Modularity in Many-sorted Term Rewriting Systems

Jaco van de Pol

Januar 1993

Abstract

Many-sorted term rewriting systems (MTRS) are an extension of the formalism of term rewriting systems (TRS). The *direct sum* of TRS's is generalized to the direct sum of MTRS's. Some equivalence between "taking the direct sum" and "eliminating the sorts" is established: Component closed reduction properties which are resistant against disjoint union (*modularity*), are also resistant against sort elimination (*persistence*). The reverse has been proved earlier.

1 Introduction

1.1 Related Work

Term rewriting systems (TRS's) have been developed in the forties as a model for computation. They were used in those days for the study on the λ-calculus (Church). Another issue formed the study on completion procedures on equational theories, started by Knuth and Bendix. In this setting, the main goal of term rewriting systems is to give an implementation of a theory of (undirected) equations. The completion procedure delivers a TRS, in which the rules have an unique direction. The applications of TRS's in computer science can be found in the analysis and implementation of functional programming languages and in the study on and implementation of algebraic specifications of abstract data types. To meet this applications several extensions are proposed: Graph and infinite rewriting, conditional term rewriting, term rewriting with rule priorities and order-sorted term rewriting [GJM85]. The extension to many-sorted term rewriting (which is a special case of order-sorted rewriting) is the main subject of this thesis.

More recently is the research on the modularity of properties of term rewriting systems. A property is called *modular* if it is preserved by the disjoint-union-operator, also called the *direct sum* on TRS's [Toy87b, Toy87a, Rus87, Mid90]. This notion is extended to many-sorted term rewriting systems.

This thesis can be seen as a continuation of the research of H. Zantema on the persistence of properties of many-sorted TRS's. A property is called *persistent* if it is preserved by the sort-elimination-operator on many-sorted TRS's. Zantema proved that persistent properties are modular, for some class of properties (reduction properties or component closed properties). In fact we can partially answer one of the "Concluding remarks" of his article [Zan91, 628]: "It is unknown whether there are modular reduction properties that are not persistent": For modular properties on many-sorted TRS's the answer is: No, there are none.

1.2 Overview and Main Results

You are reading my master thesis, which is the closure of my study in Computer Science on the "Rijksuniversiteit Utrecht". I am grateful to Hans Zantema for being present when I needed him. It sometimes took him 7 hours a day. In the rest of this section an overview is given of the contents of this thesis.

Chapter 2 gives an introduction to classical term rewriting systems. Familiar readers can jump it over, but perhaps it is better to have a look at section 2.4. This section defines the notion of "modularity" and summarizes some known results.

Chapter 3 describes the extension to many-sorted term rewriting systems.

This has been done before, only the notations are different. As far as we know, the definition of the disjoint union of many-sorted term rewriting systems is new. In section 3.5 the notion of "persistence" is defined. The definition of the problem of the thesis can be found in section 3.6.

In Chapter 4 the main result is proven: The "sort elimination"-operator on a many-sorted term rewriting system can be simulated by taking the disjoint union of modified copies of the many-sorted term rewriting system. This result can be used to prove the persistence of some class of modular properties for many-sorted term rewriting systems. This proof is given in section 4.3.

2 Term Rewriting Systems

2.1 Signatures and Terms

Terms are generated by a set of function symbols and a set of variables. The set of function symbols is typically denoted by $\mathcal{F}$. The set of variables is denoted by $\mathcal{V}$. We require the sets of function symbols and variables to be disjoint, so

$$\mathcal{V} \cap \mathcal{F} = \varnothing$$

A function symbol is 'syntactically applied' to its arguments. For every function symbol the number of arguments is fixed and it is called the *arity*. The arity is given by the function:

$$\mathsf{arity} : \mathcal{F} \to \mathbb{N},$$

in which $\mathbb{N}$ is the set of natural numbers including 0. An (unsorted) *signature* Σ is a tuple $(\mathcal{F}, \mathcal{V}, \mathsf{arity})$.

Now we can define the *terms* over $\Sigma = (\mathcal{F}, \mathcal{V}, \mathsf{arity})$ inductively:

$$\begin{cases} \mathcal{V} \subseteq \mathcal{T}(\Sigma) \\ f(t_1, \ldots, t_n) \in \mathcal{T}(\Sigma) \quad \text{iff} \quad (1) \quad f \in \mathcal{F}, \\ \qquad\qquad\qquad\qquad\quad (2) \quad \mathsf{arity}(f) = n \text{ and} \\ \qquad\qquad\qquad\qquad\quad (3) \quad t_1, \ldots, t_n \in \mathcal{T}(\Sigma) \end{cases}$$

If $n = 0$ we omit the $()$. We say that f is 'syntactically applied' to the arguments $t_1, \ldots, t_n$. If the signature Σ is clear, we use the abbreviation $\mathcal{T}$ for $\mathcal{T}(\Sigma)$.

The set of terms is defined inductively. Proofs on terms can be inductive on the structure of the term. For this reason, it is practical to have a measure for terms. This is given by the following definition:

$$\begin{aligned} \mathsf{depth}(x) &= 1 \\ \mathsf{depth}(f(t_1, \ldots, t_n)) &= 1 + \max_{1 \le i \le n} \mathsf{depth}(t_i) \end{aligned}$$

In this definition the maximum over an empty set is zero.

The *root* of a term is its outermost symbol:

$$\mathsf{root}(x) = x \quad \text{if } x \in \mathcal{V}$$

$$\mathsf{root}(f(t_1, \ldots, t_n)) = f \quad \text{if } f \in \mathcal{F}.$$

Let Σ be a signature, with variables $\mathcal{V}$. Then the set $\mathcal{T}$ consists of all terms (over Σ), as described above. A substitution is a mapping

$$\sigma : \mathcal{V} \to \mathcal{T}.$$

Instead of $\sigma(x)$ we will write: x^σ. Substitutions can be extended to general terms by defining:

$$(f(t_1, \ldots, t_n))^\sigma = f(t_1^\sigma, \ldots, t_n^\sigma)$$

If a variable occurs twice in some term, then after the application of a substitution, the variable is replaced twice by the same term, because the substitution is a function on variables.

Proposition 2.1 *Let $l \in \mathcal{T} \setminus \mathcal{V}$ and let σ be a substitution. Then* $\mathsf{root}(l) = \mathsf{root}(l^\sigma)$.

Proof: Because $l \notin \mathcal{V}$, $l = f(t_1, \ldots, t_n)$ for some $f \in \mathcal{F}$ and $n = \mathsf{arity}(f)$. Then $l^\sigma = f(t_1^\sigma, \ldots, t_n^\sigma)$. So $\mathsf{root}(l) = \mathsf{root}(l^\sigma) = f$.
□

2.2 Term Rewriting Systems

A term rewriting system T is a signature $\mathcal{F}$ and a finite set $\mathcal{R}$ of pairs of terms (l, r) for which:

- $l \notin \mathcal{V}$
- $\mathsf{Var}(r) \subseteq \mathsf{Var}(l)$

Such pairs are called rewrite rules and written as $l \to r$.

The rewrite relation $\to_\mathcal{R}$ is defined by: $s \to_\mathcal{R} t$ for $s, t \in \mathcal{T}(\mathcal{F})$ if and only if this can be proved using the following proof rules:

- $l^\sigma \to_\mathcal{R} r^\sigma$ for every $\sigma : \mathcal{V} \to \mathcal{T}(\mathcal{F})$ and $l \to r \in \mathcal{R}$
- $f(t_1, .., t_j, .., t_n) \to_\mathcal{R} f(t_1, .., t'_j, .., t_n)$ if $t_j \to_\mathcal{R} t'_j$.

The first rule is an axiom scheme, the second rule is a derivation rule scheme. The first rule closes the rewrite relation under substitution, the second under context. A rewrite step $s \to_\mathcal{R} t$ is called a reduction step. When we have to prove some property of a reduction step, this can be done with induction on the structure of the proof that it is a reduction step.

Here l^σ is called the *redex*, a reduction step of the form $l^\sigma \to_\mathcal{R} r^\sigma$ is an *elementary* reduction step (or a reduction step on top level). We say that s reduces to t if $s \to_\mathcal{R}^+ t$, where $\to^+$ is the transitive closure of $\to$. If s doesn't reduce to any term, we say s is a *normal form*, or s is in normal form. The set of normal forms of T is written as $\mathsf{NF}(T)$. A TRS T only depends on the signature $\mathcal{F}$ and the set of rules $\mathcal{R}$. This can be denoted by $T = (\mathcal{F}, \mathcal{R})$.

2.3 Properties of Term Rewriting Systems

We are going to discuss features of properties of term rewriting systems. Therefore it helps our intuition to define some concrete properties. There are some well known properties of term rewriting systems, which are particularly useful for applications. There are two kinds of properties: properties on the reduction relation and syntactic properties. The first kind can be discussed in the general setting of abstract reduction systems. The second kind depends on the form of the rules of a term rewriting system.

2.3.1 Abstract Reduction Systems

An *abstract reduction system* (ARS) $\mathcal{A}$ is a set A with a binary relation R. Term rewriting systems are a special kind of ARS, with the terms $\mathcal{T}(\mathcal{F})$ as the underlying set A and the reduction relation $\to_{\mathcal{R}}$ as binary relation R on A.

The relation R is called *reduction relation* and written as $\to$, with transitive closure $\to^+$ and transitive and reflexive closure $\to^*$. The transitive, reflexive and symmetric closure is denoted by $=_R$ and this is called the *convertibility* relation. Syntactic equality between terms is denoted by $\equiv$. An element $a \in A$ is called a normal form ($\mathsf{NF}(a)$) if there is no $x \in A$ with $a \to x$. An element $a \in A$ has a normal form if there is a $b \in A$ with $\mathsf{NF}(b)$ and $a \to^* b$. Two elements $a, b \in A$ are called *joinable* ($\downarrow$) if they have a common reduct: $a \downarrow b$ if there is a $c \in A$ with $a \to^* c$ and $b \to^* c$.

Properties on abstract reduction systems are in fact properties of relations. This sort of properties is often called *reduction properties*. Some important reduction properties are shortly described below.

strongly normalizing An ARS $\mathcal{A}$ is called strongly normalizing (SN) if there are is no infinite sequence $a_0, a_1, \ldots$ with $a_i \in \mathcal{A}$ and $a_i \to a_{i+1}$ for all $i \in \mathbb{N}$. So every reduction ends in a normal form. For term rewriting systems this means: there is no infinite reduction. Another terminology is: terminating or Noetherian.

weakly normalizing An ARS $\mathcal{A}$ is called weakly normalizing (WN) if every element $a \in \mathcal{A}$ has a normal form.

confluent An ARS $\mathcal{A}$ is called confluent or Church-Rosser (CR) if every two reducts of any term are joinable: If $a \to^* b$ and $a \to^* c$ then $b \downarrow c$.

weakly confluent An ARS $\mathcal{A}$ is called weakly confluent (WCR) if the following holds: If $a \to b$ and $a \to c$ then $b \downarrow c$.

unique normal forms An ARS $\mathcal{A}$ has unique normal forms (UN) if any two convertible normal forms are identical. Or stated otherwise: Every equivalence class under "=" has at most one normal form.

UN with respect to reduction An ARS $\mathcal{A}$ has unique normal forms with respect to reduction ($\text{UN}^{\rightarrow}$) if no element of A reduces to different normal forms.

normal form property An ARS $\mathcal{A}$ has the normal form property (NF) if every element convertible to a normal form, reduces to that normal form.

For a discussion on these properties and a proof of the relations between them we refer to [Mid90, §1.1]. Here we only state the relationship between these properties, in the same form as A. Middeldorp in the just cited book. (see figure 1)

Figure 1: The relationship between some reduction properties.

All these properties say something about the whole reduction relation. But if we analyze them, we see that it is enough to verify them for every equivalence class generated by the reduction relation. This feature on properties is called *component closed*. The formal definition is given below and can be found in [Zan92, 67]. Let $\mathcal{A} = (A, R)$ be some ARS. Then with A_i we denote the family of equivalence classes under the relation $=_R$. With R_i we denote the relation R restricted to the equivalence class A_i. We define $\mathcal{A}_i = (A_i, R_i)$.

A property $\mathcal{P}$ on abstract reduction systems is component closed, if for all ARS $\mathcal{A}$

$$\mathcal{P}(\mathcal{A}) \iff (\forall i : \mathcal{P}(\mathcal{A}_i)).$$

All of the properties of figure 1 are component closed, due to the following proposition:

Proposition 2.2 *Let $\mathcal{A} = (A, R)$ be an ARS. Let $t, t' \in \mathcal{A}$ and let $[t]$ be the equivalence class of t under $=_R$. Then*

- $t \to t'$ *in* $\mathcal{A} \iff t \to t'$ *in* $[t]$.
- $t \to^* t'$ *in* $\mathcal{A} \iff t \to^* t'$ *in* $[t]$.
- $t =_R t'$ *in* $\mathcal{A} \iff t =_R t'$ *in* $[t]$.
- $\mathsf{NF}(t)$ *in* $\mathcal{A} \iff \mathsf{NF}(t)$ *in* $[t]$.

Proof: The first three assertions follow from the fact that by definition, the equivalent classes are closed under the reflexive, symmetric and transitive closure of the reduction relation $\to$.
The last assertion follows immediately from the first.
□

Now it can be verified easily that SN, CR, UN, UN$^{\to}$, NF, WCR and WN are component closed. An example of a property that is not component closed is the (not so interesting) property OON (only one normal form): for all t and t', if $\mathsf{NF}(t)$ and $\mathsf{NF}(t')$ then $t \equiv t'$. The following example shows that OON is not component closed:

Example 1 Let $A = \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$ and let $\mathsf{a} \to \mathsf{b}$. Then the equivalence classes are $A_1 = \{\mathsf{a}, \mathsf{b}\}$ and $A_2 = \{\mathsf{c}\}$ with $\mathsf{a} \to \mathsf{b}$. Both A_1 and A_2 are OON, but A is not, because b and c are different normal forms.

. . . . . .

2.3.2 Syntactic Properties

There are some well known properties on term rewriting systems, that depend on the form of the rules. These are called *syntactic properties* in contrast with reduction properties, that depend on the reduction relation.

A term is called *linear* if every variable occurs at most once in it. A rule is called *left/right linear* if its left/right hand side is linear. A term rewriting system is called *left/right linear* if all rules are left/right linear.

A rule is a *duplicating rule* if there is a variable with more occurrences in the right hand side than in the left hand side. A *collapse-rule* is a rule with a variable as right hand side. An TRS is called duplicate/collapse free if it contains no duplicating/collapse rules.

2.4 The Direct Sum and Modularity

It is good programming practice to divide large programs into smaller modules. This insight has led to the study on the union of TRS's. However, it turns out that the union of two TRS's doesn't inherit important properties of the original TRS's. See [Mid90, 29] for examples. The reason is that the TRS's can share function symbols. Therefore only the union of disjoint

TRS's is regarded. Let $T_1 = (\mathcal{F}_1, \mathcal{R}_1, \mathcal{T})$ and $T_2 = (\mathcal{F}_2, \mathcal{R}_2, \mathcal{T})$ be two unsorted TRS's with $\mathcal{F}_1 = (F_1, \mathcal{V}_1, \mathsf{arity}_1)$ and $\mathcal{F}_2 = (F_2, \mathcal{V}_2, \mathsf{arity}_2)$. Then T_1 and T_2 are disjoint if and only if the underlying sets of function symbols are disjoint: $F_1 \cap F_2 = \varnothing$, so they share no function symbols. Without loss of generality we also require that $\mathcal{V}_1$ and $\mathcal{V}_2$ are disjoint. We can combine arity_1 and arity_2 to a new function arity:

$$\mathsf{arity} : F_1 \cup F_2 \to S^*$$

$$\mathsf{arity}(f) = \begin{cases} \mathsf{arity}_1(f) & \text{if } f \in F_1 \\ \mathsf{arity}_2(f) & \text{if } f \in F_2 \end{cases}$$

This construction is simple, because F_1 and F_2 are disjoint. Therefore we will drop the subscripts of functions like arity, sort and st in the sequel. Now $\mathcal{F}_\oplus = \mathcal{F}_1 \oplus \mathcal{F}_2 = (F_1 \oplus F_2, \mathcal{V}_1 \oplus \mathcal{V}_2, \mathsf{arity})$, where $\oplus$ denotes the union for disjoint sets. The disjoint union of the TRS's $T_1 \oplus T_2$ is defined as the new TRS $(\mathcal{F}_\oplus, \mathcal{R}_\oplus, \mathcal{T}(\mathcal{F}_\oplus))$. In other words: $T_\oplus = (\mathcal{F}_\oplus, \mathcal{R}_\oplus, \mathcal{T})$. This is known in the literature [Toy87b, Rus87] as the *direct sum*. This definition of the direct sum of TRS's can also be found in [Mid90, 31] and originates from [Toy87b, 131].

We can ask about the influence of the disjoint union operator on the properties of the two term rewriting systems. Does taking the direct sum preserve important properties, as confluence, strong normalization etc.? Properties that are resistant against the disjoint union operator are called "modular". This feature states that some property holds for the direct sum of two TRS's if and only if it holds for the original ones. So we define for a property $\mathcal{P}$ on TRS's:

$\mathcal{P}$ is modular iff for all term rewriting systems R_1 and R_2 holds:

$$\mathcal{P}(R_1 \oplus R_2) \iff (\mathcal{P}(R_1) \wedge \mathcal{P}(R_2))$$

This definition is equivalent to the definition in [Mid90, 29].

2.5 Previous Results

The modularity of many properties is known, due to recent research by Toyama, Rusinowitch and Middeldorp. For a detailed overview on these results [Mid90] can be read. In 1987 Toyama presented his famous counter example, showing that strong normalization is not a modular property for term rewriting systems [Toy87a, 141]. It reads as follows: the following TRS's are both terminating.

$R_1 = \{F(0,1,x) \to F(x,x,x)\}$

$$R_2 = \begin{cases} G(x,y) & \to & x \\ G(x,y) & \to & y \end{cases}$$

But $R_1 \oplus R_2$ contains an infinite reduction:

$$\begin{aligned} F(G(0,1),G(0,1),G(0,1)) &\rightarrow F(0,G(0,1),G(0,1)) \\ &\rightarrow F(0,1,G(0,1)) \\ &\rightarrow F(G(0,1),G(0,1),G(0,1)) \\ &\rightarrow \cdots \end{aligned}$$

Yet there are conditions under which modularity of termination holds. Rusinowitch showed in [Rus87] that if term rewriting systems R_1 and R_2 both lack collapsing rules, or if they both lack duplicating rules, then the direct sum $R_1 \oplus R_2$ is terminating if and only if R_1 and R_2 are terminating. Middeldorp showed that the following condition was sufficient to have modularity of termination: One of the term rewriting systems contains neither collapsing rules nor duplicating rules [Mid89].

Furthermore, Toyama proved the modularity of confluence (CR) in [Toy87b]. This proof is presented more accessible in [KMTdV91]. Middeldorp proved in [Mid90] the modularity of weak confluence (WCR) and unique normalization (UN). Furthermore he describes a proof for the modularity of weak normalization (WN), due to Kurihara and Kaji [KK88].

Finally, there are counterexamples for the modularity of the normal form property (NF) and the unique normal form with respect to reduction ($\mathrm{UN}^{\rightarrow}$) [Mid90, p.97, p.101]. However, NF is modular for left-linear TRS's [Mid90, p.100].

3 Many-sorted Term Rewriting

3.1 Many-sorted Signatures and Terms

We can put a *sort* restriction on terms as follows: every function symbol gets a sort (type) and a sequence of sorts, defining which sort it expects from each of its arguments. Variables get a sort too. Let S be a set of sorts and let S^n be the set of n-tuples over S. By S^* we denote the set of finite sequences over S.

As in Chapter 2.2 we define a set of function symbols F with a function $\mathsf{arity} : F \to \mathbb{N}$ and a set of variables $\mathcal{V}$. Now we define functions:

$$\mathsf{st} : F \cup \mathcal{V} \to S$$

$$\mathsf{ar} : F \to S^*,$$

with the restriction that $\mathsf{ar}(f) \in S^n$ for $n = \mathsf{arity}(f)$. To get the i-th element of $\mathsf{ar}(f)$ we use an extra argument rather than a subscript, to achieve better readability. So we write:

$$\mathsf{ar}(f) = <\mathsf{ar}(f,1), \mathsf{ar}(f,2), \ldots, \mathsf{ar}(f,n)>,$$

where $n = \mathsf{arity}(f)$ and $\mathsf{ar}(f,i) \in S$.

An *S-sorted* signature $\mathcal{F}$ is a tuple $(F, \mathcal{V}, S, \mathsf{arity}, \mathsf{st}, \mathsf{ar})$, where the domain and range of st and ar are defined before. Again we will write $f \in \mathcal{F}$ instead of $f \in F$ by convention.

Example 2 Let $S = \{0, 1, 2\}$. Let $\mathcal{V} = \{\mathsf{x}, \mathsf{y}, \mathsf{z}\}$ and $F = \{\mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{f}, \mathsf{g}, \mathsf{h}\}$. Instead of writing $\mathsf{arity}(\mathsf{f}) = 2, \mathsf{ar}(\mathsf{f}) = (0,1)$ and $\mathsf{st}(\mathsf{f}) = 2$ it is common to write: $\mathsf{f} : 0 \times 1 \to 2$. Using this notation, we can define the S-sorted signature $\mathcal{F}$ as:

x:0	a:0	f:$0 \times 1 \to 2$
y:1	b:1	g:$2 \to 0$
z:2	c:2	h:$1 \to 0$

. . . . . .

The *sort* of a term t is defined as the sort of the outermost symbol, so we have:

$$\mathsf{sort}(t) = \mathsf{st}(\mathsf{root}(t)).$$

We want to define a special subset of all terms, consisting of the terms that are *well-sorted*. The *well-sorted arguments* of a term $f(t_1, \ldots, t_n) \in \mathcal{T}(\mathcal{F})$ is a set of indexes:

$$\mathsf{wsa}(f(t_1, \ldots, t_n)) = \{i \in \{1, \ldots, n\} \mid \mathsf{sort}(t_i) = \mathsf{ar}(f, i)\}$$

Let $\mathcal{F}$ be a signature with variables $\mathcal{V}$. The set of well-sorted terms $\mathcal{WT}(\mathcal{F})$ is defined inductively by

$$\begin{cases} \mathcal{V} \subseteq \mathcal{WT}(\mathcal{F}) \\ f(t_1, \ldots, t_n) \in \mathcal{WT}(\mathcal{F}) \quad \text{iff} \quad (1) \quad f \in \mathcal{F}, \\ \qquad\qquad\qquad\qquad\quad (2) \quad \mathsf{arity}(f) = n, \\ \qquad\qquad\qquad\qquad\quad (3) \quad t_1, \ldots, t_n \in \mathcal{WT}(\mathcal{F}) \text{ and} \\ \qquad\qquad\qquad\qquad\quad (4) \quad \mathsf{wsa}(f(t_1, \ldots, t_n)) = \{1, \ldots, n\}. \end{cases}$$

If the signature is clear from the context, then $\mathcal{T}(\mathcal{F})$ and $\mathcal{WT}(\mathcal{F})$ are abbreviated to $\mathcal{T}$ and $\mathcal{WT}$.

In the S-sorted signature, the definition of $\mathcal{T}$ still makes sense. It simply does not make use of S, st and ar. Up to restriction (4) the definitions of $\mathcal{WT}$ and $\mathcal{T}$ coincide, so

$$\mathcal{WT} \subseteq \mathcal{T}.$$

This justifies the intuition that $\mathcal{T}$ is the set of all terms over $\mathcal{F}$, while $\mathcal{WT}$ is the set of well-sorted terms over $\mathcal{F}$. From the view point of many-sorted signatures, $\mathcal{T}$ contains ill-sorted terms. For a detailed discussion of order-sorted algebra we refer to [SS87].

Note also that if there is only one sort, then by definition the set $\mathsf{wsa}(f(t_1, \ldots, t_n)) = \{1, \ldots, n\}$. Therefore:

Proposition 3.1 *If $\#S = 1$ then $\mathcal{WT} = \mathcal{T}$.*

So an unsorted signature can be seen as an S-sorted signature with only one sort. This is called *one-sorted*. An S-sorted signature with $\#S > 1$ is called *many-sorted*.

The set $\mathcal{WT}$ is partitioned by the function sort: For each $\alpha \in S$ we define:

$$\mathcal{WT}^{\alpha} = \{t \in \mathcal{WT} \mid \mathsf{sort}(t) = \alpha\}.$$

3.2 Well-sorted Substitutions

Proposition 3.2 *Let $l \in \mathcal{T} \setminus \mathcal{V}$. Then $\mathsf{sort}(l) = \mathsf{sort}(l^{\sigma})$.*

Proof: From proposition 2.1 we have that $\mathsf{root}(l) = \mathsf{root}(l^{\sigma})$. But then their sorts are equal too, due to the definition of sort.
□

The notion of *well-sorted* is defined for substitutions in such a way that a well-sorted substitution on a well-sorted term returns a well-sorted term: A substitution σ is well-sorted if and only if for all $x \in \mathsf{dom}(\sigma)$:

$$x^{\sigma} \in \mathcal{WT}^{\mathsf{st}(x)}.$$

So x^σ is well-sorted and has the same sort as x. Furthermore we define:

$$\mathsf{Var}(t) = \text{ the set of variables occurring in } t.$$

We often want to restrict the influence of a substitution to the variables of a special term. This can be done by function restriction ($\restriction$). So we get:

$$(\sigma \restriction \mathsf{Var}(l))(x) = \begin{cases} x^\sigma & \text{if } x \in \mathsf{Var}(l) \\ x & \text{otherwise} \end{cases}$$

Proposition 3.3 *Let $l \in \mathcal{WT}^s$ for some sort $s \in S$ and let σ be a substitution. Then $l^\sigma \in \mathcal{WT}^s$ if and only if $\sigma \restriction \mathsf{Var}(l)$ is well-sorted.*

Proof: We have that $\mathsf{sort}(l) = s$ and that l is well-sorted.

$\Leftarrow$ Let σ be well-sorted on $\mathsf{Var}(l)$. We will prove that l^σ is well-sorted and $\mathsf{sort}(l^\sigma) = s$ by induction to the depth of r.

If $l = x \in \mathcal{V}$, then $l^\sigma = x^\sigma \in \mathcal{WT}^s$, because σ is well-sorted.

If $l = c \in \mathcal{F}$, then $l^\sigma = l$ and we are ready.

If $l = f(t_1, \ldots, t_n)$ then $l^\sigma = f(t_1^\sigma, \ldots, t_n^\sigma)$ and $\mathsf{sort}(l^\sigma) = \mathsf{st}(f) = \mathsf{sort}(l) = s$. For $i \in \{1, \ldots, n\}$: $\mathsf{Var}(t_i) \subseteq \mathsf{Var}(l)$, so σ is well-sorted on $\mathsf{Var}(t_i)$. From induction hypothesis, we have that $t_i^\sigma \in \mathcal{WT}$ and $\mathsf{sort}(t_i^\sigma) = \mathsf{sort}(t_i) = \mathsf{ar}(f, i)$. We conclude that $l^\sigma \in \mathcal{WT}^s$.

$\Rightarrow$ Let l^σ be well-sorted. For any $x \in \mathsf{Var}(l)$ we will prove that x^σ is well-sorted and that $\mathsf{st}(x) = \mathsf{sort}(x^\sigma)$. If $l = x \in \mathcal{V}$, then $\mathsf{Var}(l) = \{x\}$ and $x^\sigma \in \mathcal{WT}^s$, so $\sigma \restriction \mathsf{Var}(l)$ is well-sorted. Otherwise, if $l \notin \mathcal{V}$, each occurrence of x in l must be the argument of some function symbol, say the i-th argument of f. Because $l \in \mathcal{WT}, \mathsf{ar}(f, i) = \mathsf{st}(x)$. Because $l^\sigma \in \mathcal{WT}$ too, $\mathsf{sort}(x^\sigma) = \mathsf{ar}(f, i) = \mathsf{st}(x)$ and $x^\sigma \in \mathcal{WT}$. So σ is well-sorted for variables from $\mathsf{Var}(l)$.

□

Proposition 3.4 *Let $l, r, l^\sigma \in \mathcal{WT}$, $l \notin \mathcal{V}$ and $\mathsf{Var}(r) \subseteq \mathsf{Var}(l)$. Then $r^\sigma \in \mathcal{WT}$ and $\mathsf{sort}(r^\sigma) = \mathsf{sort}(r)$*

Proof: By proposition 3.2 $\mathsf{sort}(l) = \mathsf{sort}(l^\sigma)$. Applying proposition 3.3 we get that $\sigma \restriction \mathsf{Var}(l)$ is well-sorted. Because $\mathsf{Var}(r) \subseteq \mathsf{Var}(l)$ we have that $\sigma \restriction \mathsf{Var}(r)$ is also well-sorted. Again with proposition 3.3 we conclude that $r^\sigma \in \mathcal{WT}$ and $\mathsf{sort}(r^\sigma) = \mathsf{sort}(r)$.
□

3.3 Well-sorted Rewriting

Let an unsorted signature Σ (with variables $\mathcal{V}$) and a set of rules $\mathcal{R}$ with left and right hand sides in $\mathcal{T}(\Sigma)$ be given. In section 2.2 we have seen the following two restrictions on these rules:

- $l \notin \mathcal{V}$
- $\mathsf{Var}(r) \subseteq \mathsf{Var}(l)$

If we extend Σ with a set of sorts S and functions sort and arity to an S-sorted signature $\mathcal{F}$, we require that the rules are sorted due to the new S-sorted signature. Therefore we introduce: An signature $\mathcal{F}$ is *compatible* with some set of rules, if the following sort restrictions hold for the rules ($l \to r$):

- $l, r \in \mathcal{WT}(\mathcal{F})$
- $\mathsf{sort}(l) = \mathsf{sort}(r)$

The TRS does not really change, because we admit ill-sorted terms to be rewritten. If we introduce only one sort then the sort restrictions are satisfied implicitly (see also proposition 3.1). So extending the unsorted signature of some TRS to an one-sorted signature always gives a result which is compatible with that TRS.

Example 3 Let $\mathcal{R}$ consist of the following rules:

f(g(z),y)	→	z	(collapse rule)
f(x,b)	→	c	
f(h(y),y)	→	c	(non left linear rule)
g(c)	→	h(b)	

Then $(\mathcal{F}, \mathcal{R})$ is a TRS with a compatible signature, if we take $\mathcal{F}$ from example 2 on page 11.

. . . . . .

Proposition 3.5 *Let*

- $(\mathcal{F}, \mathcal{R})$ *be a TRS with a compatible S-sorted signature.*
- $s, t \in \mathcal{T}(\mathcal{F})$
- $s \to_{\mathcal{R}} t$
- $\mathsf{sort}(s) \neq \mathsf{sort}(t)$.

Then $s = l^\sigma$, $t = r^\sigma$ *and* $r \in \mathcal{V}$ *for some* $l \to r \in \mathcal{R}$ *and substitution* σ.

So the reduction was in fact an elementary reduction with a collapse rule.

Proof: There are two cases for a reduction $\rightarrow_{\mathcal{R}}$:

If $s = f(t_1, .., t_j, .., t_n) \rightarrow_{\mathcal{R}} f(t_1, .., t'_j, .., t_n) = t$ with $t_j \rightarrow_{\mathcal{R}} t'_j$ then $\mathsf{sort}(s) = \mathsf{st}(f) = \mathsf{sort}(t)$, but we know that this is not the case.

So $s = l^\sigma \rightarrow_{\mathcal{R}} r^\sigma = t$ for some $\sigma : \mathcal{V} \rightarrow \mathcal{T}(\mathcal{F})$ and $l \rightarrow r \in \mathcal{R}$, satisfying the sort restriction that $\mathsf{sort}(l) = \mathsf{sort}(r)$. Furthermore, $l \notin \mathcal{V}$, so $\mathsf{sort}(l^\sigma) = \mathsf{sort}(l)$, with proposition 3.2. Hence

- $\mathsf{sort}(s) = \mathsf{sort}(l^\sigma) = \mathsf{sort}(l) = \mathsf{sort}(r)$ and
- $\mathsf{sort}(s) \neq \mathsf{sort}(t) = \mathsf{sort}(r^\sigma)$.

We get $\mathsf{sort}(r) \neq \mathsf{sort}(r^\sigma)$. Applying proposition 3.2 we conclude: $r \in \mathcal{V}$.

□

Proposition 3.6 *Let $(\mathcal{F}, \mathcal{R})$ be a TRS with a compatible S-sorted signature $\mathcal{F}$, $\alpha \in S$, $s \in \mathcal{WT}^\alpha$ and $s \rightarrow_{\mathcal{R}} t$. Then $t \in \mathcal{WT}^\alpha$ too.*

Proof: (induction on the structure of the proof that $s \rightarrow_{\mathcal{R}} t$.)

If $s = l^\sigma$ and $t = r^\sigma$ for some $\sigma : \mathcal{V} \rightarrow \mathcal{T}$ and $l \rightarrow r \in \mathcal{R}$, then $l, r, l^\sigma \in \mathcal{WT}^\alpha$ (see proposition 3.2) and $\mathsf{Var}(r) \subseteq \mathsf{Var}(l)$. So we can apply proposition 3.4, giving $r^\sigma \in \mathcal{WT}$ and $\mathsf{sort}(r^\sigma) = \mathsf{sort}(r) = \mathsf{sort}(l) = \mathsf{sort}(l^\sigma) = \alpha$.

If $s = f(t_1, .., t_j, .., t_n)$ and $t = f(t_1, .., t'_j, .., t_n)$ with $t_j \rightarrow_{\mathcal{R}} t'_j$ then we know:

1. $f \in \mathcal{F}$, because $s \in \mathcal{WT}$
2. $\mathsf{arity}(f) = n$, because $s \in \mathcal{WT}$.
3. $\forall i : 1 \leq i \leq n : t_i \in \mathcal{WT}$. Therefore $t'_j \in \mathcal{WT}$ from induction hypothesis.
4. $\forall i : 1 \leq i \leq n : \mathsf{ar}(f, i) = \mathsf{sort}(t_i)$, because $s \in \mathcal{WT}$. Furthermore: $\mathsf{sort}(t'_j) = \mathsf{sort}(t_j)$ (from ih) $= \mathsf{ar}(f, j)$.

From these four statements we know that $t \in \mathcal{WT}$. Furthermore $\mathsf{sort}(t) = \mathsf{st}(f) = \mathsf{sort}(s) = \alpha$.

□

This proposition says that the set $\mathcal{WT}(\mathcal{F})$ is closed under rewriting in the TRS $(\mathcal{F}, \mathcal{R})$ if $\mathcal{F}$ is a compatible S-sorted signature. It also shows that the set $\mathcal{WT}^\alpha$ is closed under rewriting for any $\alpha \in S$. If some subset of the terms is closed under rewriting, then it is possible to restrict the rewrite relation to this subset. We introduce a new definition for restricted TRS's:

Let an S-sorted signature $\mathcal{F}$ (with variables $\mathcal{V}$), a set of rules $\mathcal{R}$ and a subset of the terms $\mathcal{A}$ be given. Then $(\mathcal{F}, \mathcal{R}, \mathcal{A})$ is a restricted TRS if:

- $l \notin \mathcal{V}$
- $\mathsf{Var}(r) \subseteq \mathsf{Var}(l)$
- $l, r \in \mathcal{WT}(\mathcal{F})$
- $\mathsf{sort}(l) = \mathsf{sort}(r)$
- $\mathcal{A}$ is closed under the unrestricted rewrite relation of $(\mathcal{F}, \mathcal{R})$.

An unrestricted TRS $(\mathcal{F}, \mathcal{R})$ can be written now as $(\mathcal{F}, \mathcal{R}, \mathcal{T}(\mathcal{F}))$, abbreviated to $(\mathcal{F}, \mathcal{R}, \mathcal{T})$. We define a many-sorted term rewriting system (MTRS) M to be a TRS with a compatible S-sorted signature, in which we only reduce well-sorted terms. This can be written as $M = (\mathcal{F}, \mathcal{R}, \mathcal{WT})$. We can also restrict our terms to the well-sorted terms of some sort α. Then we get the α-sorted term rewriting system $(\mathcal{F}, \mathcal{R}, \mathcal{WT}^\alpha)$. Note that rules of a sort different to α can also play a role in the last system.

This definition of MTRS's and well-sorted rewriting is equivalent to the definition of a many-sorted term rewriting system in [Zan91, 618], but the notations are quite different.

3.4 The Direct Sum for Many-sorted TRS

We can generalize the disjoint union operator to S-sorted term rewriting systems. Now T_1 and T_2 are S_1- and S_2-sorted TRS's, with signatures $\mathcal{F}_1 = (F_1, \mathcal{V}_1, S_1, \mathsf{arity}, \mathsf{st}, \mathsf{ar})$ and $\mathcal{F}_2 = (F_2, \mathcal{V}_2, S_2, \mathsf{arity}, \mathsf{st}, \mathsf{ar})$. Again F_1 and F_2 have to be disjoint and $\mathcal{V}_1$ and $\mathcal{V}_2$ are assumed to be disjoint without loss of generality. However, *we require no disjointness restriction on S_1 and S_2.* The functions st and ar on function symbols and variables are extended to the new domain in the same way as we extended arity in section 2.4. In this way we get the new signature $\mathcal{F}_\oplus = (F_1 \oplus F_2, \mathcal{V}_1 \oplus \mathcal{V}_2, S_1 \cup S_2, \mathsf{arity}, \mathsf{st}, \mathsf{ar})$ and the direct sum of two many-sorted TRS's $T_1 \oplus T_2 = (\mathcal{F}_\oplus, \mathcal{R}_\oplus, \mathcal{T})$. (We are speaking about unsorted rewriting.)

The set of terms in the new TRS is $\mathcal{T}(\mathcal{F}_\oplus)$ and the set of well-sorted terms is given by $\mathcal{WT}(\mathcal{F}_\oplus)$. The following lemmas say that the (well-sorted) terms of $\mathcal{F}_1$ and $\mathcal{F}_2$ are (well-sorted) terms in $\mathcal{F}_\oplus$:

Lemma 3.7 *$\mathcal{WT}(\mathcal{F}_j) \subseteq \mathcal{WT}(\mathcal{F}_\oplus)$ for $j = 1, 2$.*

Proof: Let $j = 1$ (without loss of generality). With induction to the depth of t we show that if $t \in \mathcal{WT}(\mathcal{F}_1)$ then $t \in \mathcal{WT}(\mathcal{F}_\oplus)$ too.
□

Lemma 3.8 *$\mathcal{T}(\mathcal{F}_j) \subseteq \mathcal{T}(\mathcal{F}_\oplus)$ for $j = 1, 2$.*

The proof of this lemma is analogous to the proof of the previous lemma.

Let S_1 and S_2 be the sets of sorts of $\mathcal{F}_1$ and $\mathcal{F}_2$, respectively. If $S_1 \cap S_2 = \varnothing$ then $\mathcal{WT}(\mathcal{F}_\oplus) = \mathcal{WT}(\mathcal{F}_1) \cup \mathcal{WT}(\mathcal{F}_2)$, because we cannot make mixed terms without violating the sort restrictions.

Proposition 3.9 *If T_1 is an S_1-sorted TRS and T_2 is an S_2-sorted TRS then $T_\oplus$ is an $S_1 \cup S_2$-sorted TRS.*

Proof: Every rule $l \to r$ from $\mathcal{R}_\oplus$ is in some $\mathcal{R}_i$, with $i \in \{1, 2\}$. Because T_i is S_i-sorted, we know that $l, r \in \mathcal{WT}(\mathcal{F}_i)$ and $\mathsf{sort}(l) = \mathsf{sort}(r)$. Lemma 3.7 gives us that $l, r \in \mathcal{WT}(\mathcal{F}_\oplus)$. So the sort conditions are satisfied. Moreover, the signature of $\mathcal{F}_1 \oplus \mathcal{F}_2$ is $S_1 \cup S_2$-sorted. So $T_\oplus$ is an $S_1 \cup S_2$-sorted TRS. □

The last proposition gives a justification of speaking about the direct sum of MTRS's. So we have:

$$(\mathcal{F}_1, \mathcal{R}_1, \mathcal{WT}) \oplus (\mathcal{F}_2, \mathcal{R}_2, \mathcal{WT}) = (\mathcal{F}_\oplus, \mathcal{R}_\oplus, \mathcal{WT})$$

In section 2.4 we saw the definition of modularity. A property is modular if it is resistant against taking the disjoint union. The same definition can be used in the many-sorted case. It is not known if properties that are modular in the unsorted case, are modular in the many-sorted case too. The reverse is trivial, because an unsorted TRS can be seen as an many-sorted TRS with only one sort. Therefore a new notion is introduced: MS-modular.

$\mathcal{P}$ is MS-modular iff for all MTRS's M_1 and M_2 holds:

$$\mathcal{P}(M_1 \oplus M_2) \iff (\mathcal{P}(M_1) \wedge \mathcal{P}(M_2))$$

3.5 Sort Elimination and Persistence

We are interested in what happens with some property $\mathcal{P}$ of some MTRS, if it is regarded as a TRS. This can be seen as admitting sort clashes. The sort elimination-operator (Θ) transforms an MTRS into the corresponding TRS. It is defined as a function:

$$\Theta((\mathcal{F}, \mathcal{R}, \mathcal{WT})) = (\mathcal{F}, \mathcal{R}, \mathcal{T}).$$

If eliminating the sort restrictions has no influence on the property $\mathcal{P}$, then $\mathcal{P}$ is called *persistent*. More precisely, a property $\mathcal{P}$ on TRS's is called persistent if for all many-sorted TRS's M:

$$\mathcal{P}(M) \iff \mathcal{P}(\Theta(M))$$

This definition of persistence is equivalent to the definition of [Zan91, 619] in which an MTRS is transformed into a copied version with a new one-sorted signature.

In [Zan91] a proof is given that persistent properties are modular. The same proof technique can be used for MS-modularity. This gives the following theorem:

Theorem 3.10 *Persistent and component closed properties are MS-modular.*

Proof: Let $\mathcal{P}$ be a component closed and persistent property. Let M_1 and M_2 be two disjoint many-sorted term rewriting systems. The signatures are $(\mathcal{F}_1, \mathcal{V}_1, S_1, \mathsf{arity}_1, \mathsf{st}_1, \mathsf{ar}_1)$ and $(\mathcal{F}_2, \mathcal{V}_2, S_2, \mathsf{arity}_2, \mathsf{st}_2, \mathsf{ar}_2)$ respectively. We have to prove the MS-modularity of $\mathcal{P}$, using the persistence of $\mathcal{P}$. Therefore we construct a new MTRS M, which combines M_1 and M_2. The distinction between the two MTRS's is kept in the sorts. The new signature $\mathcal{F} = (F, \mathcal{V}, S, \mathsf{arity}, \mathsf{st}, \mathsf{ar})$ with

$$
\begin{aligned}
F &= F_1 \oplus F_2 \\
\mathcal{V} &= \mathcal{V}_1 \oplus \mathcal{V}_2 \\
S &= \{(s,1) \mid s \in S_1\} \oplus \{(s,2) \mid s \in S_2\} \\
\mathsf{arity}(f) &= \begin{cases} \mathsf{arity}_1(f) & \text{if } f \in \mathcal{F}_1 \\ \mathsf{arity}_2(f) & \text{if } f \in \mathcal{F}_2 \end{cases} \\
\mathsf{st}(f) &= \begin{cases} (\mathsf{st}_1(f), 1) & \text{if } f \in \mathcal{F}_1 \cup \mathcal{V}_1 \\ (\mathsf{st}_2(f), 2) & \text{if } f \in \mathcal{F}_2 \cup \mathcal{V}_2 \end{cases} \\
\mathsf{ar}(f,i) &= \begin{cases} (\mathsf{ar}_1(f,i), 1) & \text{if } f \in \mathcal{F}_1 \\ (\mathsf{ar}_2(f,i), 2) & \text{if } f \in \mathcal{F}_2 \end{cases}
\end{aligned}
$$

The new set of rules $\mathcal{R} = \mathcal{R}_1 \oplus \mathcal{R}_2$. This gives a new MTRS $M = (\mathcal{F}, \mathcal{R}, \mathcal{WT})$. The terms of M with a sort of the form $(s,1)$, correspond one-to-one with terms of M_1. Furthermore, the rewrite relations on these terms correspond one-to-one. The same holds with 1 replaced by 2. Because terms with a sort $(s,1)$ do never reduce to terms with a sort $(s,2)$ and vice versa (proposition 3.6), they are in different equivalence classes. Because $\mathcal{P}$ is component closed, we can conclude, that

$$(\mathcal{P}(M_1) \wedge \mathcal{P}(M_2)) \iff \mathcal{P}(M).$$

From the persistence of $\mathcal{P}$, we conclude that

$$\mathcal{P}(M_1 \oplus M_2) \iff \mathcal{P}(\Theta(M_1 \oplus M_2)).$$

But, because the function symbols of $\Theta(M_1 \oplus M_2)$ and $\Theta(M)$ are equal, and also the rules are equal and finally they both have no sort restrictions, we can conclude that

$$\mathcal{P}(\Theta(M_1 \oplus M_2)) \iff \mathcal{P}(\Theta(M)).$$

Combining these three results, we get that

$$(\mathcal{P}(M_1) \wedge \mathcal{P}(M_2)) \iff \mathcal{P}(M_1 \oplus M_2)$$

and hence $\mathcal{P}$ is MS-modular.
□

3.6 Definition of the Problem

The problem of this thesis is to establish the relationship between the following three features:

1. persistence
2. modularity for TRS's
3. modularity for MTRS's.

In fact [Zan91] shows that (1) implies (2). With the same method we could prove in theorem 3.10 that (1) implies (3). Furthermore it is evident that (3) implies (2). The main theorem of this thesis is that (3) implies (1) (Theorem 4.23).

The most interesting question is whether (2) implies (1), because in that case we know that persistence holds for the same properties that are proven modular. This problem is reduced to the remaining question whether (2) implies (3). One strategy to achieve this result would be to "type-check" the modularity-proofs. Another approach would give a general proof that modular properties are MS-modular. This seems rather difficult, because it is difficult (or perhaps impossible) to construct for any MTRS a corresponding TRS with an isomorphic reduction relation. But perhaps this question leads to a closer analysis of the exact restrictions of many-sorted rewriting, which would be a fruitful item for further research.

4 Simulation of Sort Elimination by the Direct Sum

In this chapter we will prove that component closed properties $\mathcal{P}$ that are modular for MTRS's with a finite set of sorts are persistent.

The structure of the proof is the same as we have seen in theorem 3.10. There we took two MTRS's M_1 and M_2, we constructed a similar MTRS M, proved that the disjoint union of M_1 and M_2 corresponded to sort elimination in M and used persistence to establish MS-modularity. Now we take the reverse order, so the structure of the proof reads:

1. Given an MTRS M, construct a family of other MTRS's, indexed by the set of sorts: $\{M_s \mid s \in S\}$.
2. Prove that the reduction relation of M is equivalent to the reduction relation of M_s for any $s \in S$. (section 4.1)
3. Prove that the reduction relation of $\Theta(M)$ is equivalent to the reduction relation of $\bigoplus_{s\in S} M_s$. (section 4.2)
4. Use the modularity of $\mathcal{P}$ for MTRS's to observe that $\mathcal{P}(\bigoplus_{s\in S} M_s) \iff \forall s \in S : \mathcal{P}(M_s)$. Here the finiteness of the set S is used.
5. Conclude that $\mathcal{P}(M) \iff \mathcal{P}(\Theta(M))$ and hence $\mathcal{P}$ is persistent. (section 4.3)

4.1 Modified Copies of an MTRS

Let $M = (\mathcal{F}, \mathcal{R}, \mathcal{WT})$, with signature $\mathcal{F} = (F, \mathcal{V}, S, \mathsf{arity}, \mathsf{st}, \mathsf{ar})$ where S is a set of sorts. These objects are used in this whole section. Because we don't specify anything about them, they can be seen as universally quantified.

In this section a construction is given for a family of MTRS's M_s (for any $s \in S$), starting from the general MTRS M. The idea is that each M_s is a modified copy of M, in which the function symbols are the same, but the sorts are different. The sorts are permuted in a consistent way. To define this construction, we use a family of functions ϕ_s, for any $s \in S$. These functions depend on the binary operators $+$ and $-$ on sorts.

These operators assume that we can find a binary operator '$+$' on S in such a way that $(S, +)$ forms an abelian group:

- There is a $0 \in S$ such that $s + 0 = s$.
- Every element $s \in S$ has an inverse $-s$ such that $s + (-s) = 0$.
- The $+$ operator is associative.
- The $+$ operator is commutative.

We write $s - s'$ instead of $s + (-s')$. If S is a finite set, we can identify it with $\{0, 1, \ldots, n-1\}$. Then the operators $+ \pmod n$ and $- \pmod n$ satisfy. If S is a countable set, we can identify it with $\mathbb{Z}$ and the usual $0, +$ and $-$ satisfy the conditions.

We define for all $s \in S$ and for all $f \in \mathcal{F}$ a new function symbol f_s and also for any $x \in \mathcal{V}$ a new variable x_s. For any $s \in S$, the sets of new function symbols $\{f_s \mid f \in F\}$ are denoted by F_s. In the same way, the sets of variables $\{x_s \mid x \in \mathcal{V}\}$ are denoted by $\mathcal{V}_s$. We also define the functions sort, ar and arity on these sets. Let $f_s \in \mathcal{F}_s$ and $x_s \in \mathcal{V}_s$, then

$$\mathsf{arity}(f_s) = \mathsf{arity}(f)$$

$$\mathsf{st}(f_s) = \mathsf{st}(f) + s$$

$$\mathsf{st}(x_s) = \mathsf{st}(x) + s$$

$$\mathsf{ar}(f_s, i) = \mathsf{ar}(f, i) + s \qquad \text{for } 1 \leq i \leq \mathsf{arity}(f).$$

In this way we have obtained a new S-sorted signature, denoted by $\mathcal{F}_s$:

$$\mathcal{F}_s = (F_s, \mathcal{V}_s, S, \mathsf{arity}, \mathsf{st}, \mathsf{ar}).$$

Example 4 If we apply this construction to the signature of example 2, we get three new signatures (because there are three sorts).

$\mathcal{F}_0$			$\mathcal{F}_1$			$\mathcal{F}_2$		
$x_0 : 0$	$a_0 : 0$	$f_0 : 0 \times 1 \to 2$	$x_1 : 1$	$a_1 : 1$	$f_1 : 1 \times 2 \to 0$	$x_2 : 2$	$a_2 : 2$	$f_2 : 2 \times 0 \to 1$
$y_0 : 1$	$b_0 : 1$	$g_0 : 2 \to 0$	$y_1 : 2$	$b_1 : 2$	$g_1 : 0 \to 1$	$y_2 : 0$	$b_2 : 0$	$g_2 : 1 \to 2$
$z_0 : 2$	$c_0 : 2$	$h_0 : 1 \to 0$	$z_1 : 0$	$c_1 : 0$	$h_1 : 2 \to 1$	$z_2 : 1$	$c_2 : 1$	$h_2 : 0 \to 2$

. . . . . .

We define a mapping $\phi_s : \mathcal{WT}(\mathcal{F}) \to \mathcal{WT}(\mathcal{F}_s)$ for any $s \in S$:

$$\phi_s(x) = x_s$$

$$\phi_s(f(t_1, \ldots, t_n)) = f_s(\phi_s(t_1), \ldots, \phi_s(t_n))$$

Now we can prove the following lemma:

Lemma 4.1 *If $t \in \mathcal{WT}(\mathcal{F})$ then $\phi_s(t) \in \mathcal{WT}(\mathcal{F}_s)$ and $\mathsf{sort}(\phi_s(t)) = \mathsf{sort}(t) + s$.*

Proof: (induction on the depth of t)
□

Lemma 4.1 says us something about the functions ϕ_s. The intuition behind them is that they assign a label s to all function symbols of some term. Because the sorts are changed in a consistent way, the result of labeling a well-sorted term is again well-sorted. Therefore the result is in $\mathcal{WT}(\mathcal{F}_s)$. This is proved in the rest of this section.

We hope that the functions ϕ_s are bijections. Let us define a function $\tilde{\phi_s} : \mathcal{WT}(\mathcal{F}_s) \to \mathcal{WT}(\mathcal{F})$ and prove that this is the inverse of ϕ_s:

$$\tilde{\phi_s}(x_s) = x$$

$$\tilde{\phi_s}(f_s(t_1, \ldots, t_n)) = f(\tilde{\phi_s}(t_1), \ldots, \tilde{\phi_s}(t_n))$$

Now we have:

Lemma 4.2 *The functions $\tilde{\phi_s}$ are the inverse functions of ϕ_s. Hence ϕ_s are bijections from $\mathcal{WT}(\mathcal{F}) \to \mathcal{WT}(\mathcal{F}_s)$.*

Proof:

Claim 1 $\forall t \in \mathcal{WT}(\mathcal{F}) : \tilde{\phi_s}(\phi_s(t)) = t$

Claim 2 $\forall t \in \mathcal{WT}(\mathcal{F}_s) : \phi_s(\tilde{\phi_s}(t)) = t$

Proof: Induction on $\mathsf{depth}(t)$

□

We conclude from claim 1 and 2 that $\tilde{\phi_s} = \phi_s^{-1}$ and therefore ϕ_s is bijective.
□

The functions ϕ_s are extended to substitutions in a natural way. For a substitution $\sigma : \mathcal{V} \to \mathcal{WT}(\mathcal{F})$ and for all $x \in \mathcal{V}$ we define

$$\phi_s(\sigma)(x_s) = \phi_s(x^\sigma).$$

This generalizes to a substition ϕ_s on terms, as in subsection 2.1.

Lemma 4.3 *The functions $\phi_s : (\mathcal{V} \to \mathcal{WT}(\mathcal{F})) \to (\mathcal{V}_s \to \mathcal{WT}(\mathcal{F}_s))$ are surjective.*

Proof: Let $\sigma : \mathcal{V}_s \to \mathcal{WT}(\mathcal{F}_s)$ be an arbitrary substitution. Then we have to show that there exist a substitution $\rho : \mathcal{V} \to \mathcal{WT}(\mathcal{F})$ with $\phi_s(\rho) = \sigma$. From lemma 4.2 we know that ϕ_s has an inverse. So we can define $\forall x \in \mathcal{V}$:

$$\rho(x) = \phi_s^{-1}(x_s^\sigma).$$

Then we have:

	$\phi_s(\rho)(x_s)$	
$=$	$\phi_s(x^\rho)$	by the definition of ϕ_s on substitutions
$=$	$\phi_s(\phi_s^{-1}(x_s^\sigma))$	by the definition of ρ
$=$	x_s^σ	with lemma 4.2

□

The definition of these substitutions ϕ_s is made in such a way that we can prove the following lemma:

Lemma 4.4 *For all $t \in \mathcal{WT}(\mathcal{F})$, substitutions σ and $s \in S$ holds: $\phi_s(t^\sigma) = \phi_s(t)^{\phi_s(\sigma)}$*

Proof: Induction on the depth of t.
□

This is a useful result, because we now know that if the left-hand side of some rule $l \to r$ matches a term t, this matching can be translated to the labeled version $\phi_s(t)$. This is used in lemma 4.5.

We can use the functions ϕ_s on well-sorted terms, to define new MTRS's M_s, starting from a general MTRS M. Let $M = (\mathcal{F}, \mathcal{R}, \mathcal{WT})$, where $\mathcal{F}$ is an S-sorted signature, $\mathcal{R}$ is a set of rules with function symbols from $\mathcal{F}$ and $\mathcal{WT}(\mathcal{F})$ are the well-sorted terms over $\mathcal{F}$. Then we define:

$$\mathcal{R}_s = \{\phi_s(l) \to \phi_s(r) \mid l \to r \in \mathcal{R}\} \text{ and}$$

$$M_s = (\mathcal{F}_s, \mathcal{R}_s, \mathcal{WT}).$$

Note that from the signatures in M and M_s it turns out that the set $\mathcal{WT}$ in M is in fact $\mathcal{WT}(\mathcal{F})$, while the set $\mathcal{WT}$ in M_s is $\mathcal{WT}(\mathcal{F}_s)$.

Given an MTRS M we now have defined for each $s \in S$ a new MTRS M_s. This corresponds with step 1 in the overview of the proof at the beginning of this chapter. Now we have to prove, that all these MTRS's are isomorphic. This is true, because only the names of the function symbols are changed, and their sorts are permuted in a consistent way. This is proved by the following lemmas:

Lemma 4.5 *For $t, t' \in \mathcal{WT}(\mathcal{F})$ holds: if $t \to_{\mathcal{R}} t'$ then $\phi_s(t) \to_{\mathcal{R},s} \phi_s(t')$*

Proof: Induction on the structure of the proof that $t \to_{\mathcal{R}} t'$: The induction step is trivial, so let us do the case of an elementary reduction:
Let $t = l^\sigma$ and $t' = r^\sigma$ for some rule $l \to r \in \mathcal{R}$ then

	$\phi_s(l^\sigma)$	
$=$	$(\phi_s(l))^{\phi_s(\sigma)}$	by lemma 4.4
$\to_{\mathcal{R},s}$	$(\phi_s(r))^{\phi_s(\sigma)}$	because $\phi_s(l) \to \phi_s(r) \in \mathcal{R}_s$
$=$	$\phi_s(r^\sigma)$	by lemma 4.4

□

Lemma 4.6 *For $t, t' \in \mathcal{WT}(\mathcal{F})$ holds: if $\phi_s(t) \to_{\mathcal{R},s} \phi_s(t')$ then $t \to_{\mathcal{R}} t'$.*

Proof: Let $t = f(t_1, \ldots, t_n)$ (variables are normal forms and can't reduce). The proof is with induction on the structure of the proof that $\phi_s(t) \to_{\mathcal{R},s} \phi_s(t')$, but again we restrict to the base case, because the induction step is trivial.
Let $\phi_s(t) = l^\sigma$ and $\phi_s(t') = r^\sigma$ with $l \to r$ in $\mathcal{R}_s$, then we know from the definition of $\mathcal{R}_s$ that $l = \phi_s(a)$ and $r = \phi_s(b)$ for some rule $a \to b \in \mathcal{R}$. Then we have:

	$\phi_s(t)$	
$=$	l^σ	
$=$	$(\phi_s(a))^\sigma$	
$=$	$(\phi_s(a))^{\phi_s(\rho)}$	
	for some substitution ρ	from lemma 4.3
$=$	$\phi_s(a^\rho)$	by the definition of $\phi(\sigma)$

Then $t = a^\rho$, because ϕ_s is injective on terms (see lemma 4.2). In the same way we can get: $t' = b^\rho$. Because $a \to b \in \mathcal{R}$, we have: $t \to_{\mathcal{R}} t'$.
□

The conclusion of this section can be summarized in the following proposition:

Proposition 4.7 *There exist a bijection $\phi_s : \mathcal{WT}(\mathcal{F}) \to \mathcal{WT}(\mathcal{F}_s)$ (lemma 4.2) with for all $t, t' \in \mathcal{WT}(\mathcal{F})$:*

$$t \to_{\mathcal{R}} t' \iff \phi_s(t) \to_{\mathcal{R},s} \phi_s(t')$$

(lemma 4.5 and 4.6).

So the rewrite relation of sorted rewriting in the MTRS M and the rewrite relation of sorted rewriting in some MTRS M_s are "isomorphic". This corresponds with step 2 of the overview of the proof at the beginning of this chapter.

4.2 Sort Elimination is taking a Direct Sum

In this section we will see that unsorted rewriting in the TRS $\Theta(M) = (\mathcal{F}, \mathcal{R}, \mathcal{T})$ can be simulated with rewriting in the MTRS that is the direct sum of the MTRS's M_s of section 4.1. This corresponds to step 3 of the overview of the proof at the beginning of this chapter. We write $\mathcal{F}_\oplus$ for $\bigoplus_{s \in S}(\mathcal{F}_s)$. In the same way we define $\mathcal{V}_\oplus$, $\mathcal{R}_\oplus$ and $M_\oplus$. We will define a function $\psi : S \times \mathcal{T}(\mathcal{F}) \to \mathcal{WT}(\mathcal{F}_\oplus)$ taking a sort s and a term and returning a well-sorted term with its root symbol in $\mathcal{F}_s$. The $+$ and $-$ operators are the same as in section 4.1.

The function $\psi : S \times \mathcal{T}(\mathcal{F}) \to \mathcal{WT}(\mathcal{F}_\oplus)$ is defined mutually recursively:

$$\begin{aligned} \psi(s, x) &= x_s \\ \psi(s, f(t_1, \ldots, t_n)) &= f_s\Big(\psi(s_1, t_1), \ldots, \psi(s_n, t_n)\Big) \\ &\quad \text{where } s_i = s + \mathsf{ar}(f, i) - \mathsf{sort}(t_i) \end{aligned}$$

The following intuition lies behind this function: It starts labeling a term at the root with label s. It goes on labeling the sons with label s, until a subterm with the wrong sort is reached. Then the label is changed in such a way, that the resulting term gets well-sorted. In this way the function symbols within the well-sorted islands of a term get the same label, while a transition between two well-sorted parts results in a transition of the labels.

Let us prove that the range of ψ is indeed $\mathcal{WT}(\mathcal{F}_\oplus)$ (so the result is well-sorted):

Lemma 4.8 *For all $t \in \mathcal{T}(\mathcal{F})$ and $s \in S$ holds: $\psi(s, t) \in \mathcal{WT}(\mathcal{F}_\oplus)$ and $\mathsf{sort}(\psi(s, t)) = \mathsf{sort}(t) + s$.*

Proof: induction on $\mathsf{depth}(t)$
□

We define the function ψ on substitutions in the following way. For $\sigma : \mathcal{V} \to \mathcal{T}(\mathcal{F})$ and for $x \in \mathcal{V}$:

$$\psi(s, \sigma)(x_s) = \psi(s + \mathsf{st}(x) - \mathsf{sort}(x^\sigma), x^\sigma)$$

They have the same goal as the substitution $\phi_s(\sigma)$ of section 4.1. Let us write $\phi(s, t) = \phi_s(t)$. Now we can prove indeed the following lemma:

Lemma 4.9 *For all $l \in \mathcal{WT}(\mathcal{F}) \setminus \mathcal{V}$ and substitutions σ holds: $\psi(s, l^\sigma) = (\phi(s, l))^{\psi(s,\sigma)}$.*

Proof: (induction on $\mathsf{depth}(l)$)
If $l \in \mathcal{WT}(\mathcal{F}) \setminus \mathcal{V}$ then

$$\begin{array}{lll} & \psi(s, l^\sigma) & \\ = & f_s(\psi(s_1, t_1^\sigma), \ldots, \psi(s_n, t_n^\sigma)) & \text{definition of } \psi \\ & \text{where } s_i = s + \mathsf{ar}(f, i) - \mathsf{sort}(t_i^\sigma) & \end{array}$$

Claim 1 *$\psi(s_i, t_i^\sigma) = (\phi(s, t_i^\sigma))^{\psi(s,\sigma)}$ for all $1 \leq i \leq n$.*

Proof: There are two possibilities: $t_i \in \mathcal{V}$ or not. In the second case we can use the induction hypothesis, but in the first case this is impossible.

If $t_i = y \in \mathcal{V}$ then

$$\begin{array}{lll} & \psi(s_i, t_i^\sigma) & \\ = & \psi(s + \mathsf{ar}(f, i) - \mathsf{sort}(y^\sigma), y^\sigma) & \text{definition of } s_i \\ = & \psi(s + \mathsf{st}(y) - \mathsf{sort}(y^\sigma), y^\sigma) & \text{because } l \in \mathcal{WT}(\mathcal{F}) \\ = & (y_s)^{\psi(s,\sigma)} & \text{by definition of } \psi(s, \sigma) \\ = & (\phi(s, t_i))^{\psi(s,\sigma)} & \end{array}$$

If $t_i \notin \mathcal{V}$ then

$$\begin{array}{lll} & \psi(s_i, t_i^\sigma) & \\ = & \psi(s + \mathsf{ar}(f, i) - \mathsf{sort}(t_i^\sigma), t_i^\sigma) & \\ = & \psi(s + \mathsf{ar}(f, i) - \mathsf{sort}(t_i), t_i^\sigma) & \text{by proposition 3.2} \\ = & \psi(s, t_i^\sigma) & \text{because } l \in \mathcal{WT}(\mathcal{F}) \\ = & (\phi(s, t_i))^{\psi(s,\sigma)} & \text{ih} \end{array}$$

□

We resume the first computation:

$$\begin{array}{lll} & \psi(s, l^\sigma) & \\ = & \cdots & \text{see above} \\ = & f_s((\phi(s, t_1))^{\psi(s,\sigma)}, \ldots, (\phi(s, t_n))^{\psi(s,\sigma)}) & \text{by the claim} \\ = & (\phi(s, f(t_1, \ldots, t_n)))^{\psi(s,\sigma)} & \text{definition } \phi \end{array}$$

□

When we want to reason about ψ as a function of its second argument (with its first argument steady) we write $\psi_s(t) = \psi(s, t)$. Now we can formulate the following corollary:

Corollary 4.10 $\psi_s \restriction \mathcal{WT}(\mathcal{F}) = \phi_s$

Proof: We will show that for all $t \in \mathcal{WT}(\mathcal{F}) : \psi(s,t) = \phi(s,t)$:

If $t = x$ then $\psi(s,x) = x_s = \phi(s,x)$.

If $t = f(t_1, \ldots, t_n)$ then it follows from lemma 4.9 with $\sigma = \mathsf{id}$.

□

This corollary justifies the intuition that we had for the function ψ: In a well-sorted term t are no sort clashes, so $\psi(s,t)$ consists of symbols with only one label.

Example 5 Let $\mathcal{F}_\oplus$ be the union of the signatures $\mathcal{F}_0, \mathcal{F}_1$ and $\mathcal{F}_2$ from example 4. Then we can verify with the signature of this example on page 21 that the right hand side term is indeed well-sorted:

Figure 2: the action of ψ_1

. . . . . .

We hope that using the definition of $\psi(s,\sigma)$ we can prove that for every reduction $t \to_{\mathcal{R}} t'$ in M there exist a reduction $\psi(s,t) \to_{\mathcal{R},\oplus} \psi(s,t')$ in $M_\oplus$. But this is only true if there is no collapse rule in $\mathcal{R}$. In the case of a collapse rule it is possible that $\mathsf{sort}(t) \neq \mathsf{sort}(t')$ (see proposition 3.5), and the label of the reduct has to be changed. In fact things go wrong, because the previous lemma doesn't hold for variables. The label of the root symbol depends on the sort of it. Therefore we introduce a new function: χ. The only reason for its need is to compensate the influence of the sorts on the label of the root symbol. We define:

$$\chi(s,t) = \psi(s - \mathsf{sort}(t), t)$$

To give an intuition for the effect of the function χ:

Lemma 4.11 *For all terms* $t \in \mathcal{T}(\mathcal{F}) : \chi(s,t) \in \mathcal{WT}(\mathcal{F}_\oplus)$ *and* $\mathsf{sort}(\chi(s,t)) = s$.

Proof: $\chi(s,t) = \psi(s - \mathsf{sort}(t), t)$. From lemma 4.8 we get that $\chi(s,t)$ is in $\mathcal{WT}(\mathcal{F}_\oplus)$ and that $\mathsf{sort}(\chi(s,t)) = s - \mathsf{sort}(t) + \mathsf{sort}(t)$.
□

So indeed we see that the sort of $\chi(s,t)$ does not depend on the sort of t. Defining $\chi_s(t) = \chi(s,t)$, we can see the function χ_s as:

$$\chi_s : \mathcal{T}(\mathcal{F}) \to \mathcal{WT}^s(\mathcal{F}_\oplus).$$

We can prove for the domain of χ (all terms, including variables):

Lemma 4.12 *For any $l \in \mathcal{WT}(\mathcal{F})$, $\sigma : \mathcal{V} \to \mathcal{T}(\mathcal{F})$ and $s \in S$ holds:*

$$\chi(s, l^\sigma) = (\phi(p,l))^{\psi(p,\sigma)} \qquad \textit{with } p = s - \mathsf{st}(l)$$

Proof: Let $p = s - \mathsf{st}(l)$. There are two cases:

If $l = y \in \mathcal{V}$ then we have:

$$\begin{array}{lll} & (\phi(p,l))^{\psi(p,\sigma)} & \\ = & (y_{s-\mathsf{st}(y)})^{\psi(s-\mathsf{st}(y),\sigma)} & \text{definition of } \phi \text{ on variables} \\ = & \psi(s - \mathsf{st}(y) + \mathsf{st}(y) - \mathsf{sort}(y^\sigma), y^\sigma) & \text{by the definition of } \psi(s,\sigma) \\ = & \psi(s - \mathsf{sort}(y^\sigma), y^\sigma) & \\ = & \chi(s, l^\sigma) & \end{array}$$

If $l \notin \mathcal{V}$ we use lemma 4.9:

$$\begin{array}{lll} & \chi(s, l^\sigma) & \\ = & \psi(s - \mathsf{sort}(l^\sigma), l^\sigma) & \text{definition } \chi \\ = & (\phi(s - \mathsf{sort}(l^\sigma), l))^{\psi(s-\mathsf{sort}(l^\sigma),\sigma)} & \text{see lemma 4.9} \\ = & (\phi(p,l))^{\psi(p,\sigma)} & \text{by prop. 3.2 and because } p = s - \mathsf{sort}(l) \end{array}$$

□

The following lemmas show that χ_s has an inverse χ_s^{-1}. Let us define $\tilde{\chi} : \mathcal{WT}(\mathcal{F}_\oplus) \to \mathcal{T}(\mathcal{F})$:

$$\begin{array}{rcl} \tilde{\chi}(x_s) & = & x \\ \tilde{\chi}(f_s(t_1, \ldots, t_n)) & = & f(\tilde{\chi}(t_1), \ldots, \tilde{\chi}(t_n)) \end{array}$$

Lemma 4.13 *For all $t \in \mathcal{T}(\mathcal{F})$: $\tilde{\chi}(\chi_s(t)) = t$*

Proof: Induction on $\mathsf{depth}(t)$.
□

The function $\tilde{\chi}$ can not be the inverse of χ_s, because it has the wrong domain. But now we restrict the domain and we define for any $s \in S$ a function $\tilde{\chi}_s : \mathcal{WT}^s(\mathcal{F}_\oplus) \to \mathcal{T}(\mathcal{F})$ by:

$$\tilde{\chi}_s = \tilde{\chi} \restriction \mathcal{WT}^s(\mathcal{F}_\oplus).$$

Now we can prove:

Lemma 4.14 *The function $\tilde{\chi}_s$ is the inverse of χ_s for all $s \in S$.*

Proof:

Claim 1 *For all $t \in \mathcal{T}(\mathcal{F})$: $\tilde{\chi}_s(\chi_s(t)) = t$.*

Proof: From lemma 4.11 we obtain $\mathsf{sort}(\chi_s(t)) = s$ and $\chi_s(t) \in \mathcal{WT}(\mathcal{F}_\oplus)$. So $\chi_s(t) \in \mathsf{dom}(\tilde{\chi}_s)$. From lemma 4.13 we see that $\tilde{\chi}_s(\chi_s(t)) = \tilde{\chi}(\chi_s(t)) = t$.
□

Claim 2 *For all $t \in \mathcal{WT}^s(\mathcal{F}_\oplus)$: $\chi_s(\tilde{\chi}_s(t)) = t$.*

Proof: (induction to $\mathsf{depth}(t)$)

If $t \in \mathcal{V}_\oplus$ then $t = x_p$ for some $x \in \mathcal{V}$ and $p \in s$, with $\mathsf{sort}(x_p) = \mathsf{st}(x) + p = s$ (because $t \in \mathcal{WT}^s(\mathcal{F}_\oplus)$). Therefore $p = s - \mathsf{st}(x)$. But then we have $t = \chi_s(x)$. Now we can prove:

$$\begin{array}{lll} & \chi_s(\tilde{\chi}_s(t)) & \\ = & \chi_s(\tilde{\chi}_s(\chi_s(x))) & \text{see above} \\ = & \chi_s(x) & \text{from claim 1} \\ = & t & \text{see above} \end{array}$$

If $t = f_p(t_1, \ldots, t_n)$ then $p = s - \mathsf{st}(f)$, because $\mathsf{sort}(t) = \mathsf{st}(f) + p = s$. Furthermore, because t is well-sorted, we have that $\mathsf{ar}(f_p, i) = \mathsf{sort}(t_i)$. With these ingredients, we make the following derivation:

$$\begin{array}{lll} & \chi_s(\tilde{\chi}_s(f_p(t_1, \ldots, t_n))) & \\ = & \chi_s(\tilde{\chi}(f_p(t_1, \ldots, t_n))) & \text{because } \mathsf{sort}(t) = s \\ = & \chi_s(f(\tilde{\chi}(t_1), \ldots, \tilde{\chi}(t_n))) & \text{definition of } \tilde{\chi} \\ = & \chi_s(f(\tilde{\chi}(s_1, t_1), \ldots, \tilde{\chi}(s_n, t_n))) & \\ & \quad \text{for } s_i = \mathsf{sort}(t_i) & \text{definition of } \tilde{\chi}_s \\ = & \psi(p, f(\tilde{\chi}(s_1, t_1), \ldots, \tilde{\chi}(s_n, t_n))) & \text{definition of } \chi_s \\ = & f_p(\psi(p_1, \tilde{\chi}(s_1, t_1)), \ldots, \psi(p_n, \tilde{\chi}(s_n, t_n))) & \\ & \quad \text{for } p_i = p + \mathsf{ar}(f, i) - \mathsf{sort}(\tilde{\chi}(s_i, t_i)) & \text{definition of } \psi_s \\ = & f_p(t_1, \ldots, t_n), & \end{array}$$

because:

$$\begin{array}{lll} & \psi(p_i, \tilde{\chi}(s_i, t_i)) & \\ = & \chi(p_i + \mathsf{sort}(\tilde{\chi}(s_i, t_i)), \tilde{\chi}(s_i, t_i)) & \\ = & \chi(s_i, \tilde{\chi}(s_i, t_i)) & \text{because } \begin{array}{lll} p_i + \mathsf{sort}(\tilde{\chi}(s_i, t_i)) & & \\ = & p + \mathsf{ar}(f, i) & \\ = & \mathsf{ar}(f_p, i) & \\ = & \mathsf{sort}(t_i) & \\ = & s_i & \end{array} \\ = & t_i & \text{by induction hypothesis} \end{array}$$

□

These two claims prove that $\tilde{\chi}_s = \chi_s^{-1}$.
□

The following lemma says us more about the functions ψ_s on substitutions:

Lemma 4.15 *The functions ψ_s are surjective from substitutions $\mathcal{V} \to \mathcal{T}(\mathcal{F})$ onto the set of well-sorted substitutions in $\mathcal{V}_s \to \mathcal{WT}(\mathcal{F}_\oplus)$.*

Proof: Let $\sigma : \mathcal{V}_s \to \mathcal{WT}(\mathcal{F}_\oplus)$ be a well-sorted substitution. Define $x^\rho = \tilde{\chi}(x_s^\sigma)$. We have: $\mathsf{sort}(x_s^\sigma) = \mathsf{st}(x_s) = \mathsf{st}(x) + s$, and $x_s^\sigma \in \mathcal{WT}(\mathcal{F}_\oplus)$. Now we can derive for any $x \in \mathcal{V}$:

$$
\begin{array}{lll}
& \psi_s(\rho)(x) & \\
= & \psi(s + \mathsf{st}(x) - \mathsf{sort}(x^\rho), x^\rho) & \text{definition of } \psi_s(\sigma). \\
= & \chi(s + \mathsf{st}(x), x^\rho) & \text{definition of } \chi_s. \\
= & \chi(s + \mathsf{st}(x), \tilde{\chi}(x_s^\sigma)) & \text{definition of } x^\rho. \\
= & \chi(s + \mathsf{st}(x), \chi^{-1}(\mathsf{sort}(x_s^\sigma), x_s^\sigma)) & \text{definition of } \chi_s^{-1}. \\
= & x_s^\sigma & \text{by lemma 4.14.}
\end{array}
$$

So we can find a $\rho : \mathcal{V} \to \mathcal{T}(\mathcal{F})$ such that $\psi_s(\rho) = \sigma$, for any well-sorted substitution $\sigma : \mathcal{V}_s \to \mathcal{WT}(\mathcal{F}_\oplus)$, and hence ψ_s is surjective onto these well-sorted substitutions.
□

Now we can prove the goal lemmas of this section:

Lemma 4.16 *For $t, t' \in \mathcal{T}(\mathcal{F})$ holds: if $t \to_\mathcal{R} t'$ then $\chi(s,t) \to_{\mathcal{R},\oplus} \chi(s,t')$*

Proof: (induction to the structure of the proof that $t \to_\mathcal{R} t'$)

If $t = l^\sigma$ and $t' = r^\sigma$ for some substitution $\sigma : \mathcal{V} \to \mathcal{T}(\mathcal{F})$ and rule $l \to r \in \mathcal{R}$: Let $p = \mathsf{sort}(l) = \mathsf{sort}(r)$, then:

$$
\begin{array}{lll}
& \chi(s, l^\sigma) & \\
= & (\phi(p,l))^{\psi(p,\sigma)} & \text{by lemma 4.12} \\
\to_{\mathcal{R},\oplus} & (\phi(p,r))^{\psi(p,\sigma)} & \text{because } \phi(p,l) \to \phi(p,r) \in \mathcal{R}_\oplus \text{ for all } k \\
= & \chi(s, r^\sigma) & \text{by lemma 4.12}
\end{array}
$$

If $f(t_1, \ldots, t_j, \ldots, t_n) \to_\mathcal{R} f(t_1, \ldots, t'_j, \ldots, t_n)$, with $u = t_j \to_\mathcal{R} t'_j = u'$ then we have:

$$
\begin{array}{lll}
& \chi(s, f(t_1, .., u, .., t_n)) & \\
= & \psi(s - \mathsf{st}(f), f(t_1, .., u, .., t_n)) & \text{by the definition of } \chi \\
= & f_{s-\mathsf{st}(f)}\Big(\psi(s_1, t_1), \ldots, \psi(s_j, u), \ldots, \psi(s_n, t_n)\Big) & \\
& \text{where } s_i = s - \mathsf{st}(f) + \mathsf{ar}(f,i) - \mathsf{sort}(t_i) & \text{by the definition of } \psi
\end{array}
$$

Claim 1 *Let $s'_j = s - \mathsf{st}(f) + \mathsf{ar}(f,i) - \mathsf{sort}(u')$. Then $\psi(s_j, u) \to_{\mathcal{R},\oplus} \psi(s'_j, u')$.*

Proof: This can be proved straightforward:

$$
\begin{array}{lll}
& \psi(s_j, u) & \\
= & \psi(s - \mathsf{st}(f) + \mathsf{ar}(f,j) - \mathsf{sort}(u), u) & \\
= & \chi(s - \mathsf{st}(f) + \mathsf{ar}(f,j), u) & \text{by the definition of } \chi \\
\to_{\mathcal{R},\oplus} & \chi(s - \mathsf{st}(f) + \mathsf{ar}(f,j), u') & \text{ih} \\
= & \psi(s - \mathsf{st}(f) + \mathsf{ar}(f,i) - \mathsf{sort}(u'), u') & \text{by the definition of } \chi \\
= & \psi(s'_j, u') &
\end{array}
$$

□

Resuming the first computation we get:

$$
\begin{array}{lll}
 & f_{s-\mathsf{st}(f)}\Big(\psi(s_1,t_1),\ldots,\psi(s_j,u),\ldots,\psi(s_n,t_n)\Big) & \\
\rightarrow_{\mathcal{R},\oplus} & f_{s-\mathsf{st}(f)}\Big(\psi(s_1,t_1),\ldots,\psi(s'_j,u'),\ldots,\psi(s_n,t_n)\Big) & \\
 & \text{where } s'_j = s - \mathsf{st}(f) + \mathsf{ar}(f,i) - \mathsf{sort}(u') & \text{see the claim} \\
= & \psi(s-\mathsf{st}(f), f(t_1,\ldots,u',\ldots,t_n)) & \text{by the definition of } \psi \\
= & \chi(s, f(t_1,\ldots,u',\ldots,t_n)) & \text{by the definition of } \chi
\end{array}
$$

□

Lemma 4.17 *For all $t, t' \in \mathcal{T}(\mathcal{F})$: If $\chi(s,t) \rightarrow_{\mathcal{R},\oplus} \chi(s,t')$ then $t \rightarrow_{\mathcal{R}} t'$.*

Proof: The proof is analogous to the proof of lemma 4.6: Induction to the length of the derivation of the reduction step in $\mathcal{R}_\oplus$.

basis If $\chi(s,t) = l^\sigma$ and $\chi(s,t') = r^\sigma$ for some $l \rightarrow r \in \mathcal{R}_\oplus$, then $l = \phi(p,a)$ and $r = \phi(p,b)$ with $a \rightarrow b \in \mathcal{R}$. and some $p \in S$ (by the definition of $\mathcal{R}_\oplus$). Then

$$
\begin{array}{lll}
 & s & \\
= & \mathsf{sort}(\chi(s,t)) & \text{by lemma 4.11} \\
= & \mathsf{sort}(l^\sigma) & \\
= & \mathsf{sort}(l) & \text{by proposition 3.2} \\
= & \mathsf{sort}(\phi(p,a)) & \\
= & \mathsf{sort}(a) + p & \text{by lemma 4.1}
\end{array}
$$

So $p = s - \mathsf{sort}(a)$. Furthermore $\mathsf{sort}(l^\sigma) = \mathsf{sort}(r^\sigma)$ by proposition 3.6 and $\sigma \restriction \mathsf{Var}(l)$ is well-sorted by proposition 3.3. Now we have:

$$
\begin{array}{lll}
 & \chi(s,t) & \\
= & (\phi(p,a))^\sigma & \\
= & (\phi(p,a))^{\psi(p,\rho)} & \text{by lemma 4.15} \\
 & \quad \text{for some } \rho & \\
= & \chi(s, a^\rho) & \text{by lemma 4.12}
\end{array}
$$

Because χ_s is an injective function, we can conclude that $t = a^\rho$. In the same way we obtain that $t' = b^\rho$. But then we have that $t \rightarrow_{\mathcal{R}} t'$, because $a \rightarrow b \in \mathcal{R}$.

induction step If:

- $\chi(s,t) = k = f_p(k_1,..,k_j,..,k_n)$,
- $\chi(s,t') = k' = f_p(k_1,..,k'_j,..,k_n)$ and
- $k_j \rightarrow_{\mathcal{R},\oplus} k'_j$

From proposition 3.6 we obtain that $\mathsf{sort}(k_j) = \mathsf{sort}(k'_j)$. Define $s_i = \mathsf{sort}(k_i)$, then $\mathsf{sort}(k'_j) = s_j$. From induction hypothesis, we get that $\chi^{-1}(s_j,k_j) \rightarrow_{\mathcal{R}} \chi^{-1}(s_j,k'_j)$. Now we can derive:

$$
\begin{array}{lll}
 & t & \\
= & \chi^{-1}(s,k) & \text{see lemma 4.14} \\
= & f(\chi^{-1}(s_1,k_1),..,\chi^{-1}(s_j,k_j),..,\chi^{-1}(s_n,k_n)) & \text{via the definition of } \tilde{\chi}. \\
 & \quad \text{where } s_i = \mathsf{sort}(k_i) & \\
\to_{\mathcal{R}} & f(\chi^{-1}(s_1,k_1),..,\chi^{-1}(s_j,k'_j),..,\chi^{-1}(s_n,k_n)) & \text{by induction hypothesis} \\
= & \chi^{-1}(s, f(k_1,..,k'_j,..,k_n)) & \\
= & \chi^{-1}(s,k') & \text{again via } \tilde{\chi} \\
= & t' &
\end{array}
$$

□

Now we are ready to formulate that S-sorted rewriting in the direct sum $M_\oplus$ is "isomorphic" with rewriting in the TRS M (step 3 of the overview at the beginning of this chapter):

Proposition 4.18 *There exist a bijection $\chi_s : \mathcal{T}(\mathcal{F}) \to \mathcal{WT}^s(\mathcal{F}_\oplus)$ (lemma 4.14) with for all $t, t' \in \mathcal{T}(\mathcal{F})$:*

$$t \to_{\mathcal{R}} t' \iff \chi_s(t) \to_{\mathcal{R},\oplus} \chi_s(t')$$

(lemma 4.16 and 4.17).

4.3 Persistence follows from Many-sorted Modularity

In the previous two sections we saw isomorphisms between rewrite relations of term rewriting systems. In this section we will make conclusions about the properties of these systems. From proposition 4.7 and 4.18 we got the following equivalences:

$$\forall t, t' \in \mathcal{WT}(\mathcal{F}) : t \to_{\mathcal{R}} t' \iff \phi_s(t) \to_{\mathcal{R},s} \phi_s(t')$$

$$\forall t, t' \in \mathcal{T}(\mathcal{F}) : t \to_{\Theta(R)} t' \iff \chi_s(t) \to_{R,\oplus} \chi_s(t').$$

Here $\phi_s : \mathcal{WT}(\mathcal{F}) \to \mathcal{WT}(\mathcal{F}_s)$ and $\chi_s : \mathcal{T}(\mathcal{F}) \to \mathcal{WT}^s(\mathcal{F}_\oplus)$ are bijections (lemma 4.2 and 4.14).

Proposition 4.19 *For reduction properties $\mathcal{P}$ on term rewriting systems and $s \in S$ holds: $\mathcal{P}(M) \iff \mathcal{P}(M_s)$.*

Proof: From the observations above, we can conclude immediately that the rewrite relations $\to_{\mathcal{R}}$ and $\to_{\mathcal{R},s}$ have the same properties. So the term rewriting systems M and M_s share the properties that can be defined for abstract reduction systems (see section 2.3.1). These are precisely the reduction properties.
□

Examples of reduction properties are CR, SN, WCR, WN, UN, $\text{UN}^{\to}$ and NF.

Proposition 4.20 *For component closed reduction properties $\mathcal{P}$ on term rewriting systems holds:* $\mathcal{P}(\Theta(M)) \iff \mathcal{P}(\bigoplus_{s\in S} M_s)$.

Proof: The rewrite relation in the MTRS $(\mathcal{F}_\oplus, \mathcal{R}_\oplus, \mathcal{WT})$ consists of the union of the rewrite relations for each $(\mathcal{F}_\oplus, \mathcal{R}_\oplus, \mathcal{WT}^s)$. Because terms of sort s don't rewrite to terms of other sorts (proposition 3.6), the equivalence classes of the former are the union of the equivalence classes of the latter. Because the property $\mathcal{P}$ is component closed, we have for any $s \in S$ that $\mathcal{P}(\mathcal{F}_\oplus, \mathcal{R}_\oplus, \mathcal{WT}) \iff \mathcal{P}(\mathcal{F}_\oplus, \mathcal{R}_\oplus, \mathcal{WT}^s)$. The latter is equivalent to $\mathcal{P}(\mathcal{F}, \mathcal{R}, \mathcal{T})$ due to the similarity in the rewrite relations shown in proposition 4.18.
□

Proposition 4.21 *Let M be an S-sorted MTRS, with S a finite set of sorts and let $\mathcal{P}$ be an MS-modular reduction property. then* $(\forall s \in S : \mathcal{P}(M_s)) \iff \mathcal{P}(\bigoplus_{s\in S} M_s)$.

Proof: Because taking the disjoint union is associative and the disjoint union of two MTRS's is an MTRS, we can apply the modularity of $\mathcal{P}$ $|S|$ times. Here the finiteness of S is used.
□

Let $\mathcal{P}$ be a component closed property and let M be an S-sorted MTRS. Furthermore assume that $\mathcal{P}$ is an MS-modular property for MTRS's. Then we have achieved:

$\mathcal{P}$ holds for M.
$\iff$ (by proposition 4.19).
$\mathcal{P}$ holds for each M_s.
$\iff$ (by proposition 4.21 and MS-modularity).
$\mathcal{P}$ holds for $M_\oplus$.
$\iff$ (because $\mathcal{P}$ is component closed).
$\mathcal{P}$ holds for $M_\oplus$, restricted to sort s.
$\iff$ (by lemma 4.20).
$\mathcal{P}$ holds for $\Theta(M)$.

So we have proved that $\mathcal{P}$ holds for a many-sorted MTRS if and only if $\mathcal{P}$ holds for the unsorted version of it, under the assumption of MS-modularity and component closedness of $\mathcal{P}$ and the finiteness of S. In other words:

Theorem 4.22 *A component closed and MS-modular property is persistent for finitely-sorted term rewriting systems.*

Together with the result of theorem 3.10 we get:

Theorem 4.23 *For component closed properties $\mathcal{P}$ of many-sorted term rewriting systems with finitely many sorts holds:*

$\boxed{\mathcal{P} \text{ is persistent if and only if } \mathcal{P} \text{ is MS-Modular.}}$

4.4 Further Remarks

In this section two items are discussed: The generalization of the result of theorem 4.23 to an infinite set of sorts and the generalization to some syntactic properties.

Theorem 4.23 only holds for a finite set of sorts. This restriction was used in lemma 4.21 only, to establish modularity for the direct sum of more than two MTRS's. If the number of sorts is finite, we can generalize the modularity for the direct sum of two MTRS's to the direct sum $\bigoplus_{s\in S} M_s$ by induction to the number of sorts. Note that the disjoint union is associative. But what about the direct sum of infinitely many MTRS's?

Note that in every reduction sequence, only function symbols from finitely many MTRS's play a role. The reason is, that all the rules originates from exactly one MTRS, so we can't introduce new symbols. So if we restrict to properties that are not only component closed, but also *reduction tree closed* we can generalize the modularity result to the direct sum of an infinite set of terms. It is easy to see that SN, WN, CR, WCR and $\text{UN}^{\rightarrow}$ only depend on the reduction trees of all terms. But the property UN depends on a whole equivalence class. And it is possible to have an equivalence class in which infinitely many MTRS's play a role, as the following example shows:

Example 6 Let $M_i = \mathsf{f}_i(x) \to \mathsf{x}$ for every $i \in \mathbf{N}$. Then in the direct sum $\bigoplus_{i\in\mathbf{N}} M_i$ all $\mathsf{f}_i(\mathsf{b})$ are convertible, because $\mathsf{f}_i(\mathsf{b}) \to \mathsf{b} \leftarrow \mathsf{f}_j(\mathsf{b})$ is a conversion in this direct sum.

. . . . . .

Lemmas 4.7 and 4.18 don't depend on the finiteness of the set of sorts, but they depend on the existence of the operators + and − with the right properties. This leads to the following theorem:

Theorem 4.24 *Let S be a set of sorts, for which and a binary operator $+$ can be found in such a way that $(S, +)$ forms an abelian group. Let $\mathcal{P}$ be a reduction tree closed property on MTRS's, that is MS-modular. Then $\mathcal{P}$ is persistent for S-sorted MTRS's.*

Until now, we have only thought about reduction properties. But from the construction of the MTRS's M_s (making copies) it is clear that the syntactic properties that are defined in section 2.3.2 are maintained. Also, taking the direct sum does not disturb these properties. So the original MTRS M has a collapse rule if and only if all MTRS's M_s have a collapse rule. The latter

holds if and only if the direct sum of these MTRS's has a collapse rule. The same holds for a non-left-linear rule or a duplicating rule and for other well-known syntactic properties as *overlapping redexes* and *non-ambiguous rules*.

So we can extend our result with: If $\mathcal{P}$ is a modular, component closed property for collapse free (left-linear, duplicate free, etc) many-sorted term rewriting systems, then it is persistent for collapse free (left-linear, duplicate free, etc) many-sorted term rewriting systems.

This is a useful generalization, because the modularity of strong normalization is restricted to combinations of collapse free and duplicate free TRS's [Rus87].

5 Conclusions and Questions

In section 3.4 a convenient definition of the direct sum of many-sorted term rewriting systems has been given. With this definition the direct sum of two MTRS's yields an MTRS again. This led to the definition of modularity for MTRS's, MS-modularity for short, in section 3.4.

In section 4.3 and 3.5 we proved that for component closed properties and many-sorted term rewriting systems with a finite set of Sorts, persistence is equivalent to modularity. We have also seen that we can generalize to infinite sets of sorts, if we restrict to reduction tree closed properties. Also some important syntactic properties are maintained by the constructions of the proof, so we are able to restrict to left-linear, collapse free or duplicate free MTRS's.

Proving that confluence is a persistent property, only depends on checking Toyama's proof of the modularity of confluence in [KMTdV91]. As a conjecture we state that *confluence is a persistent property of many-sorted term rewriting systems.* This statement gives a strong proof technique for the confluence of concrete TRS's. The following TRS, due to Toyama, can be found in [Zan91, 629]:

$$R = \begin{cases} \mathsf{a}(\mathsf{x}, y) & \to & \mathsf{a}(\mathsf{f}(\mathsf{x}), \mathsf{f}(\mathsf{x})) \\ \mathsf{f}(\mathsf{x}) & \to & \mathsf{g}(\mathsf{x}) \\ \mathsf{b}(\mathsf{f}(\mathsf{x}), \mathsf{x}) & \to & \mathsf{b}(\mathsf{x}, \mathsf{f}(\mathsf{x})) \\ \mathsf{b}(\mathsf{g}(\mathsf{x}), \mathsf{x}) & \to & \mathsf{b}(\mathsf{x}, \mathsf{g}(\mathsf{x})) \end{cases}$$

Proving the confluence of this TRS is not easy, but we can assign the following compatible {1,2,3}-sorted signature to it:

$$\begin{array}{lcl} \mathsf{f} & : & 1 \to 1 \\ \mathsf{g} & : & 1 \to 1 \\ \mathsf{a} & : & 1 \times 1 \to 2 \\ \mathsf{b} & : & 1 \times 1 \to 3 \end{array}$$

The new MTRS is provably confluent, with standard techniques: Much fewer terms have to be taken into account. With the persistence of confluence, we obtain that the original TRS R is confluent.

There are some remaining questions. One of them is whether properties are modular for TRS's if and only if they are MS-modular for MTRS's. This can be answered by "type-checking" the existing modularity proofs, but perhaps a more general proof can be given. The notion of a well-sorted substitution could be used to give a definition of well-sorted rewriting, without using unsorted rewriting.

Another issue is, whether the notion of persistency can be generalized for order-sorted term rewriting systems [GJM85]. A related question is if the proof as given in this thesis could be generalized for the order-sorted case.

References

[GJM85] J.A. Goguen, J.-P. Jouannaud, and J. Meseguer. Operational semantics of order-sorted algebra. In *Proceedings of the 12th International Colloquium on Automata, Languages and Programming*, volume 194 of *Lecture Notes in Computer Science*, pages 221–231, Nafplion, 1985.

[KK88] M. Kurihara and I. Kaji. Modular term rewriting systems: Termination, confluence and strategies. Report, Hokkaido University, Sapporo, 1988.

[KMTdV91] J.W. Klop, A. Middeldorp, Y. Toyama, and R. de Vrijer. A simplified proof of toyama's theorem. Technical Report IR-270, Vrije Universiteit Amsterdam, dec 1991.

[Mid89] A. Middeldorp. A sufficient condition for the termination of the direct sum of term rewriting systems. In *Proceedings of the 4th IEEE Symposium on Logic in Computer Science*, pages 396–401, Pacific Grove, 1989.

[Mid90] A. Middeldorp. *Modular Properties of Term Rewriting Systems*. PhD thesis, Vrije Universiteit te Amsterdam, Amsterdam, 1990.

[Rus87] M. Rusinowitch. On termination of the direct sum of term rewriting systems. *Information Processing Letters*, 26:65–70, 1987.

[SS87] M. Schmidt-Schauß. *Computational Aspects of an Order-Sorted Logic with Term Declarations*, volume 395 of *Lecture Notes in Artificial Intelligence*. Springer-Verlag, 1987. subseries of LNCS.

[Toy87a] Y. Toyama. Counterexamples to termination for the direct sum of term rewriting systems. *Information Processing Letters*, 25:141–143, 1987.

[Toy87b] Y. Toyama. On the Church-Rosser property for the direct sum of term rewriting systems. *Journal of the ACM*, 34(1):128–143, 1987.

[Zan91] H. Zantema. Termination of term rewriting: from many-sorted to one-sorted. In J. van Leeuwen, editor, *Computing Science in The Netherlands*, volume 2 of *CSN proceedings*, pages 617–629. SION, CSN'91, nov 1991.

[Zan92] H. Zantema. Type removal in term rewriting. In M. Rusinowitch and J.L. Rémy, editors, *CTRS 92 (Extended Abstracts)*,

Conditional Term Rewriting Systems, pages 86–90, Pont-à-Mousson, july 1992. Centre de Recherche en Informatique de Nancy. To be published in the Springer-Verlag collection.