal{O}_{\mathcal{F}}(s)$*, and for all* $x \in \mathcal{D}om(\sigma|_{\mathcal{V}ar(s)})$*,* $x\sigma \xrightarrow[\mathrm{b}]{*}_R x\sigma'$ [4].

Proof. We prove this lemma by induction on the structure of term s. Since the case that s is a variable is trivial, we assume that $s \equiv f(s_1,\ldots,s_n)$.

(i) Consider the case that $s\sigma \xrightarrow[\mathrm{b}]{*}_R t$ does not contain $\to^{\varepsilon}_R$, that is, $s\sigma \xrightarrow[\mathrm{b}]{*}{}^{\varepsilon<}_R t$. Then, we can assume $s\sigma \equiv f(s_1\sigma,\ldots,s_n\sigma) \xrightarrow[\mathrm{b}]{*}{}^{\varepsilon<}_R f(t_1,\ldots,t_n) \equiv t$. By the induction hypothesis, we have a substitution σ_i such that there exists a sequence $s_i\sigma_i \xrightarrow{*}_R t_i$ based on $\mathcal{O}_{\mathcal{F}}(s_i)$ and for all $x \in \mathcal{D}om(\sigma_i|_{\mathcal{V}ar(s_i)})$, $x\sigma \xrightarrow[\mathrm{b}]{*}_R x\sigma_i$.

Since s is linear, $\sigma' = \bigcup_{i=1}^{n} \sigma_i|_{\mathcal{V}ar(s_i)}$ is a substitution and for all $x \in \mathcal{D}om(\sigma|_{\mathcal{V}ar(s)})$, $x\sigma \xrightarrow[\mathrm{b}]{*}_R x\sigma'$. From Proposition 5.3.19 (a), we have a sequence $s\sigma' \equiv f(s_1\sigma_1,\ldots,s_n\sigma_1) \xrightarrow{*}_R f(t_1,\ldots,t_n) \equiv t$ based on $\mathcal{O}_{\mathcal{F}}(s)$.

(ii) Consider the remaining case, that is, $f(s_1\sigma,\ldots,s_n\sigma) \xrightarrow[\mathrm{b}]{*}{}^{\varepsilon<}_R f(u_1,\ldots,u_n) \xrightarrow[\mathrm{b}]{\varepsilon}_R s' \xrightarrow[\mathrm{b}]{*}_R t$. As for the case (i), we can also construct σ' such that there exists a sequence $f(s_1\sigma',\ldots,s_n\sigma') \xrightarrow{*}{}^{\varepsilon<}_R f(u_1,\ldots,u_n)$ based on $\mathcal{O}_{\mathcal{F}}(s)$ and for all $x \in \mathcal{D}om(\sigma|_{\mathcal{V}ar(s)})$, $x\sigma \xrightarrow[\mathrm{b}]{*}_R x\sigma'$. From Proposition 5.3.18 (a), we have a sequence $f(s_1\sigma',\ldots,s_n\sigma') \xrightarrow{*}{}^{\varepsilon<}_R f(u_1,\ldots,u_n) \to^{\varepsilon}_R s' \xrightarrow{*}_R t$ based on $\mathcal{O}_{\mathcal{F}}(s)$.

□

[4] This implies the existence of a basic sequence $u\sigma \xrightarrow[\mathrm{b}]{*}_R u\sigma'$ for all terms u.

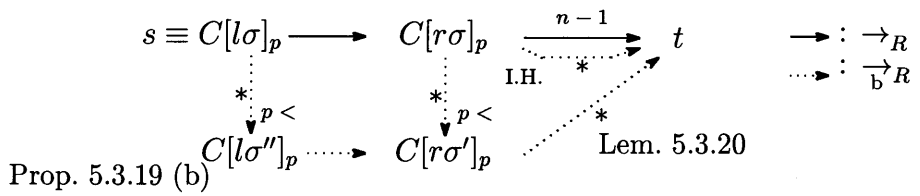

Figure 5.12: Proof sketch of Proposition 5.3.21.

For right-linear TRSs, it is known that for every rewrite sequence, there exists a basic rewrite sequence [36]. We extend this result to that on EV-TRSs in Proposition 5.3.21 and 5.3.22. We give a clear proof for the following proposition, while this can be proved similarly to the original one in [36].

Proposition 5.3.21 *Let R be a right-linear EV-TRS. Then, $s \xrightarrow{*}_R t$ implies a basic sequence $s \xrightarrow[\mathrm{b}]{*}_R t$.*

Proof. We prove this claim by induction on n that $s \xrightarrow{n}_R t$ implies $s \xrightarrow[\mathrm{b}]{*}_R t$.

Since the case of $n = 0$ is trivial, we assume that $s \equiv C[l\sigma]_p \to_R C[r\sigma]_p \xrightarrow{n-1}_R t$. By the induction hypothesis, we have $C[r\sigma]_p \xrightarrow[\mathrm{b}]{*}_R t$. From Lemma 5.3.20, we have a substitution σ' such that there exists a sequence $C[r\sigma']_p \xrightarrow{*}_R t$ based on $\mathcal{O}_{\mathcal{F}}(C[r]_p)$ and for all $x \in \mathcal{D}om(\sigma|_{\mathcal{V}ar(r)})$, $x\sigma \xrightarrow[\mathrm{b}]{*}_R x\sigma'$. Then, we have $r\sigma \overset{*}{\underset{\mathrm{b}}{\rightsquigarrow}}_R r\sigma'$. Since $l\sigma \to_R r\sigma$ and $r\sigma \overset{*}{\underset{\mathrm{b}}{\rightsquigarrow}}_R r\sigma'$, it follows from Proposition 5.3.18 (b) that $l\sigma \overset{*}{\underset{\mathrm{b}}{\rightsquigarrow}}_R r\sigma'$. From Proposition 5.3.19 (a), we have $C[l\sigma] \overset{*}{\underset{\mathrm{b}}{\rightsquigarrow}}_R C[r\sigma']$. It follows from Proposition 5.3.18 that $C[l\sigma]_p \overset{*}{\underset{\mathrm{b}}{\rightsquigarrow}}_R t$. Therefore, $s \equiv C[l\sigma]_p \overset{*}{\underset{\mathrm{b}}{\rightsquigarrow}}_R t$ (see Figure 5.12).

Let $\sigma'' = \sigma'|_{\mathcal{V}ar(r)} \cup \{\, x \mapsto x\sigma \mid x \in \mathcal{V}ar(l) \setminus \mathcal{V}ar(r) \,\}$. Then, σ'' is well-defined and $l\sigma \xrightarrow[\mathrm{b}]{*}{}^{\varepsilon<}_R l\sigma''$. From Proposition 5.3.19 (b), $C[l\sigma]_p \xrightarrow[\mathrm{b}]{*}{}^{p<}_R C[l\sigma'']_p \xrightarrow[\mathrm{b}]{}{}^{p}_R C[r\sigma'']_p \equiv C[r\sigma']_p \xrightarrow[\mathrm{b}]{*}_R t$ is basic. Therefore, we have a basic sequence $s \xrightarrow[\mathrm{b}]{*}_R t$ (see Figure 5.12). □

From Proposition 5.3.17 and Proposition 5.3.21, the following proposition is easily obtained.

Proposition 5.3.22 ([36]) *Let R be a right-linear EV-TRS over a signature $\mathcal{F}$. Let θ be an $NF^{\to_R}(\mathcal{F}, \mathcal{X})$-substitution. Then, $s\theta \xrightarrow{*}_R t$ implies a sequence $s\theta \xrightarrow{*}_R t$ based on $\mathcal{O}_{\mathcal{F}}(s)$.*

Concluding from Proposition 5.3.15 and 5.3.22, we have the following result:

Theorem 5.3.23 *Let R be a right-linear EV-TRS. Let σ be an $NF^{\rightarrow_R}(\mathcal{F},\mathcal{X})$-substitution. If $s\sigma \xrightarrow{*}_R t$ then $s \overset{*}{\underset{\mathrm{b}}{\rightsquigarrow}}_R u$ and $t \equiv u\theta$ for some term u and substitution θ.*

The above theorem means that every term which is reachable from a given term $s\sigma$ has a term which is narrowable from s by basic narrowing. Hence, $\underset{\mathrm{b}}{\rightsquigarrow}_R$ is sufficient to simulate rewrite sequences of right-linear EV-TRSs.

Theorem 5.1.10 and 5.3.23 gives the following corollary:

Corollary 5.3.24 *Let R be a right-linear EV-TRS. If $s \overset{*}{\rightsquigarrow}_R t$ then there exists a term u such that $s \overset{*}{\underset{\mathrm{b}}{\rightsquigarrow}}_R u$ and t, u are unifiable.*

In the above corollary, t, u are unifiable while $t \equiv u$ in the case of $\rightarrow_R$ in Proposition 5.3.21. Then, related to the above corollary, we have the following stronger conjecture than the above corollary.

Conjecture 5.3.25 *Let R be a right-linear EV-TRS. Let s be a linear term. Then, $s \overset{*}{\rightsquigarrow}_R t$ implies a basic sequence $s \overset{*}{\underset{\mathrm{b}}{\rightsquigarrow}}_R t$.*

Surely, the following variant of the above conjecture, which is restricted to linear EV-TRSs, holds obviously.

Theorem 5.3.26 *Let R be a linear EV-TRS. Let s be a linear term. Then, $s \overset{*}{\rightsquigarrow}_R t$ implies a basic sequence $s \overset{*}{\underset{\mathrm{b}}{\rightsquigarrow}}_R t$.*

Proof. (Sketch.) Since $\rightsquigarrow_R$ on linear terms is monotone, we can easily lift up Lemma 5.3.20 to that of $\rightsquigarrow_R$ similarly to the proof of Lemma 5.3.5 (in Lemma 5.3.20, suppose that R is right-linear, and replace $\rightarrow_R$ with $\rightsquigarrow_R$). Then, we can prove this theorem similarly to the proof of Proposition 5.3.21. □

5.3.3 Discussion on Efficiency of Inverse Computation

In this subsection, we discuss the efficiency of narrowing, innermost narrowing and basic narrowing, by using three examples.

We first consider the inverse computation of the following constructor TRS:

$$R_{20} = \{\ \mathsf{h}(0) \to 0,\ \ \mathsf{h}(\mathsf{s}(0)) \to 0,\ \ \mathsf{h}(\mathsf{s}^2(x)) \to \mathsf{s}(\mathsf{h}(x))\ \}.$$

The inverse TRS of R_{20} is generated as follows:

$$\begin{aligned} R_{21} &= \mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R_{20})) \\ &= \{\ \mathsf{h}^{\#}(0) \to 0, \quad \mathsf{h}^{\#}(0) \to \mathsf{s}(0), \quad \mathsf{h}^{\#}(\mathsf{h}(x)) \to x, \\ &\qquad \mathsf{h}^{\#}(\mathsf{s}(x)) \to \mathsf{u}_0(\mathsf{h}^{\#}(x)), \quad \mathsf{u}_0(y) \to \mathsf{s}^2(y) \qquad\qquad \}. \end{aligned}$$

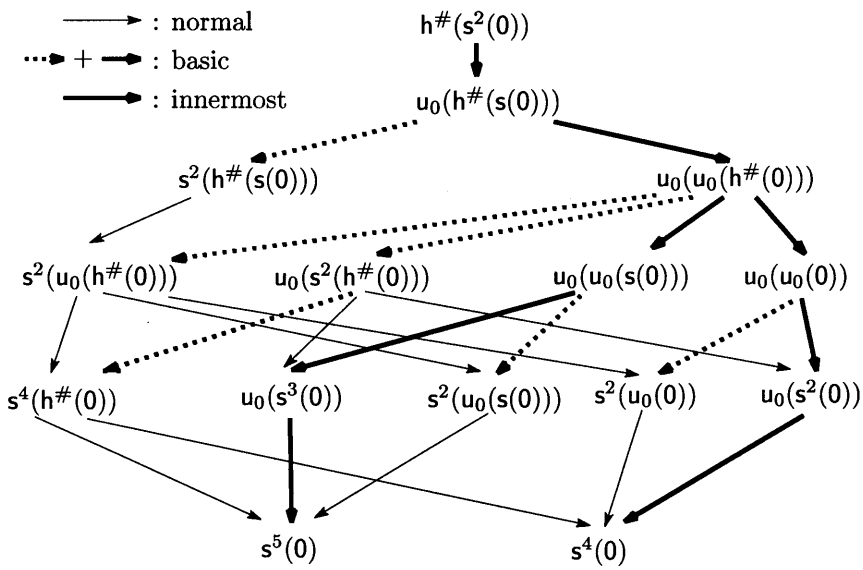

Figure 5.13: Rewrite-relation tree starting from $\mathsf{h}^{\#}(\mathsf{s}^2(0))$.

Since we have $\mathsf{h}(\mathsf{s}^4(0)) \xrightarrow{*}_{R_{20}} \mathsf{s}^2(0)$ and $\mathsf{h}(\mathsf{s}^5(0)) \xrightarrow{*}_{R_{20}} \mathsf{s}^2(0)$, the solutions of the inverse computation of $\mathsf{h}^{\#}(\mathsf{s}^2(0))$ are $\mathsf{s}^4(0)$ and $\mathsf{s}^5(0)$. The depth-first search starting from $\mathsf{h}^{\#}(\mathsf{s}^2(0))$ takes 37 steps by rewrite relation, 14 steps by basic rewrite relation, and 8 steps by innermost rewrite relation (see Figure 5.13).

As the second example, we consider to solve the equation $d(x-y) =^? 2$ where d computes twice of two natural numbers and "$-$" computes subtraction of two natural numbers. The functions d and "$-$" are defined as follows:

$$\begin{aligned} R_{22} = \{\ & \mathsf{double}(0) \to 0, && \mathsf{double}(x) \to \mathsf{s}(\mathsf{s}(\mathsf{double}(x))), \\ & \mathsf{sub}(0,y) \to 0, && \mathsf{sub}(\mathsf{s}(x),0) \to \mathsf{s}(x), \\ & \mathsf{sub}(\mathsf{s}(x),\mathsf{s}(y)) \to \mathsf{sub}(x,y) && \}. \end{aligned}$$

The inverse EV-TRS of R_{22} is generated as follows:

$$\begin{aligned} R_{23} = \{\ & \mathsf{sub}^{\#}(0) \to \mathsf{tp}_2(0,y), && \mathsf{sub}^{\#}(\mathsf{s}(x)) \to \mathsf{tp}_2(\mathsf{s}(x),0), \\ & \mathsf{sub}^{\#}(z) \to \mathsf{u}_5(\mathsf{sub}^{\#}(z)), && \mathsf{u}_5(\mathsf{tp}_2(x,y)) \to \mathsf{tp}_2(\mathsf{s}(x),\mathsf{s}(y)), \\ & \mathsf{sub}^{\#}(\mathsf{sub}(x,y)) \to \mathsf{tp}_2(x,y), && \mathsf{double}^{\#}(\mathsf{double}(x)) \to x, \\ & \mathsf{double}^{\#}(0) \to 0, && \mathsf{double}^{\#}(\mathsf{s}^2(x)) \to \mathsf{u}_4(\mathsf{double}^{\#}(x)), \\ & \mathsf{u}_4(y) \to \mathsf{s}(y) && \}. \end{aligned}$$

To simplify the discussion, we use $\mathsf{double}^{\#}$ which is well-defined to compute half of a natural number, as an inverse of double instead of $\mathsf{double}^{\#}$. To solve the equation $d(x-y) =^? 2$, we may compute $\mathsf{sub}^{\#}(\mathsf{double}^{\#}(\mathsf{s}^2(0)))$ by $\rightsquigarrow_{R_{23}}$

(see Figure 5.14). It can be proved by induction on the number m, n of $\mathsf{s}^m(0)$ and $\mathsf{s}^{2n}(0)$ that $\mathrm{WIN}^{\rightsquigarrow_{R_{23}}}_{\mathsf{sub}^\#(\mathsf{s}^m(0))}$ holds and $\mathsf{double}^\#(\mathsf{s}^{2n}(0)) \underset{\mathrm{in}}{\rightsquigarrow}_{R_{23}} \mathsf{s}^n(0)$. This fact implies that $\mathrm{WIN}^{\rightsquigarrow_{R_{23}}}_t$ holds for every term t such that $\mathsf{sub}^\#(\mathsf{double}^\#(\mathsf{s}^2(0))) \underset{\mathrm{in}}{\rightsquigarrow}_{R_{23}} t$. Then, Theorem 5.3.9 is applicable to the normal form computation of $\mathsf{sub}^\#(\mathsf{double}^\#(\mathsf{s}^2(0)))$. Hence, by the width-first search of $\underset{\mathrm{in}}{\rightsquigarrow}_{R_{23}}$, the terms $\mathsf{tp}_2(\mathsf{s}^{i+1}(0), \mathsf{s}^i(0))$ $(i \geq 0)$ which represent solutions of the above equation, are obtained by turns more efficiently than $\rightsquigarrow_{R_{23}}$ (see Figure 5.14).

The last example is to solve the equational problem $d(1 \times x) =^? 2$. Since the EV-TRS in this example is not linear, Corollary 5.3.10 cannot be applied. However, this example is valuable to discuss since all desirable normal forms can be obtained by innermost narrowing. Functions d and $\times$ are encoded as the TRS as follows:

$$
\begin{aligned}
R_{24} = \{\ & \mathsf{double}(0) \to 0, && \mathsf{double}(\mathsf{s}(x)) \to \mathsf{s}^2(\mathsf{double}(x)), \\
& \mathsf{add}(0, y) \to y, && \mathsf{add}(\mathsf{s}(x), y) \to \mathsf{s}(\mathsf{add}(x, y)), \\
& \mathsf{mul}(0, y) \to 0, && \mathsf{mul}(x, 0) \to 0, \\
& \mathsf{mul}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{s}(\mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y)) && \}.
\end{aligned}
$$

The inverse EV-TRS of R_{24} is generated as follows:

$$
\begin{aligned}
R_{25} &= \mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R_{24})) \\
&= \{\ \mathsf{add}^\#(y) \to \mathsf{tp}_2(0, y), && \mathsf{add}^\#(\mathsf{s}(z)) \to \mathsf{u}_1(\mathsf{add}^\#(z)), \\
&\quad \mathsf{add}^\#(\mathsf{add}(x, y)) \to \mathsf{tp}_2(x, y), && \mathsf{u}_1(\mathsf{tp}_2(x, y)) \to \mathsf{tp}_2(\mathsf{s}(x), y), \\
&\quad \mathsf{mul}^\#(0) \to \mathsf{tp}_2(0, y), && \mathsf{mul}^\#(\mathsf{s}(z)) \to \mathsf{u}_2(\mathsf{add}^\#(z)), \\
&\quad \mathsf{mul}^\#(0) \to \mathsf{tp}_2(x, 0), && \mathsf{u}_2(\mathsf{tp}_2(w, y)) \to \mathsf{u}_3(\mathsf{mul}^\#(w), y), \\
&\quad \mathsf{mul}^\#(\mathsf{mul}(x, y)) \to \mathsf{tp}_2(x, y), && \mathsf{u}_3(\mathsf{tp}_2(x, \mathsf{s}(y)), y) \to \mathsf{tp}_2(\mathsf{s}(x), \mathsf{s}(y)), \\
&\quad \mathsf{double}^\#(0) \to 0, && \mathsf{double}^\#(\mathsf{s}^2(x)) \to \mathsf{u}_4(\mathsf{double}^\#(x)), \\
&\quad \mathsf{u}_4(y) \to \mathsf{s}(y), && \mathsf{double}^\#(\mathsf{double}(x)) \to x \qquad \}.
\end{aligned}
$$

To solve the above equation, we compute $\mathsf{mul}^\#(\mathsf{double}^\#(\mathsf{s}^2(0)))$ by narrowing. The depth-first search starting from $\mathsf{mul}^\#(\mathsf{double}^\#(\mathsf{s}^2(0)))$ takes 42 steps by narrowing, 16 steps by basic narrowing, and 9 steps by innermost narrowing (see Figure 5.15). Although R_{25} is not linear, we obtain all normal forms by innermost narrowing.

We conclude the following table from the numbers of steps in the first and third examples above, which shows the numbers of steps of depth-first search by each derivation.

	normal	basic	innermost
$\mathsf{h}^\#(\mathsf{s}^2(0))$	37	14	8
$\mathsf{mul}^\#(\mathsf{double}^\#(\mathsf{s}^2(0)))$	42	16	9

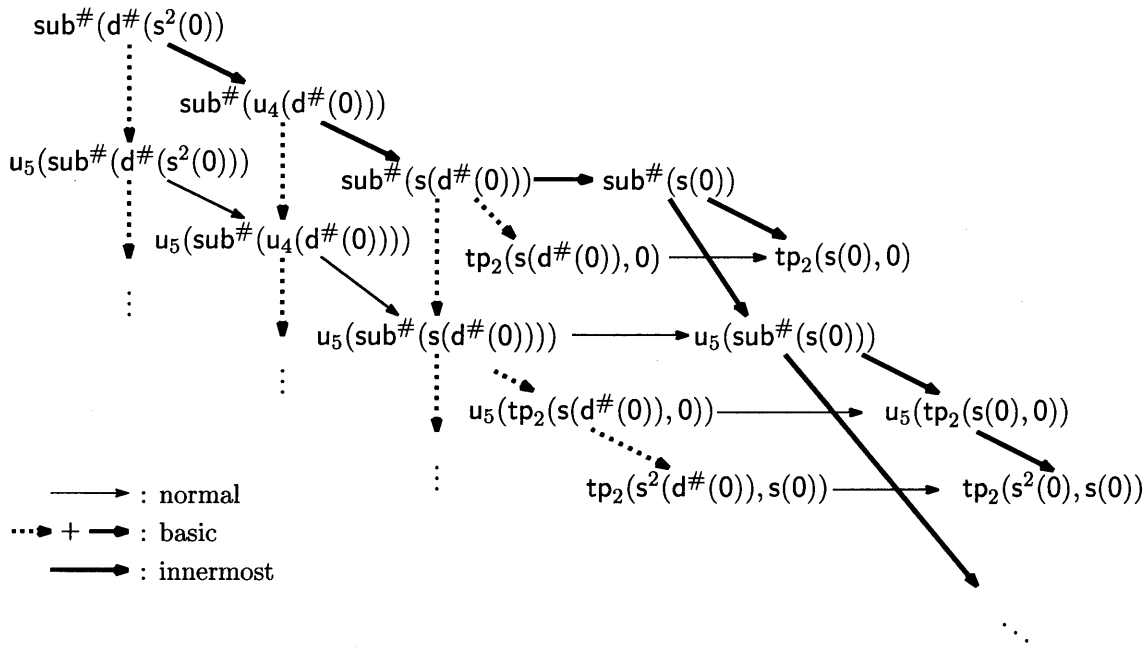

Figure 5.14: Narrowing-derivation tree starting from $\mathsf{sub}^{\#}(\mathsf{double}^{\#}(\mathsf{s}^2(0)))$.

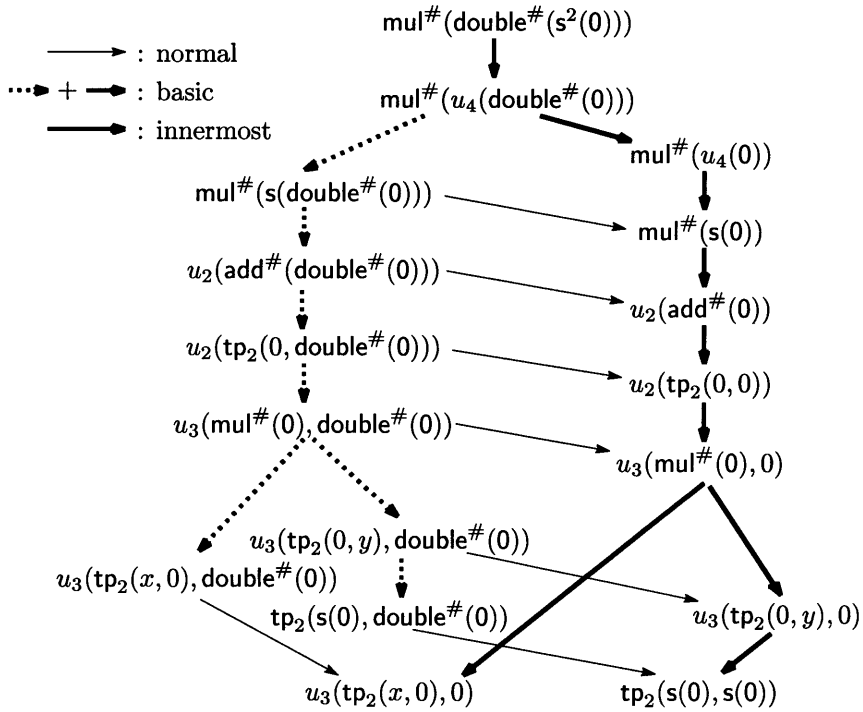

Figure 5.15: Rewrite-relation tree starting from $\mathsf{mul}^{\#}(\mathsf{double}^{\#}(\mathsf{s}^2(0)))$.

Note that we may use only narrowing since $\rightarrow_R = \rightsquigarrow_R$ on ground terms for TRSs R. Remark that basic derivation can reach all terms which is reachable from the initial terms, and innermost derivation guarantees that all normal forms can be obtained.

In this section, we have shown that for linear constructor EV-TRSs, all narrowing normal-forms of a given linear term from which no infinite narrowing sequence exists can be obtained by innermost narrowing, and also shown that basic narrowing is sufficient to simulate rewrite sequences of right-linear EV-TRSs. We conclude from the above results as follows.

- For right-linear systems, basic derivation is sufficient to simulate rewrite sequences.
- For any linear EV-TRS R, $\overset{\rightsquigarrow}{}_{\mathrm{b}R} = \rightsquigarrow_R$.
- To compute narrowing normal forms on linear constructor EV-TRSs, innermost narrowing is enough and efficient.

Chapter 6

Extension for Partial Inverse Computation

In this chapter, we tackle partial inverse computation with a transformational approach. Typical instances of partial inverse are subtraction for addition, division for multiplication and so on. We first define partial-inverse CTRSs by extending the definition of inverse systems. We then extend the inverse compiler $\mathcal{I}nv_{\mathsf{full}}$ proposed in Chapter 3 to a compiler that generates partial-inverse CTRSs. Unlike the case of full inverse, we do not transform the generated partial-inverse CTRSs to an EV-TRS although it possible. The reason is that results on the completeness of the transformation $\mathbb{U}$, which have been shown already, is weak for this case.

6.1 Partial-Inverse Systems

When the input value of an argument of a given function (defined symbol) is always known (or given), we call it a *known argument* (of the function). To represent which arguments of a given function are known, we use lists of natural numbers, written as $[i_1, \ldots, i_m]$, and we call them *indexes*. The notion of natural number lists for arguments of defined symbols is similar to the case of argument filterings. We assume that the index $[i_1, \ldots, i_m]$ for a defined symbol f satisfies $0\ m \leq n$ and $1 \leq i_1 < \cdots < i_m \leq n$ where $n = \mathtt{arity}(f)$. Note that the empty list [ ] is included in indexes. The index $[i_1, \ldots, i_m]$ is called a *known-argument index* of the defined symbol f if for $1 \leq j \leq m$, i_j-th argument of f is known. Thus, we use indexes to show which arguments is known.

Let $\mathcal{F}$ be a signature divided into the constructor set $\mathcal{C}$ and the defined-

symbol set $\mathcal{D}$: $\mathcal{F} = \mathcal{C} \uplus \mathcal{D}$. Let f be a defined symbol with $\mathtt{arity}(f) = n$, and I be an index $[i_1, \ldots, i_m]$ for f. The notation $|I|$ denotes the number of elements in I, that is, $|I| = m$. The notation I_j denotes the j-th element of I; $I_j = i_j$. Then, for a term $f(t_1, \ldots, t_n)$, t_{I_j} represents the i_j-th term t_{i_j}, that is, $t_{I_j} \equiv t_{i_j}$.

The notation $\bar{I}$ denotes the index for f consisting of the complementary natural numbers from I, that is, $I \uplus \bar{I} = [1, \ldots, n]$. Note that indexes can be treated like as sets without loss of generality. When I is a known-argument index of f, we use the pair (f, I) of f and I to represent that fact. We denote the set of such all pairs for $\mathcal{D}$ by $\mathbb{I}_{\mathcal{D}}$: $\mathbb{I}_{\mathcal{D}} = \{ (f, I) \mid f \in \mathcal{D}, I \subseteq [1, \ldots, \mathtt{arity}(f)] \}$.

Next we define partial-inverse CTRSs as follows:

Definition 6.1.1 (Partial-inverse system) *Let R be a CTRS over a signature $\mathcal{F}$, R' be a CTRS over a signature $\mathcal{F}'$, and $\mathcal{C}_R \supseteq \mathcal{C}_{R'}$. We say that a defined symbol g of R' is a* partial-inverse *of a defined symbol f of R with respect to an index $I = [i_1, \ldots, i_m]$ for f if both of the following hold:*

(a) $\mathsf{tp}_{|I|}, \mathsf{tp}_{|\bar{I}|} \in \mathcal{C}_{R'}$, *and*

(b) for all constructor terms $t, t_1, \ldots, t_n$ with respect to R, $f(t_1, \ldots, t_n) \xrightarrow{*}_R t$ iff $g(t, \mathsf{tp}_{|I|}(t_{I_1}, \ldots, t_{I_m})) \xrightarrow{*}_{R'} \mathsf{tp}_{|\bar{I}|}(t_{\bar{I}_1}, \ldots, t_{\bar{I}_{|\bar{I}|}})$.

The CTRS R' is called a partial-inverse system *of R with respect to a set $D_I \subseteq \mathbb{I}_{\mathcal{D}}$ if for every pair $(f, I) \in D_I$, R' defines a partial-inverse defined-symbol of f with respect to I.*

In the condition (b) above, we may abbreviate term $g(t, \mathsf{tp}_{|I|}(t_{I_1}, \ldots, t_{I_m}))$ to $g(t, t_{I_1}, \ldots, t_{I_m})$, and then $\mathsf{tp}_{|I|} \in \mathcal{C}_{R'}$ in (a) is not necessary. The condition $\mathsf{tp}_{|I|} \in \mathcal{C}_{R'}$ in (a) above are not necessary if $|I| = 1$ and $\mathsf{tp}_1(t)$ is abbreviated to t (similarly for $\mathsf{tp}_{|\bar{I}|}$). Note that when g is a partial-inverse of f with respect to the empty list $[\,]$, g is a (full) inverse of f. That is, full inverses are special cases of partial-inverses.

Example 6.1.2 Consider the following TRSs R_3 and R_{26} over $\{\mathsf{0}, \mathsf{s}, \mathsf{add}\}$ and $\{\mathsf{0}, \mathsf{s}, \mathsf{min}\}$, respectively:

$$R_3 = \{ \mathsf{add}(\mathsf{0}, y) \to y,\ \ \mathsf{add}(\mathsf{s}(x), y) \to \mathsf{s}(\mathsf{add}(x, y)) \},$$

$$R_{26} = \{ \mathsf{min}(x, \mathsf{0}) \to x,\ \ \mathsf{min}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{min}(x, y) \}.$$

R_{26} is a partial-inverse TRS of R_3 with respect to $\{ (\mathsf{add}, [1]), (\mathsf{add}, [2]) \}$. □

6.2 Generation of Partial-Inverse CTRSs

In this section, we extend the transformation $\mathcal{I}nv_{\mathsf{full}}$ to a transformation which generates partial-inverse CTRSs of input constructor TRSs. For a given pair (f, I) of a defined symbol f and its known-argument index I, we use $f^{\#}_I$ to represent a partial-inverse of f with respect to I.

An idea of extension is simple. Consider a rewrite rule $f(t_1, \ldots, t_n) \to r$. In the case of $\mathcal{I}nv_{\mathsf{full}}$, all arguments $t_1, \ldots, t_n$ are moved to the right-hand side of the generated rule of $f^{\#}$, such as $f^{\#}(r') \to \mathsf{tp}_n(t_1, \ldots, t_n) \Leftarrow \mathit{Cond}$. In this case, for a given pair (f, I), we leave every I_j-th argument in the left-hand side of the generated rule of $f^{\#}_I$, and move the rest to the right-hand side, like $f^{\#}_I(r'', \mathsf{tp}_{|I|}(t_{I_1}, \ldots, t_{I_{|I|}})) \to \mathsf{tp}_{|\bar{I}|}(t_{\bar{I}_1}, \ldots, t_{\bar{I}_{|\bar{I}|}}) \Leftarrow \mathit{Cond}'$. To extend in the above, $\mathcal{T}_{\mathsf{full}}$ must be extended.

To extend $\mathcal{T}_{\mathsf{full}}$, we first define, for a rewrite rule $f(t_1, \ldots, t_n) \to r$ and a known-argument index I of f, the set of variables appearing below defined symbols in the right-hand side and but not in either of known arguments, that is, I_j-th argument t_{I_j} of the left-hand side.

Definition 6.2.1 *Let a rewrite rule $\rho : f(t_1, \ldots, t_n) \to C[\![r_1, \ldots, r_m]\!]$[1] over a signature $\mathcal{F}$ and an index I for f. The set $\mathcal{UV}ar(\rho, I)$ of variables is defined as follows:*

$$\mathcal{UV}ar\left(f(t_1, \ldots, t_n) \to C[\![r_1, \ldots, r_m]\!], I\right) = \mathcal{V}ar(t_{\bar{I}_1}, \ldots, t_{\bar{I}_{|\bar{I}|}}) \setminus \Big(\mathcal{V}ar(t_{I_1}, \ldots, t_{I_{|I|}}) \cup \mathcal{V}ar(C)\Big).$$

Example 6.2.2 Consider the following constructor TRS over the signature { s, 0, add, mul }:

$$\begin{aligned} R_7 = \{\ & \mathsf{add}(0, y) \to y, \quad \mathsf{add}(\mathsf{s}(x), y) \to \mathsf{s}(\mathsf{add}(x, y)), \\ & \mathsf{mul}(0, y) \to 0, \quad \mathsf{mul}(x, 0) \to 0, \\ & \mathsf{mul}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{s}(\mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y)) \quad \}. \end{aligned}$$

For the rule $\mathsf{mul}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{s}(\mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y))$, we have the following:

$$\begin{aligned} \mathcal{UV}ar(\ & \mathsf{mul}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{s}(\mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y)), \quad [\,] \quad) = \{x, y\}, \\ \mathcal{UV}ar(\ & \mathsf{mul}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{s}(\mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y)), \quad [1] \quad) = \{y\}, \\ \mathcal{UV}ar(\ & \mathsf{mul}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{s}(\mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y)), \quad [2] \quad) = \{x\}, \\ \mathcal{UV}ar(\ & \mathsf{mul}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{s}(\mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y)), \quad [1, 2] \quad) = \emptyset. \end{aligned}$$

□

[1] It is clear that every term s can be represented as $C'[\![s_1, \ldots, s_k]\!]$.

Next, for a term $f(t_1, \ldots, t_n)$ with the defined symbol f, and for a set X of variables, we define the index for f that shows which arguments do not contain any variable in X.

Definition 6.2.3 *Let $\mathcal{F}$ be a signature divided into the constructor set $\mathcal{C}$ and the defined-symbol set $\mathcal{D}$: $\mathcal{F} = \mathcal{C} \uplus \mathcal{D}$. Let f be a defined symbol with* $\mathtt{arity}(f) = n$, *$t_1, \ldots, t_n$ be terms over $\mathcal{F}$, and X be a set of variables. Then, we define the index $\mathcal{I}ndex(f(t_1, \ldots, t_n), X)$ as follows:*

$$\mathcal{I}ndex\,(f(t_1, \ldots, t_n), X) = [\, i \mid \mathcal{V}ar(t_i) \cap X \neq \emptyset \,].$$

Example 6.2.4 Consider the the signature $\{\, 0, \mathsf{s}, \mathsf{add}, \mathsf{mul} \,\}$. Then, we have the followings:

$$\begin{array}{lrcl}
\mathcal{I}ndex(& \mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y), & \{x, y\} &) = [\,], \\
\mathcal{I}ndex(& \mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y), & \{y\} &) = [\,], \\
\mathcal{I}ndex(& \mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y), & \{x\} &) = [2], \\
\mathcal{I}ndex(& \mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y), & \emptyset &) = [1, 2],
\end{array}$$

$$\begin{array}{lrcl}
\mathcal{I}ndex(& \mathsf{mul}(x, \mathsf{s}(y)), & \{x, y\} &) = [\,], \\
\mathcal{I}ndex(& \mathsf{mul}(x, \mathsf{s}(y)), & \{y\} &) = [1], \\
\mathcal{I}ndex(& \mathsf{mul}(x, \mathsf{s}(y)), & \{x\} &) = [2], \\
\mathcal{I}ndex(& \mathsf{mul}(x, \mathsf{s}(y)), & \emptyset &) = [1, 2].
\end{array}$$

□

In the following definition, for a term t and a set X of variables, we define the term $\mathcal{I}term(t, X)$ in which each defined symbol f is labeled with the index $\mathcal{I}ndex(f(\cdots), X)$.

Definition 6.2.5 *Let $\mathcal{F}$ be a signature divided into the constructor set $\mathcal{C}$ and the defined-symbol set $\mathcal{D}$: $\mathcal{F} = \mathcal{C} \uplus \mathcal{D}$. Let t be a term over $\mathcal{F}$, and X be a set of variables. Then, the term $\mathcal{I}term(t, X)$ is recursively defined as follows:*

- $\mathcal{I}term(x, X) = x$ *where x is a variable,*
- $\mathcal{I}term(c(t_1, \ldots, t_n), X) = c\,(\mathcal{I}term(t_1, X), \ldots, \mathcal{I}term(t_m, X))$ *where c is a constructor, and*
- $\mathcal{I}term(f(t_1, \ldots, t_n), X) = f_I\,(\mathcal{I}term(t_1, X), \ldots, \mathcal{I}term(t_m, X))$ *where f is a defined symbol and $I = \mathcal{I}ndex(f(t_1, \ldots, t_n), X)$.*

Example 6.2.6 Consider the the signature $\{ \mathsf{0}, \mathsf{s}, \mathsf{add}, \mathsf{mul} \}$ again. Then, referring Example 6.2.2 and 6.2.4, we obtain the following:

$$\begin{aligned}
\mathcal{I}term(\ \mathsf{mul}(x,\mathsf{s}(y)),\ \{x,y\}\ ) &= \mathsf{mul}_{[\ ]}(x,\mathsf{s}(y)),\\
\mathcal{I}term(\ \mathsf{mul}(x,\mathsf{s}(y)),\ \{y\}\ ) &= \mathsf{mul}_{[1]}(x,\mathsf{s}(y)),\\
\mathcal{I}term(\ \mathsf{mul}(x,\mathsf{s}(y)),\ \{x\}\ ) &= \mathsf{mul}_{[2]}(x,\mathsf{s}(y)),\\
\mathcal{I}term(\ \mathsf{mul}(x,\mathsf{s}(y)),\ \emptyset\ ) &= \mathsf{mul}_{[1,2]}(x,\mathsf{s}(y)),
\end{aligned}$$

$$\begin{aligned}
\mathcal{I}term(\ \mathsf{s}(\mathsf{add}(\mathsf{mul}(x,\mathsf{s}(y)),y)),\ \{x,y\}\ ) &= \mathsf{s}(\mathsf{add}_{[\ ]}(\mathsf{mul}_{[\ ]}(x,\mathsf{s}(y)),y)),\\
\mathcal{I}term(\ \mathsf{s}(\mathsf{add}(\mathsf{mul}(x,\mathsf{s}(y)),y)),\ \{y\}\ ) &= \mathsf{s}(\mathsf{add}_{[\ ]}(\mathsf{mul}_{[1]}(x,\mathsf{s}(y)),y)),\\
\mathcal{I}term(\ \mathsf{s}(\mathsf{add}(\mathsf{mul}(x,\mathsf{s}(y)),y)),\ \{x\}\ ) &= \mathsf{s}(\mathsf{add}_{[2]}(\mathsf{mul}_{[2]}(x,\mathsf{s}(y)),y)),\\
\mathcal{I}term(\ \mathsf{s}(\mathsf{add}(\mathsf{mul}(x,\mathsf{s}(y)),y)),\ \emptyset\ ) &= \mathsf{s}(\mathsf{add}_{[1,2]}(\mathsf{mul}_{[1,2]}(x,\mathsf{s}(y)),y)).
\end{aligned}$$

□

The following proposition shows a syntactic property associated with a special form of output of $\mathcal{I}term$.

Proposition 6.2.7 *Let $\mathcal{F}$ be a signature divided into the constructor set $\mathcal{C}$ and the defined-symbol set $\mathcal{D}$: $\mathcal{F} = \mathcal{C} \uplus \mathcal{D}$. Let t be a term over $\mathcal{F}$, X be a set of variables, f be a defined symbol with $\mathtt{arity}(f) = n$, and I be an index of f. Suppose that $\mathcal{I}term(t,X) = f_I(t_1,\ldots,t_n)$. If $I = [1,\ldots,n]$, then for $1 \leq i \leq n$, for every proper subterm $u \lhd t_i$, $\mathtt{root}(u) \in \mathcal{D}$ implies $\mathtt{root}(u) = g_{[1,\ldots,\mathtt{arity}(g)]}$ for some $g \in \mathcal{D}$.*

Proof. Suppose that $I = [1,\ldots,n]$. It follows from the definition of $\mathcal{I}ndex$ that $\mathcal{V}ar(t_i) \cap X = \emptyset$. Let u be a subterm of either $t_1,\ldots,t_n$ such that $\mathtt{root}(u) \in \mathcal{D}$. Since $\mathcal{V}ar(t_i) \cap X = \emptyset$, it is clear that $\mathcal{V}ar(u) \cap X = \emptyset$. It follows from the definition of $\mathcal{I}ndex$ that there exists a defined symbol g *in* $\mathcal{D}$ such that $\mathcal{I}ndex(u,X) = [1,\ldots,\mathtt{arity}(g)]$. Therefore, this proposition holds. □

We extend $\mathcal{I}term$ over TRSs to know which argument of each defined symbol is known, and to know what partial inverse must be defined. To know it, we construct the set of defined symbols with indexes from a given $D_I \subseteq \mathbb{I}_{\mathcal{D}_R}$.

Definition 6.2.8 *Let R be a TRS over a signature $\mathcal{F}$ and $D_I \subseteq \mathbb{I}_{\mathcal{D}_R}$. We define the set $\mathcal{R}eq\mathcal{D}(R,D_I)$ to be the smallest satisfying all of the following:*

- $\mathcal{R}eq\mathcal{D}(R,D_I) \supseteq D_I$, *and*
- $\mathcal{R}eq\mathcal{D}(R,D_I) \supseteq \{\ f_I \mid \rho : g(t_1,\ldots,t_n) \to r \in R, g_J \in \mathcal{R}eq\mathcal{D}(R,D_I),$
  $f_I \in \mathcal{S}ym\,(\mathcal{I}term(r,\mathcal{U}\mathcal{V}ar(\rho,J)))\ \}.$

Example 6.2.9 Consider the following constructor TRS over the signature { s, 0, add, mul } again:

$$\begin{aligned} R_7 = \{\ & \mathsf{add}(0,y) \to y, \quad \mathsf{add}(\mathsf{s}(x),y) \to \mathsf{s}(\mathsf{add}(x,y)), \\ & \mathsf{mul}(0,y) \to 0, \quad \mathsf{mul}(x,0) \to 0, \\ & \mathsf{mul}(\mathsf{s}(x),\mathsf{s}(y)) \to \mathsf{s}(\mathsf{add}(\mathsf{mul}(x,\mathsf{s}(y)),y)) \qquad \}. \end{aligned}$$

For $\{(\mathsf{mul},[\,])\}$, $\{(\mathsf{mul},[1])\}$, $\{(\mathsf{mul},[2])\}$ and $\{(\mathsf{mul},[1,2])\}$, we obtain by $\mathcal{ReqD}$ the following:

$$\begin{aligned} \mathcal{ReqD}(R_7,\{(\mathsf{mul},[\,])\}) &= \{\ \mathsf{mul}_{[\,]},\mathsf{add}_{[\,]}\ \}, \\ \mathcal{ReqD}(R_7,\{(\mathsf{mul},[1])\}) &= \{\ \mathsf{mul}_{[1]},\mathsf{add}_{[\,]}\ \}, \\ \mathcal{ReqD}(R_7,\{(\mathsf{mul},[2])\}) &= \{\ \mathsf{mul}_{[2]},\mathsf{add}_{[2]}\ \}, \\ \mathcal{ReqD}(R_7,\{(\mathsf{mul},[1,2])\}) &= \{\ \mathsf{mul}_{[1,2]},\mathsf{add}_{[1,2]}\ \}. \end{aligned}$$

□

We define the TRS $\mathcal{I}trs(R,D_I)$ that shows which argument of each defined symbol is known, and that also shows what rules are necessary.

Definition 6.2.10 *Let R be a TRS over a signature $\mathcal{F}$ and $D_I \subseteq \mathbb{I}_{\mathcal{D}_R}$. The TRS $\mathcal{I}trs(R,D_I)$ is defined as follows:*

$$\begin{aligned} \mathcal{I}trs(R,D_I) = \{\ & f_I(t_1,\ldots,t_n) \to u \mid \rho : f(t_1,\ldots,t_n) \to r \in R, \\ & \qquad f_I \in \mathcal{ReqD}(R,D_I), |I| < n, \\ & \qquad u \equiv \mathcal{I}term(r,\mathcal{UV}ar(\rho,I)) \quad \} \\ \cup\{\ & f_I(t_1,\ldots,t_n) \to u \mid \rho : f(t_1,\ldots,t_n) \to r \in R, \\ & \qquad f_I \in D_I, |I| = n, \\ & \qquad u \equiv \mathcal{I}term(r,\mathcal{UV}ar(\rho,I)) \quad \}. \end{aligned}$$

Example 6.2.11 Consider the following TRS R_7 over $\{0,\mathsf{s},\mathsf{add},\mathsf{mul}\}$ in Example 6.2.9 again. For $\{(\mathsf{mul},[\,])\}$, $\{(\mathsf{mul},[1])\}$, $\{(\mathsf{mul},[2])\}$ and $\{(\mathsf{mul},[1,2])\}$, we obtain by $\mathcal{I}trs$ the followings:

$$\begin{aligned} &\mathcal{I}trs(R_7,\{(\mathsf{mul},[\,])\}) \\ &\quad = \{\ \mathsf{add}_{[\,]}(0,y) \to y, \\ &\qquad \mathsf{add}_{[\,]}(\mathsf{s}(x),y) \to \mathsf{s}(\mathsf{add}_{[\,]}(x,y)), \\ &\qquad \mathsf{mul}_{[\,]}(0,y) \to 0, \\ &\qquad \mathsf{mul}_{[\,]}(x,0) \to 0, \\ &\qquad \mathsf{mul}_{[\,]}(\mathsf{s}(x),\mathsf{s}(y)) \to \mathsf{s}(\mathsf{add}_{[\,]}(\mathsf{mul}_{[\,]}(x,\mathsf{s}(y)),y)) \quad \}, \end{aligned}$$

$$\begin{aligned}
&\mathcal{I}trs(R_7, \{(\mathsf{mul}, [1])\}) \\
&\quad = \{\ \mathsf{add}_{[\ ]}(0, y) \to y, \\
&\qquad \mathsf{add}_{[\ ]}(\mathsf{s}(x), y) \to \mathsf{s}(\mathsf{add}_{[\ ]}(x, y)), \\
&\qquad \mathsf{mul}_{[1]}(0, y) \to 0, \\
&\qquad \mathsf{mul}_{[1]}(x, 0) \to 0, \\
&\qquad \mathsf{mul}_{[1]}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{s}(\mathsf{add}_{[\ ]}(\mathsf{mul}_{[1]}(x, \mathsf{s}(y)), y))\ \ \},
\end{aligned}$$

$$\begin{aligned}
&\mathcal{I}trs(R_7, \{(\mathsf{mul}, [2])\}) \\
&\quad = \{\ \mathsf{add}_{[2]}(0, y) \to y, \\
&\qquad \mathsf{add}_{[2]}(\mathsf{s}(x), y) \to \mathsf{s}(\mathsf{add}_{[2]}(x, y)), \\
&\qquad \mathsf{mul}_{[2]}(0, y) \to 0, \\
&\qquad \mathsf{mul}_{[2]}(x, 0) \to 0, \\
&\qquad \mathsf{mul}_{[2]}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{s}(\mathsf{add}_{[2]}(\mathsf{mul}_{[2]}(x, \mathsf{s}(y)), y))\ \ \},
\end{aligned}$$

$$\begin{aligned}
&\mathcal{I}trs(R_7, \{(\mathsf{mul}, [1, 2])\}) \\
&\quad = \{\ \mathsf{add}_{[1,2]}(0, y) \to y, \\
&\qquad \mathsf{add}_{[1,2]}(\mathsf{s}(x), y) \to \mathsf{s}(\mathsf{add}_{[1,2]}(x, y)), \\
&\qquad \mathsf{mul}_{[1,2]}(0, y) \to 0, \\
&\qquad \mathsf{mul}_{[1,2]}(x, 0) \to 0, \\
&\qquad \mathsf{mul}_{[1,2]}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{s}(\mathsf{add}_{[1,2]}(\mathsf{mul}_{[1,2]}(x, \mathsf{s}(y)), y))\ \ \}.
\end{aligned}$$

□

Now, to generate partial-inverse CTRSs, we extend $\mathscr{T}_{\mathsf{full}}$, $\mathcal{I}nv\mathcal{R}ule_{\mathsf{full}}$ and $\mathcal{I}nv_{\mathsf{full}}$.

Definition 6.2.12 ($\mathscr{T}_{\mathsf{par}}$) *Let $\mathcal{F}$ be a signature divided into the constructor set $\mathcal{C}$ and the defined-symbol set $\mathcal{D}$: $\mathcal{F} = \mathcal{C} \uplus \mathcal{D}$. The procedure $\mathscr{T}_{\mathsf{full}}$ inputted a term which is labeled with indexes, is inductively defined as follows:*

(a) $\mathscr{T}_{\mathsf{par}}(x) = \langle\!\langle\, x\ ;\ \mathtt{true}\ \rangle\!\rangle$ *where x is a variable,*

(b) $\mathscr{T}_{\mathsf{par}}(c(t_1, \ldots, t_n)) = \langle\!\langle\, c(u_1, \ldots, u_n)\ ;\ \mathcal{C}ond_1 \wedge \cdots \wedge \mathcal{C}ond_n\ \rangle\!\rangle$
where $c \in \mathcal{C}$ and $\mathscr{T}_{\mathsf{par}}(t_i) = \langle\!\langle\, u_i\ ;\ \mathcal{C}ond_i\ \rangle\!\rangle$,

(c) $\mathscr{T}_{\mathsf{par}}(f_I(t_1, \ldots, t_n))$
$= \langle\!\langle\, y\ ;\ f_I^{\#}(y, \mathsf{tp}_{|I|}(t_{I_1}, \ldots, t_{I_{|I|}})) \twoheadrightarrow \mathsf{tp}_{|\bar{I}|}(u_{\bar{I}_1}, \ldots, u_{\bar{I}_{|\bar{I}|}}) \wedge \mathcal{C}ond_{\bar{I}_1} \wedge \mathcal{C}ond_{\bar{I}_{|\bar{I}|}}\ \rangle\!\rangle$
where $f \in \mathcal{D}$, $|I| < n$, y is a fresh variable and $\mathscr{T}_{\mathsf{par}}(t_{\bar{I}_i}) = \langle\!\langle\, u_{\bar{I}_i}\ ;\ \mathcal{C}ond_{\bar{I}_i}\ \rangle\!\rangle$, and

(d) $\mathscr{T}_{\mathsf{par}}(f_{[1,\ldots,n]}(t_1, \ldots, t_n)) = \langle\!\langle\, f(u_1, \ldots, u_n)\ ;\ \mathtt{true} \wedge \mathcal{C}ond_1 \wedge \cdots \wedge \mathcal{C}ond_n\ \rangle\!\rangle$
where $f \in \mathcal{D}$ and $\mathscr{T}_{\mathsf{par}}(t_i) = \langle\!\langle\, u_i\ ;\ \mathcal{C}ond_i\ \rangle\!\rangle$.

The word 'fresh' in (c) means that in (c), $y \notin \mathcal{V}ar(t_{\bar{I}_j}, u_{\bar{I}_j}, \mathcal{C}ond_{\bar{I}_j})$ and $y \notin \mathcal{V}ar(t_{I_j})$, and in (b) and (c), variables introduced in each $\mathscr{T}_{\mathsf{par}}(t_i) = \langle\!\langle\, u_i \,;\, \mathcal{C}ond_i \,\rangle\!\rangle$ are disjoint, that is, $\mathcal{V}ar(u_i, \mathcal{C}ond_i) \cap \mathcal{V}ar(t_j, u_j, \mathcal{C}ond_j) = \emptyset$ for each i and j with $i \neq j$.

The above procedure $\mathscr{T}_{\mathsf{par}}$ terminates and return a pair of a term and a condition since any input term is finite. Note that for $\mathscr{T}_{\mathsf{par}}(t) = \langle\!\langle\, u \,;\, \mathcal{C}ond \,\rangle\!\rangle$, u can be represented as $C[\![u_1, \ldots, u_n]\!]$ with a constructor context C and terms u_i with $\mathtt{root}(u_i) \in \mathcal{D}$.

Example 6.2.13 Consider the signature $\{\mathsf{0}, \mathsf{s}, \mathsf{add}, \mathsf{mul}\}$ again. Then, we have the followings by $\mathscr{T}_{\mathsf{par}}$:

$$\begin{aligned}
&\mathscr{T}_{\mathsf{par}}(\mathsf{s}(\mathsf{add}_{[\,]}(\mathsf{mul}_{[\,]}(x, \mathsf{s}(y)), y))) \\
&\quad = \langle\!\langle\, \mathsf{s}(z) \,;\, \mathsf{add}^{\#}_{[\,]}(z) \twoheadrightarrow \mathsf{tp}_2(w, y) \wedge \mathsf{mul}^{\#}_{[\,]}(w) \twoheadrightarrow \mathsf{tp}_2(x, \mathsf{s}(y)) \,\rangle\!\rangle, \\
&\mathscr{T}_{\mathsf{par}}(\mathsf{s}(\mathsf{add}_{[\,]}(\mathsf{mul}_{[1]}(x, \mathsf{s}(y)), y))) \\
&\quad = \langle\!\langle\, \mathsf{s}(z) \,;\, \mathsf{add}^{\#}_{[\,]}(z) \twoheadrightarrow \mathsf{tp}_2(w, y) \wedge \mathsf{mul}^{\#}_{[1]}(w, x) \twoheadrightarrow \mathsf{tp}_1(\mathsf{s}(y)) \,\rangle\!\rangle, \\
&\mathscr{T}_{\mathsf{par}}(\mathsf{s}(\mathsf{add}_{[2]}(\mathsf{mul}_{[2]}(x, \mathsf{s}(y)), y))) \\
&\quad = \langle\!\langle\, \mathsf{s}(z) \,;\, \mathsf{add}^{\#}_{[2]}(z, y) \twoheadrightarrow \mathsf{tp}_1(w) \wedge \mathsf{mul}^{\#}_{[2]}(w, \mathsf{s}(y)) \twoheadrightarrow \mathsf{tp}_1(x) \,\rangle\!\rangle, \\
&\mathscr{T}_{\mathsf{par}}(\mathsf{s}(\mathsf{add}_{[1,2]}(\mathsf{mul}_{[1,2]}(x, \mathsf{s}(y)), y))) = \langle\!\langle\, \mathsf{s}(\mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y)) \,;\, \mathtt{true} \,\rangle\!\rangle.
\end{aligned}$$

□

The following proposition shows a property associated with the case (d) in Definition 6.2.12, which is not in the definition of $\mathscr{T}_{\mathsf{full}}$.

Proposition 6.2.14 *Let $\mathcal{F}$ be a signature divided into the constructor set $\mathcal{C}$ and the defined-symbol set $\mathcal{D}$: $\mathcal{F} = \mathcal{C} \uplus \mathcal{D}$. Let t be a term over $\mathcal{F}$, X be a set of variables. If $\mathcal{V}ar(t) \cap X = \emptyset$, then $\mathscr{T}_{\mathsf{par}}(\mathcal{I}term(t, X)) = \langle\!\langle\, t \,;\, \mathtt{true} \,\rangle\!\rangle$.*

Proof. This proposition follows obviously from the definitions of $\mathcal{I}ndex$, $\mathcal{I}term$ and $\mathscr{T}_{\mathsf{par}}$, and Proposition 6.2.7. □

Using the extended procedure $\mathscr{T}_{\mathsf{par}}$, we extend $\mathcal{I}nv_{\mathsf{full}}$ for partial inverse computation, as follows:

Definition 6.2.15 ($\mathcal{I}nv_{\mathsf{par}}$) *Let R be a constructor TRS over a signature $\mathcal{F}$ and $D_I \subseteq \mathbb{I}_{\mathcal{D}_R}$. For a rewrite rule $\rho : f_I(w_1, \ldots, w_n) \to u \in \mathcal{I}trs(R)$, its corresponding conditional rewrite rule $\mathcal{I}nv\mathcal{R}ule_{\mathsf{par}}(\rho)$ of $f^{\#}_I$ is defined as follows:*

$$\begin{aligned}
\mathcal{I}nv\mathcal{R}ule_{\mathsf{par}}(f_I(w_1, \ldots, w_n) \to u) = f^{\#}_I(C[y_1, \ldots, y_m]) \to \mathsf{tp}_{|\bar{I}|}(w_{\bar{I}_1}, \ldots, w_{\bar{I}_{|\bar{I}|}}) \\
\Leftarrow u_1 \twoheadrightarrow y_1 \wedge \cdots \wedge u_m \twoheadrightarrow y_m \wedge \mathcal{C}ond
\end{aligned}$$

where y_i *is a variable with* $y_i \notin \mathcal{V}ar(w_1, \ldots, w_n)$, $\mathscr{T}_{\mathsf{par}}(u) = \langle\!|C[\![u_1, \ldots, u_m]\!]; \mathcal{C}ond|\!\rangle$ *and* $(\mathcal{V}ar(u_1, \ldots, u_m, \mathcal{C}ond) \setminus \mathcal{V}ar(u)) \cap \mathcal{V}ar(y_1, \ldots, y_m, w_1, \ldots, w_n) = \emptyset$.

The partial-inverse CTRS $\mathcal{I}nv_{\mathsf{par}}(R, D_I)$ *is defined as follows:*

$$\begin{aligned}\mathcal{I}nv_{\mathsf{par}}(R, D_I) = \{\, & \mathcal{I}nv\mathcal{R}ule_{\mathsf{par}}(s \to u) \mid u \to u \in \mathcal{I}trs(R) \,\} \\ & \cup\{\, f_I^{\#}(f(x_0, x_{I_1}, \ldots, x_{I_{|I|}})) \to \mathsf{tp}_{|\bar{I}|}(x_{\bar{I}_1}, \ldots, x_{\bar{I}_{|\bar{I}|}}) \\ & \qquad\qquad \mid f_I \in \mathcal{R}eq\mathcal{D}(R, D_I) \,\} \\ & \cup\{\, f(w_1, \ldots, w_n) \to r \mid f(w_1, \ldots, w_n) \to r \in R, \\ & \qquad\qquad f_{[1,\ldots,n]} \in \mathcal{R}eq\mathcal{D}(R, D_I) \quad \}.\end{aligned}$$

The above rewrite rule $f_I^{\#}(f(x_0, x_{I_1}, \ldots, x_{I_{|I|}})) \to \mathsf{tp}_{|\bar{I}|}(x_{\bar{I}_1}, \ldots, x_{\bar{I}_{|\bar{I}|}})$ *is called the* partial-inverse-property rule *of the defined symbol* f *with respect to* I.

It is clear that $\mathcal{I}nv\mathcal{R}ule_{\mathsf{par}}(\rho)$ is a conditional rewrite rule and hence $\mathcal{I}nv_{\mathsf{full}}(R)$ is a CTRS. In Subsection 3.6.3, we show the correctness of $\mathcal{I}nv_{\mathsf{par}}$ with D_I, that is, that $\mathcal{I}nv_{\mathsf{par}}(R, D_I)$ is a partial-inverse CTRS of R with respect to D_I.

Example 6.2.16 Consider the following constructor TRS over the signature {s, 0, add, mul} again:

$$\begin{aligned}R_7 = \{\; & \mathsf{add}(0, y) \to y, \quad \mathsf{add}(\mathsf{s}(x), y) \to \mathsf{s}(\mathsf{add}(x, y)), \\ & \mathsf{mul}(0, y) \to 0, \quad \mathsf{mul}(x, 0) \to 0, \\ & \mathsf{mul}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{s}(\mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y)) \qquad \}.\end{aligned}$$

For { (mul, [ ]) }, { (mul, [1]) }, { (mul, [2]) } and { (mul, [1, 2]) }, referring Example 6.2.11, we obtain by $\mathcal{I}nv_{\mathsf{par}}$ the following CTRSs:

$$\begin{aligned}&\mathcal{I}nv_{\mathsf{par}}(R_7, \{(\mathsf{mul}, [\,])\}) \\ &\quad = \{\; \mathsf{add}^{\#}_{[\,]}(y) \to \mathsf{tp}_2(0, y), \\ &\qquad \mathsf{add}^{\#}_{[\,]}(\mathsf{s}(z)) \to \mathsf{tp}_2(\mathsf{s}(x), y) \Leftarrow \mathsf{add}^{\#}_{[\,]}(z) \twoheadrightarrow \mathsf{tp}_2(x, y), \\ &\qquad \mathsf{add}^{\#}_{[\,]}(\mathsf{add}(x, y)) \to \mathsf{tp}_2(x, y), \\ &\qquad \mathsf{mul}^{\#}_{[\,]}(0) \to \mathsf{tp}_2(0, y), \\ &\qquad \mathsf{mul}^{\#}_{[\,]}(0) \to \mathsf{tp}_2(x, 0), \\ &\qquad \mathsf{mul}^{\#}_{[\,]}(\mathsf{s}(z)) \to \mathsf{tp}_2(\mathsf{s}(x), \mathsf{s}(y)) \\ &\qquad\qquad \Leftarrow \mathsf{add}^{\#}_{[\,]}(z) \twoheadrightarrow \mathsf{tp}_2(w, y) \wedge \mathsf{mul}^{\#}_{[\,]}(w) \twoheadrightarrow \mathsf{tp}_2(x, \mathsf{s}(y)), \\ &\qquad \mathsf{mul}^{\#}_{[\,]}(\mathsf{mul}(x, y)) \to \mathsf{tp}_2(x, y) \qquad\qquad\qquad\qquad\qquad\qquad \},\end{aligned}$$

$$
\begin{aligned}
&\mathcal{I}nv_{\mathsf{par}}(R_7, \{(\mathsf{mul}, [1])\}) \\
&\quad = \{\ \mathsf{add}^{\#}_{[\,]}(y) \to \mathsf{tp}_2(0, y), \\
&\qquad \mathsf{add}^{\#}_{[\,]}(\mathsf{s}(z)) \to \mathsf{tp}_2(\mathsf{s}(x), y) \Leftarrow \mathsf{add}^{\#}_{[\,]}(z) \twoheadrightarrow \mathsf{tp}_2(x, y), \\
&\qquad \mathsf{add}^{\#}_{[\,]}(\mathsf{add}(x, y)) \to \mathsf{tp}_2(x, y), \\
&\qquad \mathsf{mul}^{\#}_{[1]}(0, 0) \to \mathsf{tp}_1(y), \\
&\qquad \mathsf{mul}^{\#}_{[1]}(0, x) \to \mathsf{tp}_1(0), \\
&\qquad \mathsf{mul}^{\#}_{[1]}(\mathsf{s}(z), \mathsf{s}(x)) \to \mathsf{tp}_1(\mathsf{s}(y)) \\
&\qquad\qquad \Leftarrow \mathsf{add}^{\#}_{[\,]}(z) \twoheadrightarrow \mathsf{tp}_2(w, y) \wedge \mathsf{mul}^{\#}_{[1]}(w, x) \twoheadrightarrow \mathsf{tp}_1(\mathsf{s}(y)), \\
&\qquad \mathsf{mul}^{\#}_{[1]}(\mathsf{mul}(x, y), x) \to \mathsf{tp}_1(y) \qquad\qquad \},
\end{aligned}
$$

$$
\begin{aligned}
&\mathcal{I}nv_{\mathsf{par}}(R_7, \{(\mathsf{mul}, [2])\}) \\
&\quad = \{\ \mathsf{add}^{\#}_{[2]}(y, y) \to \mathsf{tp}_1(0), \\
&\qquad \mathsf{add}^{\#}_{[2]}(\mathsf{s}(z), y) \to \mathsf{tp}_1(\mathsf{s}(x)) \Leftarrow \mathsf{add}^{\#}_{[2]}(z, y) \twoheadrightarrow \mathsf{tp}_1(x), \\
&\qquad \mathsf{add}^{\#}_{[2]}(\mathsf{add}(x, y), y) \to \mathsf{tp}_1(x), \\
&\qquad \mathsf{mul}^{\#}_{[2]}(0, y) \to \mathsf{tp}_1(0), \\
&\qquad \mathsf{mul}^{\#}_{[2]}(0, 0) \to \mathsf{tp}_1(x), \\
&\qquad \mathsf{mul}^{\#}_{[2]}(\mathsf{s}(z), \mathsf{s}(y)) \to \mathsf{tp}_1(\mathsf{s}(x)) \\
&\qquad\qquad \Leftarrow \mathsf{add}^{\#}_{[2]}(z, y) \twoheadrightarrow \mathsf{tp}_1(w) \wedge \mathsf{mul}^{\#}_{[2]}(w, \mathsf{s}(y)) \twoheadrightarrow \mathsf{tp}_1(x), \\
&\qquad \mathsf{mul}^{\#}_{[2]}(\mathsf{mul}(x, y), y) \to \mathsf{tp}_1(x) \qquad\qquad \},
\end{aligned}
$$

$$
\begin{aligned}
&\mathcal{I}nv_{\mathsf{par}}(R_7, \{(\mathsf{mul}, [1, 2])\}) \\
&\quad = \{\ \mathsf{mul}^{\#}_{[1,2]}(0, 0, y) \to \mathsf{tp}_0, \\
&\qquad \mathsf{mul}^{\#}_{[1,2]}(0, x, 0) \to \mathsf{tp}_0, \\
&\qquad \mathsf{mul}^{\#}_{[1,2]}(\mathsf{s}(z), \mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{tp}_0 \\
&\qquad\qquad \Leftarrow \mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y) \twoheadrightarrow \mathsf{tp}_1(z), \\
&\qquad \mathsf{mul}^{\#}_{[1,2]}(\mathsf{mul}(x, y), x, y) \to \mathsf{tp}_0, \\
&\qquad \mathsf{add}(0, y) \to y, \\
&\qquad \mathsf{add}(\mathsf{s}(x), y) \to \mathsf{s}(\mathsf{add}(x, y)), \\
&\qquad \mathsf{mul}(0, y) \to 0, \\
&\qquad \mathsf{mul}(x, 0) \to 0, \\
&\qquad \mathsf{mul}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{s}(\mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y)) \qquad \}.
\end{aligned}
$$

□

6.3 Correctness of Partial Inverse Compiler

In this subsection, we prove the correctness of our partial inverse compiler $\mathcal{I}nv_{\mathsf{par}}$, that is, that for a convergent constructor TRS R and $D_I \subseteq \mathbb{I}_{\mathcal{D}_R}$, $\mathcal{I}nv_{\mathsf{par}}(R, D_I)$

is a partial-inverse CTRS of R with respect to D_I.

We prove the correctness of $\mathcal{I}nv_{\mathsf{par}}(R, D_I)$ similarly to that of $\mathcal{I}nv_{\mathsf{par}}(R)$, such as Lemma 3.6.15, 3.6.16 and Theorem 3.6.17.

Lemma 6.3.1 *Let R be a constructor TRS over a signature $\mathcal{F}$, and $D_I \subseteq \mathbb{I}_{\mathcal{D}_R}$. Let t_0 be a term over $\mathcal{F}$, X be a variable set, and $t = \mathcal{I}term(t_0, X) \in \mathcal{T}(\mathcal{R}eq\mathcal{D}(R, D_I) \cup \mathcal{C}_R, \mathcal{X})$. Let $\mathscr{T}_{\mathsf{par}}(t) = \langle\!\langle\, u \,;\, \mathcal{C}ond \,\rangle\!\rangle$. If $t \xrightarrow[\text{in}]{*}_R s \in NF_R(\mathcal{F}, \mathcal{X})$, then there exists an $NF_R(\mathcal{F}, \mathcal{X})$-substitution σ such that $\mathcal{D}om(\sigma) \subseteq \mathcal{V}ar(\mathcal{C}ond) \setminus \mathcal{V}ar(t)$, $u\sigma \xrightarrow{*}_{\mathcal{I}nv_{\mathsf{par}}(R)} s$, and $\mathcal{C}ond(\sigma, \to_{\mathcal{I}nv_{\mathsf{par}}(R)})$.*

Proof. (Sketch.) Similarly to Lemma 6.3.1, this lemma can be proved by induction on the lexicographic combination of the number k of steps of $t \xrightarrow[\text{in}]{k}_R s$, and the structure of term t. □

Lemma 6.3.2 *Let R be a constructor TRS over a signature $\mathcal{F}$, and $D_I \subseteq \mathbb{I}_{\mathcal{D}_R}$. Let t_0 be a term over $\mathcal{F}$, X be a variable set, and $t = \mathcal{I}term(t_0, X) \in \mathcal{T}(\mathcal{R}eq\mathcal{D}(R, D_I) \cup \mathcal{C}_R, \mathcal{X})$. Let $\mathscr{T}_{\mathsf{par}}(t) = \langle\!\langle\, u \,;\, \mathcal{C}ond \,\rangle\!\rangle$. Let σ be an $NF_R(\mathcal{F}, \mathcal{X})$-substitution such that $\mathcal{D}om(\sigma) \subseteq \mathcal{V}ar(\mathcal{C}ond) \setminus \mathcal{V}ar(t)$. Then, $\mathcal{C}ond(\sigma, \to_{\mathcal{I}nv_{\mathsf{par}}(R)})$ implies $t \xrightarrow[\text{in}]{*}_R u\sigma$ $(\in NF_R(\mathcal{F}, \mathcal{X}))$.*

Proof. (Sketch.) Similarly to Lemma 6.3.2, this lemma can be proved by induction on the lexicographic combination of the level k of rewrite relation of $\mathcal{C}ond(\sigma, \xrightarrow[k]{}_{\mathcal{I}nv_{\mathsf{par}}(R)})$, and the structure of term t. □

Theorem 6.3.3 *Let R be a constructor TRS over a signature $\mathcal{F}$, and $D_I \subseteq \mathbb{I}_{\mathcal{D}_R}$. Let $(f, I) \in D_I$ and $t, t_1, \ldots, t_n$ are normal forms with respect to $\to_R$. Then, $f(t_1, \ldots, t_n) \xrightarrow[\text{in}]{*}_R t$ iff $f_I^{\#}(t, t_{I_1}, \ldots, t_{I_{|I|}}) \to_{\mathcal{I}nv_{\mathsf{par}}(R, D_I)} \mathsf{tp}_{|\bar{I}|}(t_{\bar{I}_1}, \ldots, t_{\bar{I}_{|\bar{I}|}})$.*

Proof. (Sketch.) This claim can be proved similarly to Theorem 3.6.17, using Lemma 6.3.1 and 6.3.2. □

From Proposition 3.6.18, we obtain the following corollary:

Corollary 6.3.4 *Let R be a convergent constructor TRS over a signature $\mathcal{F}$, and $D_I \subseteq \mathbb{I}_{\mathcal{D}_R}$. Let $(f, I) \in D_I$ and $t, t_1, \ldots, t_n$ are normal forms of R. Then, $f(t_1, \ldots, t_n) \xrightarrow{*}_R t$ iff $f_I^{\#}(t, t_{I_1}, \ldots, t_{I_{|I|}}) \to_{\mathcal{I}nv_{\mathsf{par}}(R, D_I)} \mathsf{tp}_{|\bar{I}|}(t_{\bar{I}_1}, \ldots, t_{\bar{I}_{|\bar{I}|}})$.*

From the above corollary and $\mathcal{T}(\mathcal{C}_R, \mathcal{X}) \subseteq NF_R(\mathcal{F}, \mathcal{X})$, it is shown that the CTRS $\mathcal{I}nv_{\mathsf{par}}(R, D_I)$ is a partial-inverse system of the convergent constructor TRS R with respect to D_I in the sense of Definition 6.1.1.

As we have shown the correctness of $\mathcal{I}nv_{\mathsf{par}}$, it is said that $\mathsf{add}^{\#}_{\{2\}}$ computes subtraction of two natural numbers (compare min in Example 6.1.2), each of $\mathsf{mul}^{\#}_{\{1\}}$ and $\mathsf{mul}^{\#}_{\{2\}}$ computes division of two natural numbers.

Chapter 7

Applications of Inverse EV-TRSs

In this chapter, we give three applications of the inverse EV-TRSs generated by our transformations $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(\cdots))$ and $\mathcal{I}nv_{\mathsf{par}}(\cdots)$.

The first example shows that inverse programs are used to transform a given program (CTRS) which is correctly defined but is not executable, to an equivalent program which is executable. This example is the solution of our motivation problem as shown in Section 1.1.

The second application shows that generated inverse EV-TRSs are usable to solve conditional equations. This is a mathematical application of inverse programs.

The last one is an interesting example in which we try to generate an inverse TRS from another inverse TRS generated from a given convergent non-erasing constructor TRS. We also discuss the equivalence between the input TRS and the final output TRS.

7.1 Program Transformation

In this section, we introduce an example in which the generated inverse TRS plays a useful role in a program transformation. In this example, we consider the CTRS whose definition is correct in the sense of specifications, and on which some terms cannot be rewritten to desirable normal forms. By transforming it using inverse TRSs, we construct an equivalent CTRS on which every term can be rewritten to its desirable normal form. This example shows a solution of our motivation problem.

Consider the following TRS which represents our motivation program *gcd* in

Section 1.1 as a TRS:

$$\begin{aligned} R_{27} = \{\ & \mathsf{gcd}(\mathsf{add}(x,y),y) \to \mathsf{gcd}(x,y), \\ & \mathsf{gcd}(y,\mathsf{add}(x,y)) \to \mathsf{gcd}(x,y), \\ & \mathsf{gcd}(x,\mathsf{0}) \to x, \\ & \mathsf{gcd}(\mathsf{0},x) \to x, \\ & \mathsf{gcd}(x,y) \to \mathsf{gcd}(y,x) \Leftarrow \mathsf{leq}(y,x) \twoheadrightarrow \mathsf{F}, \\ & \mathsf{add}(\mathsf{0},y) \to y, \\ & \mathsf{add}(\mathsf{s}(x),y) \to \mathsf{s}(\mathsf{add}(x,y)) \qquad \}. \end{aligned}$$

As we described in Section 1.1, the function gcd computes the greatest common divisor (G.C.D.) of two natural numbers. The above definition is correct as G.C.D. in the sense of Euclid's Algorithm. However, the terms $\mathsf{gcd}(\mathsf{s}^m(\mathsf{0}),\mathsf{s}^n(\mathsf{0}))$ with $m,n>0$ cannot be rewritten to each desirable normal form $\mathsf{s}^k(\mathsf{0})$ where k is the G.C.D. of m and n. For example, the term $\mathsf{gcd}(\mathsf{s}^4(\mathsf{0}),\mathsf{s}^2(\mathsf{0}))$ is a normal form and hence cannot be rewritten to $\mathsf{s}^2(\mathsf{0})$. The cause is that the first argument $\mathsf{add}(x,y)$ of gcd in the left-hand side of the first rule cannot be matched with $\mathsf{s}^m(\mathsf{0})$ in the target term $\mathsf{gcd}(\mathsf{s}^4(\mathsf{0}),\mathsf{s}^2(\mathsf{0}))$ since one of them contains add. In other words, the above TRS R_{27} is not a constructor system.

To solve this problem, we transform R_{27} to a constructor CTRS which really computes G.C.D., in the following ways using the inverse $\mathsf{add}^{\#}$:

1. replace the term $\mathsf{add}(x,y)$ in the left-hand side of the first (and second) rule of R_{27} with a fresh variable z, and then adding the condition $\mathsf{add}^{\#}(z) \twoheadrightarrow \mathsf{tp}_2(x,y)$ to the conditional part, and then
2. remove the rules of add and adding the rules of $\mathsf{add}^{\#}$.

Consequently, we obtain the following CTRS:

$$\begin{aligned} R_{28} = \{\ & \mathsf{gcd}(z,y) \to \mathsf{gcd}(x,y) \Leftarrow \mathsf{add}^{\#}(z) \twoheadrightarrow \mathsf{tp}_2(x,y), \\ & \mathsf{gcd}(y,z) \to \mathsf{gcd}(x,y) \Leftarrow \mathsf{add}^{\#}(z) \twoheadrightarrow \mathsf{tp}_2(x,y), \\ & \mathsf{gcd}(x,\mathsf{0}) \to x, \\ & \mathsf{gcd}(\mathsf{0},x) \to x, \\ & \mathsf{add}^{\#}(y) \to \mathsf{tp}_2(\mathsf{0},y), \\ & \mathsf{add}^{\#}(\mathsf{s}(z)) \to \mathsf{tp}_2(\mathsf{s}(x),y) \Leftarrow \mathsf{add}^{\#}(z) \twoheadrightarrow \mathsf{tp}_2(x,y), \\ & \mathsf{add}^{\#}(\mathsf{add}(x,y)) \to \mathsf{tp}_2(x,y) \qquad \}. \end{aligned}$$

Now we can obtain the solution $\mathsf{s}^2(\mathsf{0})$ of $\mathsf{gcd}(\mathsf{s}^4(\mathsf{0}),\mathsf{s}^2(\mathsf{0}))$ by rewriting on R_{28} as follows:

$$\mathsf{gcd}(\mathsf{s}^4(\mathsf{0}),\mathsf{s}^2(\mathsf{0})) \to_{R_{28}} \mathsf{gcd}(\mathsf{0},\mathsf{s}^2(\mathsf{0})) \to_{R_{28}} \mathsf{s}^2(\mathsf{0}).$$

The reason why we do not use the unconditional TRSs (R_6 and $R_{11}/_{\{\mathsf{add}^\#,\mathsf{u}_1\}}$) which define $\mathsf{add}^\#$, is that the completeness of the transformations $\mathbb{U}_{\mathsf{PT}}$ and $\mathbb{U}$ is not guaranteed for both R_{28} and $R_{28}/_{\{\mathsf{gcd}\}} \cup R_{11}/_{\{\mathsf{add}^\#,\mathsf{u}_1\}}$. If either Conjecture 3.6.25 or 3.6.26 is proved, then the following TRS is more useful than R_{28}:

$$
\begin{aligned}
R_{29} = \{\ & \mathsf{gcd}(z,y) \to \mathsf{gcd}(x,y) \Leftarrow \mathsf{add}^\#(z) \twoheadrightarrow \mathsf{tp}_2(x,y),\\
& \mathsf{gcd}(y,z) \to \mathsf{gcd}(x,y) \Leftarrow \mathsf{add}^\#(z) \twoheadrightarrow \mathsf{tp}_2(x,y),\\
& \mathsf{gcd}(x,\mathsf{0}) \to x,\\
& \mathsf{gcd}(\mathsf{0},x) \to x,\\
& \mathsf{add}^\#(y) \to \mathsf{tp}_2(\mathsf{0},y),\\
& \mathsf{add}^\#(\mathsf{s}(z)) \to \mathsf{u}_1(\mathsf{add}^\#(z)),\\
& \mathsf{u}_1(\mathsf{tp}_2(x,y)) \to \mathsf{tp}_2(\mathsf{s}(x),y),\\
& \mathsf{add}^\#(\mathsf{add}(x,y)) \to \mathsf{tp}_2(x,y) \qquad \}.
\end{aligned}
$$

On the other hand, the second (and first) argument y of $\mathsf{add}(x,y)$ in the left-hand side of the first (second, respectively) rule is given as the second (first, respectively) argument of gcd of the left-hand side of the first (second, respectively) rule. This means that the second argument of add in the first rule is known. Hence we can transform R_{27} using the partial-inverse $\mathsf{add}^\#_{[2]}$ as follows:

$$
\begin{aligned}
R_{30} = \{\ & \mathsf{gcd}(z,y) \to \mathsf{gcd}(x,y) \Leftarrow \mathsf{add}^\#_{[2]}(z,y) \twoheadrightarrow x,\\
& \mathsf{gcd}(y,z) \to \mathsf{gcd}(x,y) \Leftarrow \mathsf{add}^\#_{[2]}(z,y) \twoheadrightarrow x,\\
& \mathsf{gcd}(x,\mathsf{0}) \to x, \quad \mathsf{gcd}(\mathsf{0},x) \to x,\\
& \mathsf{add}^\#_{[2]}(y,y) \to \mathsf{0},\\
& \mathsf{add}^\#_{[2]}(\mathsf{s}(z),y) \to \mathsf{s}(x) \Leftarrow \mathsf{add}^\#_{[2]}(z,y) \twoheadrightarrow x,\\
& \mathsf{add}^\#_{[2]}(\mathsf{add}(x,y),y) \to x \qquad \}.
\end{aligned}
$$

Thus, every CTRS which is not a constructor system, may be transformed into an equivalent constructor CTRS.

7.2 Solving Equation

In this section, we solve the following equation [14] using a partial-inverse CTRS generated by $\mathcal{I}nv_{\mathsf{par}}$:

$$
N =^? \mathsf{n}(d,m,y) = d + 30m + 360y, \quad (\,0 \le d < 30,\ 0 \le m < 12\,). \tag{7.1}
$$

The function n inputed d,m,y computes the number of elapsed days from the initial date encoded by $(0,0,0)$ to the given date (d,m,y) with a fixed year-length

of 360 days, a simple scheme of 12-equal months and 0-based dates. We encoded a system computes n as the following CTRS R_{31} over $\{0, \mathsf{s}, \mathsf{add}, \mathsf{mul}, \mathsf{leq}\}$:

$$\begin{aligned} R_{31} = \{\ \mathsf{n}(x_d, x_m, x_y) \to \mathsf{add}(x_d, \mathsf{add}(\mathsf{mul}(\mathsf{s}^{30}(0), x_m), \mathsf{mul}(\mathsf{s}^{360}(0), x_y))) \\ \Leftarrow \mathsf{leq}(0, x_d) \twoheadrightarrow \mathsf{T} \\ \wedge \mathsf{leq}(x_d, \mathsf{s}^{29}(0)) \twoheadrightarrow \mathsf{T} \\ \wedge \mathsf{leq}(0, x_m) \twoheadrightarrow \mathsf{T} \\ \wedge \mathsf{leq}(x_m, \mathsf{s}^{359}(0)) \twoheadrightarrow \mathsf{T}\} \cup R_{32}, \end{aligned}$$

$$\begin{aligned} R_{32} = \{\ & \mathsf{add}(0, y) \to y, \quad \mathsf{add}(\mathsf{s}(x), y) \to \mathsf{s}(\mathsf{add}(x, y)), \\ & \mathsf{mul}(0, y) \to 0, \quad \mathsf{mul}(x, 0) \to 0, \\ & \mathsf{mul}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{s}(\mathsf{add}(\mathsf{mul}(x, \mathsf{s}(y)), y)), \\ & \mathsf{leq}(0, 0) \to \mathsf{T}, \\ & \mathsf{leq}(\mathsf{s}(x), 0) \to \mathsf{F}, \\ & \mathsf{leq}(0, \mathsf{s}(y)) \to \mathsf{T}, \\ & \mathsf{leq}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{leq}(x, y) \qquad \}. \end{aligned}$$

Since the rule of n is deterministic Type 3, the above CTRS R_{31} can be transformed by $\mathbb{U}_{\mathsf{PT}}$ to the following TRS R_{33}:

$$\begin{aligned} R_{33} = \{\ & \mathsf{n}(x_d, x_m, x_y) \to \mathsf{un}(x_d, x_m, x_y, \mathsf{leq}(0, x_d), \mathsf{leq}(0, x_m), \\ & \qquad\qquad \mathsf{leq}(0, x_m), \mathsf{leq}(x_m, \mathsf{s}^{359}(0))), \\ & \mathsf{un}(x_d, x_m, x_y, \mathsf{T}, \mathsf{T}, \mathsf{T}, \mathsf{T}) \\ & \qquad \to \mathsf{add}(x_d, \mathsf{add}(\mathsf{mul}(\mathsf{s}^{30}(0), x_m), \mathsf{mul}(\mathsf{s}^{360}(0), x_y)))\} \cup R_{32}. \end{aligned}$$

The above TRS is a constructor system, and then we obtain the partial inverse CTRS R_{34} with respect to $\{\ (\mathsf{n}, [\ ])\ \}$ shown in Figure 7.1, by the transformation $\mathcal{I}nv_{\mathsf{par}}$.

The solution of the case $N = 400$ in the equation (7.1) is only $(10, 1, 1)$. By rewriting the term $\mathsf{n}(\mathsf{s}^{400}(0))$ on R_{34}, we can also obtain the normal form $\mathsf{tp}_3(\mathsf{s}^{10}(0), \mathsf{s}(0), \mathsf{s}(0))$: $\mathsf{n}(\mathsf{s}^{400}(0)) \xrightarrow{*}_{R_{34}} \mathsf{tp}_3(\mathsf{s}^{10}(0), \mathsf{s}(0), \mathsf{s}(0))$.

Thus, our transformational approach to (partial) inverse computation — inverse computation by inverse CTRSs — can solve conditional equations, easily handling a condition of equations.

7.3 Inverse of Inverse TRSs

In this section, we try to generate an inverse TRS of another inverse TRS which is generated from a given TRS. We have already shown that inverse EV-TRSs

$$
\begin{aligned}
R_{34} &= \mathcal{I}nv_{\mathsf{par}}(R_{33}, \{(\mathsf{n}, [\,])\}) \\
&= \{\ \mathsf{n}^{\#}_{[\,]}(z) \to \mathsf{tp}_3(x_d, x_m, x_y) \\
&\qquad \Leftarrow \mathsf{leq}^{\#}_{[1]}(y_1, 0) \twoheadrightarrow x_d \wedge \mathsf{leq}^{\#}_{[2]}(y_2, \mathsf{s}^{29}(0)) \twoheadrightarrow x_d \\
&\qquad\quad \wedge \mathsf{leq}^{\#}_{[1]}(y_3, 0) \twoheadrightarrow x_m \wedge \mathsf{leq}^{\#}_{[2]}(y_4, \mathsf{s}^{359}(0)) \twoheadrightarrow x_m, \\
&\quad \mathsf{un}(z) \to \mathsf{tp}_7(x_d, x_m, x_y, \mathsf{T}, \mathsf{T}, \mathsf{T}, \mathsf{T}) \\
&\qquad \Leftarrow \mathsf{add}^{\#}_{[\,]}(z) \twoheadrightarrow \mathsf{tp}_2(x_d, z_1) \wedge \mathsf{add}^{\#}_{[\,]}(z_1) \twoheadrightarrow \mathsf{tp}_2(z_2, z_3) \\
&\qquad\quad \wedge \mathsf{mul}^{\#}_{[1]}(z_2, \mathsf{s}^{30}(0)) \twoheadrightarrow x_m \wedge \mathsf{mul}^{\#}_{[1]}(z_3, \mathsf{s}^{360}(0)) \twoheadrightarrow x_y), \\
&\quad \mathsf{add}^{\#}_{[\,]}(y) \to \mathsf{tp}_2(0, y), \\
&\quad \mathsf{add}^{\#}_{[\,]}(\mathsf{s}(z)) \to \mathsf{tp}_2(\mathsf{s}(x), y) \Leftarrow \mathsf{add}^{\#}_{[\,]}(z) \twoheadrightarrow \mathsf{tp}_2(x, y), \\
&\quad \mathsf{add}^{\#}_{[\,]}(\mathsf{add}(x, y)) \to \mathsf{tp}_2(x, y), \\
&\quad \mathsf{mul}^{\#}_{[1]}(0, 0) \to y, \\
&\quad \mathsf{mul}^{\#}_{[1]}(0, x) \to 0, \\
&\quad \mathsf{mul}^{\#}_{[1]}(\mathsf{s}(z), \mathsf{s}(x)) \to \mathsf{s}(y) \\
&\qquad \Leftarrow \mathsf{add}^{\#}_{[\,]}(z) \twoheadrightarrow \mathsf{tp}_2(w, y) \wedge \mathsf{mul}^{\#}_{[1]}(w, x) \twoheadrightarrow \mathsf{s}(y), \\
&\quad \mathsf{mul}^{\#}_{[1]}(\mathsf{mul}(x, y), x) \to y, \\
&\quad \mathsf{leq}^{\#}_{[1]}(\mathsf{T}, 0) \to 0, \\
&\quad \mathsf{leq}^{\#}_{[1]}(\mathsf{F}, \mathsf{s}(x)) \to 0, \\
&\quad \mathsf{leq}^{\#}_{[1]}(\mathsf{T}, 0) \to \mathsf{s}(x), \\
&\quad \mathsf{leq}^{\#}_{[1]}(z, \mathsf{s}(x)) \to \mathsf{s}(y) \Leftarrow \mathsf{leq}^{\#}_{[1]}(z, x) \twoheadrightarrow y, \\
&\quad \mathsf{leq}^{\#}_{[1]}(\mathsf{leq}(x, y), x) \to y, \\
&\quad \mathsf{leq}^{\#}_{[2]}(\mathsf{T}, 0) \to 0, \\
&\quad \mathsf{leq}^{\#}_{[2]}(\mathsf{F}, 0) \to \mathsf{s}(x), \\
&\quad \mathsf{leq}^{\#}_{[2]}(\mathsf{T}, \mathsf{s}(x)) \to 0, \\
&\quad \mathsf{leq}^{\#}_{[2]}(^{\#}(z, \mathsf{s}(y)) \to \mathsf{s}(x) \Leftarrow \mathsf{leq}^{\#}_{[1]}(z, y) \twoheadrightarrow x, \\
&\quad \mathsf{leq}^{\#}_{[2]}(\mathsf{leq}(x, y), y) \to x \qquad\qquad \}.
\end{aligned}
$$

Figure 7.1: Partial inverse CTRS of R_{31} with respect to $\{\,(\mathsf{n}, [\,])\,\}$.

generated by $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(\cdots))$ from non-erasing constructor TRSs are non-erasing constructor TRSs. Hence the transformation $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(\cdots))$ is again applicable to inverse TRS of non-erasing constructor TRSs.

Consider the following convergent non-erasing constructor TRS R_1 over the signature $\{\mathsf{0}, \mathsf{s}, \mathsf{double}\}$:

$$R_1 = \{\, \mathsf{double}(\mathsf{0}) \to \mathsf{0},\ \ \mathsf{double}(\mathsf{s}(x)) \to \mathsf{s}^2(\mathsf{double}(x)) \,\}.$$

Applying $\mathcal{I}nv_{\mathsf{full}}$ and $\mathbb{U}$, we obtain the following inverse TRS of R_1 over the signature $\{\, \mathsf{0}, \mathsf{s}, \mathsf{double}, \mathsf{double}^{\#}, \mathsf{u}_4 \,\}$:

$$\begin{aligned} R_{35} &= \mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R_1)) \\ &= \{\ \ \mathsf{double}^{\#}(\mathsf{0}) \to \mathsf{0},\ \ \mathsf{double}^{\#}(\mathsf{s}^2(x)) \to \mathsf{u}_4(\mathsf{double}^{\#}(x)), \\ &\qquad \mathsf{u}_4(y) \to \mathsf{s}(y),\qquad \mathsf{double}^{\#}(\mathsf{double}(x)) \to x \qquad\qquad \}. \end{aligned}$$

Moreover, we apply $\mathcal{I}nv_{\mathsf{full}}$ and $\mathbb{U}$ to the convergent non-erasing constructor TRS above, denoting $(\mathsf{double}^{\#})^{\#}$ by D, as follows:

$$\begin{aligned} R_{36} &= \mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R_{35})) \\ &= \{\ \ \mathsf{D}(\mathsf{0}) \to \mathsf{0}, \\ &\qquad \mathsf{D}(y) \to \mathsf{u}_5(\mathsf{u}_4^{\#}(y)), \\ &\qquad \mathsf{u}_5(z) \to \mathsf{u}_6(z), \\ &\qquad \mathsf{u}_6(x) \to \mathsf{s}^2(x), \\ &\qquad \mathsf{u}_4^{\#}(\mathsf{s}(y)) \to y, \\ &\qquad \mathsf{D}(x) \to \mathit{double}(x), \\ &\qquad \mathsf{D}(\mathsf{double}^{\#}(x)) \to x \quad \}. \end{aligned}$$

Now we have a question whether the computations of R_1 and R_{36} equivalent or not. The answer is "yes" for those of double and D on the constructor terms over $\{\, \mathsf{0}, \mathsf{s} \,\}$. Because Theorem 5.1.19 guarantees that for all normal forms s and t over $\{\mathsf{0}, \mathsf{s}, \mathsf{double}\}$,

- $\mathsf{double}(s) \xrightarrow{*}_{R_1} t$ iff $\mathsf{double}^{\#}(t) \xrightarrow{*}_{R_{35}} s$, and
- $\mathsf{double}^{\#}(t) \xrightarrow{*}_{R_{35}} s$ iff $\mathsf{D}(s) \xrightarrow{*}_{R_{36}} t$,

and hence for all normal forms s and t over $\{\mathsf{0}, \mathsf{s}, \mathsf{double}\}$,

$$\mathsf{double}(s) \xrightarrow{*}_{R_1} t \text{ iff } \mathsf{D}(s) \xrightarrow{*}_{R_{36}} t.$$

Chapter 8

Conclusion

In this thesis, we have proposed the inverse compilers of PT-TRSs and constructor TRSs. Since the inverse compilers generate EV-TRSs as inverse programs, we have shown that the narrowing which is restricted to substitute a fresh variable for each extra variable, can simulate a rewrite sequence of an EV-TRSs if the EV-TRS is right-linear or the rewrite sequence is EV-safe. As a termination proof technique of inverse computation using generated inverse EV-TRSs, we have also proposed two termination proof methods of narrowing (starting from ground terms), one of which is based on the notion of active chains, and the other of which is based on the dependency pair method. To improve efficiency of inverse computation, we have shown that basic narrowing is sufficient to simulate rewrite sequences of right-linear EV-TRSs. We have also shown that for linear constructor EV-TRSs, all normal forms of a given terminating term can be obtained by innermost narrowing. We have approached to partial inverse computation, by extending the inverse compiler of constructor TRSs. We have shown tree examples as applications of the generated inverse EV-TRSs, the first of which was a solution of our motivation problem.

We have analyzed the relationships between the syntactic properties of the input TRSs and the generated EV-TRSs. We have also given a condition of the input TRSs to generate the innermost-terminating inverse TRSs, that is, constructor-increasing. These works make it possible and easier to use several results in term rewriting for our inverse EV-TRSs.

Our transformational approach to inverse computation — the inverse compilers proposed in this thesis — is the first one in term rewriting, whose input and output works on the same interpretation (narrowing starting form ground terms). Through this thesis, we have shown the necessity of studies on EV-TRSs. It is

because the generated inverse EV-TRSs, such as the inverse of multiplication, are practical.

The results on computation by inverse EV-TRSs, which have shown in this thesis, are also general results on narrowing of EV-TRSs as follows;

- Every ground rewrite sequence which is of right-linear EV-TRSs or is EV-safe, can be simulated by narrowing.
- The termination proof theorems, Theorem 5.2.7 and 5.2.5, are usable
  - to prove $\mathrm{GSN}^{\rightsquigarrow_R}$, and
  - to prove that $\mathrm{SN}_t^{\rightsquigarrow_R}$ for all linear terms t,

  for a right-linear EV-TRS R, and
  - to prove $\mathrm{GSN}^{\rightsquigarrow_R}$, and
  - to prove $\mathrm{SN}^{\rightsquigarrow_R}$

  for a constructor EV-TRS R.
- For linear constructor EV-TRSs, all normal forms of a given linear term which is innermost terminating with respect to innermost narrowing, can be obtained by innermost narrowing.
- For right-linear EV-TRSs, basic narrowing is sufficient to simulate ground rewrite sequences.
- For linear EV-TRSs, narrowing and basic narrowing, both of which are starting from ground terms, are equivalent.

We would like to generate inverse programs which are incorporable with other programs. To use inverse EV-TRSs with another (C)TRS, the completeness of $\mathbb{U}$ is necessary. As a future work, we will first prove Conjecture 3.6.25 and 3.6.26 on completeness of the transformation $\mathbb{U}$. Then, to transform a partial-inverse CTRS into an equivalent EV-TRS, we will investigate a condition for completeness of $\mathbb{U}$.

We are interested in a transformation which generates a rule $f(x) \to g(h(x))$ from $f(x) \to g(y) \Leftarrow h(x) \twoheadrightarrow y$. Consider the following TRSs in Example 3.1.2:

$$R_1 = \{\, \mathsf{double}(0) \to 0,\ \mathsf{double}(\mathsf{s}(x)) \to \mathsf{s}^2(\mathsf{double}(x)) \,\},$$

$$R_2 = \{\ \mathsf{half}(0) \to 0,\ \ \mathsf{half}(\mathsf{s}^2(x)) \to \mathsf{s}(\mathsf{half}(x))\ \}.$$

The transformation $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}())$ in this thesis generates the following EV-TRS from R_1 (see Section 7.3):

$$\begin{aligned} R_{35} &= \mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(R_1)) \\ &= \{\ \ \mathsf{double}^{\#}(0) \to 0,\ \ \mathsf{double}^{\#}(\mathsf{s}^2(x)) \to \mathsf{u}_4(\mathsf{double}^{\#}(x)), \\ &\qquad \mathsf{u}_4(y) \to \mathsf{s}(y),\qquad \mathsf{double}^{\#}(\mathsf{double}(x)) \to x \qquad\qquad \}. \end{aligned}$$

Although both of half and $\mathsf{double}^{\#}$ compute half of natural numbers, their rewrite rules are different even if ignoring the inverse-property rule $\mathsf{double}^{\#}(\mathsf{double}(x)) \to x$. The above example motivate us to improve the transformation $\mathbb{U}(\mathcal{I}nv_{\mathsf{full}}(\cdots))$.

We would like to study more on partial inverse CTRSs. Such a work will help study on full inverse programs. For example, we are interested in extending the transformation $\mathcal{I}nv_{\mathsf{par}}$ to a transformation that generates an inverse CTRS which computes more efficiently than an equivalent CTRS generated by the previous transformation. Consider the sixth rule in the following partial inverse CTRS:

$$\begin{aligned} &\mathcal{I}nv_{\mathsf{par}}(R_7, \{(\mathsf{mul}, [\,])\}) \\ &\quad = \{\ \ \mathsf{add}^{\#}_{[\,]}(y) \to \mathsf{tp}_2(0, y), \\ &\qquad \mathsf{add}^{\#}_{[\,]}(\mathsf{s}(z)) \to \mathsf{tp}_2(\mathsf{s}(x), y) \Leftarrow \mathsf{add}^{\#}_{[\,]}(z) \twoheadrightarrow \mathsf{tp}_2(x, y), \\ &\qquad \mathsf{add}^{\#}_{[\,]}(\mathsf{add}(x, y)) \to \mathsf{tp}_2(x, y), \\ &\qquad \mathsf{mul}^{\#}_{[\,]}(0) \to \mathsf{tp}_2(0, y), \\ &\qquad \mathsf{mul}^{\#}_{[\,]}(0) \to \mathsf{tp}_2(x, 0), \\ &\qquad \mathsf{mul}^{\#}_{[\,]}(\mathsf{s}(z)) \to \mathsf{tp}_2(\mathsf{s}(x), \mathsf{s}(y)) \\ &\qquad\qquad \Leftarrow \mathsf{add}^{\#}_{[\,]}(z) \twoheadrightarrow \mathsf{tp}_2(w, y) \wedge \mathsf{mul}^{\#}_{[\,]}(w) \twoheadrightarrow \mathsf{tp}_2(x, \mathsf{s}(y)), \\ &\qquad \mathsf{mul}^{\#}_{[\,]}(\mathsf{mul}(x, y)) \to \mathsf{tp}_2(x, y) \qquad\qquad \}. \end{aligned}$$

After the evaluation of the first condition $\mathsf{add}^{\#}_{[\,]}(z) \twoheadrightarrow \mathsf{tp}_2(w, y)$, we know value of the variable y. Then, we know value of y before the evaluation of the second condition $\mathsf{mul}^{\#}_{[\,]}(w) \twoheadrightarrow \mathsf{tp}_2(x, \mathsf{s}(y))$. This fact means that the value of y can be used for evaluation of the second condition, replacing the second condition with a condition $\mathsf{mul}_{[2]}(w, y) \twoheadrightarrow \mathsf{tp}_1(x)$. We will also study the efficiency of a CTRS obtained by such an extended transformation, that is, that the extension is actually effective (or not).

We have shown that inverse TRSs obtained from NECI constructor TRSs are innermost terminating with respect to rewrite relation. Since a lot of practical TRSs (such as multiplication) have an erasing rule, we will lift up this result to narrowing, that is, that inverse EV-TRSs obtained from constructor-increasing

constructor TRS is innermost terminating with respect to narrowing starting from ground terms. Unfortunately, we already have a counterexample which is the inverse EV-TRS R_{11} (see Section 5.2.4). Hence we guess a conjecture that $f^{\#}(t)$ (where t is a ground constructor term) is innermost terminating with respect to narrowing. As a future work, we will prove this conjecture.

The transformations $\mathcal{I}nv_{\mathsf{PT}}$ and $\mathcal{I}nv_{\mathsf{full}}$ are applicable to pure treeless EV-TRSs and constructor EV-TRSs, respectively. We are interested in the type of a generated CTRS (type 3, or type 4), the syntactic properties of it, and whether it is an inverse program of the input (or not).

We will also study a termination proof method of narrowing, a strategy of narrowing derivation. The termination proof methods in this thesis are weak than those of rewrite relation of TRSs. Since there are part of the dependency pair method [4, 5] which this thesis did not deal with, we would like to improve the method in this thesis, based on the dependency pair method. In this thesis, we have shown that innermost narrowing is useful for the computation of linear constructor TRSs. However, most of practical programs (such as multiplication) are not right-linear but left-linear, that is, their inverse program is not linear (but right-linear). Hence, we would also like to investigate a condition that innermost narrowing is sufficient for right-linear EV-TRSs.

In Section 7.1, we have shown the transformation of a CTRS into an equivalent constructor CTRS. In future, we will formalize such a transformation.

Bibliography

[1] S. Abramov and R. Glück. The universal resolving algorithm: Inverse computation in a functional language. In R. C. Backhouse and J. N. Oliveira, editors, *Proceedings of Mathematics of Program Construction*, volume 1837 of *Lecture Notes in Computer Science*, pages 187–212, Springer-Verlag, 2000.

[2] S. Abramov and R. Glück. Principles of inverse computation and the universal resolving algorithm. In T. Æ. Mogensen, D. A. Schmidt, and I. H. Sudborough, editors, *The Essence of Computation*, volume 2566 of *Lecture Notes in Computer Science*, pages 269–295, Springer-Verlag, 2002.

[3] S. Abramov and R. Glück. The universal resolving algorithm and its correctness: Inverse computation in a functional language. *Science of Computer Programming*, 43(2–3):193–229, 2002.

[4] T. Arts. *Automatically Proving Termination and Innermost Normalization of Term Rewriting Systems.* PhD thesis, Universiteit Utrecht, 1997.

[5] T. Arts and J. Giesl. Termination of term rewriting using dependency pairs. *Theoretical Computer Science*, 236:133–178, 2000.

[6] L. Augustsson. Compiling pattern-matching. In J.-P. Jouannaud, editor, *Proceedings of Conference on Functional Programming Languages and Computer Architecture (FPCA)*, volume 201 of *Lecture Notes in Computer Science*, pages 368–381, Springer, September 1985.

[7] F. Baader and T. Nipkow. *Term Rewriting and All That.* Cambridge University Press, 1998.

[8] W.-N. Chin. Safe fusion of functional expressions. In *Proceedings of the 1992 ACM Conference on LISP and Functional Programming*, pages 11–20, ACM Press, New York, 1992.

[9] J. Christian. Some termination criteria for narrowing and E-narrowing. In D. Kapur, editor, *Proceedings of the 11th International Conference on Automated Deduction (CADE-11)*, volume 607 of *Lecture Notes in Artificial Intelligence*, pages 582–588, 1992.

[10] N. Cutland. *Computability, an Introduction to Recursive Function Theory.* Cambridge University Press, New York, 1980.

[11] N. Dershowitz. Termination of linear rewriting systems. In S. Even and O. Kariv, editors, *Proceedings of the 8th International Colloquium on Automata, Languages and Programming (ICALP'81)*, volume 115 of *Lecture Notes in Computer Science*, pages 448–458, Springer, 1981. Preliminary version.

[12] N. Dershowitz and C. Kirchner. Inversion strategies. In *Third Workshop on Rule-Based Constraint Reasoning and Programming (RCoRP'01)*, Paphos, Cyprus, December 2001.

[13] N. Dershowitz and S. Mitra. Function inversion. In R. J. G. B. de Queiroz and M. Finger, editors, *Proceedings of the 5th Workshop of Logic, Language, Information and Computation*, pages 55–59, Sao Paulo, Brazil, July 1999.

[14] N. Dershowitz and S. Mitra. Jeopardy. In P. Narendran and M. Rusinowitch, editors, *Proceedings of the 10th International Conference on Rewriting Techniques and Applications (RTA'99)*, volume 1631 of *Lecture Notes in Computer Science*, pages 16–29, Trento, Italy, July 1999.

[15] H. Ganzinger. Order-sorted completion: the many-sorted way. *Theoretical Computer Science*, 89:3–32, 1991.

[16] J. Giesl, R. Thiemann, P. Schneider-Kamp, and S. Falke. AProVE: A system for proving termination. In A. Rubio, editor, *Proceedings of the 6th International Workshop on Termination (WST'03)*, pages 68–70, Valencia, Spain, June 2003.

[17] J. Giesl, R. Thiemann, P. Schneider-Kamp, and S. Falke. Improving dependency pairs. In *Proceedings of the 10th International Conference on Logic for Programming, Artificial Intelligence and Reasoning (LPAR'03)*, volume 2850 of *Lecture Notes in Artificial Intelligence*, pages 167–182, Almaty, Kazakhstan, 2003. Extended version available as Technical Report AIB-2003-04, RWTH Aachen, Germany.

[18] J. Giesl, R. Thiemann, P. Schneider-Kamp, and S. Falke. The termination prover AProVE. In *Proceedings of the 4th International Workshop on the Implementation of Logics (WIL'03)*, pages 46–54, Almaty, Kazakhstan, 2003.

[19] R. Glück and M. Kawabe. A program inverter for a functional language with equality and constructors. In A. Ohori, editor, *Programming Languages and Systems, First Asian Symposium (APLAS 2003)*, volume 2895 of *Lecture Notes in Computer Science*, pages 246–264, Springer, 2003.

[20] R. Glück, Y. Kawada, and T. Hashimoto. Transforming interpreters into inverse interpreters by partial evaluation. In *Proceedings of the 2003 ACM SIGPLAN Workshop on Partial Evaluation and Semantics-based Program Manipulation (PEPM'03)*, pages 10–19, ACM, USA, 2003.

[21] R. Glück and M. Leuschel. Abstraction-based partial deduction for solving inverse problems — a transformational approach to software verification. In D. Bjørner, M. Broy, and A. V. Zamulin, editors, *Proceedings of Perspectives of System Informatics, Third International Andrei Ershov Memorial Conference (PSI'99)*, volume 1755 of *Lecture Notes in Computer Science*, pages 93–100, Springer-Verlag, Russia, 2000.

[22] M. Hanus. On extra variables in (equational) logic programming. In *Proceedings of the 12th International Conference on Logic Programming (ICLP'95)*, pages 665–679, MIT Press, Cambridge, 1995.

[23] P. G. Harrison. Function inversion. In D. Bjørner, A. P. Ershov, and N. D. Jones, editors, *Proceedings of the IFIP TC2 Workshop on Partial Evaluation and Mixed Computation*, pages 153–166, North-Holland, 1988.

[24] P. G. Harrison and H. Khoshnevisan. On the synthesis of function inversion. *Acta Informatica*, 29(3):211–239, 1992.

[25] N. Hirokawa and A. Middeldorp. Tsukuba termination tool. In R. Nieuwenhuis, editor, *Proceedings of the 14th International Conference on Rewriting Techniques and Applications (RTA'03)*, volume 2706 of *Lecture Notes in Computer Science*, pages 311–320, Valencia, Spain, Springer-Verlag, 2003.

[26] G. Huet and J.-J. Lèvy. Call-by-need computations in non-ambiguous linear term rewriting systems. Technical Report 359, INRIA, 1979.

[27] G. Huet and J.-J. Lèvy. Computations in orthogonal rewriting systems, I and II. In J.-L. Lassez and G. Plotkin, editors, *Computational Logic: Essays in Honor of Alan Robinson*, chapter 11-12, pages 395–443, MIT Press, Cambridge, 1991.

[28] J.-M. Hullot. Canonical forms and unification. In *Proceedings of the 5th International Conference on Automated Deduction*, volume 87 of *Lecture Notes in Computer Science*, pages 318–334, 1980.

[29] H. Khoshnevisan and K. M. Sephton. InvX: An automatic function inverter. In N. Dershowitz, editor, *Proceedings of the 3rd International Conference on Rewriting Techniques and Applications (RTA'89)*, volume 355 of *Lecture Notes in Computer Science*, pages 564–568, Springer-Verlag, USA, 1989.

[30] J. W. Klop. Term rewriting systems. In S. Abramsky, D. M. Gabbay, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 2, pages 2–116, Oxford University Press, New York, 1992.

[31] J. W. Klop and A. Middeldorp. Sequentiality in orthogonal term rewriting systems. *Journal of Symbolic Computation*, 12:161–195, 1991.

[32] K. Kusakari. *Termination, AC-Termination and Dependency Pairs of Term Rewriting Systems.* PhD thesis, Japan Advanced Institute of Science and Technology (JAIST), March 2000.

[33] K. Kusakari, M. Nakamura, and Y. Toyama. Argument filtering transformation. In *Proceedings of International Conference on Principles and Practice of Declarative Programming (PPDP'99)*, volume 1702 of *Lecture Notes in Computer Science*, pages 47–61, 1999.

[34] M. Marchiori. Unravelings and ultra-properties. In *Proceedings of the 5th International Conference on Algebraic and Logic Programming*, volume 1139 of *Lecture Notes in Computer Science*, pages 107–121, Springer-Verlag, 1996.

[35] J. McCarthy. The inversion of functions defined by Turing machines. In C. E. Shannon and J. McCarthy, editors, *Automata Studies*, pages 177–181, Princeton University Press, 1956.

[36] A. Middeldorp and E. Hamoen. Completeness results for basic narrowing. *Applicable Algebra in Engineering, Communication and Computing*, 5:213–253, 1994.

[37] A. Middeldorp, T. Suzuki, and M. Hamada. Complete selection functions for a lazy conditional narrowing calculus. *Journal of Functional and Logic Programming*, 2002(3):(43 pages), 2002.

[38] T. Nagaya, M. Sakai, and Y. Toyama. NVNF-sequentiality of left-linear term rewriting systems. Technical Report 918, RIMS, 1995.

[39] N. Nishida, M. Sakai, and T. Sakabe. Improving efficiency of of computation of right-linear constructor term rewriting systems with extra variables. In *Proceedings of the 20th Conference of Japan Society for Software Science and Technology (JSSST)*, pages 5B–3 (5 pages), September 2003 (in Japanese).

[40] N. Nishida, M. Sakai, and T. Sakabe. Normal-form computation by left-most innermost narrowing on right-linear overlay term rewriting systems with extra variables. In *LA Symposium 2003 Summer*, pages 24-1–24-6 (6 pages), Mie, July 2003 (in Japanese).

[41] E. Ohlebusch. Termination of logic programs: Transformational methods revisited. *Applicable Algebra in Engineering, Communication and Computing*, 12(1–2):73–116, 2001.

[42] E. Ohlebusch. *Advanced Topics in Term Rewriting.* Springer-Verlag, 2002.

[43] M. Oyamaguchi. NV-sequentiality: a decidable condition for call-by-need computations in term rewriting systems. *SIAM Journal on Computing*, 22(1):114–135, 1993.

[44] A. Romanenko. The generation of inverse functions in Refal. In D. Bjørner, A. P. Ershov, and N. D. Jones, editors, *Proceedings of the IFIP TC2 Workshop on Partial Evaluation and Mixed Computation*, pages 427–444, North-Holland, 1988.

[45] A. Romanenko. Inversion and metacomputation. In *Proceedings of the Symposium on Partial Evaluation and Semantics-Based Program Manipulation (PEPM'91)*, volume 26 of *SIGPLAN Notices*, number 9, pages 12–22, ACM Press, September 1991.

[46] M. Sakai. Innermost terminating right-linear overlay term rewriting systems are terminating, October 2003. Research note available from `http://www.sakabe.nuie.nagoya-u.ac.jp/~sakai/papers/`.

[47] M. Sakai, K. Okamoto, and T. Sakabe. Innermost reductions find all normal forms on right-linear terminating overlay TRSs. In B. Gramlich and S. Lucas, editors, *Proceedings of the 3rd International Workshop on Reduction Strategies in Rewriting and Programming (WRS'03)*, pages 79–88, Valencia, Spain, June 2003.

[48] J. P. Secher and M. H. Sørensen. From checking to inference via driving and dag grammars. In *Proceedings of the 2002 ACM SIGPLAN Workshop on Partial Evaluation and Semantics-Based Program Manipulation*, volume 37 of *SIGPLAN Notices*, number 3, pages 41–51, Portland, Oregon, USA, ACM, March 2002.

[49] P. Wadler. Deforestation: Transforming programs to eliminate trees. *Theoretical Computer Science*, 73(2):231–248, 1990.

Publications

[1] N. Nishida, M. Sakai, and T. Sakabe. Designing unlimited size resource c-libraries freeing users from GC annoyance. Technical Report of IEICE, SS 2000-9, Vol. 100, No. 64, pages 25–32, May 2000 (in Japanese).

[2] N. Nishida, M. Sakai, and T. Sakabe. Generation of a conditional TRS implements the inverses of pure treeless functions. Technical Report of IEICE, COMP 2001-14, Vol. 101, No. 133, pages 9–16, June 2001 (in Japanese).

[3] N. Nishida, M. Sakai, and T. Sakabe. Generation of a TRS implementing the inverses of pure treeless functions. In *Proceedings of the 18th Conference of Japan Society for Software Science and Technology (JSSST)*, 6C–3 (5 pages), September 2001 (in Japanese).

[4] N. Nishida, M. Sakai, and T. Sakabe. Generation of inverse term rewriting systems for pure treeless functions. In Y. Toyama, editor, *Proceedings of the International Workshop on Rewriting in Proof and Computation (RPC'01)*, pages 188–198, Sendai, Japan, October 2001.

[5] N. Nishida, M. Sakai, and T. Sakabe. Generation of a TRS implementing the inverses of the functions with specified arguments fixed. Technical Report of IEICE, COMP 2001-67, Vol. 101, No. 488, pages 33–40, December 2001.

[6] N. Nishida, M. Sakai, and T. Sakabe. Generation of a TRS implementing the inverses of pure treeless functions. *Computer Software*, Vol. 19, No. 1, pages 29–33, Japan Society for Software Science and Technology, January 2002 (in Japanese).

[7] N. Nishida, M. Sakai, and T. Sakabe. A computation model of term rewriting systems with extra variables. In *Proceedings of the 19th Conference*

of Japan Society for Software Science and Technology (JSSST), 6F–3 (5 pages), September 2002 (in Japanese).

[8] N. Nishida, M. Sakai, and T. Sakabe. A narrowing-based reduction of term rewriting systems with extra variables. In *Record of 2002 Tokai-section Joint Conference of the Eight Institutes of Electrical and Related Engineers*, page 292, September 2002 (in Japanese).

[9] N. Nishida, M. Sakai, and T. Sakabe. Narrowing-based effective rewriting and its termination for term rewriting systems with extra variables. Technical Report of IEICE, COMP 2003-68, Vol. 102, No. 593, pages 45–52, January 2003 (in Japanese).

[10] N. Nishida, M. Sakai, and T. Sakabe. Narrowing-based effective rewriting and its termination for term rewriting systems with extra variables. In *LA Symposium 2003 Winter*, pp. 43-1–43-1 (8 pages), Kyoto, February 2003 (in Japanese).

[11] N. Nishida, M. Sakai, and T. Sakabe. Narrowing-based effective rewriting and its termination for term rewriting systems with extra variables. RIMS Kokyuroku 1325, Kyoto University, pages 238–243, May 2003 (in Japanese).

[12] N. Nishida, M. Sakai, and T. Sakabe. Narrowing-based simulation of term rewriting systems with extra variables and its termination proof. In G. Vidal, editor, *Proceedings of the 12th International Workshop on Functional and (Constraint) Logic Programming (WFLP'03)*, pages 198–211, Valencia, Spain, June 2003.

[13] N. Nishida, M. Sakai, and T. Sakabe. Normal-form computation by leftmost innermost narrowing on right-linear overlay term rewriting systems with extra variables. In *LA Symposium 2003 Summer*, pp. 24-1–24-6 (6 pages), Mie, July 2003 (in Japanese).

[14] N. Nishida, M. Sakai, and T. Sakabe. Computation model of term rewriting systems with extra variables. *Computer Software*, Vol. 20, No. 5, pages 85–89, Japan Society for Software Science and Technology, September 2003 (in Japanese).

[15] N. Nishida, M. Sakai, and T. Sakabe. Improving efficiency of of computation of right-linear constructor term rewriting systems with extra variables. In *Proceedings of the 20th Conference of Japan Society for Software Science and Technology (JSSST)*, 5B–3 (5 pages), September 2003 (in Japanese).

[16] N. Nishida, M. Sakai, and T. Sakabe. Narrowing-based simulation of term rewriting systems with extra variables and its termination proof. In G. Vidal, editor, *Functional and Constraint Logic Programming (WFLP'03)*, *Electronic Notes in Theoretical Computer Science*, Vol. 86, Issue 3, 18 pages. Elsevier, November 2003.

[17] N. Nishida, M. Sakai, and T. Sakabe. Improving efficiency of of computation of right-linear constructor term rewriting systems with extra variables. *Computer Software*, Japan Society for Software Science and Technology, accepted (in Japanese).

Index

Printed on January 29, 2004.