Equational Reasoning with Non-Monotonic Orderings

Tesi doctoral presentada al
Departament de Llenguatges i Sistemes Informàtics
de la Universitat Politècnica de Catalunya

per a optar al grau de
Doctor en Informàtica

per
Miquel Bofill Arasa

sota la direcció del doctor
Albert Rubio

Barcelona, 6 de juny de 2004

a l'Alba

Agraïments

El meu principal agraïment, evidentment, és per a l'Albert. En primer lloc, per haver acceptat de dirigir-me la tesi. En segon lloc, per haver-me tractat més com a un amic que no pas com a un doctorand, pel seu optimisme encomanadís, i per la seva total disponibilitat al llarg d'aquests anys. El seu grau d'implicació no podia haver estat major. Agraeixo també molt especialment a en Roberto la seva col·laboració desinteressada i els seus savis consells. I a en Guille, el seu mestratge i la seva amistat. A tots tres moltes gràcies pel que m'han ensenyat.

Agraeixo a en Mateu les bones estones que hem passat junts i el seu suport en les teràpies de grup. A la Cristina, el seu ajut i suport incondicional en moltíssims aspectes al llarg d'aquests anys. A en Joan, les bones estones de conversa que hem passat. I a l'Alba, especialment, tot el seu amor i el seu esforç per ajudar-me fins més enllà de les seves possibilitats.

Als familiars, amics i companys de feina, gràcies per donar-me suport. Agraeixo especialment el seu recolzament a l'Àngel, l'Ascensió, en Xavi, l'Imma, els Jordis, en Raymond, en Joan Carles (gràcies pel consell de la llibreta), en Robert, la Montse, en Miquel, en Jaume, l'Horacio, en Marc, la Maria, en Joaquim, la Llúcia i la Puri. Voldria donar també les gràcies a altres persones que no he citat, i que possiblement m'hagin donat l'oportunitat, indirectament, de desenvolupar aquesta tesi.

Finalment agrair a Larry Wos l'amabilitat d'explicar-me per què va escollir les paraules demodulació i paramodulació per anomenar les seves regles d'inferència, ara fa 35 anys.

Aquesta tesi ha estat parcialment finançada per l'ESPRIT Basic Research Action CCL-II, ref. WG # 22457 i pels projectes CICYT HEMOSS ref. TIC98-0949-C02-01, MAVERISH ref. TIC2001-2476-C03-01, i CADVIAL ref. TIC2001-2392-C03-01.

Acknowledgments

My main gratitude is obviously addressed to Albert. First of all, for accepting me as a PhD student. And for treating me as a friend rather than as a student, for his infectious optimism and his full availability along these years. His involvement could not have been greater. I am also especially grateful to Roberto for his unselfishness help and his wise advice. And to Guille for his teaching and friendship. I would like to thank all three of them for all that they have taught me.

I am grateful to Mateu for the good times we have had together and his support in our group therapies. To Cristina, for her help and her unconditional support in many aspects along these years. To Joan, for the pleasant chats we have had. And especially to Alba, for all her love and her effort to help me at her maximum.

I want to thank the members of my family, friends and colleagues for supporting me. I am especially grateful to Àngel, Ascensió, Xavi, Imma, the Jordis, Raymond, Joan Carles (thank you for the piece of advice about the notebook), Robert, Montse, Miquel, Jaume, Horacio, Marc, Maria, Joaquim, Llúcia and Puri. I also would like to thank the people I have not cited and that possibly have indirectly given me the opportunity to develop this thesis.

Finally, let me thank Larry Wos for the kindness of telling me the reason why he chose, 35 years ago, the words demodulation and paramodulation to name his inference rules.

This thesis has been partially supported by the ESPRIT Basic Research Action CCL-II, ref. WG # 22457 and the spanish CICYT projects HEMOSS ref. TIC98-0949-C02-01, MAVERISH ref. TIC2001-2476-C03-01, and CADVIAL ref. TIC2001-2392-C03-01.

Contents

Chapter 1

Introduction

Reasoning with equality is of great importance in mathematics and logic. Therefore, it is not surprising that automated reasoning techniques for first-order logic with equality have applications in many areas of computer science, which include, among others, verification and synthesis of hardware and software, security of communications, database query processing and integrity checking, knowledge-based systems, and (constraint) logic or functional programming.

During the last two decades, automated equational reasoning has importantly progressed through new Knuth-Bendix-like completion techniques and their extensions to ordered paramodulation for first-order clauses. At the present time most, if not all, state-of-the-art theorem provers for first-order logic with equality are based on the *ordered paramodulation* inference rule, that is parametrized by an ordering on terms. In this context, powerful redundancy notions, which include, for instance, tautology deletion, subsumption, or demodulation (i.e., simplification by rewriting), can be applied without losing refutation completeness.

In this thesis, a detailed study of the ordered paramodulation inference rule is made. To this end, two aspects are considered:

- which properties of the ordering used to paramodulate with are essential for ensuring refutation completeness of an ordered paramodulation based inference system, and
- what kind of redundancy notions are then compatible with refutation completeness in the resulting calculus.

In this introduction, based on [BG98a, NR01, DV01], we first shortly overview the research area and some aspects of its rapid development during the last years. We mainly concentrate on refinements of paramodulation, that have yielded the most satisfactory results to date in automated equational reasoning. After this, we give an outline of the structure and contents of this thesis.

1.1 Equational reasoning in resolution-based calculi

Finding general procedures for proving theorems has been a challenge for many scientists along the history. With the invention of the computer, the interest in automated deduction grew considerably among mathematicians, logicians and computer scientists, and the need of reasonably efficient procedures, and simple enough to be implemented, became evident. The first significant result in this direction was achieved by J.A. Robinson in 1965 [Rob65], who defined a semi-decision procedure for automated theorem proving in first-order clause logic, based on the resolution principle. This is a *refutation* procedure: to prove the validity of a formula, what is proved is the *unsatisfiability* (or inconsistency) of its negation, represented by an equivalent, with respect to unsatisfiability, set of clauses. Robinson showed that by systematically enumerating all consequences of an unsatisfiable set of clauses by *(binary) resolution*

$$\frac{A \vee C \qquad \neg A' \vee D}{(C \vee D)\sigma} \quad \text{if } \sigma = mgu(A, A')$$

where $mgu(A,A')$ denotes the *most general unifier* of A and A', and *factoring*

$$\frac{A \vee A' \vee C}{(A \vee C)\sigma} \quad \text{if } \sigma = mgu(A, A')$$

the empty clause will eventually be inferred, i.e., resolution combined with factoring is *refutation complete*. It must be noted that the procedure may not halt if the set of clauses is satisfiable, since satisfiability of a set of clauses in first-order logic is an undecidable property.

Herbrand's theorem states that for every unsatisfiable set of non-ground clauses there is a finite set of ground instances that is propositionally unsatisfiable. Hence, it establishes a link between propositional clauses and general clauses with variables.

But resolution is not primarily a method for deciding the unsatisfiability of propositional formulas (the procedure by Davis and Putnam [DP60] is better suited for that purpose). Its main advantage over other early theorem proving methods, is that unification, as a selection mechanism for inferences, provides an effective way of interleaving the two processes of (i) the identification of suitable (ground) instances of clauses and (ii) a demonstration of their unsatisfiability.

Many efforts have been made to improve the efficiency of this procedure, by adding some control to the application of the rule or by improving the rule itself (see [BG01a]). Some successful approaches, although sometimes by restricting the logical language (for instance allowing only *Horn clauses*) or at the expense of losing completeness, have led to an increasing number of applications of automated theorem proving in artificial intelligence and many other areas of computer science such as program verification and synthesis, database query processing and integrity constraint checking, or as a basis for logic programming. For example, in integrity constraint checking, we are interested in knowing whether a given specification (a set of clauses) S is consistent. In other cases, like in program verification, logic programming or database query processing, we may be interested in proving that a certain theorem T is a logical consequence of S, i.e., $S \cup \neg T$ is inconsistent, sometimes requiring solutions for the existentially quantified variables.

1.1.1 Axiomatization of equality

The necessity of handling equality in automated reasoning has been recognized already in the very early papers in the area [Wan60, Kan63, WRCS67, Rob68, Dar68, RW69].

Equality is a relation which appears in many domains and, hence, it appears in many specifications. For example, common data structures such as lists and stacks can often be described by sets of equations. But resolution itself provides no way of using sentences involving the equality relation. For example, given a set of clauses consisting of only $P(a)$ and $a \simeq b$, it cannot prove $P(b)$. This is because, as the set of clauses is concerned, the relation $\simeq$ is arbitrary. To give $\simeq$ its standard interpretation requires additional axiom schemata. One can specify the equality predicate $\simeq$ in first-order logic by means of the following *congruence axioms* $\mathcal{E}$:

$$\begin{aligned}
&\rightarrow x \simeq x && \textit{(reflexivity)}\\
&x \simeq y \rightarrow y \simeq x && \textit{(symmetry)}\\
&x \simeq y \wedge y \simeq z \rightarrow x \simeq z && \textit{(transitivity)}\\
&x_1 \simeq y_1 \wedge \ldots \wedge x_n \simeq y_n \rightarrow f(x_1,\ldots,x_n) \simeq f(y_1,\ldots,y_n) && \textit{(monotonicity-I)}\\
&x_1 \simeq y_1 \wedge \ldots \wedge x_n \simeq y_n \wedge P(x_1,\ldots,x_n) \rightarrow P(y_1,\ldots,y_n) && \textit{(monotonicity-II)}
\end{aligned}$$

The monotonicity axioms are in fact axiom schemes: one *monotonicity-I* axiom is required for each n-ary function symbol f, and one *monotonicity-II* axiom is required for each n-ary predicate symbol P. A set S of clauses is satisfiable in first-order logic (FOL) with equality if, and only if, $S \cup \mathcal{E}$ is satisfiable in FOL without equality. (FOL *with equality* means with equality being built-in, that is, having a predefined interpretation where $a \simeq b$ is true whenever a and b correspond to the same element in the domain of the interpretation.)

The *monotonicity-II* axioms are obviously not needed for equality clauses, that is, when equality is the only predicate. Also, the symmetry and transitivity axioms may be replaced by a single *commutation axiom*

$$x \simeq y \wedge x \simeq z \rightarrow y \simeq z$$

as proposed by Brand [Bra75]. And, as a matter of fact, the *monotonicity-I* axioms are not needed for Skolem functions, as they may be added to the original formulae, before any clause transformations (including skolemization) are applied.

However, even with these improvements, this approach has important deficiencies. It notably increases the number of clauses in the specification, and it also tends to cause the generation of many useless consequences.

1.1.2 Built-in equality: paramodulation

Due to the combinatorial explosion caused by the extensive applicability of equality axioms, even the early methods of handling equality in automated reasoning tried to avoid the necessity of using them. George Robinson (no relation with J.A.) and Larry Wos explored the possibility to avoid the necessity of specifying equality, by treating it as part of the logical language, i.e., directly considering FOL with equality. This required the design of dedicated inference rules for it, such as *paramodulation* [RW69]:

$$\frac{C' \vee (e \simeq e') \qquad C}{(C' \vee C[e']_p)\sigma} \quad \text{if } \sigma = mgu(e, C|_p)$$

where $C|_p$ is the subterm at position p of C, and $C[e']_p$ denotes the result of replacing this subterm in C by e'. The conclusion results from replacing an occurrence of $e\sigma$ in the right premise by the term $e'\sigma$ and adding "side conditions" $C'\sigma$. In descriptions of paramodulation calculi one usually identifies the symmetric equations $e \simeq e'$ and $e' \simeq e$, so that one may also replace $e'\sigma$ by $e\sigma$.

Example 1.1.1 *The following is an inference by paramodulation*

$$\frac{P(x) \vee f(g(y),x) \simeq y \qquad Q(f(z,h(z)),z)}{P(h(g(y))) \vee Q(y,g(y))}$$

since $\sigma = mgu(f(g(y),x), f(z,h(z))) = \{z \mapsto g(y), x \mapsto h(g(y))\}$.

Equality resolution is another inference rule, which encodes resolution with the reflexivity axiom $x \simeq x$:

$$\frac{C \vee (e \not\simeq e')}{C\sigma} \quad \text{if } \sigma = mgu(e, e')$$

One of the aims of Robinson and Wos [RW69] was to design a more "immediate" inference system that combined several resolution steps involving congruence axioms, but ignored the intermediate results, so as to achieve better "convergence" of the deduction process (meaning fewer derived clauses).

But paramodulation was not the first inference rule that was formulated by Robinson and Wos in their way to a better inference system. *Demodulation* [WRCS67] is a related former rule. It has the following restrictions with respect to paramodulation: first, the equality clause must be unit (i.e., C' must be empty); second, *matching* is required instead of full unification (i.e., $C|_p = e\sigma$). On the other hand, it has the side effect that the right premise is deleted. Demodulation is applied when the term $e'\sigma$ is in some sense simpler (e.g., smaller) than $e\sigma$, and hence it encodes the idea of simplification by rewriting, which is the basis for the semantics of functional programming, and is of fundamental importance in automated theorem proving. In fact, without a method of simplifying terms, a theorem prover becomes bogged down with many equivalent terms. But demodulation is an incomplete inference rule. Hence, paramodulation was formulated later with the aim of obtaining a refutation complete inference system. In fact, demodulation is used nowadays as a simplification method within paramodulation-based systems, rather than as an inference rule on its own.

The following is a personal communication from Larry Wos, as an answer to a question about the naming of demodulation and paramodulation:

> After reading 2000 clauses with the conjecture that many were somewhat trivial, in the sense that they contained something like $0+a$ as a subterm, and discovering that such was not the case, I made a second conjecture. I conjectured that, although not very many clauses were themselves trivial, those few trivial clauses spawned many clauses. In other words, if trivial clauses were not kept, then far fewer than 2000 clauses would have been kept, and the small theorem under study would have been proved. Such turned out to be the case; more than 1900 clauses would be avoided if the so-called trivial clauses had not been kept to be parents, grandparents, and so forth. Demodulation thus was born: simply include equations to simplify and canonicalize potentially trivial clauses. I needed a name for the procedure. At the time, the style was that of mathematics, quite formal; therefore, a formal name was in order. So, since modifying was what was being done, and to "modulate" a sound, so it seemed to me at the time, meant (mainly) to modify it, demodulation occurred to me. As for paramodulation, one might wonder why I simply did not call the procedure equality-oriented reasoning. But such a term would have been incorrect. The reason rests with the fact that paramodulation permits nontrivial substitution in both the *from* and the *into* items. Why choose the term paramodulation? If one considers paramodulation in the context of demodulation, one might decide that the former in a certain sense generalizes the latter. Indeed, demodulation permits nontrivial substitution in the *from* but not in the *into* item. But paramodulation permits both, generalizes equality substitution in this sense. Therefore, it occurred to me to connect the two terms demodulation and paramodulation. The "para" may have come to me in the fashion of para-psychology.

Paramodulation, together with equality resolution and factoring, was proved to be refutation complete [RW69] for first-order equational clauses under the presence of a *functional reflexivity equation*

$$f(x_1,\ldots,x_n) \simeq f(x_1,\ldots,x_n)$$

for every n-ary function symbol f of the alphabet.

Unfortunately, the claims for fewer derived clauses with respect to resolution in the presence of the equality congruence axioms are unfounded for unrestricted paramodulation. Indeed, paramodulation in the presence of the functional reflexivity equations is actually a *much* more prolific inference mechanism than (hyper-)resolution in the presence of the congruence axioms: in particular, paramodulation inferences into proper subterms of input clauses actually represent *additional* inferences with no analogous resolution inferences [BG98a].

The technical reason for introducing the functional reflexivity equations is that they enable lifting. The *lifting* argument, which is used in the completeness proof, roughly states that for every clause C' that can be derived from the set of ground instances S' of a set of clauses S, there is a clause C that can be derived from S such that C' is an instance of C. For resolution lifting is straightforward: if C' and D' are ground instances of C and D, respectively, then any resolvent G' that can be obtained from C' and D' is an instance of the corresponding resolvent G from C and D. That is, we may either first instantiate C and D and then resolve; or else first resolve C and D, and the instantiate the conclusion of the inference. This commutation property does not hold for paramodulation. Consider the inference

$$\frac{a \simeq b \qquad g(f(a),f(a)) \simeq c}{g(f(a),f(b)) \simeq c}$$

where the right premise is an instance of $g(x,x) \simeq c$ under the substitution $\{x \mapsto f(a)\}$. There is no paramodulation inference from $a \simeq b$ and $g(x,x) \simeq c$ from which $g(f(a),f(b)) \simeq c$ can be obtained by instantiation: from these two clauses at not ground level we can only obtain $g(a,b) \simeq c$ and $g(b,a) \simeq c$. The problem is that the ground inference replaces a subterm in the "substitution part" of the instantiated clause $(g(x,x) \simeq c)\{x \mapsto f(a)\}$. The functional reflexivity equations allow further instantiation of clauses, and then lifting is possible by two consecutive paramodulations *into variables*:

$$\frac{\dfrac{a \simeq b \quad f(x) \simeq f(x)}{f(a) \simeq f(b)} \qquad g(x,x) \simeq c}{g(f(a),f(b)) \simeq c}$$

The functional reflexivity equations are only useful for paramodulation *into* variables. However, this kind of paramodulation, as well as paramodulation *from* variables, was already avoided in the early paramodulation-based theorem provers (although it was not clear whether this compromised the completeness of the calculus), since such inferences are not constrained by the unification (that is, any term unifies with a variable) and hence are difficult to control. Moreover, the functional reflexivity equations are instances of, and hence are subsumed by, the reflexivity axiom $x \simeq x$. If they were really needed, the paramodulation calculus would not be compatible with the eager elimination of subsumed clauses—another disadvantage compared to resolution.

It was not until 1975 that functional reflexivity equations were proved unnecessary for refutation completeness of paramodulation [Bra75]. Brand's *modification method* transforms a set of equational clauses N to a set N', such that $N \cup \mathcal{C}$ is satisfiable if and only if $N' \cup \{x \simeq x\}$ is satisfiable, where $\mathcal{C}$ denotes the set of *reflexivity*, *commutation*, and all *monotonicity-I* axioms (the *monotonicity-II* axioms are not necessary, provided that equality is the only predicate). Hence, it eliminates all equality axioms except reflexivity. The redundancy of the functional reflexivity equations is obtained as a side effect. Brand showed that resolution inferences with clauses in N'—for a specific selection strategy—can be simulated by paramodulation with clauses in N.

Brand left open whether paramodulation into variables is necessary. The difficulty to prove, within Brand's method, that paramodulation into variables is not necessary, relies in that paramodulation *from* variables and paramodulation *into* variables cannot be clearly distinguished as separate resolution inferences on transformed clauses; and the former inferences are in general necessary. Later (and possibly simpler) proofs, like the one from Peterson [Pet83], confirmed that paramodulation into variables is completely unnecessary for completeness of paramodulation, and dispensed with the reflexivity axiom $x \simeq x$, among other improvements commented in Section 1.2.1.

Brand's results also point out other redundancies in unrestricted paramodulation. *Most* of the paramodulation inferences needed to simulate resolution (with the specific selection strategy) take place at non-variable positions that were present in the initial set of clauses. Almost (but not quite) all paramodulations into variables or into terms introduced by unification in previous inference steps are unnecessary. This is interesting since it implies that the paramodulations obtained from Brand's transformation are only done on term structure near the top of terms, that is, term structure that was originally present in the input clauses. This idea is very attractive, especially on problems where the input equations are small but the equations produced during inference are large. At the moment of Brand's proof nobody appreciated that his proof implied that this restriction on paramodulation (called *basic paramodulation* at present) is complete, as it was proved later (see Section 1.2.4).

The study of restrictions on the applicability of paramodulation is interesting since, even if we get rid of equality axioms and avoid paramodulation into variables, many times the too general applicability of paramodulation quickly produces a large amount of unnecessary clauses, which expand the search tree in an inadmissible way.

A fairly straightforward approach to paramodulation is to simulate it by transforming the input in a manner similar to that of Brand [Bra75]. However, transformation methods typically *flatten* terms, and then the filtering effect of unification is much weaker on transformed clauses. Also, this simulation tends to reduce the possibilities of simplification. For example, if

$$f(g(x)) \simeq t$$

is flattened into

$$z \not\simeq g(x) \vee f(z) \simeq t,$$

then simplification with, e.g., $f(g(x)) \to x$ can no longer be applied. There have been also approaches to deal efficiently with equality in frameworks not covered by this thesis, such as tableau and sequent-based calculi (see [DV01]).

Concerning paramodulation-based theorem proving, fruitful practical research has been done at Argonne by Wos, Overbeek, Henschen and others (see, e.g., [Wos88, Lus92, Wos96] for references), and especially by McCune with his provers Otter [McC94] and EQP [McC97a] and his celebrated automated proof of a long-standing open problem, the Robbins conjecture [McC97c, McC97b] (this success required clever prover engineering and heuristics, but probably depended to an even larger degree on some insights provided by a long line of theoretical research on paramodulation-based calculi and related deductive methods, as mentioned in Section 1.2.5). Other state-of-the-art systems using (some variations of) paramodulation and which have recently succeeded, in particular, in (some category of) the CADE systems competition, are SPASS [WBH+02], E-SETHEO [MIL+97], Vampire [RV02] and Waldmeister [Hil03].

Further theoretical research on paramodulation-based theorem proving has been aiming at restricting the applicability of the paramodulation rule while preserving refutation completeness. In the following sections we review some of these refinements.

1.2 Paramodulation calculi: further refinements

1.2.1 Ordered paramodulation

As said in the previous section, even if paramodulation into a variable is avoided, paramodulation quickly produces a large amount of unnecessary clauses. Often, paramodulation allows one to repeatedly apply the same equation obtaining larger and larger literals.

Example 1.2.1 *The equation $x \simeq fx$ can repeatedly be applied*[1] *as in the derivation below:*

$$\dfrac{x \simeq fx \quad \dfrac{x \simeq fx \quad P(a)}{P(fa)}}{\dfrac{P(ffa)}{\dfrac{\vdots}{P(f\ldots fa)}}}$$

obtaining larger and larger literals of the form $P(f^n a)$.

Now the basic idea of *ordered* paramodulation, originally considered in [Pet83] and [HR86], is to only perform replacements of terms by *simpler* ones, where the definition of what is simple may be fairly subtle. However, as a first approximation, smaller terms are typically simpler. If we have an equation $x+0 \simeq x$ then we would typically orient it as $x+0 \rightarrow x$. This reduces the generation of terms like $((x+0)+0)+0$ which can appear in proofs if no such directionality is applied. The question arises how to orient equations and what conditions guarantee that the resulting systems have the same computational power as the systems they came from.

An important step forwards in this context was the introduction of well-founded (*reduction*) orderings on terms with respect to which paramodulation is applied. A reduction ordering $\succ$ is a transitive relation which is well-founded (there are no infinite descending chains $s_1 \succ s_2 \succ \ldots$), monotonic ($s \succ t$ implies $f(u_1,\ldots,u_i,s,v_1,\ldots,v_j) \succ f(u_1,\ldots,u_i,t,v_1,\ldots,v_j)$, for every symbol f of arity $i+j+1$ and all terms $u_1,\ldots,u_i,v_1,\ldots,v_j$) and stable under substitutions ($s \succ t$ implies $s\sigma \succ t\sigma$, for every substitution σ). Stability under substitution implies that reduction orderings cannot be total on terms with variables. But many reduction orderings are total (or totalizable) on ground terms, and this suffices in many completeness proofs, since one can often reason about non-ground formulae by considering the corresponding ground instances.

If we use ordered paramodulation with respect to a reduction ordering, no derivation from $P(a)$ and $x \simeq fx$ is possible (compare with Example 1.2.1).

But even more restrictions based on orderings can be imposed while preserving refutation completeness. For example, it is possible to consider only paramodulations involving maximal (with respect to a lifted version of the ordering $\succ$) literals. This can also be seen as a *literal selection strategy* (see Section 1.2.3 for details about selection strategies). On the other hand, *superposition* is a refinement of ordered paramodulation involving only the equality predicate, in which only maximal terms of equations are considered (that is, not only paramodulation *with* small sides, but also

[1] Observe that this is not an inference by paramodulation *into a variable* but *from a variable*.

paramodulation *on* small sides is avoided). Some versions of superposition were proposed in [ZK88] and [Rus91]. *Strict superposition* [BG90, BG94b] is the strongest refinement, in which only maximal terms of maximal equations are considered.

These ideas came from the hope of extending the results on equational theories, started with the so-called *Knuth-Bendix completion* method [KB70], to full first-order logic. Such a completion procedure attempts to transform a set of equations E into an equivalent *convergent* (i.e., terminating and confluent) term rewrite system (TRS) R, such that rewriting using R becomes a decision procedure for deduction in the theory of E, i.e., for deciding the *word problem* in E.

Example 1.2.2 *Let E consist of the equations*

$$plus(0,x) \simeq x$$
$$plus(s(x),y) \simeq s(plus(x,y))$$

Then an equivalent convergent TRS R for deduction in the theory of E is

$$plus(0,x) \to x$$
$$plus(s(x),y) \to s(plus(x,y))$$

Since R is terminating and confluent then to decide, e.g., whether $E \models plus(s(s(0)),s(0)) \simeq s(s(s(0)))$, one just needs to check if each of two terms rewrites to the same normal form *(a term which cannot be rewritten any more):*

$$plus(s(s(0)),s(0)) \to_R s(plus(s(0),s(0))) \to_R s(s(plus(0,s(0)))) \to_R s(s(s(0)))$$

A TRS R terminates if, and only if, its rules are included in a reduction ordering (that is, if there exists a reduction ordering $\succ$ such that $l \succ r$ for every rule $l \to r$ in R). Indeed, by definition of rewriting ($C[l\sigma]_p$ rewrites with R into $C[r\sigma]_p$ if $l \to r \in R$), if a TRS R terminates then the rewrite relation generated by R is itself a reduction ordering (see [Der87, DP01] for details).

Two crucial points in the transformation process of a set of equations E to an equivalent set of rewrite rules R are the *orientation* of equations and the computation of *critical pairs* between rules. *Orienting* an equation is turning it into a rewrite rule, by proving that one term of the equation, which becomes the left-hand side of the rule, is greater with respect to the given ordering on terms than the other (right-hand) side. A *critical pair* is an equation obtained by the restricted version of paramodulation, called superposition, in which paramodulations take place only with left-hand sides of rules, i.e., only the big terms (with respect to the ordering) are considered. During the completion process, equations are simplified by rewriting, and tautologies, i.e., equations of the form $s \simeq s$, are removed.

Note that, since the word problem is not decidable in general, not always such a rewrite system can be obtained. This can be because of a failure, due to the impossibility of orienting an equation (like the commutativity axiom $f(x,y) \simeq f(y,x)$), or because the process does not terminate. To completely avoid failure, the completion procedure can be extended to deal with unorientable equations [Lan75, HR87, BDP89] obtaining the so-called *ordered* (or *unfailing*) completion procedures, for which the ordering is required to be total on ground terms (also called complete) or at least completable (i.e., there exists a total ordering extending the given ordering). Ordered rewriting consists then of applying the equation $l \simeq r$ as a rewrite rule $l\sigma \to r\sigma$ for all those instances $l\sigma \simeq r\sigma$ of $l \simeq r$ such that $l\sigma \succ r\sigma$. This leads to complete ordered theorem provers for equational theories E: for every $s \simeq t$ such that $E \models s \simeq t$, enough equations for a rewrite proof will exist in E after a finite number of steps of the (possibly infinite) completion procedure. Moreover, if the process terminates it produces a convergent system for ordered rewriting.

Example 1.2.3 *Let E consist of the equations*

$$\begin{array}{rclcrcl} x \cdot 1 & \simeq & x & \quad & x \cdot y & \simeq & y \cdot x \\ x \cdot (y \cdot z) & \simeq & (x \cdot y) \cdot z & & x \cdot x^{-} & \simeq & 1 \end{array}$$

Then ordered completion w.r.t. a lexicographic path ordering will generate

$$\begin{array}{rclcrcl} 1^{-} & \to & 1 & \quad & x \cdot y & \leftrightarrow & y \cdot x \\ x \cdot 1 & \to & x & & x \cdot (y \cdot z) & \leftrightarrow & y \cdot (x \cdot z) \\ 1 \cdot x & \to & x & & (x \cdot y) \cdot z & \to & x \cdot (y \cdot z) \\ (x^{-})^{-} & \to & x & & x \cdot (x^{-} \cdot z) & \to & z \\ x \cdot x^{-} & \to & 1 & & (y \cdot x)^{-} & \to & x^{-} \cdot y^{-} \end{array}$$

(Those equations that can only be used to rewrite in one direction are given as rules.)

To reduce the number of cases of non-termination, and to improve efficiency, powerful simplification methods and critical pair criteria for detecting redundant inferences have been developed [BDH86, Pet90, MN90, Bac91, BD94, CNNR98]. On the other hand, many completion procedures for dealing with more powerful languages have been designed. For instance, to obtain convergent sets for conditional rewriting [Kap84, JW86] some completion procedures transforming sets of conditional equations (clauses of the form $e_1 \simeq e_1' \wedge \ldots \wedge e_n \simeq e_n' \to e \simeq e'$) have been proposed [KR91, Gan91]. Note that this language is a subset of the one of equality Horn clauses, since each clause must have exactly one positive literal. Completion procedures for equality Horn clauses have also been proposed [KR91, NN91], as well as for other extensions of the purely equational case, such as the language of restricted equality clauses (clauses of the form $l_1 \vee \ldots \vee l_n \vee e \simeq e'$, or simply $l_1 \vee \ldots \vee l_n$, where $l_1, \ldots, l_n$ are positive or negative non-equality literals), which leads to convergent systems for clausal rewriting [NO90].

The generalization of completion procedures to full first-order clauses with equality turned out to be extremely difficult and required the development of more powerful techniques for the completeness proofs. Using the *transfinite semantic tree* method, Hsiang and Rusinowitch [HR91] proved the refutation completeness of ordered paramodulation, while Bachmair and others applied an extension of *proof orderings* [BDH86, BD94] for obtaining similar results. An important breakthrough was Bachmair and Ganzinger's work [BG90, BG94b] where, by means of their *model generation* proof method (extending the so-called *forcing* techniques developed by Zhang [Zha88] and Pais and Peterson [PP91]), an inference system for full first-order clauses with equality, based on strict superposition and compatible with powerful redundancy notions, was proved refutation complete.

1.2.2 Redundancy notions

The concept of saturation in theorem proving is a well-known useful concept. The main idea of *saturation* is to compute the closure of a set of formulae, with respect to a given inference system I, only *up to redundant inferences*. Roughly, a saturation procedure adds conclusions of non-redundant inferences and removes redundant formulae, obtaining in the limit a *complete* or *saturated* set.

Knuth-Bendix completion attempts to transform sets of equations into *complete* ones: sets that are closed under the addition of non-joinable critical pairs, where a critical pair is *joinable* if it can be rewritten into a tautology $s \simeq s$. In this context, joinable critical pairs do not need to be added, and existing critical pairs can be simplified by demodulation (with other equations), and even removed if a tautology has been obtained after the demodulation process. Examples of early notions of redundancy (in the context of resolution) are the elimination of tautologies and subsumption. In the context of paramodulation, concrete simplification methods were discussed in [WRCS67, Sla74, Lov78, Pet83]. Bachmair and Ganzinger gave more general abstract notions of redundancy for inferences and formulae (see, e.g., [BG94b]) where, roughly, a formula is redundant if it is a logical consequence of smaller formulae (with respect to a given ordering on formulae). In provers based on ordered resolution, paramodulation or superposition with respect to a total reduction ordering on ground terms $\succ$, apart from standard methods like removing tautologies, subsumption, or demodulation with respect to $\succ$, these abstract redundancy notions also cover other practical techniques (see, e.g., the *Saturate* system [NN93, GNN99], a paramodulation-based deduction system where the abstract redundancy notions have been instantiated by several concrete simplification and deletion methods).

Simplification of formulae and elimination of redundant formulae are essential components of automated theorem provers. In fact, in most successful automated provers, simplification is the primary mode of computation, whereas expensive deduction rules are used only sparingly. Finding the right combination of deduction and simplification is a fundamental issue in automated deduction.

In the setting of first-order clauses, proving the refutation completeness of a saturation procedure amounts to showing that the empty clause is in S for every (possibly infinite) unsatisfiable saturated set of clauses S. Due to the restrictive inference rules and the powerful redundancy notions, using these saturation procedures it is sometimes possible to compute a finite saturated set (not containing the empty clause) for a given theory. In this case the *satisfiability* (or consistency) of the theory has been proved. This kind of satisfiability proving has many applications and is closely related to inductive theorem proving [CN00, Com01]. The SPASS system [WBH$^+$02] successfully applied saturation to prove satisfiability of all problems in the corresponding category of the 1997 CADE theorem proving competition [SS98].

In the monadic class with equality, saturation can always be made terminating, producing decidability results [BGW93]. For some classes of first-order formulae, saturated sets can provide decision procedures, like rewrite proofs do in the equational case. For instance, saturated sets of conditional equations provide decision procedures for conditional rewriting, provided they fulfill some syntactic requirements (e.g., in certain clauses the maximal terms must contain all variables). Furthermore, theorem proving in theories expressed by saturated sets of axioms is also interesting because more restrictive proof strategies become refutation complete. E.g., the set-of-support strategy, which is incomplete in general for paramodulation-based inference systems and also for ordered[2] inference systems (even in absence of equality), becomes complete for saturated sets S due to the fact that no inferences between clauses in S are then needed.

1.2.3 Selection strategies

Another way for reducing (and having better control of) the search space while preserving refutation completeness are the so-called *selection strategies*. In such strategies, the possible inferences between clauses are restricted by allowing only inferences involving *selected* literals.

Well-known examples of selection strategies are the *maximal* (or *ordered*) strategies for a given atom ordering. For example, in a maximal resolution strategy, a (ground) inference between $A \vee C$ and $\neg A \vee D$ is performed only if A is larger in the given atom ordering than all other atoms in C and D. This strategy extends well to paramodulation-based inference systems (see Section 1.2.1).

Another well-known selection strategy is the so-called *eager* negative selection strategy [Der91, NN91, BG94b], where in each clause a single negative literal is selected whenever there is any. This leads to the so-called *positive* strategies (positive unit strategies in the Horn case), in which always

[2] That is, with an *ordered* selection strategy; see Section1.2.3.

the left premise of each (resolution or paramodulation) inference is a positive (unit) clause.

Selection strategies are even more desirable for paramodulation than for resolution since, in paramodulation, the positive equations may have to be paramodulated not only into negative equations, but also into positive ones.

1.2.4 Basic strategies and constraints

A *constraint* is a relation among several unknowns (or variables), each taking a value in a given domain. A constraint thus restricts the possible values that variables can take. *Constraint programming* (CP) is the study of computational systems based on constraints, and has interdisciplinary origins (see [Bar99] for a survey). The main step towards the modern view of CP was achieved when Gallaire [Gal85] and Jaffar and Lassez [JL87] noted that *logic programming* (LP) is just a particular kind of constraint programming, since unifiability corresponds to solvability of constraints containing equations between terms. From this point on, *constraint logic programming* (CLP) began as a merger of two declarative paradigms: constraint solving and logic programming[3] (see [JM94, Ros99] for a survey). An abstract framework for CLP, called CLP($\mathcal{X}$), was developed in [JL87] (where $\mathcal{X}$ refers to an arbitrary constraint system), while a general concept of constraints and constraint resolution in this context was given in [HS88]. As an example, it is natural to view logic programming as an instantiation of the CLP scheme: LP = CLP($\mathcal{FT}$) where $\mathcal{FT}$ is the algebra of finite trees (a.k.a. the Herbrand universe) [Mah92]. CLP languages are LP languages in which unification is replaced by constraint solving. The combination of constraints and logic programming turns out to be very natural, and CLP makes a nice declarative environment for solving problems by means of constraints. Moreover, constraints are special predicates whose satisfiability can be established for various domains using efficient algorithms. As a result, many CLP-related systems have been successfully applied in real-life applications to deal with hard problems such as scheduling, timetabling, resource allocation, and many combinatorial problems.

Due to the close relation between logic programming and theorem proving, many results of one field are applicable to the other. In theorem proving, the idea of using constraints is to reduce the search space without losing completeness. Some use of constraints in resolution-based theorem proving was already present in [Hue72]. Regarding paramodulation-based systems, the first ideas for specific applications of constraints were given in [Pet90, KKR90]. In first order (clausal) logic, a clause C with a constraint T, written $C \mid T$, represents the set of all ground instances $C\sigma$ of C such that the variable assignment of σ is a solution of the constraint T. For example, the constrained clause $P(x) \mid x = f(y) \wedge y > a$ denotes all ground atoms of the form $P(f(t))$ such that $t \succ a$ in the given ordering $\succ$. It must be noted that the symbols $=$ and $>$ used in the constraint language are interpreted as a given congruence (usually the syntactic equality of terms) and as a given term ordering $\succ$, respectively.

In [NR95], the use of both equality and ordering constraints in theorem proving procedures is put in a general framework, based on the notion of *constraint inheritance strategies*. The main idea is to keep unification and ordering restrictions in constraints, that are inherited between clauses. Then, an inference rule like strict superposition can be expressed in a compact way by:

$$\frac{C' \vee (e \simeq e') \mid T' \qquad C \mid T}{C' \vee C[e']_p \mid T' \wedge T \wedge C|_p = e \wedge OC}$$

where T and T' are the constraints inherited from the premises, the *equality constraint* $C|_p = e$ stores

[3] However, it does not mean that constraint programming is restricted to CLP.

the unification restriction, and OC is an *ordering constraint* of the form $e > e' \wedge \ldots$ encoding the ordering restrictions imposed by this inference.

Observe that the fact that the unifier is not applied in the conclusion already reduces the search space, provided that inferences into variables are forbidden. Moreover, since every constrained clause $C \mid T$ represents the set of all instances of C such that σ is a solution of T, every constrained clause for which the constraint T is unsatisfiable can be removed. And, if some inference is not compatible with the required restrictions (applied on the current inference and on the previous ones), then it can be discarded. On the other hand, if we obtain the empty clause, a contradiction has been derived if, and only if, its associated constraint is satisfiable. Therefore, for taking advantage of the constraints it is necessary to have (efficient) decision algorithms for their satisfiability.

Constraints are closely related to *basic* strategies, where no inferences have to be computed on subterms generated in unifiers of ancestor inference steps. In fact, an inference system with a basic strategy can be modeled as working with constrained clauses, inheriting them and simply storing the unification restrictions in the constraint part (i.e., without applying the constraints to the clause part). Basic strategies can also be seen as *term selection strategies*. These strategies are important since, in principle, all subterms in a clause are possible candidates for replacement via paramodulation and, since the instantiation of subterms tends to produce larger and larger terms, the inherent branching factor in the proof search may be huge.

Nieuwenhuis and Rubio proved the completeness of *basic superposition* in [NR92a]. Bachmair, Ganzinger, Lynch and Snyder [BGLS92, BGLS95] independently developed a comprehensive framework for the basic strategy using *closure substitutions*, which play the role of equality constraints. These developments took place independently of much earlier work in Russia by Degtyarev [Deg79], who used conditional clauses (which can be seen as clauses with syntactic equality constraints) for describing a form of basic paramodulation without ordering restrictions (see also [DV86]).

In [NR92b], Nieuwenhuis and Rubio showed that by means of *ordering constrained clauses* one can restrict the search space even further without losing completeness, whereas Lynch and Snyder [LS93] applied equality, disequality and irreducibility constraints to obtain more powerful redundancy methods in basic equational completion.

1.2.5 Paramodulation with built-in equational theories

The aforementioned paramodulation-based inference systems apply in the same way to any set of clauses with equality. However, there are some cases in which is preferable to give an special treatment to a subset of the axioms. Historically, these special treatments were motivated by the fact that equations like the commutativity axiom $f(x,y) \simeq f(y,x)$ cannot be oriented into a terminating rewrite rule. Although this problem can be overcome by the ordered rewriting approach (see Section 1.2.1), there are two other good reasons for using specific methods for particular equations. On the one hand, some axioms generate many slightly different permuted versions of clauses, and for efficiency reasons it is many times better to treat all these clauses together as a single one representing the whole class. As observed by Bachmair in [Bac91], some equations like the commutativity axiom are more naturally viewed as "structural" axioms (defining a congruence relation on terms) rather than as "simplifiers" (defining a reduction relation). On the other hand, special treatments can avoid non-termination of completion procedures, like happens in some cases in the presence of associativity and commutativity axioms [Wer92]. Then the idea is to extend completion procedures to deal with congruence classes of terms instead of single terms, i.e., working with a *built-in* equational theory E, and performing rewriting and completion with special E-matching and E-unification algorithms.

The first results on *rewriting modulo E* were given by Plotkin [Plo72], Slagle [Sla74], Lankford and Ballantine [LB77], and Peterson and Stickel [PS81]. For these new rewriting relations, the notion of critical pair has to be extended and, as said, unification replaced by E-unification. Several E-completion procedures for the equational case were developed in [LB77, Hue80, PS81, Jou83, JK86, BD89]. Special attention has always been devoted to the case where E includes axioms of associativity (A) and commutativity (C), as they occur very frequently in practical equational (or clausal) specifications, and are well-suited for being built in, due to their permutative nature.

The generalization of these E-completion techniques to full first-order clauses with equality has been studied by Wertz [Wer92] for general sets of equations E within Bachmair and Ganzinger's model generation framework, giving a particular treatment to the AC case. Other different approaches for the AC case by Paul, and by Rusinowitch and Vigneron, can be found in [Pau92, RV95]. A variant by Bachmair and Ganzinger of their model generation method for the AC case is given in [BG94a].

In general, there is no unique most general E-unifier for a given E-unification problem, and new variables may appear: for example, if f is an AC-symbol, then $f(x,a)$ and $f(y,b)$ have the two AC-unifiers $\sigma_1 = \{x \mapsto b, y \mapsto a\}$ and $\sigma_2 = \{x \mapsto f(b,z), y \mapsto f(a,z)\}$. Moreover, the number of E-unifiers may be large: for example, the cardinality of a minimal complete set of AC-unifiers of two terms may be doubly exponential [Dom92, KN92] and therefore a single inference can generate a double exponential number of new clauses.

In this context, equality constraints are extremely useful: instead of E-unifying the terms, the idea is to keep the unification problem in an E-equality constraint, that is, a constraint where equality is interpreted as the congruence on terms corresponding to the equational theory E. Dealing with a constrained clause $C \mid s =_E t \wedge T$, where $=_E$ denotes E-equality, can be much more efficient than having n clauses $C_1\sigma_1, \ldots, C_n\sigma_n$, one for each solution σ_i of $s =_E t \wedge T$ since, e.g., many inferences can be computed at once, and each inference generates one single conclusion. Furthermore, computing E-unifiers is not needed at all. On the one hand, a clause $C \mid T$, where T is an E-equality constraint, can be proved redundant by means of efficient (possibly incomplete) methods for detecting unsatisfiability of T. On the other hand, semi-decision procedures for checking E-unifiability exist for every finite E and, hence, when C is the empty clause it is possible to know whether a contradiction has been derived.

The first results on (almost basic) constrained deduction methods modulo AC were reported by Vigneron [Vig94]. The completeness of a basic strategy (see Section 1.2.4) for the AC case, combined with ordering constraints, was first proved in [NR94, NR97]. The basicness restriction is considered to "have been a key strategy" by McCune [McC97c] in his automated AC-paramodulation-based proof of the Robbins conjecture.

Another well-known equational theory that is interesting for being built-in is the one of abelian groups (AG), as they appear in many applications of automated reasoning and, moreover, they behave better than theories like AC in some aspects (for example, due to the fact that diophantine equation solving is easier in the integers than in the natural numbers, AG-unification is easier than AC-unification). Paramodulation with built-in abelian groups has been investigated by many authors [Che86, Zha93, Mar94, Mar96, GW96, Wal98, Wal99, Stu98]. An approach where abelian groups are fully built-in, in the sense that no inferences with the AG axioms or abstraction rules are needed, can be found in [GN00, GN03], and related work concerning constraint solving techniques can be found in [GN01].

1.3 Thesis outline

In this section we give an overview of the structure and contents of this thesis. Following the introduction, there are the following chapters:

1.3.1 Chapter 2: Notation and preliminaries

In this chapter we introduce the notation we use, and give some basic notions on logic, orderings and rewriting.

1.3.2 Chapter 3: Paramodulation with west orderings

This chapter is the first one introducing new results. All the completeness results for ordered paramodulation mentioned in Section 1.2.1 require the term ordering $\succ$ to be well-founded, monotonic and total (or extendable to a total ordering) on ground terms. In many practical situations, these requirements are too strong, and hence weakening them has been a well-known research challenge.

In this chapter it is shown that ordered paramodulation remains complete when using orderings without the monotonicity requirement. The only properties required for the ordering are well-foundedness and the subterm property. (Note that any such ordering can be totalized without losing these two properties. Then the totalized ordering is called *west* after well-founded, subterm and total.) This solves a well-known open problem in the RTA list of open problems [RTA01] (problem no. 85) since 1995.

Part of the results given in this chapter have been published at the LICS'99 conference [BGNR99] and further developed in the Journal of Automated Reasoning [BGNR03].

1.3.3 Chapter 4: Knuth-Bendix completion with west orderings

Here, by a careful further analysis of the technique presented in Chapter 3, we obtain the first practical Knuth-Bendix completion procedure that finds a convergent TRS R for a given set of equations E, logically equivalent to E, and contained in a (possibly non-totalizable) reduction ordering $\succ$, whenever it exists. This solves another well-known open problem, posed by N. Dershowitz and J-P. Jouannaud in the RTA list of open problems [RTA01] (problem no. 34) since its creation in 1991.

Part of the results given in this chapter have been published as well at the LICS'99 conference [BGNR99] and further developed in the Journal of Automated Reasoning [BGNR03].

1.3.4 Chapter 5: Paramodulation with well-founded orderings

In Chapter 3 we have shown that well-foundedness and the subterm property are enough for ensuring completeness of ordered paramodulation, i.e., the monotonicity requirement is not necessary. After this, the fundamental question arises whether more requirements can be dropped.

Here we show that the subterm property is not necessary either. By using a new restricted form of rewriting we obtain a completeness proof of ordered paramodulation where well-foundedness of the ordering suffices. Apart from the theoretical interest of this result, some potential applications motivating the interest of dropping the subterm property are given.

Part of the results given in this chapter have been published at the CADE'02 conference [BR02].

1.3.5 Chapter 6: Modular redundancy

There is some evidence that crucial simplification techniques like demodulation are not fully applicable in combination with ordered paramodulation when the ordering is not monotonic.

As a first solution to this problem, in this chapter we introduce a notion of modular redundancy for theorem proving. Modular redundancy allows us to exploit redundancy elimination techniques (like tautology elimination, subsumption, demodulation or other more refined methods) in combination with arbitrary existing theorem provers, in a refutation complete way, even if these provers are not (or not known to be) complete in combination with the redundancy techniques when applied in the usual sense.

The results given in this chapter have been published at the FroCos'00 workshop [BGNR00].

1.3.6 Chapter 7: Redundancy in paramodulation with west orderings

In Chapters 3, 4 and 5, we have shown that ordered paramodulation and Knuth-Bendix completion remain complete when using orderings with less requirements than usual. However, the considered redundancy notions are too weak and, in particular, demodulation cannot be applied at all. Even though modular notions of redundancy, like the ones presented in Chapter 6, can be applied, it is always better to have specialized notions of redundancy if possible. To this end, in this chapter, a second solution to the problem of finding compatible redundancy notions is described.

Here, we present a calculus which strictly improves the one given in Chapter 3. On the one hand, regarding the amount of inferences needed to be performed, our inference system is basically the same as the one in Chapter 3 but, on the other hand, our calculus is compatible with powerful redundancy notions, like demodulation.

Part of the results given in this chapter have been published at the IJCAR'04 conference [BR04].

1.3.7 Chapter 8: Conclusions and future work

In this chapter we summarize our conclusions and outline several open questions.

Chapter 2

Notation and preliminaries

In this chapter we give some basic definitions on terms, equations, clauses, inference systems, orderings and related issues, which have been adapted from [NR01, DP01, Pla93, DJ90, GN87]. Most of our notation is introduced here.

2.1 Terms

Let $\mathcal{F}$ be a *signature*, a (finite) set of function symbols with an arity function *arity*: $\mathcal{F} \to \mathbb{N}$, denoting its number of arguments, and $\mathcal{X}$ a (denumerable) set of variable symbols, with $\mathcal{F} \cap \mathcal{X} = \emptyset$. Function symbols f with $arity(f) = n$ are denoted by $\mathcal{F}_n$, and are called *n-ary* symbols (when $n = 1$ one says *unary* and when $n = 2$, *binary*). If $arity(f) = 0$ then f is a *constant symbol*. Observe that the $\mathcal{F}_n$ are disjoint by definition of *arity*. That is, we do not allow *varyadic* function symbols. We use $a, b, c \dots$ to denote constant symbols, $f, g, h \dots$ to denote (non-constant) function symbols, and $x, y, z \dots$ to denote variables.

The set of *free*, *first-order terms* over $\mathcal{F}$ and $\mathcal{X}$, denoted by $T(\mathcal{F}, \mathcal{X})$, is the smallest set containing $\mathcal{X}$ such that $f(t_1, \dots, t_n) \in T(\mathcal{F}, \mathcal{X})$ whenever $f \in \mathcal{F}_n$ and $t_1, \dots, t_n \in T(\mathcal{F}, \mathcal{X})$ for some n (possibly zero). Terms in $T(\mathcal{F}_0 \cup \mathcal{F}_1, \mathcal{X})$ are called *monadic*. The set $T(\mathcal{F}, \emptyset)$ of variable free or *ground* terms will be denoted by $T(\mathcal{F})$. Note that, if there are no constant symbols in $\mathcal{F}$, that is, $\mathcal{F}_0 = \emptyset$, then $T(\mathcal{F}) = \emptyset$ as well. Therefore, as usual, we will make the (rarely critical) assumption that there is at least one constant symbol in $\mathcal{F}$.

Let t be a term $f(t_1, \dots, t_n)$ in $T(\mathcal{F}, \mathcal{X})$ with $n \geq 0$. Then the size of t, denoted by $|t|$, is defined as $|t_1| + \cdots + |t_n| + 1$. Observe that a variable has size 1 (since $n = 0$).

A term t in $T(\mathcal{F}, \mathcal{X})$ can be viewed as a finite ordered tree, whose leaves are labeled with variables (from $\mathcal{X}$) or constants (from $\mathcal{F}_0$) and whose internal nodes are labeled with function symbols (from $\mathcal{F}_1 \cup \mathcal{F}_2 \cup \dots$) of positive arity, with out-degree equal to the arity of the label. A *position* p within t can then be represented as a sequence of positive integers (based on Dewey's decimal notation), describing the path from the outermost, "root" symbol to the head of the subterm at that position. By $t|_p$ we denote the *subterm* of t (rooted) at position p. We have $t|_\lambda = t$ (where λ denotes the empty sequence) and $f(t_1, \dots, t_n)|_{i.p} = t_i|_p$ if $1 \leq i \leq n$ (and is undefined if $i > n$). For example, if t is $f(a, g(b, h(c)), d)$ then $t|_{2.2.1}$ is c. To avoid confusion, we will use the symbol $\equiv$ for syntactic equality of terms. We use $=$ in the metalanguage.

We say that a term s (resp. a function symbol f) *occurs* at position p in a term t if $t|_p \equiv s$ (resp. $t|_p$ is rooted by f). By $vars(t)$ we denote the set of all variables occurring in a term t. We write $t \trianglerighteq s$ to mean that a term s is a subterm of t, that is, s occurs in t at some position p. If $p \neq \lambda$ we write

$t \rhd s$ and we say that s is a *proper* subterm of t. We speak of position p as being (strictly) *above* q if p, represented as a sequence of numbers, is a (proper) prefix of q. At the same time, we speak of q as being (strictly) *below* p.

Reasoning with equations requires replacing subterms by other terms. The result of replacing $t|_p$ by s in t is denoted by $t[s]_p$. A *context* $t[\,]_p$ is a term t with a "hole" (i.e., a lambda-bound variable) at a distinguished position p. Then $t[s]_p$ is a term for any term s. An *equivalence relation* is a reflexive, symmetric and transitive relation. Replacement leads to the important notion of congruence: an equivalence relation $\sim$ on a set of terms is a *congruence* if $f(s_1,\ldots,s_n) \sim f(t_1,\ldots,t_n)$ whenever $s_i \sim t_i$ for all i in $1\ldots n$.

A *substitution* is a partial mapping from variables (in $\mathcal{X}$) to terms (in $T(\mathcal{F},\mathcal{X})$). It can be represented explicitly as a function by a set of bindings of variables in its domain. That is, if $\sigma = \{x_1 \mapsto s_1,\ldots,x_n \mapsto s_n\}$, then $\sigma(x_i) = s_i$ for all i in $1\ldots n$, and $\sigma(x) = x$ for every other variable $x \in \mathcal{X}$. The *domain* of a substitution σ, denoted by $Dom(\sigma)$, is the set of variables $\{x \in \mathcal{X} \mid \sigma(x) \neq x\}$. (Though traditional, there is usually no need to restrict the domain of a substitution to be a finite subset of variables.) The *range* of a substitution σ, denoted by $Ran(\sigma)$, is the set of terms $\bigcup_{x\in Dom(\sigma)}\{\sigma(x)\}$.

Substitutions $\sigma\colon \mathcal{X} \to T(\mathcal{F},\mathcal{X})$ are extended homomorphically to a total mapping $\sigma\colon T(\mathcal{F},\mathcal{X}) \to T(\mathcal{F},\mathcal{X})$ from terms to terms; we use the postfix notation $t\sigma$ for the image of a term t under a substitution σ. This is defined inductively on the structure of terms as follows:

$$t\sigma = \begin{cases} \sigma(t) & \text{if } t \text{ is a variable} \\ f(t_1\sigma,\ldots,t_m\sigma) & \text{if } t \text{ is of the form } f(t_1,\ldots,t_m) \end{cases}$$

In the second case of this definition, $m = 0$ is allowed: in this case, f is a constant symbol and $f\sigma$ is f. Thus $t\sigma$ is t with all variables replaced by terms as specified by σ. The image of an equation, rewrite rule, or formula under a substitution is defined similarly.

For example, if $\sigma = \{x \mapsto f(b,y), y \mapsto a\}$ then $g(x,y)\sigma$ is $g(f(b,y),a)$.

Note that the replacement of variables is *simultaneous*. Hence, we have $t[s]_p\sigma \equiv t\sigma[s\sigma]_p$ for all terms t and s, positions p and substitutions σ.

The composition of two substitutions σ_1 and σ_2, denoted by juxtaposition, is defined as the composition of two functions, that is, $t\sigma_1\sigma_2 = (t\sigma_1)\sigma_2$.

For example, if $\sigma_1 = \{x \mapsto f(b,y)\}$ and $\sigma_2 = \{y \mapsto a\}$, then $g(x,y)\sigma_1\sigma_2$ is $g(f(b,a),a)$. (Compare with $g(x,y)\sigma$ in the previous example.)

Given two substitutions σ_1 and σ_2 with $Dom(\sigma_1)\cap Dom(\sigma_2) = \emptyset$, by $\sigma_1 \cup \sigma_2$ we denote the substitution σ such that $x\sigma \equiv x\sigma_1$ for every variable $x \in Dom(\sigma_1)$, $x\sigma \equiv x\sigma_2$ for every variable $x \in Dom(\sigma_2)$, and $x\sigma \equiv x$ for all other variables $x \in \mathcal{X}$.

A term t *matches* a term s if $s\sigma \equiv t$ for some substitution σ. Then t is called an *instance* of s. A term t is a *(renamed) variant* of a term s if they are instances of each other. A substitution σ is *ground* if $Ran(\sigma) \subseteq T(\mathcal{F})$. Unless stated otherwise, we will assume that ground substitutions σ applied to a term t are also *grounding*, that is, $vars(t) \subseteq Dom(\sigma)$, and hence $t\sigma$ is ground. Then we say that $t\sigma$ is a *ground instance* of t. Similar terminology applies to instances of equations, rewrite rules and formulae.

A term t is *unifiable* with a term s if $s\sigma \equiv t\sigma$ for some substitution σ. Then σ is called *unifier* of s and t. Furthermore, a substitution σ is called a *most general unifier* of s and t, denoted $mgu(s,t)$, if $s\sigma \equiv t\sigma$, and for every other unifier θ of s and t, it holds that $s\theta \equiv s\sigma\sigma' \equiv t\theta \equiv t\sigma\sigma'$ for some σ', that is, roughly, if every other unifier θ is a particular instance of σ. We sometimes speak about *the mgu* of two terms because it is unique up to variable renaming. See [BS01] for details and for unification algorithms computing *mgu*'s.

A *multiset* over a set S is a function $M\colon S \to \mathbb{N}$. The union and intersection of multisets are defined as usual by $M_1 \cup M_2(x) = M_1(x) + M_2(x)$, and $M_1 \cap M_2(x) = min(M_1(x), M_2(x))$. We also use a set-like notation: $M = \{a, a, b\}$ denotes $M(a) = 2$, $M(b) = 1$, and $M(x) = 0$ for $x \not\equiv a$ and $x \not\equiv b$. A multiset M is *empty* if $M(x) = 0$ for all $x \in S$.

2.2 Equations and (equality) clauses

An *equation* is a multiset of terms $\{s, t\}$, denoted $s \simeq t$ or, equivalently, $t \simeq s$. By $gnd(s \simeq t)$ we denote the set of ground instances of $s \simeq t$. A set of equations is an *equational system*. Given an equational system E, by $gnd(E)$ we denote the set of ground instances of all equations in E. Similar terminology applies to rewrite rules and formulae in general.

A *first-order equality clause* is a pair of finite multisets of equations Γ (the *antecedent*) and Δ (the *succedent*), denoted by $\Gamma \to \Delta$. We use a comma in clauses to denote the union of multisets of equations. For example, we write $\Gamma, \Gamma' \to \Delta$ instead of $\Gamma \cup \Gamma' \to \Delta$. Within a clause, we also write single equations as an equivalent multiset. That is, we write $s \simeq t, \Gamma \to \Delta$ for $\{s \simeq t\} \cup \Gamma \to \Delta$.

Clauses $e_1, \ldots, e_n \to e'_1, \ldots, e'_m$ correspond to the equivalent (quantifier free) disjunction of equations and negated equations $\neg e_1 \vee \ldots \vee \neg e_n \vee e'_1 \vee \ldots \vee e'_m$. Hence, given a clause $\Gamma \to \Delta$, equations in Γ are called *negative equations* and equations in Δ are called *positive equations*. We also speak of negative and positive *literals*, though we assume that equality (denoted by the symbol $\simeq$) is the only predicate (cf. section 2.12).

If there is no restriction on the number of positive and negative literals, we speak of *general clauses*. A Horn clause is a clause with at most one positive literal. The empty clause $\Box$ is a clause where both Γ and Δ are empty. A *positive* (resp. *negative*) clause is a clause $\Gamma \to \Delta$ where Γ (resp. Δ) is empty, and a *unit* clause is a clause with exactly one literal.

2.3 Interpretations

An *interpretation* I is a mapping between elements of the language and elements of a conceptualization. It first-order logic, it consists of:

- A non-empty set D, which is called the *domain* of I.
- A function $f_I\colon D^n \to D$ for every n-ary symbol $f \in \mathcal{F}$.
- A relation $P_I \subseteq D^n$ for every n-ary predicate symbol $P \in \mathcal{P}$.

(In our setting, $\mathcal{P}$ consists only of the equality predicate $\simeq$; cf. section 2.12.) An interpretation gives us a meaning for every symbol: it determines the possible values for the variables, the conceptualization of every function symbol, and which predicates hold under this conceptualization.

A *variable assignment* U is a function from variables in $\mathcal{X}$ to objects in D. (Note that this is different from a substitution, which is a function from variables to terms; cf. section 2.1.) Given an interpretation I and a variable assignment U, the *term assignment* T_{IU} corresponding to I and U is a mapping from terms in $T(\mathcal{F}, \mathcal{X})$ to objects in D, defined inductively on the structure of terms as follows:

$$T_{IU}(t) = \begin{cases} U(t) & \text{if } t \text{ is a variable} \\ f_I(T_{IU}(t_1), \ldots, T_{IU}(t_n)) & \text{if } t \text{ is of the form } f(t_1, \ldots, t_n) \end{cases}$$

The notions of interpretation and variable assignment allow us to define a relative notion of truth called *satisfaction*. By convention, the fact that a formula (e.g., a clause) F is satisfied by an interpretation I and a variable assignment U is written $I[U] \models F$. In this case, we say that the formula F is *true* relative to the interpretation I and the assignment U. Under some interpretations, a formula can be true; under other interpretations, it can be false. An interpretation I and a variable assignment U satisfy an atomic formula (a formula of the form $P(t_1,\ldots,t_n)$, where P is an n-ary predicate and $t_1,\ldots,t_n$ are terms) if and only if the tuple formed from the objects designated by the terms in the formula is an element of the relation designated by the predicate:

$$I[U] \models P(t_1,\ldots,t_n) \text{ if and only if } \langle T_{IU}(t_1),\ldots,T_{IU}(t_n)\rangle \in P_I.$$

Hence, the relation P_I can also be seen as a function which assigns a truth value to an atomic formula $P(t_1,\ldots,t_n)$ under the interpretation and variable assignment of its arguments.

In the context of first-order logic with equality, the interpretation of the equality predicate $\simeq$ is predefined, so that an interpretation and variable assignment satisfy an equation if and only if the corresponding term assignment maps the equated terms to the same object:

$$I[U] \models s \simeq t \text{ if and only if } T_{IU}(s) = T_{IU}(t).$$

That is, we have that $=_I$ is predefined as $\bigcup_{d\in D}\{\langle d,d\rangle\}$.

The satisfiability of formulas depends on the logical operators involved. E.g., the negation of a formula is satisfied if and only if the formula itself is not satisfied. A conjunction is satisfied if and only if all the conjuncts are satisfied. And a disjunction is satisfied if and only if at least one of the disjuncts is satisfied:

- $I[U] \models \neg F$ if and only if $I[U] \not\models F$.
- $I[U] \models F_1 \wedge \ldots \wedge F_n$ if and only if $I[U] \models F_i$ for all $i = 1,\ldots,n$.
- $I[U] \models F_1 \vee \ldots \vee F_n$ if and only if $I[U] \models F_i$ for some i, $1 \le i \le n$.

An universally quantified formula is satisfied if and only if the enclosed formula is satisfied for all assignments of the quantified variable. An existentially quantified formula is satisfied if and only if the enclosed formula is satisfied for some assignment of the quantified variable.

- $I[U] \models \forall x F$ if and only if for all $d \in D$ it is the case that $I[V] \models F$, where $V(x) = d$ and $V(y) = U(y)$ for all other variables y.
- $I[U] \models \exists x F$ if and only if for some $d \in D$ it is the case that $I[V] \models F$, where $V(x) = d$ and $V(y) = U(y)$ for all other variables y.

If an interpretation I satisfies a formula F for all variable assignments, then I is said to be a *model* of F, written $I \models F$. This corresponds to the intention that variables are implicitly universally quantified. Obviously, a variable assignment has no effect on the satisfaction of a formula that contains no free variables (variables without an enclosing quantifier).

A formula is said to be *satisfiable*, or *consistent*, if and only if there is *some* interpretation and variable assignment that satisfy it. Otherwise, it is *unsatisfiable*, or *inconsistent*. A formula is *valid*, or a *tautology*, if and only if it is satisfied by *every* interpretation and variable assignment.

If F and F' are two formulas such that F' is true in all models of F, then we say that F' is a *logical consequence* of F, or *logically follows* from F, or that F *logically implies* F', and we denote it by

$F \models F'$. Two formulas F and F'' are said to be *(logically) equivalent*, written $F \models\!\!\!\models F''$, if, and only if, they have the same truth value in each interpretation (i.e., are either both true or both false). The *equational theory* of an equational system E is the set of equations that are logical consequences of E.

As mentioned in section 2.2, clauses are quantifier free disjunctions of literals. It is folk knowledge that, in first-order logic, every formula F can be transformed into a set (conjunction) of clauses S preserving satisfiability: F is satisfiable if and only if S is. It is said that this transformation process puts F into *conjunctive normal form* (CNF).

2.4 Herbrand interpretations

A *Herbrand interpretation* for a set of clauses is an interpretation that maps ground terms into themselves. More specifically, an interpretation I is a Herbrand interpretation if and only if the following conditions hold:

- The domain D of I is just $T(\mathcal{F})$ (a.k.a. the *Herbrand universe*).[1]
- For every n-ary function symbol f, the f_I function is defined as $f_I(t_1,\ldots,t_n) = f(t_1,\ldots,t_n)$.

Note that this definition includes no restriction on the interpretation of predicate symbols; so we can choose any interpretation we like. The *Herbrand base* for a set of clauses is its set of ground instances. For any satisfiable Herbrand base, we can produce a Herbrand interpretation that satisfies it as follows. Since the Herbrand base is satisfiable, it has a model J. We construct our Herbrand interpretation I by making true those atomic formulas that are true under J and making false those atomic formulas that are false under J. This observation leads to the following theorem.

Theorem 2.4.1 (Herbrand theorem) *If a finite set S of clauses is unsatisfiable, then the Herbrand base for S is unsatisfiable.*

Therefore, a set of clauses is satisfiable if and only if it has a Herbrand model. It is also important to notice that, for a Herbrand interpretation I and a clause C, we have that $I \models C$ if and only if $I \models C\sigma$ for every ground substitution σ.

Since Herbrand interpretations map ground terms to themselves, they can also be seen simply as a set of ground atoms: the set of ground atoms that are true under the interpretation. Then a ground atom A is said to be true under a Herbrand interpretation I if $A \in I$, and false otherwise. In this setting, I satisfies a ground clause $\Gamma \to \Delta$ if $I \not\supseteq \Gamma$ or $I \cap \Delta \neq \emptyset$. The empty clause $\Box$ is hence satisfied by no interpretation, and is called a *contradiction*.

2.5 Constrained clauses

An *(ordering and equality) constraint* is a quantifier-free first-order formula built over the binary predicate symbols $>$ and $=$ relating terms in $T(\mathcal{F},\mathcal{X})$. Regarding semantics, the constraints are interpreted in $T(\mathcal{F})$: the symbol $=$ is interpreted as some congruence $=_c$ on $T(\mathcal{F})$ (like syntactic equality or AC-equality) and $>$ is interpreted as a given ordering $\succ$ (see section 2.11) on $T(\mathcal{F})$ that is total up to $=_c$. Hence a *solution* of a constraint T is a ground substitution σ whose domain is $vars(T)$

[1] The Herbrand universe requires at least one element. Recall that, in our setting, $T(\mathcal{F}) \neq \emptyset$, since we make the assumption that there is at least one constant symbol in $\mathcal{F}$.

and such that $T\sigma$ evaluates to true for the given $=_c$ and $\succ$. If a solution for T exists, then T is called *satisfiable*.

A *constrained clause* is a pair $C \mid T$ where C is a clause and T is a constraint. A *ground instance* of $C \mid T$ is a ground clause $C\sigma$ where σ is a solution of T. The semantics of $C \mid T$ is the set of all its ground instances. Hence, by definition, an interpretation I satisfies $C \mid T$ if $I \models C\sigma$ for every ground instance $C\sigma$ of $C \mid T$. Therefore, clauses with unsatisfiable constraints are tautologies, and $C \mid T$ is the *constrained empty clause* (satisfied by no interpretation), denoted as well by $\square$, if C is empty and T is satisfiable. Constrained clauses $C \mid T$ where T is a tautology are sometimes denoted by C, omitting the constraint part T.

2.6 Inference systems

A (clausal) logical *inference* is a step by which from a multiset of zero or more constrained clauses (the *premises*) a new clause (the *conclusion*) is obtained. An *inference rule* $\mathcal{R}$

$$\frac{C_1 \mid T_1 \ \dots \ C_n \mid T_n}{D \mid T} \quad \text{if} \quad \textit{condition}$$

is a finite representation of the set of inferences where from the multiset of clauses of the form $\{C_1 \mid T_1 \ \dots \ C_n \mid T_n\}$ one can infer $D \mid T$ if *condition* holds. One such an inference is called an *inference by* $\mathcal{R}$. An *inference system Inf* is a set of inference rules. An *inference by Inf* is an inference by one of the rules of *Inf*. We mainly consider inference rules where premises and conclusions have constraints that are tautologies, and hence these constraints are omitted.

An inference rule $\mathcal{R}$ is *correct*, or *sound*, if, for all inferences by $\mathcal{R}$, the conclusion is a logical consequence of the premises, and an inference system is correct if all its rules are correct. A set of clauses S is *closed* under an inference system *Inf* if, for every inference by *Inf* with premises in S, the corresponding conclusion is in S. An inference system *Inf* is *refutation complete* if $\square \in S$ for every unsatisfiable set of clauses S which is closed under *Inf*. All inference systems in this thesis are easily proved correct, and we will focus on completeness.

2.7 Term Rewrite Systems

If $\rightarrow$ is a binary relation, then $\leftarrow$ is its inverse[2], $\leftrightarrow$ is its symmetric closure, $\xrightarrow{+}$ is its transitive closure and $\xrightarrow{*}$ is its reflexive-transitive closure. A *derivation* for a binary relation $\rightarrow$ is a (finite or infinite) sequence of the form $s_1 \rightarrow s_2 \rightarrow \cdots$. An element t is *reducible* w.r.t. a given binary relation $\rightarrow$ if there is an element u such that $t \rightarrow u$; otherwise, t is called *irreducible*. We write $s \xrightarrow{!} t$ if $s \xrightarrow{*} t$ and t is *irreducible* w.r.t. $\rightarrow$. Then t is called a *normal form* of s (w.r.t. $\rightarrow$). A relation $\rightarrow$ is called *normalizing* (or *weakly terminating*) if every element has at least one normal form. If an element s has exactly one normal form then it is denoted by $s!_{\rightarrow}$. A relation $\rightarrow$ is *terminating* (or *strongly normalizing*) if there exists no endless derivation $s_1 \rightarrow s_2 \rightarrow \cdots$. Termination demands more than that the relation be normalizing, since the latter allows some derivations to be infinite. The composition of two relations is indicated by $\circ$. Thus $s \xleftarrow{*} \circ \xrightarrow{*} t$ if there is an element r such that $s \xleftarrow{*} r$ and $r \xrightarrow{*} t$. A relation $\rightarrow$ is *confluent* if the relation $\xleftarrow{*} \circ \xrightarrow{*}$ is contained in the joinability relation $\xrightarrow{*} \circ \xleftarrow{*}$. It is *locally confluent* if $\leftarrow \circ \rightarrow \ \subseteq \ \xrightarrow{*} \circ \xleftarrow{*}$. By Newman's lemma, terminating locally-confluent relations are confluent. Confluence means that any two derivations from a given element r can always be "brought together",

[2]Here, and everywhere, we use the mirror image of a binary relation symbol for its inverse.

that is, if $s \overset{*}{\leftarrow} r$ and $r \overset{*}{\rightarrow} t$ for some s and t then $s \overset{*}{\rightarrow} u$ and $u \overset{*}{\leftarrow} t$ for some u, which is equivalent to the *Church-Rosser* property that $s \overset{*}{\rightarrow} u$ and $u \overset{*}{\leftarrow} t$ for some u whenever $s \overset{*}{\leftrightarrow} t$.

A relation $\rightarrow$ on terms is *monotonic*, or *stable under contexts*, if $s \rightarrow t$ implies $u[s]_p \rightarrow u[t]_p$ for all terms s and t, contexts u, and positions p in u. It is *stable* (or *closed*) *under substitutions* if $s \rightarrow t$ implies $s\sigma \rightarrow t\sigma$ for all terms s and t and substitutions σ. It is a *rewrite relation* if it is both monotonic and stable under substitutions. A *rewrite rule* is an ordered pair of terms (s,t), written $s \rightarrow t$, and a set of rewrite rules R is a *rewrite system*, or (more specifically) a *term rewrite system* (TRS). The *rewrite relation* with R on $T(\mathcal{F},\mathcal{X})$, denoted $\rightarrow_R$, is the smallest monotonic relation such that for all rules $l \rightarrow r$ in R and substitutions σ, $l\sigma \rightarrow_R r\sigma$. That is, $s \rightarrow_R t$ if and only if $s \equiv u[l\sigma]_p$ and $t \equiv u[r\sigma]_p$ for some context u, position p in u, rule $l \rightarrow r$ in R and substitution σ; then we say that *s rewrites into t* with R, and that s is *rewritable*, or *reducible*, by R. We use R as an abbreviation for the binary relation $\rightarrow_R$. A subterm $s|_p$ at which a rewrite (with R) can take place is called a (R-)*redex*. A redex r is *innermost* for R if it is an R-redex but no proper subterms of r are R-redexes. An occurrence of a redex r in a term t is *outermost* for R if the occurrence of r is not a proper subterm of any other R-redexes of t. A derivation $s_1 \rightarrow_R s_2 \rightarrow_R \cdots$ for a rewrite system R is called a *rewrite sequence* with R. It is called *innermost* if all rewrite steps take place at innermost redexes, and *outermost* if they take place at outermost redexes. A rewrite system R is called terminating, confluent, etc. if $\rightarrow_R$ is. It is called *convergent* if it is confluent and terminating. It is said to be *(inter-)reduced* if, for each rule $l \rightarrow r$ in R, the right-hand side r is irreducible by R and the left-hand side l is not reducible by any other rule in R. We will reserve the word *canonical* for reduced convergent rewrite systems, though, in the literature, “canonical” is sometimes synonymous with “convergent”.

2.8 Termination of rewriting

Confluence and termination are, among all the properties of term rewriting, the most important (cf. Section 2.9). Rewrite systems are directed equations used to compute by repeatedly replacing subterms of a given formula with “equal” terms until the simplest possible form is obtained. Hence, termination of rewriting is a crucial property. Within automated theorem proving, equations can be used freely for *simplification* provided progress toward normal forms is guaranteed. The trick is to ensure that one can delete pre-simplified formulae without compromising the completeness of the prover (cf. Section 1.2.2). Confluence is important as well, since it guarantees that, regardless of the nondeterministic choices we make in a rewrite sequence, the same normal form will be reached at the end, if any. If a rewrite system R is terminating, then the local confluence (and confluence) of R is decidable. However, these properties are undecidable in general. It is not our intention to discuss here any technique for proving termination (nor confluence), since this is out of the scope of this thesis, but just to give some basic facts that characterize non-termination.

As mentioned in the previous section, a rewrite system R is called terminating if $\rightarrow_R$ is. That is, if there is no endless rewrite sequence $t_1 \rightarrow_R t_2 \rightarrow_R \cdots$ of terms. The following is a well-known example of a non-terminating rewrite system [Toy87]:

$$\begin{array}{lll} f(a,b,x) & \rightarrow & f(x,x,x) \\ g(x,y) & \rightarrow & x \\ g(x,y) & \rightarrow & y \end{array}$$

The following endless sequence *cycles*, i.e., there is a term in the sequence which coincides with a previous one:

$$f(a,b,g(a,b)) \rightarrow f(g(a,b),g(a,b),g(a,b)) \rightarrow f(a,g(a,b),g(a,b)) \rightarrow f(a,b,g(a,b)) \rightarrow \cdots$$

Although cycling rewrite systems are obviously non-terminating, a non-cycling rewrite system may be non-terminating as well. The following is a trivial non-cycling and non-terminating rewrite system:

$$a \;\rightarrow\; f(a)$$

The following endless sequence *loops*, i.e., there is a term in the sequence with a (not necessarily proper) subterm which coincides with a previous term in the sequence:

$$a \rightarrow f(a) \rightarrow f(f(a)) \rightarrow \cdots$$

It is also obvious that looping systems do not terminate, since we can infinitely repeat the same steps in the subterm which creates the loop. However a TRS needs not be looping to be non-terminating.

The *embedding relation*, defined as the monotonic closure of the (non-strict) subterm relation, is a weaker relation which characterizes non-termination. Roughly speaking, a term s is embedded in a term t if s is syntactically simpler or equal to t, i.e., s is obtained by removing some symbols of t, or $s \equiv t$. For example, $f(g(a))$ embeds $f(a)$. The following theorem from Kruskal allows us to characterize non-termination:

Theorem 2.8.1 (Tree theorem [Kru60]) *Any infinite sequence $t_1, t_2, \ldots$ of terms in a set $\mathcal{T}$ with finitely many function symbols and constants contains two terms t_i and t_j with $i < j$ such that t_i is embedded in t_j.*

When a derivation contains two terms t_i and t_j with $i < j$ such that t_i is embedded in t_j, it is said that contradicts the embedding relation. *Self-embedding* derivations are defined as derivations which contradict the embedding relation. A rewrite system is *self-embedding* if it allows a self-embedding derivation [Der82]. Thus self-embedding characterizes non-termination of rewriting, as a consequence of Kruskal's theorem:

Corollary 2.8.2 (Non-termination theorem [Der82]) *If a finite rewrite system is non-terminating, then it must be self-embedding.*

The non-terminating TRS

$$f(g(x)) \;\rightarrow\; g(g(f(f(x))))$$

contains the following self-embedding derivation

$$f(f(g(a))) \rightarrow f(g(g(f(f(a))))) \rightarrow g(g(f(f(g(f(f(a))))))) \rightarrow \cdots$$

However, self-embeddingness does not imply non-termination. That is, not every TRS containing a self-embedding derivation is non-terminating. The following TRS

$$f(f(x)) \;\rightarrow\; f(g(f(x)))$$

is terminating (take as a measure that decreases in every step the number of consecutive occurrences of the function symbol f) even though all instances of the rule are self-embedding.

2.9 Rewriting as a deductive mechanism

If E is an equational system, then by $E_{\rightarrow}$ we denote the set $\bigcup_{s\simeq t\in E}\{s \rightarrow t\}$. We define $u \rightarrow_E v$ as $u \rightarrow_{E_{\rightarrow}} v$. Note that $\rightarrow_E$ coincides with $\leftrightarrow_E$, since we view the equations as unordered pairs (cf. Section 2.2). By Birkhoff's theorem, $E \models s \simeq t$ (E logically implies $s \simeq t$) iff $s \overset{*}{\leftrightarrow}_E t$. If R is a TRS, then by $R_{\simeq}$ we denote the set $\bigcup_{l\rightarrow r\in R}\{l \simeq r\}$. We write $s =_E t$ for $E \models s \simeq t$, and $s =_R t$ for $R_{\simeq} \models s \simeq t$. Sometimes the congruence relations $\overset{*}{\leftrightarrow}_R$ (or $\overset{*}{\leftrightarrow}_E$) are denoted by R^* (E^*).

Convergent TRS are especially interesting, since such systems R can be used for theorem proving in their equational theory $R_{\simeq}$. It is not difficult to see that, when R is convergent, then every term t has a unique normal form w.r.t. $\rightarrow_R$, denoted $nf_R(t)$: termination implies the existence of, at least, one normal form, and confluence implies the impossibility of more than one normal form. This fact, together with Birkhoff's theorem and the Church-Rosser property, allow us to connect the equational theory of R with rewriting in the following way: $s =_R t$ iff $s \overset{*}{\leftrightarrow}_R t$ (by Birkhoff's theorem) iff $s \overset{*}{\rightarrow} u$ and $u \overset{*}{\leftarrow} t$ for some u (by Church-Rosser property) iff $nf_R(s) \equiv nf_R(t)$ (by termination). The latter is a directed form of theorem proving for such an equational theory. In other words: since, for convergent systems, the choice among possible rewrites at each step does not affect the normal form computed for any given input term (because it is unique), then $s =_R t$ can be decided by (nondeterminalistically) rewriting s and t with R to any normal form and checking if these are identical. A *rewrite ("valley") proof* of an equation $s \simeq t$ with a rewrite system R is a finite sequence $s_1, s_2, \ldots, s_n, t_m, t_{m-1}, \ldots, t_1$ where $s_1 \equiv s$, $t_1 \equiv t$, $s_n \equiv t_m$ and for all i, $1 \leq i < n$, $s_i \rightarrow_R s_{i+1}$ and for all j, $1 \leq j < m$, $t_j \rightarrow_R t_{j+1}$.

2.10 Critical pairs

Let $l \rightarrow r$ and $s \rightarrow t$ be two rules. We say that the left-hand side l *overlaps* the left-hand side s if there is a *non-variable* subterm $s|_p$ of s such that l and $s|_p$ are both instances of a common term u. To determine overlap, the variables in the two (not necessarily distinct) rules are renamed, if necessary, so that they are disjoint. Then, l overlaps s if there exists a unifying substitution σ such that $l\sigma \equiv s\sigma|_p$. When there is an overlap, the overlapped term $s\sigma$ can be rewritten to either $t\sigma$ or $s\sigma[r\sigma]_p$. The two-step proof $t\sigma \leftarrow_R s\sigma[l\sigma]_p \rightarrow_R s\sigma[r\sigma]_p$ is called a *critical peak.*

Definition 2.10.1 *If $l \rightarrow r$ and $s \rightarrow t$ are two rewrite rules (with distinct variables), p is the position of a non-variable subterm of s, and μ is a most general unifier of $s|_p$ and l, then the equation $t\mu \simeq s\mu[r\mu]_p$ is a* critical pair *formed from those rules.*

Let $\mathbf{cp}(R)$ denote the set of all critical pairs between (not necessarily distinct, but perhaps renamed) rules in R and let $\underset{\mathbf{cp}(R)}{\longleftrightarrow}$ denote its symmetric rewrite closure.

Lemma 2.10.2 (Critical Pair Lemma [KB70]) *For any rewrite system R and peak $s \leftarrow_R u \rightarrow_R t$, there either exists a rewrite proof $s \overset{*}{\rightarrow}_R v \overset{*}{\leftarrow}_R t$ or a critical-pair proof $s \underset{\mathbf{cp}(R)}{\longleftrightarrow} t$.*

The proof considers all relative positions of the two redexes. As stressed in [Hue80], no assumption of termination is necessary for this lemma. It follows from this and Newman's lemma, that a terminating system R is confluent iff $\mathbf{cp}(R)$, regarded as a relation, is a subset of the joinability relation $\overset{*}{\rightarrow}_R \circ \overset{*}{\leftarrow}_R$. Since finite systems have a finite number of critical pairs, their confluence is decidable, provided they are terminating. This criterion for confluence is called the *superposition test* [KB70].

Without termination, a system may have no critical pairs (hence be locally confluent), and still be non-confluent. A system without critical pairs (except trivial ones of the form $t \simeq t$) is called *non-overlapping*.

2.11 Term orderings

A (strict partial) *ordering* is a transitive and irreflexive binary relation. An ordering $\succ$ on $T(\mathcal{F},\mathcal{X})$ is *well-founded* if there exists no infinite descending chain $t_1 \succ t_2 \succ \cdots$ of elements of $T(\mathcal{F},\mathcal{X})$. Thus, a relation $\rightarrow$ is terminating iff its transitive closure $\xrightarrow{+}$ is a well-founded ordering. The importance of terminating relations lies in the possibility of inductive proofs in which the hypothesis is assumed to hold for all elements t such that $s \xrightarrow{+} t$ when proving it for an arbitrary s. An ordering $\succ$ on terms fulfills the *subterm property* if $\succ \supseteq \rhd$, where $s \rhd t$ if t is a proper subterm of s. It is total on $T(\mathcal{F})$ if for all s and t in $T(\mathcal{F})$, either $s \equiv t$ or $s \succ t$ or $t \succ s$. If we have either $s \sim t$ or $s \succ t$ or $t \succ s$, where $\sim$ is an equivalence relation different from syntactic equality, we speak of totality *up to* $\sim$.

A *rewrite ordering* is an ordering which is both monotonic and stable under substitutions. In other words, it is a transitive and irreflexive rewrite relation. A *reduction ordering* is a well-founded rewrite ordering, and a *simplification ordering* is a rewrite ordering with the subterm property. The following properties are not difficult to check:

- A reduction ordering which is total on $T(\mathcal{F})$ is necessarily a simplification ordering on $T(\mathcal{F})$.
- By Kruskal's theorem, simplification orderings are well-founded (for finite, fixed-arity signatures). The crucial point is that they contain the strict embedding relation.
- A rewrite system R is terminating if, and only if, all its rules are contained in a reduction ordering $\succ$, i.e., $l \succ r$ for every $l \rightarrow r$ in R (symbolically: $R \subseteq \succ$). In fact, then $\xrightarrow{+}_R$ is itself a reduction ordering.

A *quasi-ordering* $\succeq$ is a reflexive and transitive binary relation; the associated equivalence relation $\sim$ is the intersection of $\succeq$ with its inverse; the associated strict partial ordering $\succ$ is their difference. Hence, $\succeq$ is the disjoint union of $\succ$ and $\sim$. We say that a relation $\succ$ is compatible with an equivalence relation $\sim$ if $s \sim s' \succ t' \sim t$ implies $s \succ t$. It follows that, if $\succ$ is an ordering which is compatible with an equivalence relation $\sim$, then $\succ \cup \sim$ is a quasi-ordering whose associated strict partial ordering is $\succ$. If E is a set of equations and $\succ$ is an ordering that is compatible with $=_E$, then $\succ$ is called *E-compatible*. Note that if $\succ$ is E-compatible, then $s =_E t$ implies $s \not\succ t$ and $t \not\succ s$.

Some quasi-orderings play an important role in our results. The *subsumption* relation, denoted by $\geq$, is a quasi-ordering defined as $s \geq t$ if s is an instance of t, i.e., $s \equiv t\sigma$ for some σ. The equivalence relation associated with $\geq$ is denoted by $\doteq$. Note that if $s \doteq t$ then t is a variable renamed version (or a *variant*) of s. The *encompassment* relation, denoted by $\trianglerighteq$, is a quasi-ordering defined as $s \trianglerighteq t$ if a subterm of s is an instance of t, i.e., $s \trianglerighteq t\sigma$ for some σ; therefore, encompassment is the composition of the subterm and the subsumption relations.

Let $\succ$ be an ordering and $\sim$ an equivalence relation such that $\succ$ is compatible with $\sim$. Then these relations induce relations on tuples and multisets of terms as follows.

The *lexicographic (left to right) extension of $\succ$ with respect to $\sim$* is the relation $\succ^{lex}$ on n-tuples of terms defined by

$$\langle s_1,\ldots,s_n\rangle \succ^{lex} \langle t_1,\ldots,t_n\rangle \quad \text{if} \quad s_1 \sim t_1,\ldots,s_{k-1} \sim t_{k-1} \text{ and } s_k \succ t_k$$

for some k in $1\ldots n$. It is well-known that, if $\succ$ is well-founded and compatible with $\sim$, then $\succ^{lex}$ is well-founded.

This definition can be generalized in several ways. We are interested in the one of having not only n-tuples, but also n ordering relations $\succ_1,\ldots,\succ_n$ (and the corresponding equivalence relations $\sim_1,\ldots,\sim_n$). Then each component is compared with its own ordering, i.e.,:

$$\langle s_1,\ldots,s_n\rangle \succ^{lex} \langle t_1,\ldots,t_n\rangle \quad \text{if} \quad s_1 \sim_1 t_1,\ldots,s_{k-1} \sim_{k-1} t_{k-1} \text{ and } s_k \succ_k t_k$$

for some k in $1 \dots n$. Again, we have that, if $\succ_1, \dots, \succ_n$ are well-founded and respectively compatible with $\sim_1, \dots, \sim_n$, then $\succ^{lex}$ is well-founded.

The *multiset extension of* $\sim$ is defined as the smallest relation $\sim^{mul}$ on multisets of terms such that

$$\emptyset \sim^{mul} \emptyset$$

and

$$S \cup \{s\} \sim^{mul} S' \cup \{t\} \quad \text{if} \quad S \sim^{mul} S' \wedge s \sim t.$$

The *multiset extension of* $\succ$ *with respect to* $\sim$ is defined as the smallest ordering $\succ^{mul}$ on multisets of terms such that

$$M \cup \{s\} \succ^{mul} N \cup \{t_1, \dots, t_n\} \quad \text{if} \quad M \sim^{mul} N \text{ and } s \succ t_i \text{ for all } i \in 1 \dots n.$$

Sometimes the notation $\succ^{mul}$ is used without explicitly indicating which is the equivalence relation $\sim$. In these cases $\sim$ is assumed to be the syntactic equality relation $\equiv$. If $\succ$ is well founded on a set S, so is $\succ^{mul}$ on finite multisets over S [DM79].

2.11.1 Path orderings

A way to define suitable orderings for practical purposes (like proving termination or automated deduction) is to construct them directly from a well-founded *precedence*, an ordering $\succ_{\mathcal{F}}$ on $\mathcal{F}$. This is done in the so-called *path orderings*, like the *lexicographic path ordering* (LPO) or the *recursive path ordering (with status)* (RPO) [KL80, Der82].

Let $\succ_{\mathcal{F}}$ be a precedence and let $\mathcal{F}$ be the disjoint union of two sets *lex* and *mul*, the symbols with lexicographic and multiset *status*, respectively. By $=_{mul}$ we denote the equality of ground terms up to permutation of direct arguments of symbols with multiset status: $f(s_1, \dots, s_n) =_{mul} g(t_1, \dots, t_m)$ if $f \equiv g$ (and hence $n = m$) and $s_i = t_{\pi(i)}$ for $1 \le i \le n$, where π is a permutation of $1 \dots n$ which is the identity if $f \in lex$.

In this setting, RPO is defined as follows: $s \succ_{rpo} x$ if x is a variable that is a proper subterm of s or else $s \equiv f(s_1, \dots, s_n) \succ_{rpo} g(t_1, \dots, t_m) \equiv t$ if at least one of the following conditions holds:

- $s_i \succ_{rpo} t$ or $s_i =_{mul} t$ for some i in $1 \dots n$
- $f \succ_{\mathcal{F}} g$, and $s \succ_{rpo} t_j$ for all j in $1 \dots m$
- $f \equiv g$ (and hence $n = m$), $f \in mul$ and $\{s_1, \dots, s_n\} \succ_{rpo}^{mul} \{t_1, \dots, t_n\}$
- $f \equiv g$ (and hence $n = m$), $f \in lex$, $\langle s_1, \dots, s_n \rangle \succ_{rpo}^{lex} \langle t_1, \dots, t_n \rangle$, and $s \succ_{rpo} t_j$ for all j in $1 \dots n$

where $\succ_{rpo}^{lex}$ and $\succ_{rpo}^{mul}$ are, respectively, the lexicographic and multiset extensions of $\succ_{rpo}$ with respect to $=_{mul}$.

The *lexicographic path ordering* (LPO) is defined as the particular case of an RPO where $\mathcal{F} = lex$, i.e., where all symbols have a lexicographic status.

It is known that RPO is a reduction ordering on $T(\mathcal{F}, \mathcal{X})$, which is moreover total on $T(\mathcal{F})$ up to $=_{mul}$ (and hence, in case of LPO, total up to $\equiv$) if $\succ_{\mathcal{F}}$ is total on $\mathcal{F}$ [KL80, Der82]. LPO's are useful for extending reduction orderings $\succ$ that are total up to an equivalence relation $\sim$ (like RPO is total up to $=_{mul}$), to reduction orderings total up to $\equiv$. This extension is obtained by a lexicographic combination $\succ_t$, whose first component is $\succ$, and whose second component is a total LPO $\succ_{lpo}$. That is, $s \succ_t t$ if either $s \succ t$ or $s \sim t$ and $s \succ_{lpo} t$.

Since RPO is a reduction ordering, it can be used to prove termination. Recall that checking that the rules of a term rewrite system R are contained in a reduction ordering, like an RPO, suffices to prove termination of R. However, since RPO includes the subterm relation, it cannot prove termination of rules like $f(f(x)) \rightarrow f(g(f(x)))$, because the term $g(f(x))$ is greater than its subterm $f(x)$ and, hence, by monotonicity $f(g(f(x)))$ will always be greater than $f(f(x))$.

The *semantic path ordering* (SPO, [KL80]) is a well-known generalization of the RPO, where the precedence on function symbols is replaced by any (well-founded) underlying (quasi-)ordering involving the whole term rather than only its head symbol. This makes the ordering much more powerful. In fact, for every terminating TRS R, there is always some SPO that includes $\rightarrow_R^+$. Hence, it can handle rules like $f(f(x)) \rightarrow f(g(f(x)))$. SPO includes the subterm relation, but it is not monotonic in general (i.e., the monotonicity property is lost, even if the underlying quasi-ordering is monotonic). Then, although SPO characterizes termination, it turns out to be not so useful in practice. Note that, in absence of monotonicity, checking that the rules are contained in the ordering is not enough to ensure termination: one needs to check that the rules decrease when applied under any context.

An ordering which is better suited for practical purposes is the *monotonic semantic path ordering* (MSPO, [BFR00]). Unlike SPO, MSPO preserves the monotonicity of the underlying quasi-ordering which is built on, and still characterizes termination.

2.11.2 Combined systems

Termination of the union of term rewrite systems can be reduced to the termination of each. Let R and S be two binary relations contained in well-founded orderings $\succ_R$ and $\succ_S$, respectively. If $\succ_S$ *commutes over* $\succ_R$, i.e., if $\succ_R \circ \succ_S$ is contained in $\succ_S \circ \succ_R$, then the union $R \cup S$ is terminating. Actually, it suffices if $\succ_S$ *quasi commutes over* $\succ_R$, by which we mean that $\succ_R \circ \succ_S$ is contained in $\succ_S \circ \succeq_R$. For example, if R is a terminating rewrite system, then the union of $\rightarrow_R$ with the proper subterm relation $\rhd$ is well-founded, since taking subterms is well-founded and if a subterm rewrites then so does the superterm. More generally, the union of any terminating rewrite relation with the (proper) encompassment ordering $\blacktriangleright$ is also well-founded, since encompassment is just $>$ (subsumption) and/or $\rhd$, both of which are well-founded, and any rewrite relation commutes over both. To prove termination of a combined term rewrite system $R \cup S$, it is necessary and sufficient that R and S be contained in reduction orderings that commute as above.

We also have that, if R is a binary relation contained in a well-founded ordering $\succ_R$, S is a binary relation contained in a congruence $\sim_S$, and $\succ_R$ commutes over $\sim_S$, then the composite relation $R \circ S$ is well-founded.

2.12 Equality Herbrand interpretations and non-equality predicates

Let R be a ground TRS. Then the congruence $\overset{*}{\leftrightarrow}_R$ defines an *equality* Herbrand interpretation where $\simeq$ is interpreted by $s \simeq t$ iff $s \overset{*}{\leftrightarrow}_R t$ (cf. Section 2.4). Such an interpretation will be denoted by R^*.

In this framework, equality can be considered to be the only predicate, since for every other predicate symbol P, (positive or negative) atoms $P(t_1, \ldots, t_n)$ can be expressed as (positive or negative) equations $P(t_1, \ldots, t_n) \simeq true$, where *true* is a new constant symbol, and where P is considered as a function symbol rather than as a predicate symbol. Note however that, in order to avoid meaningless expressions in which predicate symbols occur at proper subterms, one should adopt a two-sorted type discipline on terms in the encoding.

It is easy to see that this transformation preserves satisfiability. Very roughly: one can "translate" the interpretations such that a ground atom is true in a Herbrand interpretation I if and only if in the equality Herbrand interpretation I' over the modified signature the term $P(t_1,\ldots,t_n)$ is congruent to *true*. We remark that I and I' are not isomorphic since two ground atoms that are false in I need not be in the same congruence class of I'.

After this satisfiability preserving transformation, resolution (ground) inferences of the form

$$\frac{\Gamma' \rightarrow A \qquad \Gamma, A \rightarrow \Delta}{\Gamma', \Gamma \rightarrow \Delta}$$

become a special case of paramodulation

$$\frac{\Gamma' \rightarrow A \simeq \mathit{true} \qquad \Gamma, A \simeq \mathit{true} \rightarrow \Delta}{\Gamma', \Gamma, \mathit{true} \simeq \mathit{true} \rightarrow \Delta}$$

combined with equality resolution for eventually eliminating the trivial equation $\mathit{true} \simeq \mathit{true}$.

In the context of ordered paramodulation, it is convenient to make *true* small in the ordering for efficiency reasons. Sometimes it is also useful to take into account that P is a predicate symbol when handling the ordering restrictions. For example, in orderings like RPO, if the predicate symbols are bigger in the precedence than function symbols then $P \succ_{\mathcal{F}} Q$ always implies $P(t_1,\ldots,t_n)\sigma \succ_{rpo} Q(s_1,\ldots,s_m)\sigma$ for all ground σ.

Chapter 3

Paramodulation with west orderings

All the completeness results for ordered paramodulation mentioned in Section 1.2.1 require the term ordering $\succ$ to be well-founded, monotonic and total (or extendable to a total ordering) on ground terms. For several applications, these requirements are too strong, and hence weakening them has been a well-known research challenge.

Here we introduce a new completeness proof technique for ordered paramodulation where the only properties required on $\succ$ are well-foundedness and the subterm property. This solves a well-known open problem posed by Michaël Rusinowitch in the RTA list of open problems [RTA01] (problem no. 85) since 1995: "Ordered paramodulation is known to be complete for simplification orderings that are total on ground terms [HR86]. Other theorem proving strategies are similarly restricted. How can these restrictions be relaxed?".

The technique is a variant of the model generation method of [BG94b], involving a relatively simple application of some fundamental results on the termination and confluence of ground term rewrite systems.

3.1 Introduction

All main completeness proof techniques for ordered paramodulation, like the transfinite semantic tree method [HR91], the proof ordering method [BDH86, BD94], and the model generation method [BG94b] require the term ordering $\succ$ to be well-founded, monotonic and total(izable) on ground terms.

But, in many practical situations, these requirements are too strong. On the one hand, in deduction modulo built-in equational theories E (see Section 1.2.5) , the existence of a total (up to E-equal ground terms) E-compatible reduction (i.e., well-founded, monotonic) ordering is a very strong requirement. For example, the existence of such an ordering for the case where E consists of associativity and commutativity (AC) properties for some symbols remained open for a long time.[1] Unfortunately, for many E such orderings cannot exist. For example, when E contains an idempotency axiom $f(x,x) \simeq x$, then, if $s \succ t$, by monotonicity one should have $f(s,s) \succ f(s,t)$, which by E-compatibility implies $s \succ f(s,t)$ and hence non-well-foundedness. On the other hand, for efficiency reasons, one may wish to use an ordering that cannot be extended to a total one: e.g., an ordering $\succ$ including $f(a) \succ f(b)$ and $g(b) \succ g(a)$, for which a and b must be incomparable in any monotonic extension, or an ordering including $f(f(x)) \succ f(g(f(x)))$, for which $f(x)$ and $g(f(x))$ must be

[1] Once it was found [NR91, RN95], it immediately led to a number of follow-up results, like the decidability of the ground AC-word and -unification problems [Mar91, NR93].

incomparable in any monotonic extension.

In this chapter we introduce techniques for dropping the monotonicity requirement. The only properties required for $\succ$ are well-foundedness and the subterm property. Note that any such ordering can be totalized without losing these two properties. Then the totalized ordering is called *west* after well-founded, subterm and total.

Our technique is a variant of the model generation method of [BG94b]. In this standard technique for proving completeness of ordered paramodulation calculi, a ground TRS R is *generated* from a closed (or saturated) set of clauses S, such that R^* defines an equality Herbrand interpretation (see Section 2.12) which is a model of S if $\square \notin S$ (hence the name of "model generation"). The rules in R are oriented by the ordering $\succ$ used to paramodulate with, and confluence and termination of R are crucial properties for the completeness proof. In the standard model generation method, termination of R follows directly from the fact that $R \subseteq \succ$, since it assumed that $\succ$ is a total reduction ordering on ground terms. Then the completeness proof is done by induction w.r.t. $\succ$.

In our variant, termination of the ground rewrite system R that defines the model is not a consequence of the ordering. Instead, termination of R follows from other properties. In one of the settings treated here, it follows from the irreducibility, w.r.t. R itself, of the right-hand sides of all rules of R; in another setting, if R is contained in a well-founded ordering $\succ$ with the subterm property, its termination follows from the irreducibility of the right-hand sides at non-topmost positions only. Since each terminating TRS R induces a reduction (i.e., well-founded, monotonic) ordering $\xrightarrow{+}_R$, we can then use induction w.r.t. (an extension of) this reduction ordering for proving the main completeness results. The fact that the ordering used to paramodulate with and the ordering used for the induction in the completeness proof do not need to coincide is the key for establishing completeness of ordered paramodulation with non-monotonic orderings. Note that, if $\succ$ is not monotonic, it may happen that $\xrightarrow{+}_R$ is not included in $\succ$ although $R \subseteq \succ$, and in fact $\xrightarrow{+}_R$ may even contradict $\succ$.

The results given here are for paramodulation with general first-order clauses with eager selection of negative literals. In Section 3.6 we briefly mention other possible strategies.

3.2 West orderings

Here we define the kind of non-monotonic ordering that we assume, in this Chapter, it is used to paramodulate with. We also give some related properties.

Definition 3.2.1 *A* west ordering *is a well-founded ordering on $T(\mathcal{F})$ that fulfills the subterm property and that is total on $T(\mathcal{F})$ (it is called* west *after well-founded, subterm and total).*

Not all well-founded orderings on terms can be extended to west orderings, even if they do not contradict the subterm property. For example, if $a \succ_1 f(b)$ and $b \succ_1 f(a)$, then, if $\succ$ is $(\succ_1 \cup \rhd)^+$, we get $a \succ f(b) \succ b \succ f(a) \succ a$. But every well-founded ordering can be totalized [Wec91], and hence every well-founded ordering satisfying the subterm property can be extended to a west ordering. We also have the following:

Lemma 3.2.2 *Every reduction ordering $\succ_r$ can be extended to a west ordering.*

Proof. It is well-known that $\succ_r \cup \rhd$ is well-founded (c.f. Section 2.11.2). Then $(\succ_r \cup \rhd)^+$ is a well-founded ordering and, since it satisfies the subterm property, every totalization of it gives us a west ordering. $\square$

Lemma 3.2.3 *Let R be a ground TRS such that for all rules $l \to r$ in R the term r is irreducible by R. Then R is terminating.*

Proof. Assume R is non-terminating. Then there exists an infinite rewrite sequence $t_1 \to_R t_2 \to_R \cdots$ It is easy to extract an infinite subsequence $s_1 \to_R s_2 \to_R \cdots$ of it where there is at least one rewrite step $s_i \to_R s_{i+1}$ at the topmost position, i.e., where $s_i \equiv l$ and $s_{i+1} \equiv r$ for some rule $l \to r$ in R. But then s_{i+1} is irreducible by R, contradicting the infiniteness assumption. $\square$

Lemma 3.2.4 *Let $\succ$ be a well-founded ordering, and let R be a ground TRS such that, for all $l \to r$ in R, $l \succ r$ and r is irreducible by R at non-topmost positions. Then R is terminating.*

Proof. If R is non-terminating there is an infinite sequence $s_1 \to_R s_2 \to_R \cdots$ with at least one rewrite step $s_i \to_R s_{i+1}$ at the topmost position, i.e., where $s_i \equiv l$ and $s_{i+1} \equiv r$ for some rule $l \to r$ in R. But then $s_{i+1} \to_R s_{i+2} \to_R \cdots$ is an infinite sequence with steps only at topmost positions, where $s_{i+1} \succ s_{i+2} \succ \cdots$ contradicting the well-foundedness of $\succ$. $\square$

3.2.1 West orderings in practice

In practical applications (theorem provers, implementations of Knuth-Bendix completion) the west ordering $\succ$ can be defined and dealt with in different ways.

One possibility is that an approximation of $\succ$ is available by a non-total ordering $\succ_v$ on terms with variables such that $s \succ_v t$ implies $s\sigma \succ t\sigma$ for all ground σ (e.g., when $\succ_v$ is a reduction ordering of which $\succ$ is an extension, as in Lemma 3.2.2). Then, inferences with ordering restrictions like $l\sigma\theta \succ r\sigma\theta$ for some ground substitution θ can be proved redundant by showing that $r\sigma \succeq_v l\sigma$. Indeed, the actual west ordering needs not always be really built for application purposes. It suffices for completeness that it exists, and for practice that a reasonably good approximation is available.

We now mention two general-purpose techniques for defining $\succ$ such that in practice it can be used or approximated efficiently.

Semantic path orderings

Recall from Section 2.11 that the *recursive path ordering* with status (RPO) —which includes the *lexicographic path ordering* (LPO)— is a well-known, easy to implement, general-purpose ordering for deduction purposes, and that the *semantic path ordering* (SPO) is a well-known powerful generalization of the RPO. SPO includes the subterm relation, but it is not monotonic in general.

Since the results of this chapter imply that for deduction by ordered paramodulation monotonicity is not needed, and any SPO can be extended to a west ordering, SPO is an interesting candidate for our purposes. That is, one can apply SPO in our context as a sufficient condition for ruling out unnecessary inferences.

Furthermore, in Chapter 4 of this thesis we will show that if R is a convergent TRS for some set of equations E, and $\xrightarrow{+}_R$ is included in a west ordering (like an SPO), then unfailing Knuth-Bendix completion on E with this west ordering will compute this R.

Non-monotonic E-compatible orderings

As mentioned in the beginning of this chapter, an important bottleneck for defining deduction techniques modulo built-in equational theories E is to find the required total (up to E-equal ground terms),

E-compatible reduction ordering. But if the monotonicity requirement can be dropped, this becomes a much simpler task.

Let us consider as an example such an ordering for the case where E consists of associativity and commutativity (AC) properties. The AC-compatibility of such an ordering $\succ$ means that $s =_{\mathrm{AC}} s' \succ t =_{\mathrm{AC}} t'$ implies $s' \succ t'$. It is easy to check whether two ground terms are AC-equal by using their *flattened* forms: a term s can be *flattened* by removing all AC-operators f that are immediately below another f. For example, if f and g are AC-operators, then the term $h(f(f(a,a),f(b,g(c,g(d,e)))))$ is flattened into $h(f(a,a,b,g(c,d,e)))$. Two terms are AC-equal if, and only if, their flattened forms are equal up to permutation of arguments of AC-operators.

Therefore, when trying to define total AC-compatible orderings, the first idea that comes to mind is to apply general-purpose orderings on the flattened forms of the terms to be compared, like an RPO where AC-symbols have multiset status. But the resulting ordering may not be monotonic: if f is an AC-operator that is larger than g, then $f(a,a)$ will be larger than $g(a,a)$, but $f(g(a,a),a)$ will be larger than $f(f(a,a),a)$, because the latter term becomes $f(a,a,a)$ after flattening. However, it is not difficult to see that one obtains an AC-compatible west ordering from this simple approach.

3.3 Paramodulation with equations

In this section we introduce part of our ideas for the purely equational case. Dealing with this simple case first is useful not only for explanation purposes, but also because its results will be used in Chapter 4 on Knuth-Bendix completion. In the following, let $\succ$ be a given west ordering.

Definition 3.3.1 *The inference rule of (equational)* ordered paramodulation *with respect to $\succ$ is*

$$\frac{l \simeq r \qquad s \simeq t}{(s[r]_p \simeq t)\sigma},$$

where $\sigma = mgu(s|_p, l)$, the most general unifier of $s|_p$ and l, where $s|_p$ is not a variable, and l is maximal in its premise, that is, for some ground substitution θ, it holds that $l\sigma\theta \succ r\sigma\theta$.

It is sometimes said that the inference takes place *with* the term l *on* the term s, and that the inference *involves* l and s. Note that, as usual, it may be necessary to rename variables in the premises in order to avoid name clashes and make the unification possible.

As said in the previous section, the usefulness of the ordering restrictions in practical applications depends on the way the west ordering $\succ$ is given. For example, if only an approximation is given by a non-total ordering $\succ_v$ on terms with variables such that $s \succ_v t$ implies $s\theta \succ t\theta$ for all ground θ, then the inference is not needed if, for instance, $r\sigma \succeq_v l\sigma$.

Note also that the inference is not necessary if $s|_p$ is a variable (this will extend to Horn clauses and general clauses as well). It shows that certain inferences between ground instances of the premises turn out to be redundant: at the non-ground level we do not perform, for an instance with θ, the inferences *inside* θ (also called inferences *into variables*, that is, on positions p of $s\theta$ where $s|_{p'}$ is a variable for some prefix p' of p). As mentioned in the introduction of this thesis, avoiding such inferences is especially desirable, since they are not constrained by unification (that is, any term unifies with a variable) and hence are difficult to control.

We now define, by induction w.r.t. $\succ^{mul}$, a (possibly infinite) ground TRS R_E *generated* by a set of equations E. We will see that, by construction, it follows that R_E is convergent and, moreover, R_E^* is a model of E whenever E is closed under ordered paramodulation with respect to $\succ$.

Definition 3.3.2 *Let E be a set of equations. Then, for every ground instance e of the form $l \simeq r$ of an equation in E, we define $Gen(e) = \{l \to r\}$, and say that e* generates *the rule $l \to r$ if, and only if,*

1. *$l \succ r$ and*
2. *l and r are irreducible by R_e,*

where R_e is the set of rules generated by all instances d of equations in E such that $e \succ^{mul} d$, that is, $R_e = \bigcup_{e \succ^{mul} d} Gen(d)$. In all other cases, $Gen(e) = \emptyset$. We denote by R_E the set of rules generated by all ground instances of E, that is, $R_E = \bigcup_{e \in gnd(E)} Gen(e)$.

The previous construction is similar to the one of [BG94b], but here the rules are oriented only by a (non-monotonic) west ordering. Hence the termination of R_E has to be ensured otherwise, and therefore we require not only the left-hand sides to be irreducible (which is required for confluence), but also the right-hand sides.

Property 3.3.3 *Let E be a set of equations. Then for all rules $l \to r$ in R_E we have*

1. *r is irreducible by R_E, and*
2. *l is irreducible by $R_E \setminus \{l \to r\}$.*

Proof. For the first property, by construction, if an instance e generates $l \to r$, the term r is irreducible by R_e. Since $l \succ r$, and $\succ$ is irreflexive and fulfills the subterm property, clearly $l \to r$ itself does not reduce r either. Finally, for every rule $l' \to r'$ generated by an instance d with $d \succ^{mul} e$, we must have $l' \succeq l$ and hence $l' \succ r$ which implies that l' cannot be a subterm of r either.
For the second property, by construction, if an instance e generates $l \to r$, then the term l is irreducible by R_e. Now, as before, for every rule $l' \to r'$ generated by an instance d with $d \succ^{mul} e$ we must have $l' \succeq l$. Since l' must be irreducible by R_d which contains $l \to r$ we have $l' \not\equiv l$, and hence $l' \succ l$, which implies that l' cannot be a subterm of l. □

Lemma 3.3.4 *For every set of equations E, the ground TRS R_E is convergent.*

Proof. Termination follows by Property 3.3.3.1 and Lemma 3.2.3. For confluence, since R_E is terminating, by Newman's lemma we only need to show that R_E is locally confluent, which holds by Property 3.3.3.2. □

Theorem 3.3.5 *Let E be a set of equations closed under ordered paramodulation with respect to a west ordering $\succ$. Then $R_E^* \models E$.*

Proof. Since R_E is terminating, it follows that $\xrightarrow{+}_{R_E}$ is a reduction ordering. We now proceed by induction w.r.t. the well-founded ordering on ground equations $(\xrightarrow{+}_{R_E})^{mul}$, denoted in the remainder of this proof by $\succ_R$. A contradiction is derived from the existence of a minimal w.r.t. $\succ_R$ ground instance e of the form $s\sigma \simeq t\sigma$ of an equation $s \simeq t$ in E such that $R_E^* \not\models e$. Since $R_E^* \not\models e$, the equation e is not a tautology of the form $u \simeq u$, and hence one of its sides is strictly larger w.r.t. the (total) ordering $\succ$ than the other one. Moreover, again since $R_E^* \not\models e$, the equation e has not generated any rule of R_E. This must be because either $s\sigma$ or $t\sigma$ is reducible by R_e. Then there exists some equation $l \simeq r$ that has generated a rule $l\sigma \to r\sigma$ reducing $s\sigma$ or $t\sigma$. We consider the case where $s\sigma$ is reducible; the other one is analogous. Now we have $s\sigma|_p \equiv l\sigma$, and there are two possibilities:

(i) An inference: $s|_p$ is a non-variable position of s.
Then there exists an inference by ordered paramodulation,

$$\frac{l \simeq r \qquad s \simeq t}{(s[r]_p \simeq t)\theta},$$

whose conclusion has an instance d of the form $(s[r]_p \simeq t)\sigma$, such that $e \succ_R d$ and $R_E^* \not\models d$, contradicting the minimality of e. Note that this inference satisfies the ordering constraints of ordered paramodulation, since $l\sigma \succ r\sigma$.

(ii) Lifting: $s|_{p'}$ is a variable x for some prefix p' of p.
Then $p = p' \cdot p''$ and $x\sigma|_{p''}$ is $l\sigma$. Now let σ' be the ground substitution with the same domain as σ but where $x\sigma' \equiv x\sigma[r\sigma]_{p''}$ and $y\sigma' \equiv y\sigma$ for all other variables y. Then $R_E^* \not\models s\sigma' \simeq t\sigma'$ and $e\sigma \succ_R e\sigma'$, contradicting the minimality of e. □

3.3.1 A slightly stronger paramodulation rule

In the proof of termination of R_E (Lemma 3.3.4) we can as well use Lemma 3.2.4 instead of Lemma 3.2.3. More precisely, we only need the right-hand sides of the rules in R_E to be irreducible at non-topmost positions (instead of being completely irreducible). Based on this observation, we can restrict our paramodulation rule avoiding its application at topmost positions of small sides of equations. Then we obtain the following *strict ordered paramodulation* rule.

Definition 3.3.6 *The inference rule of (equational)* strict ordered paramodulation *with respect to $\succ$ is*

$$\frac{l \simeq r \qquad s \simeq t}{(s[r]_p \simeq t)\sigma},$$

where $\sigma = mgu(s|_p, l)$, the most general unifier of $s|_p$ and l, where $s|_p$ is not a variable, and l is maximal in its premise, and, if $p = \lambda$, then s is also maximal in its premise, that is, for some ground substitution θ, it holds that $l\sigma\theta \succ r\sigma\theta$, and, if $p = \lambda$, then also $s\sigma\theta \succ t\sigma\theta$.

Now to prove Theorem 3.3.5, we slightly modify Definition 3.3.2, the generation of R_E, obtaining the following variant of it.

Definition 3.3.7 *Let E be a set of equations. Then, for every ground instance e of the form $l \simeq r$ of an equation in E, we define $Gen(e) = \{l \to r\}$, and say that e* generates *the rule $l \to r$ if, and only if,*

1. *$l \succ r$,*
2. *l is irreducible by R_e, and*
3. *r is irreducible by R_e at non-topmost positions.*

where R_e is the set of rules generated by all instances d of equations in E such that $e \succ^{mul} d$, that is, $R_e = \bigcup_{e \succ^{mul} d} Gen(d)$. In all other cases, $Gen(e) = \emptyset$. We denote by R_E the set of rules generated by all ground instances of E, that is, $R_E = \bigcup_{e \in gnd(E)} Gen(e)$.

We can now adapt Property 3.3.3, and prove it analogously. From this property, Lemma 3.3.4 holds as before but using Lemma 3.2.4 to conclude termination.

Property 3.3.8 *Let E be a set of equations. Then for all rules $l \to r$ in R_E we have*

1. *r is irreducible by R_E at non-topmost positions, and*
2. *l is irreducible by $R_E \setminus \{l \to r\}$.*

Now we adapt the proof of Theorem 3.3.5 for the strict ordered paramodulation rule.

Theorem 3.3.9 *Let E be a set of equations closed under strict ordered paramodulation with respect to a west ordering $\succ$. Then $R_E^* \models E$.*

Proof. As in Theorem 3.3.5, we proceed by induction w.r.t. the well-founded ordering on ground equations $\succ_R$, deriving a contradiction from the existence of a minimal w.r.t. $\succ_R$ ground instance e of the form $s\sigma \simeq t\sigma$ of an equation $s \simeq t$ in E such that $R_E^* \not\models e$. Since $s\sigma$ cannot be equal to $t\sigma$, we assume w.l.o.g. that $s\sigma \succ t\sigma$.
The equation e has not generated any rule because $s\sigma$ is reducible by R_e or $t\sigma$ is reducible by R_e at a non-topmost position. Then there exists some equation $l \simeq r$ that has generated a rule $l\sigma \to r\sigma$, which reduces $s\sigma$ or $t\sigma$. The case where $s\sigma$ is reducible is analogous as in Theorem 3.3.5. For the other case we have $t\sigma|_p \equiv l\sigma$, for some position $p \neq \lambda$. If p is below a variable position in t then we apply the same lifting argument as in Theorem 3.3.5, and otherwise, since $p \neq \lambda$, we can conclude as well by the existence of an ordered paramodulation inference with a smaller conclusion. □

This strict ordered paramodulation rule of Definition 3.3.6 will be adapted for the Horn case and the case of general clauses in Sections 3.4 and 3.5 respectively, and the non-strict one of Definition 3.3.1 will be used to obtain the results on unfailing Knuth-Bendix completion in Chapter 4.

3.4 The Horn case

In this section we generalize the results of the previous section to Horn clauses. In the following inference system it is assumed that in each clause with a non-empty antecedent one of these negative equations, the one that is written underlined, has been *selected* (see Section 1.2.3 and [BG98a]). In the Horn case this leads to positive unit strategies (and in the non-Horn case to positive strategies): left premises of paramodulations are unit clauses, and the only inferences involving non-unit clauses are equality resolution or paramodulation left on its selected equation.

Definition 3.4.1 *The inference system $\mathcal{H}$ for Horn clauses with respect to the west ordering $\succ$ is defined as follows:*

paramodulation right:

$$\frac{\to l \simeq r \qquad \to s \simeq t}{\to (s[r]_p \simeq t)\sigma} \qquad \text{where } \sigma = mgu(l, s|_p)$$

paramodulation left:

$$\frac{\to l \simeq r \qquad \Gamma, \underline{s \simeq t} \to \Delta}{(\Gamma, s[r]_p \simeq t \to \Delta)\sigma} \qquad \text{where } \sigma = mgu(l, s|_p)$$

equality resolution:

$$\frac{\Gamma, \underline{s \simeq t} \to \Delta}{(\Gamma \to \Delta)\sigma} \qquad \text{where } \sigma = mgu(s, t)$$

where, moreover, in both paramodulation rules $s|_p$ *is not a variable,* l *is maximal in its premise and, if* $p = \lambda$*, then* s *is also maximal in* $s \simeq t$*, that is, for some ground substitution* θ*, it holds that* $l\sigma\theta \succ r\sigma\theta$*, and, if* $p = \lambda$*, then also* $s\sigma\theta \succ t\sigma\theta$*.*

Let us remark that the equality resolution rule is named after the fact that it encodes a resolution inference with the reflexivity axiom of equality $x \simeq x$.

We prove completeness of $\mathcal{H}$ by showing that, if S is a set of Horn clauses closed under $\mathcal{H}$ and $\Box \notin S$, then S is satisfiable. The satisfiability proof is of a constructive nature. First, an equality Herbrand interpretation is built, and second, it is shown that this interpretation is a model of S. The choosen equality Herbrand interpretation is R_S^* (see Section 2.12), where R_S denotes the ground TRS generated as in Definition 3.3.7, but taking the set of positive unit clauses of S as the set of equations E. Then, since R_S is convergent, the completeness result quite easily follows.

Definition 3.4.2 *Let* S *be a set of Horn clauses. By* E_S *we denote the set of equations occurring as positive unit clauses in* S*, that is,* $E_S = \{\ s \simeq t\ \mid \rightarrow s \simeq t \in S\ \}$*, and by* R_S *we denote the set of rules generated by* E_S *as in Definition 3.3.7 (i.e.,* $R_S = R_{E_S}$*).*

We use multiset extensions for lifting orderings $\succ$ on terms to orderings on equations and clauses. Let C be a ground clause, and let $emul(s \simeq t)$ be $\{s,t\}$ if $s \simeq t$ is a positive equation in C, and $\{s,s,t,t\}$ if it is negative. Then, if $\succ$ is an ordering, we define the ordering $\succ^e$ on (occurrences of) ground equations in a clause by $e \succ^e e'$ if $emul(e) \succ^{mul} emul(e')$. Similarly, $\succ^c$ on ground clauses is defined by $C \succ^c D$ if $mse(C)\ (\succ^{mul})^{mul}\ mse(D)$, where $mse(C)$ is the multiset of all $emul(e)$ for occurrences e of equations in C. The result is a total ordering on ground clauses.[2]

Theorem 3.4.3 (Refutation completeness of $\mathcal{H}$ for Horn clauses) *Let* S *be a set of Horn clauses closed under* $\mathcal{H}$*. Then* $\Box \in S$ *if, and only if,* S *is unsatisfiable.*

Proof. The left-to-right implication is trivial. For the other one, let S be a set of clauses closed under $\mathcal{H}$ such that $\Box \notin S$. As in Theorem 3.3.5, we proceed by induction w.r.t. $(\xrightarrow{+}_{R_S})^c$, which will be denoted in the remainder of this proof by $\succ_R$. This ordering is monotonic where needed (see the cases below) and well-founded. Again it is proved that R_S^* is a model of S by deriving a contradiction from the existence of a minimal w.r.t. $\succ_R$ ground instance C of a clause in S such that $R_S^* \not\models C$. Since E_S is closed under strict ordered paramodulation, by Theorem 3.3.9 we have $R_S^* \models E_S$, and therefore C cannot be a positive unit clause. Two cases have to be considered:

1. C is an instance $\Gamma\sigma, \underline{s\sigma \simeq t\sigma} \rightarrow \Delta\sigma$ of a clause $\Gamma, \underline{s \simeq t} \rightarrow \Delta$, where $s\sigma \equiv t\sigma$.

   Then there is an inference by equality resolution,

$$\frac{\Gamma, \underline{s \simeq t} \rightarrow \Delta}{(\Gamma \rightarrow \Delta)\theta},$$

   whose conclusion has an instance D of the form $\Gamma\sigma \rightarrow \Delta\sigma$ such that $C \succ_R D$ and, moreover, D is in S and $R_S^* \not\models D$, which is a contradiction.

[2] Roughly, $\succ^c$ compares the multisets of all equations occurring in the clauses, but where in addition terms occurring negatively have slightly more weight than the ones occurring positively; in fact, in order to make $\succ^c$ total on ground clauses, the information about which equations are positive and which ones are negative has to be present anyway.

2. C is an instance $\Gamma\sigma, \underline{s\sigma \simeq t\sigma} \to \Delta\sigma$ of a clause $\Gamma, \underline{s \simeq t} \to \Delta$, where, w.l.o.g., $s\sigma \succ t\sigma$.

   Then, since $R_S^* \not\models C$, we have $R_S^* \models s\sigma \simeq t\sigma$, and since R_S is convergent, there must be a rewrite proof of $s\sigma \simeq t\sigma$ by R_S, that is, $s\sigma$ and $t\sigma$ must rewrite into the same normal form with R_S. This implies that either $t\sigma$ is reducible at a non-topmost position or else $s\sigma$ is reducible. (Note that it cannot be the case that the only possible reduction step on $s\sigma \simeq t\sigma$ is at the topmost position of $t\sigma$. By such a step, a new term t' is obtained with $t\sigma \succ t'$ and t' again irreducible at non-topmost positions; since $s\sigma \succ t\sigma$, such a sequence of topmost steps on $t\sigma$ can never produce $s\sigma$).

   Then either the lifting argument applies like in Theorem 3.3.5, or else there is an inference by paramodulation left,
$$\frac{\to l \simeq r \qquad \Gamma, \underline{s \simeq t} \to \Delta}{(\Gamma, s[r]_p \simeq t \to \Delta)\theta},$$
   whose conclusion has an instance D of the form $(\Gamma, s[r]_p \simeq t \to \Delta)\sigma$ such that $C \succ_R D$ and $R_S^* \not\models D$, which is a contradiction.

□

3.5 General clauses

Now we consider general clauses. As for the Horn case, we consider that in each clause with a non-empty antecedent one of its negative equations, the one that is written underlined, has been selected.

Definition 3.5.1 *The inference system I for general clauses with respect to the west ordering $\succ$ is defined as follows:*

paramodulation right:
$$\frac{\to l \simeq r, \Delta \qquad \to s \simeq t, \Delta'}{(\to s[r]_p \simeq t, \Delta, \Delta')\sigma} \qquad \text{where } \sigma = mgu(l, s|_p)$$

paramodulation left:
$$\frac{\to l \simeq r, \Delta \qquad \Gamma, \underline{s \simeq t} \to \Delta'}{(\Gamma, s[r]_p \simeq t \to \Delta, \Delta')\sigma} \qquad \text{where } \sigma = mgu(l, s|_p)$$

equality resolution:
$$\frac{\Gamma, \underline{s \simeq t} \to \Delta}{(\Gamma \to \Delta)\sigma} \qquad \text{where } \sigma = mgu(s, t)$$

equality factoring:
$$\frac{\to s \simeq t, s' \simeq t', \Delta}{(t \simeq t' \to s \simeq t, \Delta)\sigma} \qquad \text{where } \sigma = mgu(s, s')$$

where in both paramodulation rules $s|_p$ is not a variable. The ordering restrictions are as follows:

In paramodulation right, l is strictly maximal in $l \simeq r$, and $l \simeq r$ is the strictly maximal equation in its premise, and if $p = \lambda$, then s is strictly maximal in $s \simeq t$ and $s \simeq t$ is the strictly maximal equation in its premise. Formally: for some ground substitution θ it holds that $l\sigma\theta \succ r\sigma\theta$ and $(l \simeq r)\sigma\theta \succ^e e\sigma\theta$ for all equations e in Δ, and if $p = \lambda$, then $s\sigma\theta \succ t\sigma\theta$ and $(s \simeq t)\sigma\theta \succ^e e\sigma\theta$ for all equations e in Δ'.

In paramodulation left, l is strictly maximal in $l \simeq r$, and $l \simeq r$ is strictly maximal in its premise, and if $p = \lambda$, then s is strictly maximal in $s \simeq t$.

Finally, in equality factoring, s is strictly maximal in $s \simeq t$ and $s \simeq t$ is maximal.

Here, the paramodulation rules and the equality resolution rule play the same role as their counterparts in the inference system $\mathcal{H}$. The equality factoring rule is new. Intuitively, it expresses that, if s and s' are syntactically equal, and t and t' are semantically equal, then the two equations in the succedent express the same information, and one of them can be omitted.

Note also that several terms in a premise C may be involved in the inferences; e.g., for a term t it may be the case that for some ground instances $C\theta$ the term $t\theta$ is the maximal one, and for other instances it is not. We now define the ground TRS R_S generated by a set of clauses S.

Definition 3.5.2 *Let S be a set of clauses. Then, for every ground instance C of the form $\rightarrow l \simeq r, \Delta$ of a clause in S, we define $Gen(C) = \{l \rightarrow r\}$, and say that C* generates *the rule $l \rightarrow r$ if, and only if,*

1. $R_C^* \not\models C$,
2. $l \succ r$, *and* $(l \simeq r) \succ^e e$ *for all equations e in Δ,*
3. *l is irreducible by R_C,*
4. *r and Δ are irreducible at non-topmost positions by R_C, and*
5. $R_C^* \models r \simeq t$ *for no equation $l \simeq t$ in Δ,*

where R_C is the set of rules generated by all instances D of clauses in S such that $C \succ^c D$, that is, $R_C = \bigcup_{C \succ^c D} Gen(D)$. In all other cases, $Gen(C) = \emptyset$. We denote by R_S the set of rules generated by all ground instances of S, that is, $R_S = \bigcup_{C \in gnd(S)} Gen(C)$.

Let us remark that the first condition states that a clause only contributes to the model if it does not hold in the partial model built so far and hence we are *forced* to extend this partial model. Conditions 2, 3 and 4, state that only maximal equations (at the ground level) can generate a rule; these conditions are also the key for showing that R_S is convergent:

Property 3.5.3 *Let S be a set of clauses. Then for all rules $l \rightarrow r$ in R_S we have*

1. *r is irreducible by R_S at non-topmost positions.*
2. *l is irreducible by $R_S \setminus \{l \rightarrow r\}$.*

Lemma 3.5.4 *For every set of clauses S, the ground TRS R_S is convergent.*

For the completeness proof, it is crucial that clauses generating rules have only one positive literal satisfied by the interpretation. Although by the first condition to generate rule in R_S, Δ does not hold in the partial model built so far when an instance $\rightarrow l \simeq r, \Delta$ generates the rule $l \rightarrow r$, condition 5 is necessary to enforce that Δ does not hold in the final interpretation either:

Lemma 3.5.5 *Let S be a set of clauses. If $\rightarrow l \simeq r, \Delta$ is an instance C of a clause in S that generates the rule $l \rightarrow r$ in R_S, then $R_S^* \not\models \Delta$.*

Proof. We first prove that if $R_S^* \models s \simeq t$ for ground s and t that are irreducible at non-topmost positions and such that $s \succ t$, then no rules with left-hand sides greater than s (w.r.t. $\succ$) are used in the rewrite proof. We proceed by induction on the size of $s \simeq t$ w.r.t. $\succ^{mul}$. The first step can only apply at topmost position of s or t. If it is on t, then we obtain $s \simeq r$, where $t \rightarrow r$ is the applied rule in R_S, and hence $s \succ t \succ r$, by definition of R_S, and r is irreducible at non-topmost positions by Property 3.5.3.1, and

then we can conclude by the induction hypothesis. Otherwise, we obtain $r \simeq t$, where $s \to r$ is the applied rule in R_S, and hence, as before, $s \succ r$ and r is irreducible at non-topmost positions. If $r \equiv t$, we are done; otherwise, if $r \succ t$ or $t \succ r$, we can conclude by induction.
Now, assume $s \simeq t$ is a ground equation in Δ with a rewrite proof using R_S and such that $s \succ t$. The rules generated by clauses D with $D \succ^c C$ cannot be used, since they have left-hand sides greater than l and hence greater than s w.r.t. $\succ$, and s and t are irreducible at non-topmost positions. But since $R_C^* \not\models s \simeq t$, the rule $l \to r$ is used. Hence $s \equiv l$, and $s \simeq t$ rewrites into $r \simeq t$, to which $l \to r$ can no longer be applied, which contradicts the last condition of Definition 3.5.2. □

We now introduce a well-founded ordering $\succ_R$ that will be used in the proof of Theorem 3.5.9. In contrast to the Horn and equational cases, it does not coincide with $\xrightarrow{+}_{R_S}$.

Definition 3.5.6 *Let S be a set of clauses. By $\succ_R$ we denote the smallest transitive relation such that $s \succ_R t$ whenever (i) $s \xrightarrow{+}_{R_S} t$ or (ii) $s \rhd t$ or (iii) s and t are irreducible by R_S at non-topmost positions and $s \succ t$.*

The ordering $\succ_R$ fulfills the following properties w.r.t. R_S.

Property 3.5.7 *Let S be a set of clauses. Then for all ground terms s and t such that s is irreducible by R_S at non-topmost positions and $s \succ_R t$, we have*

1. *$s \succ t$ and*
2. *t is irreducible by R_S at non-topmost positions.*

Proof. If $s \succ_R t$ by case (ii) or (iii), then it trivially holds. Otherwise, $s \to_{R_S} s_1 \to_{R_S} \cdots \to_{R_S} s_n \to_{R_S} t$. Since s is irreducible at non-topmost positions and, by Property 3.5.3.1, all right-hand sides of rules in R_S are irreducible at non-topmost positions, all steps in the sequence are at topmost position, which implies on the one hand that t is irreducible at non-topmost positions, and on the other hand, by definition of R_S, that $s \succ s_1 \succ \cdots \succ s_n \succ t$. □

Lemma 3.5.8 *Let S be a set of clauses. Then $\xrightarrow{+}_{R_S}$ is a reduction ordering and $\succ_R$ is well-founded.*

Proof. Since, by Lemma 3.5.4, R_S is a terminating TRS, then $\xrightarrow{+}_{R_S}$ is a reduction ordering. Now assume $\succ_R$ is not well-founded. Since $(\xrightarrow{+}_{R_S} \cup \rhd)^+$ is well-founded by Lemma 3.2.2, there is also some such infinite sequence $t_1 \succ_R t_2 \succ_R t_3 \succ_R \cdots$ starting with case (iii): where t_1 and t_2 are irreducible by R_S at non-topmost positions and $t_1 \succ t_2$. Then, by Property 3.5.7, t_3, t_4, and so on are all irreducible at non-topmost positions, and $t_1 \succ t_2 \succ t_3 \cdots$ w.r.t. the well-founded west ordering $\succ$, which is a contradiction. □

We finally prove refutation completeness of I by induction w.r.t. $\succ_R^c$.

Theorem 3.5.9 (Refutation completeness of I for general clauses) *Let S be a set of clauses closed under I with respect to a west ordering $\succ$. Then $\Box \in S$ if, and only if, S is unsatisfiable.*

Proof. We prove that if $\Box \notin S$, then R_S^* is a model of S, by induction w.r.t. $\succ_R^c$, which is a well-founded ordering on clauses. In the following, we (ambiguously) write $\succ_R$ for terms, equations, and clauses instead of $\succ_R$, $\succ_R^e$, and $\succ_R^c$, respectively. We derive a contradiction from the existence of a minimal w.r.t. $\succ_R$ ground instance C of a clause in S such that $R_S^* \not\models C$.

1. We first consider the case where C is an instance with σ of a positive clause $\rightarrow s \simeq t, \Delta$, where $s\sigma \simeq t\sigma$ is strictly maximal with respect to $\succ^e$ in C and w.l.o.g. $s\sigma \succ t\sigma$. Since $R_S^* \not\models C$, we know that C has not generated any rule because of one of the following reasons:

   (a) $t\sigma$ is reducible by R_C at a non-topmost position. Then there exists some clause $\rightarrow l \simeq r, \Delta'$ in S whose instance C' with $C \succ^c C'$ generates the rule $l\sigma \rightarrow r\sigma$ reducing $t\sigma$.

   If, for some prefix p' of p, the term $t|_{p'}$ is a variable x, then the same lifting argument as in proof of Theorem 3.3.5 applies.

   Otherwise, $t|_p$ is a non-variable subterm of t, and there exists an inference by paramodulation right,

   $$\frac{\rightarrow l \simeq r, \Delta' \qquad \rightarrow s \simeq t, \Delta}{(\rightarrow s \simeq t[r]_p, \Delta, \Delta')\theta},$$

   whose conclusion has an instance D of the form $(\rightarrow s \simeq t[r]_p, \Delta, \Delta')\sigma$, such that, by Lemma 3.5.5, $R_S^* \not\models D$. This contradicts the minimality of C, since $C \succ_R D$ for the following reasons: $mse(C) = mse(\Delta) \cup \{ \{s\sigma, t\sigma\} \}$ and $mse(D) = mse(\Delta) \cup \{ \{s\sigma, t\sigma[r\sigma]_p\} \} \cup mse(\Delta')$. Then we need to prove that $\{s\sigma, t\sigma\} \succ_R^{mul} \{s\sigma, t\sigma[r\sigma]_p\}$ and $(s\sigma \simeq t\sigma) \succ_R e\sigma$ for all equations e in Δ'. The first one holds because, by monotonicity of $\xrightarrow{+}_{R_S}$, we have $t\sigma[l\sigma]_p \succ_R t\sigma[r\sigma]_p$. For the second one, we have $l\sigma \succeq u\sigma$ for all terms u in Δ', and since they are also irreducible at non-topmost positions, we have $l\sigma \succeq_R u\sigma$. Furthermore $t\sigma \succ_R l\sigma$, since $t\sigma \rhd l\sigma$, which implies $t\sigma \succ_R u\sigma$ for all terms u in Δ', and hence we can conclude.

   (b) $s\sigma$ is reducible by R_C and case 1a does not apply. Then we argue as in the previous case by lifting or an inference in s whose conclusion has an instance D of the form $(\rightarrow s[r]_p \simeq t, \Delta, \Delta')\sigma$, contradicting the minimality of C. Here $C \succ_R D$ if $p \neq \lambda$ as in the previous case. If $p = \lambda$, then $(s\sigma \simeq t\sigma) \succ_R e\sigma$ for all equations e in Δ': since $C \succ^c C'$, and $s\sigma \simeq t\sigma$ is the strictly maximal equation of C, we have $(s\sigma \simeq t\sigma) \succeq^e (l\sigma \simeq r\sigma) \succ^e e\sigma$, and since all these equations are irreducible at non-topmost positions, $(s\sigma \simeq t\sigma) \succ_R e\sigma$.

   (c) An equation $u\sigma \simeq v\sigma$ in Δ is reducible by R_C at a non-topmost position. The proof is like that of case 1a.

   (d) None of the previous cases applies, and Δ is of the form $s' \simeq t', \Delta'$, where $s\sigma \equiv s'\sigma$ and $R_S^* \models t\sigma \simeq t'\sigma$, that is, the last condition of Definition 3.5.2 fails. Then there exists some inference by equality factoring,

   $$\frac{\rightarrow s \simeq t, s' \simeq t', \Delta'}{(t \simeq t' \rightarrow s \simeq t, \Delta')\theta},$$

   whose conclusion has an instance D such that $R_S^* \not\models D$. This contradicts the minimality of C: since $s\sigma$, $t\sigma$ and $t'\sigma$ are irreducible at non-topmost positions, and $s'\sigma \succ t\sigma$ and $s'\sigma \succ t'\sigma$, we have $C \succ_R D$.

2. If C is an instance with σ of $\rightarrow s \simeq t, \Delta$, where $s\sigma \simeq t\sigma$ is maximal but not strictly maximal with respect to $\succ^e$ in C, then condition 2 of Definition 3.5.2 fails. If $s\sigma \simeq t\sigma$ is reducible at non-topmost positions, then the same reasoning as in case 1a applies. Otherwise, the proof of case 1d applies.

3. If C is an instance with σ of a clause $\Gamma, \underline{s \simeq t} \rightarrow \Delta$, where $s\sigma \equiv t\sigma$, we conclude, as in the proof of Theorem 3.4.3, by equality resolution.

4. If C is an instance with σ of a clause $\Gamma, \underline{s \simeq t} \to \Delta$, where $s\sigma \not\equiv t\sigma$, then $R_S^* \models s\sigma \simeq t\sigma$. Then, since R_S is convergent, there is a rewrite proof of $s\sigma \simeq t\sigma$ with R_S. Assume w.l.o.g. that $s\sigma \succ t\sigma$. This implies that either $s\sigma$ is reducible or $t\sigma$ is reducible at a non-topmost position (otherwise the only possible reduction step is at the top of $t\sigma$ and a new term t' is obtained with $t\sigma \succ t'$ and t' again irreducible except at the top; since $s\sigma \succ t\sigma$, a sequence of topmost steps on $t\sigma$ can never produce $s\sigma$).

   Then, as before, either the lifting argument applies or there is an inference by paramodulation left (we develop only the case where the step takes place in s),

$$\frac{\to l \simeq r, \Delta' \qquad \Gamma, \underline{s \simeq t} \to \Delta}{(\Gamma, s[r]_p \simeq t \to \Delta', \Delta)\theta},$$

   whose conclusion has an instance D of the form $(\Gamma, s[r]_p \simeq t \to \Delta, \Delta')\sigma$, such that $R_S^* \not\models D$. This is a contradiction as before, since also here $C \succ_R D$, for the following reasons: on the one hand, we clearly have $s\sigma \succ_R s\sigma[r\sigma]_p$; on the other hand, we have $s\sigma \trianglerighteq l\sigma$ and hence $s\sigma \succeq_R l\sigma$, and moreover $l\sigma \succ_R r\sigma$, and $l\sigma \simeq r\sigma \succ_R u\sigma \simeq v\sigma$ for all equations $u \simeq v$ in Δ', and hence $\{s\sigma, s\sigma, t\sigma, t\sigma\} \succ_R^{mul} \{l\sigma, r\sigma\} \succ_R^{mul} \{u\sigma, v\sigma\}$.

□

3.6 Conclusions

We have proved that ordered paramodulation remains complete when dealing with non-monotonic orderings. This opens the door to deduction in practical situations where the usual requirements are too strong. Alternatively, it frees us from checking that the ordering which is being used to paramodulate with fulfills some of the requirements: for example, a non-total reduction ordering could be used, without the necessity of checking that it can be totalized while preserving monotonicity and well-foundedness (which makes sense especially if the ordering is going to be supplied by the user).

We have developed a variant of the model generation method of [BG94b], where termination of the ground TRS R that defines the model is not a consequence of the ordering.

In the following we mention some possible extensions and limitations of our results.

3.6.1 Redundancy elimination

In this section we briefly mention the applicability of techniques for *redundancy elimination* in the context of this chapter. We refer to Section 1.2.2 and [BG94b, BG98a, NR01] for all details on general redundancy elimination techniques, for which it would make no sense to repeat them here.

There are standard ways for uniformly covering simplification and deletion techniques that are compatible with refutation completeness by notions of redundancy for inferences and clauses, where *saturation* amounts to the closure under I (where I is an inference system) *up to redundant inferences*.

In the setting of this chapter, there is an important difference, however, because two different orderings are considered. The inferences are computed w.r.t. the west ordering $\succ$, and the completeness proof uses the reduction ordering $\succ_R$. Unfortunately, redundancy should hence be defined w.r.t. $\succ_R$, which is unknown during the saturation process. In some cases $\succ_R$ can be sufficiently approximated. For example, practical redundancy notions like tautology deletion or subsumption are (trivially) correct w.r.t. any $\succ_R$. But it is not clear to what extent crucial redundancy notions like demodulation (i.e., simplification by rewriting) can be applied. For this reason, in Chapter 6 we develop a general framework to exploit redundancy elimination techniques in combination with arbitrary theorem provers, in a refutation complete way, even if these provers are not (or not known to be) complete in combination with the redundancy techniques when applied in the usual sense. Moreover, in Chapter 7, we present a calculus which strictly improves the one given here in that, roughly, it is compatible with demodulation w.r.t. $\succ_r$, where $\succ_r$ is assumed to be a reduction ordering that is included in the (possibly non-monotonic) west ordering $\succ$ w.r.t. which ordered paramodulation is applied.

3.6.2 Basicness and constraints

Ordered paramodulation is a very adequate inference rule for dealing with constrained clauses and the *basic* strategy (see Section 1.2.4). All the results given in this chapter can be easily shown to be compatible with equality constraint inheritance and hence with the basic strategy, thus further restricting the search space: subterms created by unifiers of inferences on ancestors can be *blocked* for inferences, and in our setting this is true also for proper subterms of the term $r\sigma$ (which are irreducible by R_S in our proof) in the conclusions of paramodulation inferences.

All the proofs of this extension can be found in the long version of the paper [BG01b].

3.6.3 Arbitrary selection strategies

As mentioned in the introduction of this thesis, a crucial way for reducing the search space in automated deduction are the so-called *selection strategies*. In this chapter, we have dealt with inference systems with eager selection of negative literals, that is, a negative literal is selected whenever there is any. These strategies are usually easy to prove complete, since the left premise of each (resolution or paramodulation) inference can be restricted to be a positive clause (cf. Section 1.2.3). But sometimes this is not very efficient, because, roughly speaking, one enumerates all solutions of its conditions before using the positive information of a clause (as discussed in [Der91]). Hence, completeness within other selection strategies should be investigated.

In [BG01b], a relatively simple proof transformation technique for the Horn case is given, by which the completeness is shown for arbitrary selection strategies and arbitrary paramodulation-based inference systems, provided that they are complete with equality constraint inheritance and eager selection of negative literals (like, in particular, it happens in our setting). However, Example 3.6.4 of Section 3.6.4 shows that arbitrary selection strategies can cause incompleteness if certain tautologies are deleted. Hence, it is unlikely that a standard model construction-based proof, where such tautologies are redundant, can be found.

3.6.4 Further restrictions

One may wonder whether more restrictive inference systems could be complete as well. In Chapter 5, we show that the subterm property is not necessary either for completeness of ordered paramodulation, leaving as the last question whether even well-foundedness is necessary. A number of negative results have been obtained in other directions, which are best described by the following counterexamples.

Example 3.6.1 *Inferences on non-maximal positive atoms are needed: if the inconsistent set S consists of the clauses*

$$\begin{array}{rl} & \rightarrow\ P(b,b),P(a,b) \\ \underline{P(b,b)},P(a,b) & \rightarrow \\ & \rightarrow\ a \simeq b \end{array}$$

with a west ordering $\succ$ where $a \succ b$ and $P(b,b) \succ P(a,b)$, then the only other possible inference produces the tautology $P(a,b) \to P(a,b)$.

Example 3.6.2 *Also inferences on small sides of positive equations are needed: if the inconsistent set S consists of the clauses*

$$\begin{aligned} &\to\ a \simeq b \\ &\to\ g(b) \simeq g(g(a)) \\ x \simeq g(x)\ &\to \end{aligned}$$

with a west ordering $\succ$ where $a \succ b$ and $g(b) \succ g(g(a))$, then the only other possible inferences produce clauses of the form $b \simeq g^n(b) \to$ and $b \simeq g^n(a) \to$ for $n > 1$.

Example 3.6.3 *Also inferences on small sides of negative equations are needed: if the inconsistent set of clauses S is*

$$\begin{aligned} &\to\ a \simeq b \\ f(x,x) \simeq f(a,b)\ &\to \end{aligned}$$

with a west ordering $\succ$ where $f(a,a) \succ f(a,b)$ and $f(b,b) \succ f(a,b)$, then S is closed.

Example 3.6.4 ([Lyn97]) *For arbitrary selection strategies, these techniques are incompatible with tautology deletion. Consider the set*

$$\begin{array}{lrcl} (1) & & \to & P(c,b,b) \\ (2) & P(c,c,b), P(c,b,c) & \to & b \simeq c \\ (3) & P(x,y,y) & \to & P(x,y,x) \\ (4) & P(x,y,y) & \to & P(x,x,y) \\ (5) & P(c,c,c) & \to & \end{array}$$

This clause set is inconsistent: from (1) and (3) we get $P(c,b,c)$, and from (1) and (4) we get $P(c,c,b)$; these two atoms together with (2) produce $b \simeq c$, which gives the empty clause with (1) and (5). But the empty clause cannot be obtained by ordered paramodulation on non-tautology clauses with an ordering where $b \succ c$, and where always the positive literals are selected (except in clause 5, which has none). In fact, the only new clauses obtained are tautologies.

Chapter 4

Knuth-Bendix completion with west orderings

In this chapter, as a side effect of the technique presented in Chapter 3, we obtain the first practical Knuth-Bendix completion procedure that finds a convergent TRS R for a given set of equations E, logically equivalent to E, and contained in an ordering $\succ$, whenever it exists. This solves a well-known open problem posed by N. Dershowitz and J-P. Jouannaud at the RTA list of open problems [RTA01] (problem no. 34) since its creation in 1991: "Ordered rewriting computes a given convergent set of rewrite rules for an equational theory E and an ordering $\succ$ whenever such a set R exists for $\succ$, provided $\succ$ can be made total on ground terms. Unfortunately, this is not always possible, even if $\succ$ is derivability ($\xrightarrow{+}_R$) in R. Is there a set of inference rules that will always succeed in computing R whenever R exists for $\succ$?".

4.1 Introduction

All current state-of-the-art theorem provers in pure equational logic, like Waldmeister [HBVL97], are based on variations of the Knuth-Bendix completion procedure. When given a reduction ordering $\succ$ that can be extended to a total reduction ordering on ground terms, this procedure is complete in the sense that it always finds a (possibly infinite) convergent TRS R for a set of equations E, logically equivalent to E, and contained in $\succ$, provided it exists (see Section 1.2.1 and [DJ90] for details). As mentioned in Section 1.2.1, this is very useful for automatically proving that an equation $s \simeq t$ follows from E, since it can then be decided by rewriting s and t with R into a (unique) normal form.

It is well-known that, for a given reduction ordering $\succ$, a convergent TRS contained in $\succ$ exists for an equational theory E if, and only if, each E-congruence class of terms has a single minimal element. Note that being a reduction ordering is the minimal possible requirement on $\succ$, since a TRS terminates if, and only if, it is contained in a reduction ordering. Hence a very natural (frequently asked) question is: what happens if we apply completion with a given desirable orientation by a reduction ordering that cannot be extended to a total one, like $f(a) \to f(b)$ and $g(b) \to g(a)$, for which a and b must be incomparable in any monotonic extension? Here this question is answered affirmatively. By a careful further analysis of the technique presented in Chapter 3, we obtain the first practical Knuth-Bendix-like[1] completion procedure that finds a convergent TRS R for a given

[1]We write "Knuth-Bendix-like", because, unlike the classical Knuth-Bendix procedure, our completion procedure is based only on the paramodulation rule.

set of equations E, logically equivalent to E, and contained in a (possibly non-totalizable) reduction ordering $\succ$, whenever it exists.

4.2 A theoretical procedure

In what follows, E denotes a set of equations and $\succ_r$ a reduction ordering on $T(\mathcal{F},\mathcal{X})$ (the symbol $\succ$ will be reserved to denote a west ordering extending $\succ_r$). Then a *convergent TRS for E and $\succ_r$* is a convergent TRS, logically equivalent to E, and such that $l \succ_r r$ for all its rules $l \to r$. The problem we deal with is finding a convergent TRS for the given E and $\succ_r$ whenever it exists, and find it in finite time if it is finite.

It is not difficult to devise a procedure of theoretical nature, i.e., without much practical value, for finding a convergent TRS for E and $\succ_r$. The idea is to systematically enumerate all equational consequences of E, say $s_1 \simeq t_1, s_2 \simeq t_2, \ldots$. If a finite convergent TRS R exists, there exists some (probably huge) i, such that R is contained in (the orientations of) a subset of $\{s_1 \simeq t_1, \ldots, s_i \simeq t_i\}$. One can find this R by periodically checking during the enumeration process whether (i) the subset R of orientable (with $\succ_r$) rules of $\{s_1 \simeq t_1, \ldots s_i \simeq t_i\}$ is confluent and (ii) whether R entails E. The confluence of R can be decided by checking joinability of all its critical pairs. After this, entailment of E can be decided by rewriting.

The following lemma states that, after enumerating i equational consequences of E, it is not necessary to check for confluence of the orientations for each subset of $\{s_1 \simeq t_1, \ldots, s_i \simeq t_i\}$, but that it suffices to consider the orientations of the whole set $\{s_1 \simeq t_1, \ldots, s_i \simeq t_i\}$, because unnecessary rules do not destroy convergence.

Lemma 4.2.1 *Let E be a set of equations, let $\succ_r$ be a reduction ordering on $T(\mathcal{F},\mathcal{X})$, and let R be a convergent TRS for E and $\succ_r$. Let R' be any set of rules $l \to r$ such that $l \succ_r r$ and $E \models l \simeq r$. Then $R \cup R'$ is a convergent TRS for E and $\succ_r$.*

Proof. Clearly all critical pairs between rules in $R \cup R'$ are logical consequences of E. Since R is a convergent TRS for E, all these critical pairs have rewrite proofs with R; that is, they are joinable. Hence $R \cup R'$ is a convergent TRS for E and $\succ_r$. □

The TRS R found in this way may not be minimal; that is, it may have some proper subset R' that is also a convergent TRS for E and $\succ_r$, but such a minimal TRS always exists, and it can be effectively computed from a finite R:

Lemma 4.2.2 ([DMT88]) *For every convergent TRS its unique* canonical[2] *version can be obtained by* interreducing *it by (i) normalizing all right-hand sides and (ii) removing all rules whose left-hand sides are reducible by other rules.*

4.3 Practical procedures

Regarding practically useful procedures, Devie showed that for left- and right *linear E* (i.e., no variable occurs more than once in a side of an equation) standard Knuth-Bendix completion [BD94] finds R [Dev90]. For the general problem, the previously existing procedures still relied on the enumeration of all equational consequences (see, e.g., [Dev92]).

[2]Unfortunately, the word canonical is sometimes also used as equivalent of convergent.

In the following, we denote by R the canonical TRS for E and $\succ_r$ (we assume that R exists), and we denote by R_g the canonical TRS for $gnd(R)$, the set of all ground instances of rules of R.

The following lemma strengthens the uniqueness result of [DMT88] to TRS included in a west ordering, but only for the ground case.

Lemma 4.3.1 *Let $\succ$ be a west ordering, and let R_1 and R_2 be two interreduced TRS over $T(\mathcal{F})$, both included in $\succ$, and such that $R_1^* = R_2^*$. Then $R_1 = R_2$.*

Proof. Consider the rule $l \to r$ in $(R_1 \cup R_2) \setminus (R_1 \cap R_2)$ with minimal l w.r.t. $\succ$. Assume w.l.o.g. that $l \to r \in R_1$. Since $R_1^* \models l \simeq r$ and hence $R_2^* \models l \simeq r$ and R_2 is convergent, there must be a rewrite proof with R_2 for $l \simeq r$. But, since R_1 is interreduced, r and all strict subterms of l are irreducible w.r.t. R_1, and hence also w.r.t. R_2, since any rule reducing them has a left-hand side smaller than l w.r.t. $\succ$. Hence there is a rule $l \to r'$ in R_2. But then r' and r are both irreducible by R_2 and $R_2 \models r \simeq r'$. This implies that r and r' are the same term, contradicting the assumption that $l \to r \notin R_2$. $\square$

In the following we consider the ordered equational paramodulation rule given in Definition 3.3.1 w.r.t. a west ordering $\succ$ extending $\succ_r$, and the ground TRS R_E as in Definition 3.3.2. Then, by Property 3.3.3, for all rules $l \to r$ in R_E we have that $l \succ r$, that r is irreducible by R_E and that l is irreducible by $R_E \setminus \{l \to r\}$.

Lemma 4.3.2 *Let E' be the closure of E under ordered equational paramodulation w.r.t. a west ordering $\succ$ extending $\succ_r$. Then $R_g = R_{E'}$.*

Proof. By Theorem 3.3.5 we have $R_{E'}^* \models E'$, and hence $R_{E'}^* \models E$, and hence $E^* = R_{E'}^*$, since all rules in $R_{E'}$ are logical consequences of E. Furthermore, we clearly have $E^* = R^* = R_g^*$, and hence $R_g^* = R_{E'}^*$. We also have $R_g \subseteq \succ_r \subseteq \succ$, and $R_{E'} \subseteq \succ$. Therefore, since both are interreduced we conclude by Lemma 4.3.1 that $R_{E'} = R_g$. $\square$

Now we come to the main theorem of this chapter. It says that R is a subset of the closure of E under the inference rule of ordered paramodulation (which we recall here from Definition 3.3.1):

$$\frac{l \simeq r \qquad s \simeq t}{(s[r]_p \simeq t)\sigma},$$

where $\sigma = mgu(s|_p, l)$ and $s|_p$ is not a variable, and with ordering restrictions saying that the inference is not needed if $r\sigma \succeq_r l\sigma$ (or, more generally, that the inference is needed only if $l\sigma\theta \succ r\sigma\theta$ for some ground substitution θ, where $\succ$ is the west ordering extending $\succ_r$).

Theorem 4.3.3 *Let E' be the closure of E under equational ordered paramodulation w.r.t. a west ordering $\succ$ extending $\succ_r$. Then $E' \supseteq R$.*

Proof. W.l.o.g. assume there are sufficiently many new constants in $\mathcal{F}$ that do not occur in E or R. Let $l \to r$ be an arbitrary rule in R. We prove that $l \simeq r \in E'$. Let σ be the ground substitution replacing each variable with a distinct new constant. Then $l\sigma \to r\sigma$ is in $gnd(R)$ and also in R_g: since R is interreduced, r is irreducible by R (and hence $r\sigma$ by $gnd(R)$), and l is irreducible by $R \setminus \{l \to r\}$ (and hence $l\sigma$ by $gnd(R) \setminus \{l\sigma \to r\sigma\}$). If $l\sigma \to r\sigma$ is in R_g and by the previous lemma $R_g = R_{E'}$, then $l\sigma \to r\sigma$ is in $R_{E'}$. Since the new constants do not occur in E, if $l\sigma \to r\sigma \in R_{E'}$, then some $l' \simeq r'$ is in E', and $l\sigma \equiv l'\theta\sigma$ and $r\sigma \equiv r'\theta\sigma$ for some θ. We conclude by showing that θ is the identity substitution: there is no equation $l' \simeq r'$ that strictly subsumes a rule $l \to r$ in R; otherwise, the rewrite proof with R of $l' \simeq r'$ would apply some rule different from $l \to r$, which would then also reduce $l \to r$, contradicting $l \to r \in R$. $\square$

4.4 Conclusions

We have shown that, whenever a convergent TRS R exists for a given set of equations E and a (possibly non-totalizable) reduction ordering $\succ_r$, then equational ordered paramodulation w.r.t. a west ordering $\succ$ extending $\succ_r$ will eventually find R. That is, if $\xrightarrow{+}_R$ is included in a west ordering (like an SPO), then unfailing Knuth-Bendix completion on E with this west ordering will compute this R.

In Chapter 7 we show that Theorem 4.3.3 still holds when simplification by rewriting with respect to the reduction ordering $\succ_r$ is applied, thus obtaining a Knuth-Bendix completion procedure compatible with demodulation.

Chapter 5

Paramodulation with well-founded orderings

In Chapter 3 we showed that well-foundedness and the subterm property are enough for ensuring completeness of ordered paramodulation, i.e., the monotonicity requirement is not necessary. After this, the fundamental question arises whether more requirements can be dropped.

In this chapter we show that, for the case of Horn clauses, the subterm property is not necessary either, and hence well-foundedness of the ordering suffices for completeness of ordered paramodulation. This result somehow shows the power of ordered paramodulation regardless of the properties of the ordering which is used, and leaves as the last question whether even well-foundedness is necessary.

5.1 Introduction

As explained in the introduction of this thesis, one of our aims is to study which properties of the ordering used in the ordered paramodulation inference rule are essential for ensuring refutation completeness of the inference system. In Chapter 3 we proved that well-foundedness and the subterm property are sufficient for completeness, thus dropping the monotonicity requirement. Here we show that, for the case of Horn clauses with equality, the subterm property is not necessary either, that is, equations may be ordered from subterms to superterms such as in $a \to f(a)$ and ordered paramodulation is still refutation complete. However, well-foundedness of the ordering is required.

Apart from its theoretical value, there are at least two other potential applications of this result. The first one is in deduction modulo built-in equational theories E, since for many E, finding E-*compatible* orderings fulfilling the required properties is extremely complex or even impossible (see Chapter 3). Requiring only well-foundedness allows us to consider any theory E, since there is always a well-founded E-compatible ordering (note that, as any set, the set of E-congruence classes admits a well-founded ordering). The second potential application comes from the fact that, sometimes, a goal-oriented (ordered) paramodulation proof can only be obtained if the ordering contradicts the subterm property. Let us illustrate this with a simple example:

Example 5.1.1 *Consider the set of equations*

$$y^{\log_y x} \simeq x$$
$$\log_y y^x \simeq x$$
$$x^{y+z} \simeq x^y \cdot x^z$$

and the goal

$$\log_a b + \log_a c \simeq \log_a(b \cdot c).$$

With the following well-founded ordering

$$\begin{array}{rll} y^{\log_y x} & \succ x & \\ x & \succ \log_y y^x & \textit{if } x \textit{ is not headed by } \log \\ \log_y y^x & \succ x & \textit{if } x \textit{ is headed by } \log \\ x^{y+z} & \succ x^y \cdot x^z & \end{array}$$

which contradicts the subterm property, we have a goal-oriented proof by ordered paramodulation (note that some variable renamings have been applied):

$$\cfrac{\boxed{y_1^{\log_{y_1} x_1}} \simeq x_1 \qquad \cfrac{\boxed{y_1^{\log_{y_1} x_1}} \simeq x_1 \qquad \cfrac{\boxed{x_1^{y_1+z_1}} \simeq x_1^{y_1} \cdot x_1^{z_1} \qquad \cfrac{\boxed{x} \simeq \log_y y^x \qquad \boxed{\log_a b + \log_a c} \not\simeq \log_a(b \cdot c)}{\log_y \boxed{y^{\log_a b + \log_a c}} \not\simeq \log_a(b \cdot c)}}{\log_y(\boxed{y^{\log_a b}} \cdot y^{\log_a c}) \not\simeq \log_a(b \cdot c)}}{\log_a(b \cdot \boxed{a^{\log_a c}}) \not\simeq \log_a(b \cdot c)}}{\cfrac{\log_a(b \cdot c) \not\simeq \log_a(b \cdot c)}{\square}}$$

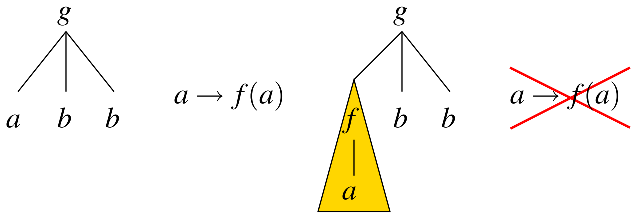
Note that there is no such fully goal-oriented proof with respect to a well-founded ordering including the subterm relation, because no paramodulation step could be applied on the goal: for the first two equations, due to the subterm property, only the term on the left can be used and it does not unify with any subterm of the goal; for the last equation neither side overlaps with the goal.

This simple example illustrates that certain expansions of terms, which are possible only if the ordering contradicts the subterm property, can help in obtaining goal-oriented (ordered) paramodulation proofs. Observe, however, that the "unusual" orientation $x \to \log_y y^x$ of the equation $\log_y y^x \simeq x$ is only taken if x is not headed by log*. Therefore, we are not suggesting that it is better to orient rules so that left-hand sides are variables, but only under certain circumstances. Roughly speaking, this approach can be seen as a kind of* definition unfolding *which takes place at the equational level.*

The technique presented in this chapter is a new variant of the model generation proof method of [BG94b], where the ground rewrite system R that defines the model may be non-terminating (note that rules of the form $a \to f(a)$ necessarily cause non-termination of standard rewriting). For this reason, we define a *new* restricted form of rewriting, called *blocked rewriting* ($\mathcal{B}$-rewriting for short), which is terminating by definition.

$\mathcal{B}$-rewriting works on *abstracted terms*, denoted by $t \cdot \gamma$, which are ground terms split into two parts: the *abstraction (or term) part* t and the *substitution part* γ, such that $t\gamma$ is a ground term (this formalism corresponds, at the ground level, to the *closures* defined for clauses in [BGLS95]).

No rewrite step can be applied to the γ part and when we rewrite $t \cdot \gamma$ with a ground rule $l \to r$ at a non-variable position p of t we obtain $t[x]_p \cdot (\gamma \cup \{x \mapsto r\})$ for some fresh variable x. It is obvious that this rewrite relation terminates since each step strictly reduces the number of non-variable symbols of the abstraction part t.



$$g(a,b,b) \curvearrowright g(x,b,b) \cdot \{x \mapsto f(a)\}$$

Figure 5.1: Illustration of blocked rewriting on abstracted terms.

Example 5.1.2 *Consider the set of equations E containing only*

$$f(a) \simeq a$$

and an ordering such that $a \succ f(a)$ *and* $f^k(a) \succ f^j(a)$ *with* $k \geq 2$ *and* $k > j \geq 0$. *In this ordering, the minimal term is* $f(a)$.

If we close this set E with respect to the ordered paramodulation rule (see Section 3.3.1), we obtain a set of equations of the form

$$f^k(a) \simeq f^j(a) \text{ with } k > j \geq 0,$$

which are oriented as

$$\begin{aligned} a &\to f(a) \\ f^k(a) &\to f^j(a) \quad \text{for all } k \geq 2 \text{ and } k > j \geq 0. \end{aligned}$$

Then the following set R containing the two rules

$$\begin{aligned} a &\to f(a) \\ f(f(a)) &\to f(a) \end{aligned}$$

is a confluent (although overlapping) rewrite system for $\mathcal{B}$*-rewriting, and furthermore all equations in the closure can be proved by* $\mathcal{B}$*-rewriting using R. In particular, all* $f^k(a) \simeq f^j(a)$ *can be proved by joining both sides into the same abstracted term* $x \cdot \{x \mapsto f(a)\}$. *Therefore, for every consequence of E, there is a* $\mathcal{B}$*-rewrite proof using R (see Figure 5.2).*

As seen in Chapter 3, within the model generation method for proving refutation completeness, confluence of the rewrite system R which defines the model is essential. In this new approach with (just) well-founded orderings, a (possibly infinite) rewrite system like the R above is generated. Then, since R may be overlapping and non-terminating, confluence does not follow easily.

Besides the proof of confluence (which is the hardest and more technical part), the core of the completeness proof is rather simple and similar to the one in Chapter 3, since it uses induction on the $\mathcal{B}$-rewrite relation with R, which is well-founded by definition.

This chapter is structured as follows. The next section introduces $\mathcal{B}$-rewriting. Section 5.3 is devoted to the equational case. There, the original part of our proof technique is presented and confluence of the limit rewrite system that defines the model is proved. In Section 5.4, using the results obtained for the equational case, we prove refutation completeness of (positive) ordered paramodulation with well-founded orderings for the case of Horn clauses with equality. Finally, in Section 5.5, conclusions and possible future developments, like in particular the extension to the case of general first order clauses, are discussed.

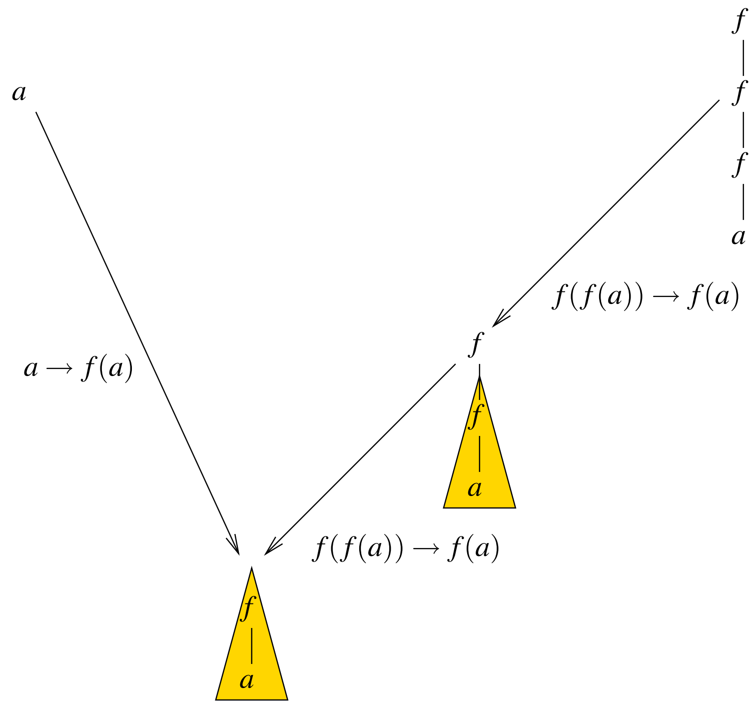


Figure 5.2: $\mathcal{B}$-rewrite proof from Example 5.1.2.

5.2 Blocked rewriting

Here we define a special kind of rewrite relation, called *blocked rewriting* ($\mathcal{B}$-rewriting for short), which is always terminating. This relation restricts standard rewriting by mainly forbidding rewrite steps on right-hand sides introduced by previous rewrite steps. Roughly speaking, after every rewrite step the introduced right-hand side becomes blocked.

In order to express this blocking we use *abstracted terms*, which have variables in the blocked positions. The blocked part of the term is stored in a substitution.

In what follows we only consider ground rewrite systems, which is the only case we need. However, all definitions can easily be extended to deal with a non-ground TRS.

Given a rewrite system R, by $Left(R)$ and $Right(R)$ we denote, respectively, the set of left-hand sides of rules in R and the set of right-hand sides of rules in R.

Definition 5.2.1 (Abstracted term) *Let R be a ground TRS. Let γ be a ground substitution and s a linear term with $vars(s) \subseteq Dom(\gamma)$ and $Ran(\gamma) \subseteq Right(R)$. Then $s \cdot \gamma$ is called an* abstracted term *w.r.t. R, and an* abstraction *of $s\gamma$ w.r.t. R. (We will write s instead of $s \cdot \gamma$ whenever γ is the identity substitution.)*

Two abstracted terms $s \cdot \gamma$ and $t \cdot \delta$ are called equivalent, *denoted by $s \cdot \gamma \equiv t \cdot \delta$, if $s \doteq t$ and $s\gamma \equiv t\delta$.*

An abstracted term $t' \cdot \delta$ is called an argument abstraction *w.r.t. R of a term $t = f(t_1, \ldots, t_n)$ if $t' = f(t'_1, \ldots, t'_n)$ and every $t'_i \cdot \delta$ is an abstraction of t_i which cannot be further abstracted. We will (ambiguously) write $arg\_abs(t, R)$ to denote any argument abstraction of t w.r.t. R, since all argument abstractions of t w.r.t. R are equivalent.*

Let t be a term in $T(\mathcal{F},\mathcal{X})$ and M a set of terms in $T(\mathcal{F},\mathcal{X})$. By $P_M(t)$ we denote the set of positions p of t s.t. there is some prefix p' of p s.t. $t|_{p'} \in M$, i.e., if $t|_p \in M$ then $P_M(t)$ includes all positions below p. By $P_{\mathcal{X}}(t)$ we denote the set of variable positions of t, i.e., $P_{vars(t)}(t)$.

Given an abstracted term $s \cdot \gamma$ w.r.t. R, the set of *blocked positions* is the set of positions q of $s\gamma$ such that q is below a variable position p of s.

Now, we can define $\mathcal{B}$-rewriting as a rewrite relation on abstracted terms.

Definition 5.2.2 ($\mathcal{B}$-rewriting) *Let R be a ground TRS and $s \cdot \gamma$ an abstracted term w.r.t. R. Then*

$$s \cdot \gamma \curvearrowright_R t \cdot \delta \quad \textit{iff}$$

1. $t \cdot \delta \equiv s[x]_p \cdot (\gamma \cup \{x \mapsto r\})$,
2. $s|_p \notin \mathcal{X}$,
3. $x \in \mathcal{X} \setminus (vars(s) \cup Dom(\gamma))$, *and either*
   (a) $s\gamma|_p \to r \in R$ *or*
   (b) $s\gamma|_p \equiv r \in Right(R)$.

If $s\gamma|_p \equiv r \in Right(R)$ we refer to the rewrite step as an abstraction step. *Sometimes we will (ambiguously) write $s \cdot \gamma \curvearrowright_S t \cdot \delta$ to denote a* non-abstraction step *using a rule of a subset S of R.*

Lemma 5.2.3 *Let R be a ground TRS. Then $\curvearrowright_R^+$ is a well-founded ordering.*

Proof. It suffices to show that $\curvearrowright_R$ is terminating. We have that (i) at every step the number of non-variable symbols of the term part strictly decreases and (ii) every step must "take place" at a non-variable position of the term part. □

Now we define two restricted versions of $\curvearrowright_R$, denoted by $\curvearrowright_{\underline{R}}$ and $\curvearrowright_{\overline{R}}$, which respectively consist of rewriting (by non-abstraction steps) only above blocked positions and only above non-blocked positions. Recall that the only forbidden steps for $\curvearrowright_R$ are the ones below blocked positions. Then, for example, in the abstracted term $f(a,x) \cdot \{x \mapsto g(a)\}$, we can rewrite with $\curvearrowright_{\underline{R}}$ only at position λ and with $\curvearrowright_{\overline{R}}$ only at position 1.

Definition 5.2.4 *Let R be a ground TRS and $s \cdot \gamma$ an abstracted term w.r.t. R.*
We write $s \cdot \gamma \curvearrowright_{\underline{R}} s[x]_p \cdot (\gamma \cup \{x \mapsto r\})$ iff $s \cdot \gamma \curvearrowright_R s[x]_p \cdot (\gamma \cup \{x \mapsto r\})$, $s\gamma|_p \to r \in R$ and $vars(s|_p) \neq \emptyset$.
We write $s \cdot \gamma \curvearrowright_{\overline{R}} s[x]_p \cdot (\gamma \cup \{x \mapsto r\})$ iff $s \cdot \gamma \curvearrowright_R s[x]_p \cdot (\gamma \cup \{x \mapsto r\})$, $s\gamma|_p \to r \in R$ and $vars(s|_p) = \emptyset$.

Note that one relation is the complement of the other one if we restrict to non-abstraction steps.

5.3 Paramodulation with equations

In the following, let $\succ$ be a given (total) well-founded ordering on ground terms.

Definition 5.3.1 *The inference rule of (equational)* ordered paramodulation *with respect to $\succ$ is:*

$$\frac{l \simeq r \qquad s \simeq t}{(s[r]_p \simeq t)\sigma}$$

where $\sigma = mgu(s|_p, l)$ *is the most general unifier of* $s|_p$ *and* l*, where* l *is maximal in its premise (that is, for some ground substitution* θ*, it holds that* $l\sigma\theta \succ r\sigma\theta$*) and* $s|_p$ *is not a variable or, if* $s|_p$ *is a variable* x*, then* $p = \lambda$*,* $x \in vars(t)$*, and* l *is also a variable.*

Note that this last condition implies that in very restrictive cases we have to perform an inference on a variable position. But these potential inferences on variable positions are not prolific at all, since both s *and* l *should be variables (with, moreover,* s *occurring in* t*). Hence the only possible inference has* $r\sigma \simeq t$ *as conclusion where* σ *is the variable renaming of* l *by* s*.*

We associate the following ground TRS $\vec{E}$ to a set of equations E.

Definition 5.3.2 *Let* E *be a set of equations. We define* $\vec{E}$ *as the ground TRS obtained by orienting w.r.t.* $\succ$ *the ground instances of all equations in* E*. That is,*

$$\vec{E} = \{s \to t \mid s \simeq t \in gnd(E) \wedge s \succ t\}.$$

Example 5.3.3 *Following Example 5.1.2 of the introduction, as said, the closure by ordered paramodulation of the equation* $a \simeq f(a)$*, consists of all equations of the form* $f^k(a) \simeq f^j(a)$ *with* $k > j \geq 0$*. Then, with the considered well-founded ordering,* $\vec{E}$ *consists of the rules*

$$\begin{aligned} a &\to f(a) \\ f^k(a) &\to f^j(a) \quad \textit{for all } k \geq 2 \textit{ and } k > j \geq 0. \end{aligned}$$

Now we define a well-founded ordering $\succ_E$ on $\vec{E}$, which will be used in the inductive definition of the limit TRS that determines the model.

Definition 5.3.4 *Let* E *be a set of equations. Let* $\succ_{\triangleright}$ *be an arbitrary west ordering on* $T(\mathcal{F})$*, and* $\succ$ *be the well-founded ordering used in the ordered paramodulation inference rule. Then we define the ordering* $\succ_E$ *on* $\vec{E}$ *as follows: given two rules* $l \to r$ *and* $l' \to r'$ *in* $\vec{E}$*, we have*

$$l \to r \succ_E l' \to r' \quad \textit{iff}$$

1. $l \succ_{\triangleright} l'$*, or*
2. $l \equiv l'$ *and* $r \succ r'$*.*

Note that $\succ_E$ is a well-founded ordering, since it is the lexicographic combination of two well-founded orderings $\succ_{\triangleright}$ and $\succ$, and moreover $\succ_{\triangleright}$ is compatible with the equivalence relation $\equiv$ on ground terms. Moreover, $\succ_E$ is a total ordering, since so are $\succ_{\triangleright}$ and $\succ$.

Now we define, by induction on $\succ_E$, a ground TRS $\mathcal{R}$ *generated* by $\vec{E}$, such that $\mathcal{R} \subseteq \vec{E}$. We refer as *back-terms*, or simply as $\mathcal{B}$*-terms*, to all terms that occur as right-hand sides of rules in $\mathcal{R}$.

Definition 5.3.5 *Let* E *be a set of equations. Then, a ground rule* e *of the form* $s \to t$ *in* $\vec{E}$ generates *the rule* $s \to t$ *in* $\mathcal{R}$ *if, and only if,*

1. s *is irreducible by* $\mathcal{R}^e$ *at positions not in* $P_{\mathcal{B}^e}(s)$ *and*
2. t *is irreducible by* $\mathcal{R}^e$ *at positions not in* $P_{\mathcal{B}^e}(t)$*,*

where $\mathcal{R}^e$ is the set of rules generated by all rules e' in $\vec{E}$ such that $e \succ_E e'$, and $\mathcal{B}^e = Right(\mathcal{R}^e)$.

We denote by $\mathcal{R}$ the set of rules generated by all rules in $\vec{E}$, and we define $\mathcal{B} = Right(\mathcal{R})$ as the set of $\mathcal{B}$-terms.

Given a term s, we define the $\mathcal{B}$-positions of *s as all positions in $P_{\mathcal{B}}(s)$, and the* non-$\mathcal{B}$-positions of *s as all other positions of s. We denote by $\mathcal{B}^{\lhd s}$ the set of terms t in $\mathcal{B}$ s.t. $s \rhd t$.*

Using a west ordering —which includes the subterm relation— for comparing the left-hand sides of the rules is crucial in order to ensure that a rule in $\vec{E}$ is included in $\mathcal{R}$ if, and only if, it is irreducible at non-$\mathcal{B}$-positions by other rules in $\mathcal{R}$.

The way we define $\mathcal{R}$ is quite similar to the definition in [BG94b], and especially to Definition 3.3.2 of Chapter 3. The main difference is that only non-$\mathcal{B}$-positions are required to be irreducible. As a consequence, $\mathcal{R}$ can be non-terminating and overlapping. However, if the ordering $\succ$ is a west ordering, i.e., it fulfills the subterm property as well, then, taking $\succ$ as $\succ_{\rhd}$, the generated $\mathcal{R}$ coincides with the rewrite system generated in Definition 3.3.2 of Chapter 3. In this case, by similar arguments as in Property 3.3.3, it is not difficult to see that the right-hand sides of $\mathcal{R}$ (i.e., the $\mathcal{B}$-terms) turn out to be irreducible, which makes irrelevant the fact that we only consider non-$\mathcal{B}$-positions to be irreducible.

Example 5.3.6 *For the closed set of equations in Example 5.3.3, $\mathcal{R}$ consists of the rules*

$$a \rightarrow f(a)$$
$$f(f(a)) \rightarrow f(a).$$

Recall that $\mathcal{R} \subseteq \vec{E}$ and we have defined $\mathcal{B} = Right(\mathcal{R})$ as the set of $\mathcal{B}$-terms. Then, a subset of $\vec{E}$ which is closely related to $\mathcal{R}$, is the set of rules in $\vec{E}$ whose right-hand side is a $\mathcal{B}$-term:

Definition 5.3.7 *By $E_{\mathcal{B}}$ we denote the subset of rules $l \rightarrow r$ in $\vec{E}$ such that $r \in \mathcal{B}$.*

Example 5.3.8 *Again, following Example 5.3.3, $E_{\mathcal{B}}$ consists of the rules*

$$a \rightarrow f(a)$$
$$f^k(a) \rightarrow f(a) \quad \textit{for all } k \geq 2.$$

5.3.1 Some relevant properties

In the following, we assume E to be a set of equations closed under the ordered paramodulation inference rule of Definition 5.3.1, with respect to a well-founded ordering $\succ$. We also assume, w.l.o.g., that the variables of all equations in E are disjoint. The following lemmas state several relevant properties of $\mathcal{R}$, $E_{\mathcal{B}}$ and $\mathcal{B}$, which will be used in the proof of confluence of $\mathcal{R}$, and in particular to show the so-called *Return Property* (see Section 5.3.2).

Lemma 5.3.9 roughly states that, if a term is reducible by $\mathcal{R}$ (at non-$\mathcal{B}$-positions) at some point of the construction of $\mathcal{R}$, then it will stay reducible by $\mathcal{R}$ from this point on. As a consequence, all rules that have not been included in $\mathcal{R}$ will be reducible at some non-$\mathcal{B}$-position by $\mathcal{R}$. This happens in particular for the left-hand sides of rules in $E_{\mathcal{B}} \setminus \mathcal{R}$ (see Lemma 5.3.10).

Lemma 5.3.9 *Let e be a rule in $\vec{E}$ and t be a term in $T(\mathcal{F})$ which is reducible by a rule $l \rightarrow r$ in $\mathcal{R}^e$ at some position not in $P_{\mathcal{B}^e}(t)$. Then t is reducible by $l \rightarrow r$ at some position not in $P_{\mathcal{B}^d}(t)$ for all $d \succ_E e$ in $\vec{E}$.*

Proof. Suppose t is irreducible by $l \to r$ at positions not in $P_{\mathcal{B}^d}(t)$ for some $d \succ_E e$. Then necessarily there is some term $s \in \mathcal{B}^d \setminus \mathcal{B}^e$ with $t \trianglerighteq s \trianglerighteq l$. And, consequently, there is a rule $e' = u \to s \in \mathcal{R}$ for some u, with $e' \succ_E e$. W.l.o.g., we choose such an e' to be minimal w.r.t. $\succ_E$. Now, given that $u \to s \in \mathcal{R}$, we have that s is irreducible by $\mathcal{R}^{e'}$ at positions not in $P_{\mathcal{B}^{e'}}(s)$. We also have $s \trianglerighteq l$ and, moreover, $l \to r \in \mathcal{R}^{e'}$, since $e' \succ_E e$ and, hence, $\mathcal{R}^e \subseteq \mathcal{R}^{e'}$. Therefore, since $l \to r$ reduces s, then necessarily there exists a term $w \in \mathcal{B}^{e'}$ with $s \trianglerighteq w \trianglerighteq l$. And, consequently, there is a rule $e'' = v \to w \in \mathcal{R}^{e'}$. But then $e' \succ_E e''$, which contradicts the minimality of e'. $\square$

Lemma 5.3.10 *Let $l \to r$ be a rule in $E_{\mathcal{B}} \setminus \mathcal{R}$. Then l is reducible by $\mathcal{R}$ at some non-$\mathcal{B}$-position.*

Proof. We proceed by contradiction. Let $e = l \to r$ be a rule in $E_{\mathcal{B}} \setminus \mathcal{R}$. Suppose l is irreducible by $\mathcal{R}$ at non-$\mathcal{B}$ positions. Then l is irreducible by $\mathcal{R}^e$ at non-$\mathcal{B}$-positions, since $\mathcal{R}^e \subseteq \mathcal{R}$. Moreover l is irreducible by $\mathcal{R}^e$ at positions not in $P_{\mathcal{B}^e}(l)$: otherwise l would be reducible by $\mathcal{R}^e$ at positions not in $P_{\mathcal{B}^d}(l)$ for all $d \succ_E e$ by Lemma 5.3.9, contradicting the assumption. Now note that $e \in \vec{E}$, since $E_{\mathcal{B}} \subseteq \vec{E}$. Therefore, by definition of $\mathcal{R}$, if l is irreducible by $\mathcal{R}^e$ at positions not in $P_{\mathcal{B}^e}(l)$ and $e \notin \mathcal{R}$ then we must have that r is reducible by $\mathcal{R}^e$ at some position not in $P_{\mathcal{B}^e}(r)$. Then, by Lemma 5.3.9, we have that r is reducible by $\mathcal{R}^e$ at some position not in $P_{\mathcal{B}^d}(r)$ for all $d \succ_E e$. Therefore, r is reducible by $\mathcal{R}$ at some position not in $P_{\mathcal{B}}(r)$. But then we have a contradiction, since $e \in E_{\mathcal{B}}$ implies $r \in \mathcal{B}$ and hence the set of positions not in $P_{\mathcal{B}}(r)$ is empty. $\square$

From the two previous lemmas we can infer that $\mathcal{B}$-terms do not appear as left-hand sides neither in $\mathcal{R}$ nor in $E_{\mathcal{B}}$.

Lemma 5.3.11 *Let t be a term in $\mathcal{B}$. Then $t \notin Left(\mathcal{R})$.*

Proof. We proceed by induction on t w.r.t. $\succ_{\triangleright}$. If $t \in \mathcal{B}$ then there is a rule $e = s \to t$ in $\mathcal{R}$. W.l.o.g., we choose such an e with t as right-hand side to be minimal w.r.t. $\succ_E$. Then $t \notin \mathcal{B}^e$ and, therefore, $\lambda \notin P_{\mathcal{B}^e}(t)$. On the other hand, $s \to t \in \mathcal{R}$ implies that t is irreducible by $\mathcal{R}^e$ at positions not in $P_{\mathcal{B}^e}(t)$ and, given that $\lambda \notin P_{\mathcal{B}^e}(t)$, we have that $t \notin Left(\mathcal{R}^e)$. Moreover, we have that $s \not\equiv t$, since $s \succ t$. Hence, to conclude, we just need to show that there is no rule $e' = t \to u$ in $\mathcal{R}$ with $e' \succ_E e$, i.e., that no greater rule than e in $\mathcal{R}$ can have t as left-hand side. Then, since $s \not\equiv t$, we just have to consider the case where $t \succ_{\triangleright} s$. If $s \to t$ and $t \to u$ are two rules in $\mathcal{R}$, then there are two equations $s' \simeq t'$ and $t'' \simeq u'$ in E s.t., for some ground substitution σ, we have $s'\sigma \equiv s$, $t'\sigma \equiv t''\sigma \equiv t$ and $u'\sigma \equiv u$. Two cases have to be considered:

1. $t' \notin \mathcal{X}$. Then there is an inference by ordered paramodulation (recall that, as said at the beginning of this section, we assume that E is a set of equations closed under the ordered paramodulation inference rule of Definition 5.3.1)

$$\frac{t'' \simeq u' \qquad s' \simeq t'}{(s' \simeq u')\theta},$$

where $\theta = mgu(t', t'')$. Note that, w.l.o.g., we can choose σ s.t. $s'\theta\sigma \equiv s'\sigma$ and $u'\theta\sigma \equiv u'\sigma$. Moreover, note that $s \simeq u$ is not a tautology of the form $w \simeq w$ since (although $t \succ_{\triangleright} s$) we have $s \succ t \succ u$. Then $s \simeq u \in E$ and hence $s \to u \in \vec{E}$.

Now note that $s \to t \succ_E s \to u$, since $t \succ u$. And, as stated before, $t \to u \succ_E s \to t$. Moreover, by the induction hypothesis, we have that $s \notin \mathcal{B}$, since $t \succ_{\triangleright} s$ and $s \to t \in \mathcal{R}$. Therefore $s \to u$ has not generated a rule in $\mathcal{R}$: otherwise, $s \to t$ would have failed condition 1 to generate a rule in $\mathcal{R}$. Let us now analyze the possible reasons why $e'' = s \to u$ has not generated a rule:

(a) s is reducible by $\mathcal{R}^{e''}$ at positions not in $P_{\mathcal{B}^{e''}}(s)$. Then, by Lemma 5.3.9, s is reducible by $\mathcal{R}^{e''}$ (and hence by $\mathcal{R}^{e}$) at positions not in $P_{\mathcal{B}^{e}}(s)$, contradicting $s \to t \in \mathcal{R}$.

(b) u is reducible by $\mathcal{R}^{e''}$ at positions not in $P_{\mathcal{B}^{e''}}(u)$. Then, by Lemma 5.3.9, u is reducible by $\mathcal{R}^{e''}$ (and hence by $\mathcal{R}^{e'}$) at positions not in $P_{\mathcal{B}^{e'}}(u)$, contradicting $t \to u \in \mathcal{R}$.

2. $t' = x \in \mathcal{X}$. In this case we have $x \notin vars(s')$, given that $t \succ_{\rhd} s$ implies $s \not\trianglerighteq t$. Then let σ' be a substitution s.t. $x\sigma' \equiv u$ and $s'\sigma' \equiv s\sigma$. We have $(s' \simeq t')\sigma' \equiv s \simeq u$. Then $s \to u \in \vec{E}$ as in the previous case, and the rest of the proof is analogous.

□

Lemma 5.3.12 *Let t be a term in $\mathcal{B}$. Then $t \notin Left(E_{\mathcal{B}})$.*

Proof. We proceed by contradiction. Assume that t is a term in $\mathcal{B}$ and there is some rule of the form $t \to u$ in $E_{\mathcal{B}}$. Since t is a term in $\mathcal{B}$ we have that $t \notin Left(\mathcal{R})$ by Lemma 5.3.11. Then $t \to u \in E_{\mathcal{B}} \setminus \mathcal{R}$. But then by Lemma 5.3.10 we have t is reducible by $\mathcal{R}$ at some non-$\mathcal{B}$-position, contradicting $t \in \mathcal{B}$ since all positions of a $\mathcal{B}$-term are $\mathcal{B}$-positions. □

The fact that $\mathcal{B}$-terms do not occur as left-hand side in $E_{\mathcal{B}}$ lets us conclude that there are no two different rules in $E_{\mathcal{B}}$ with the same left-hand side (by showing contradiction with the previous lemma after applying a paramodulation step between the left-hand sides of both rules).

Lemma 5.3.13 *If $l \to r$ is a rule in $E_{\mathcal{B}}$, then l is irreducible by $E_{\mathcal{B}} \setminus \{l \to r\}$ at topmost position.*

Proof. We proceed by contradiction. Assume that $l \to r$ and $l \to s$ are two rules in $E_{\mathcal{B}}$ with $r \succ s$. Then there are two equations $l' \simeq r'$ and $l'' \simeq s'$ in E such that, for some substitution σ, we have $l'\sigma \equiv l$, $r'\sigma \equiv r$, $l''\sigma \equiv l$ and $s'\sigma \equiv s$. There are the following possibilities:

1. $l' \notin \mathcal{X}$. Then there is an inference by ordered paramodulation,

$$\frac{l'' \simeq s' \qquad l' \simeq r'}{(s' \simeq r')\theta},$$

where $\theta = mgu(l', l'')$, whose conclusion has an instance of the form $s \simeq r$. Then $r \to s \in E_{\mathcal{B}}$ since $r \succ s$. But this contradicts Lemma 5.3.12 since $r \in \mathcal{B}$.

2. $l' = x \in \mathcal{X}$ and $l'' \notin \mathcal{X}$. Then there is an inference by ordered paramodulation,

$$\frac{l' \simeq r' \qquad l'' \simeq s'}{(r' \simeq s')\theta},$$

where $\theta = mgu(l', l'') = \{x \mapsto l''\}$, whose conclusion has an instance of the form $r \simeq s$. Then we conclude as in the previous case.

3. $l' = x \in \mathcal{X}$, $l'' = y \in \mathcal{X}$ and $y \in vars(s')$. Then we proceed exactly as in case 2. (Recall that inferences into variables are allowed in this special case; see Definition 5.3.1.)

4. $l' = x \in \mathcal{X}$, $l'' = y \in \mathcal{X}$ and $y \notin vars(s')$. Then let σ' be a substitution such that $y\sigma' \equiv r$ and $z\sigma' \equiv z\sigma$ for all other variables z. We have $l''\sigma' \equiv r$ and $s'\sigma' \equiv s$, and we conclude as in the previous cases.

□

The following is an easy consequence of Lemma 5.3.10.

Lemma 5.3.14 *Let t be a term in $T(\mathcal{F})$. If t is reducible by $E_{\mathcal{B}}$ at some non-$\mathcal{B}$-position p, then t is reducible by $\mathcal{R}$ at some non-$\mathcal{B}$-position below p.*

Proof. Let $l \to r$ be a rule in $E_{\mathcal{B}}$ reducing t at some non-$\mathcal{B}$-position p. If $l \to r \in \mathcal{R}$ then we are done. If, otherwise, $l \to r \notin \mathcal{R}$, then l is reducible by $\mathcal{R}$ at some non-$\mathcal{B}$-position p' by Lemma 5.3.10. But then p' is also a non-$\mathcal{B}$-position of t, since $t|_p \equiv l$ and p is a non-$\mathcal{B}$-position of t. □

Finally, all these previous properties let us prove the following property, which characterizes $\mathcal{R}$ in terms of $E_{\mathcal{B}}$ and $\mathcal{B}$. Recall that, by definition of $\mathcal{R}$ and $E_{\mathcal{B}}$, we already know that $\mathcal{R} \subseteq E_{\mathcal{B}}$.

Lemma 5.3.15 *Let $l \to r$ be a rule in $E_{\mathcal{B}}$. Then $l \to r \in \mathcal{R}$ if, and only if, l is irreducible by $E_{\mathcal{B}}$ at every non-topmost non-$\mathcal{B}$-position.*

Proof. We first prove the left-to-right implication by contradiction. Let $e = l \to r$ be a rule in $\mathcal{R}$. Suppose l is reducible by $E_{\mathcal{B}}$ at some non-$\mathcal{B}$-position $p \neq \lambda$. Then, by Lemma 5.3.14, l is reducible by some rule $e' = l' \to r'$ in $\mathcal{R}$ at some non-$\mathcal{B}$-position below p. Moreover, we have that $e \succ_E e'$, since $p \neq \lambda$ and, hence, $l \rhd l'$. Therefore l is reducible by $\mathcal{R}^e$ at some non-$\mathcal{B}$-position. Then, in particular, it is reducible by $\mathcal{R}^e$ at some position not in $P_{\mathcal{B}^e}$. But this contradicts $l \to r \in \mathcal{R}$.
W.r.t. the right-to-left implication, by Lemma 5.3.13, we have that the only rule in $E_{\mathcal{B}}$ reducing l at topmost position is precisely $l \to r$. Now recall that $\mathcal{R} \subseteq E_{\mathcal{B}}$. Therefore, if l is reducible by $\mathcal{R}$ at topmost position then $l \to r \in \mathcal{R}$. If, otherwise, l is irreducible by $\mathcal{R}$ at topmost position, then l is irreducible by $\mathcal{R}$ at every non-$\mathcal{B}$-position. Then, by Lemma 5.3.10, we have that $l \to r \notin E_{\mathcal{B}} \setminus \mathcal{R}$, which implies $l \to r \in \mathcal{R}$ given that $l \to r \in E_{\mathcal{B}}$. So in either case we have $l \to r \in \mathcal{R}$. □

This property generalizes the notion in Chapter 3 of irreducibility of rules of the ground TRS that defines the model (see Property 3.3.3). Here, irreducibility amounts to irreducibility at non-$\mathcal{B}$-positions of the left- and right-hand sides. Recall that right-hand sides of $\mathcal{R}$ define the set of $\mathcal{B}$-terms (and hence all of their positions are $\mathcal{B}$-positions). Therefore, in particular, if the right-hand sides of rules in $\mathcal{R}$ are irreducible by $\mathcal{R}$, then left-hand sides are irreducible as well, and both notions of irreducibility coincide.

5.3.2 The Return Property

In this subsection we prove the so-called *Return Property*, which roughly states that if we reduce a $\mathcal{B}$-term (i.e., a right-hand side of a rule in $\mathcal{R}$) t into a term u with $\mathcal{R}$, then there is a rewrite sequence from u to t again with $\mathcal{R}$ (without reducing any $\mathcal{B}$-subterm). This powerful property is crucial to prove that $\mathcal{R}$ is confluent, since it is a potentially overlapping and non-terminating TRS. See Figure 5.3 for an example.

The intuition behind this property (at the ground level) comes from the fact that $\mathcal{B}$-terms are minimal w.r.t. $\succ$ in their congruence class and, for every chain $t_1 \simeq t_2 \cdots \simeq t_n$ of equations with $t_1 \succ t_2 \cdots \succ t_n$, ordered paramodulation gives at last $t_1 \simeq t_n$. Then, if t_n is minimal w.r.t. $\succ$ and moreover it is reducible by some rule, for example $t_1 \to t_n$ itself, at some position $p \neq \lambda$ (which may happen since $\succ$ can contradict the subterm property), then we can obtain $t_1 \simeq t_n[t_n]_p$ by an ordered paramodulation step, which gives us that $t_n[t_n]_p$ is in the same congruence class and, hence, $t_n[t_n]_p \simeq t_n$ is in the closure as well, with $t_n[t_n]_p \succ t_n$.

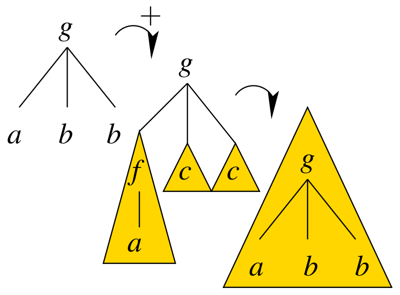
Example of generated $\mathcal{R}$ ($\mathcal{B}$-terms are underlined)	Example of returning path
$\begin{array}{rcl} d & \rightarrow & \underline{g(a,b,b)} \\ b & \rightarrow & \underline{c} \\ a & \rightarrow & \underline{f(a)} \\ f(\underline{f(a)}) & \rightarrow & \underline{f(a)} \\ g(\underline{f(a)},\underline{c},\underline{c}) & \rightarrow & \underline{g(a,b,b)} \end{array}$	

Figure 5.3: Illustration of the Return Property.

Following the example, if we have $a \simeq f(a)$ with $f(a)$ minimal and $f(a)$ reducible by $a \rightarrow f(a)$, we can obtain $a \simeq f(f(a))$ and then $f(f(a)) \simeq f(a)$, with $f(f(a)) \succ f(a)$. Thus, after reducing $f(a)$ into $f(f(a))$ with $a \rightarrow f(a)$, we can return to $f(a)$ using $f(f(a)) \rightarrow f(a)$.

Thanks to this Return Property, t_n can be chosen as a representative (not a normal form, since it can be reducible) of all $t_1, \ldots, t_n$ in its congruence class. Moreover, for all these terms t_i, their representative can be obtained by an innermost rewrite sequence consisting in replacing every reducible subterm by its representative. These representatives are the $\mathcal{B}$-terms.

As usual, the proof becomes rather technical and especially long due to the reductions applied at positions that are below a variable in the equation in E that has generated the rule $l \rightarrow t_n$ in $\mathcal{R}$ (recall that the $\mathcal{B}$-terms have been defined as the right-hand sides of rules in $\mathcal{R}$). For this reason, many times we have to reduce in one shot all redexes involving the same subterm. (This will be denoted by a kind of parallel rewriting.)

Now, we give the main definitions and lemmas which will allow us to prove the Return Property. (The proof of confluence is given in Section 5.3.3.) First of all, we define the set of "reducible positions" with respect to a given rewrite relation on abstracted terms.

Definition 5.3.16 (Reducible positions) *Let R be a ground TRS, let $\curvearrowright$ denote a binary relation on abstracted terms such that $\curvearrowright \subseteq \curvearrowright_R^+$, and let $s \cdot \gamma$ be an abstracted term w.r.t. R. Then $red\_pos(s \cdot \gamma, \curvearrowright)$ is defined as the set of positions of $s \cdot \gamma$ that can be rewritten by $\curvearrowright$. Formally,*

$$red\_pos(s \cdot \gamma, \curvearrowright) = \bigcup_{t \cdot \delta} P_X(t) \setminus P_X(s) \text{ for all } t \cdot \delta \text{ such that } s \cdot \gamma \curvearrowright t \cdot \delta.$$

Definition 5.3.18 establishes a kind of innermost rewrite notion, which fairly mimics the aforementioned process for obtaining the representative of a given term. Since this special rewriting strategy is intended to be used in the proof of the Return Property, we are obliged to define it not only as an innermost, but also as a kind of parallel rewrite notion (see Definition 5.3.17) in order to express substitution replacements at the non-ground level. Fortunately, every "illegal" rewrite subsequence (that is, a rewrite subsequence below a $\mathcal{B}$-position) due to the parallel rewriting strategy can be omitted by applying the Return Property by induction on smaller $\mathcal{B}$-subterms (this is mainly what Lemma 5.3.23 states).

Definition 5.3.17 *Let R be a non-overlapping ground TRS and let $\curvearrowright$ denote a binary relation on abstracted terms, such that $\curvearrowright \subseteq \curvearrowright_R$. Then we write*

$$s\cdot\gamma \curvearrowright^{\parallel} t\cdot\delta \quad \textit{iff}$$

1. *$s\cdot\gamma \curvearrowright^{+} t\cdot\delta$ and*
2. *$\forall p \in red\_pos(s\cdot\gamma, \curvearrowright) : t|_p \in \mathcal{X}$.*

Note that requiring $t|_p \in \mathcal{X}$ implies p to be a valid position in t and hence the rewrite steps not being overlapped.

By this definition we have that every rewrite step with $\curvearrowright^{\parallel}$ represents a rewrite sequence with $\curvearrowright$.

Definition 5.3.18 *Let R be a ground TRS and let $u\cdot\gamma$ be an abstracted term w.r.t. R. Then we write*

$$u\cdot\gamma \rightsquigarrow_R v\cdot\delta \quad \textit{iff}$$

1. *$u\cdot\gamma \curvearrowright^{\parallel}_{\underline{\{l\to r\}}} v\cdot\delta$ for some rule $l\to r$ in R,*
2. *$v \notin \mathcal{X}$, and*
3. *there is no rule $s\to t$ in R with $l \rhd s$ such that $u\cdot\gamma \curvearrowright_{\underline{\{s\to t\}}} w\cdot\rho$ for any $w\cdot\rho$.*

Recall from Definition 5.2.4 that $\curvearrowright_{\underline{\{l\to r\}}}$ means a non-abstraction step with $l\to r$ above a blocked position. The reason why we only consider these kind of steps is that $\rightsquigarrow$ will be used in the proof of the Return Property, where it will suffice to consider only steps which are overlapped with previous ones, which will have introduced some blocking.

By definition of $\rightsquigarrow$, every non-empty rewrite sequence with $\rightsquigarrow_R$ represents a rewrite sequence with $\curvearrowright_R$ (i.e., $u\cdot\gamma \rightsquigarrow^{+}_R v\cdot\delta$ implies $u\cdot\gamma \curvearrowright^{+}_R v\cdot\delta$). Given a rewrite sequence D which includes steps with $\curvearrowright_R$ and $\rightsquigarrow_R$ for some TRS R, we will write $D(\curvearrowright_R)$ to denote the corresponding rewrite sequence with $\curvearrowright_R$ steps.

The following is a rather simple but useful lemma, since it guarantees us that during the transformation of a $\mathcal{B}$-term t towards itself in the proof of the Return Property, no tautology of the form $t \simeq t$ is obtained at the equational level till the last step. It is very important to show that we get back to t at the last step, since then all "illegal" steps on smaller $\mathcal{B}$-subterms are guaranteed to be already "repaired" and hence they can be omitted, as Lemma 5.3.23 states.

Lemma 5.3.19 *Let t be a term in $T(\mathcal{F})$. If $t \curvearrowright_{E_{\mathcal{B}}} t[x]_p\cdot\delta \curvearrowright^{*}_{E_{\mathcal{B}}} t'\cdot\gamma$ with $p \notin P_{\mathcal{B}^{\lhd}}(t)$ and $t' \notin \mathcal{X}$, then $t'\gamma \not\equiv t$.*

Proof. Assume that $t \curvearrowright_{E_{\mathcal{B}}} t[x]_p\cdot\delta \curvearrowright^{*}_{E_{\mathcal{B}}} t'\cdot\gamma$ with $p \notin P_{\mathcal{B}^{\lhd}}(t)$ and $t' \notin \mathcal{X}$. Having $t \curvearrowright_{E_{\mathcal{B}}} t[x]_p\cdot\delta \curvearrowright^{*}_{E_{\mathcal{B}}} t'\cdot\gamma$ implies that $p \in P_{\mathcal{B}}(t'\gamma)$ since, by definition, all right-hand sides of rules in $E_{\mathcal{B}}$ are $\mathcal{B}$-terms. More precisely, we have that $p \in P_{\mathcal{B}^{\lhd t'\gamma}}(t'\gamma)$, since $t' \notin \mathcal{X}$. Therefore it is not possible that $t'\gamma \equiv t$, since otherwise we would have $p \in P_{\mathcal{B}^{\lhd}}(t)$, contradicting the assumption. □

The following two lemmas roughly state that the innermost rewriting strategy used to prove the Return Property leads to rewrite sequences which are unique up to permutation of non-overlapped steps (although this is not a crucial property for the proof, it somehow simplifies the reasoning).

Lemma 5.3.20 *Let R be a non-overlapping ground TRS. Then every term s in $T(\mathcal{F})$ has exactly one normal form $s_n\cdot\gamma$ w.r.t. $\curvearrowright_{\overline{R}}$, and all the rewrite sequences with $\curvearrowright_{\overline{R}}$ from s to $s_n\cdot\gamma$ are a permutation of the same steps.*

Proof. Since R is non-overlapping and, by definition of $\curvearrowright_{\overline{R}}$, in a rewrite sequence with $\curvearrowright_{\overline{R}}$ there are no steps at overlapped positions, then we can rearrange the steps of every rewrite sequence D with $\curvearrowright_{\overline{R}}$ leading to a normal form from s, and nothing else but a permutation of the steps in D can lead to a normal form w.r.t. $\curvearrowright_{\overline{R}}$ of s from s. □

Lemma 5.3.21 *Let $u \cdot \gamma$ be an abstracted term w.r.t. $E_{\mathcal{B}}$. Then for every term $v \cdot \delta$ there is at most one possible rewrite sequence $u \cdot \gamma \rightsquigarrow^*_{E_{\mathcal{B}}} v \cdot \delta$, up to permutation of non-overlapped steps.*

Proof. It follows from the innermost rewriting strategy of $\rightsquigarrow_{E_{\mathcal{B}}}$ (see Definition 5.3.18) and from the fact that, by Lemma 5.3.13, there are no two different rules in $E_{\mathcal{B}}$ with the same left-hand side. □

Now we show that, thanks to the fact that our rewriting strategy is innermost, the steps which are applied below non-topmost $\mathcal{B}$-positions can be omitted. We prove this by assuming that the Return Property holds for rewrite sequences (using $E_{\mathcal{B}}$) beginning with a smaller (w.r.t. size) $\mathcal{B}$-term. We split the proof into two lemmas.

Lemma 5.3.22 *Let t be a term in $\mathcal{B}$ and $\mathcal{S}$ be a non-overlapping subset of $E_{\mathcal{B}}$. Then for every rewrite sequence D of the form $t \curvearrowright^!_{\overline{\mathcal{S}}} t_1 \cdot \rho \rightsquigarrow^!_{E_{\mathcal{B}}} t_m \cdot \beta$ there is a rewrite sequence D' of the form $t \curvearrowright^*_{E_{\mathcal{B}}} t_n \cdot \alpha \curvearrowright^*_{E_{\mathcal{B}}} t_m \cdot \beta$ which is equal to $D(\curvearrowright_{E_{\mathcal{B}}})$ up to permutation of non-overlapped steps, and such that $t_n\alpha \equiv t$ and $t_n \cdot \alpha \curvearrowright^*_{E_{\mathcal{B}}} t_m \cdot \beta$ consists of the steps in $D(\curvearrowright_{E_{\mathcal{B}}})$ which are applied at positions not in $P_{\mathcal{B}^{\triangleleft}}(t)$.*

Proof. We assume that Lemma 5.3.37 holds for terms whose size is smaller than $|t|$. Note that, by Lemmas 5.3.20 and 5.3.21, D is unique up to permutation of non-overlapped steps, and hence so is $D(\curvearrowright_{E_{\mathcal{B}}})$. By definition of $\curvearrowright_{E_{\mathcal{B}}}$, no step in $D(\curvearrowright_{E_{\mathcal{B}}})$ is applied below an already rewritten position, that is, every position p becomes blocked when a step is applied at such position, and subsequent steps are applied only strictly above p or at some position which is not overlapped with p. Therefore, the steps in $D(\curvearrowright_{E_{\mathcal{B}}})$ can be rearranged so that all steps at positions in $P_{\mathcal{B}^{\triangleleft}}(t)$ are applied before than others. That is, there exists a rewrite sequence D' of the form $t \curvearrowright^*_{E_{\mathcal{B}}} t_n \cdot \alpha \curvearrowright^*_{E_{\mathcal{B}}} t_m \cdot \beta$ for some $t_n \cdot \alpha$, such that D' is equal to $D(\curvearrowright_{E_{\mathcal{B}}})$ up to permutation of non-overlapped steps, and where $t_n \cdot \alpha \curvearrowright^*_{E_{\mathcal{B}}} t_m \cdot \beta$ consists of the steps in $D(\curvearrowright_{E_{\mathcal{B}}})$ which are applied at positions not in $P_{\mathcal{B}^{\triangleleft}}(t)$.
We conclude by showing that $t_n\alpha \equiv t$. If no rule in $\mathcal{S}$ reduces t at any position in $P_{\mathcal{B}^{\triangleleft}}(t)$, then we are done. Otherwise, let P denote the set of non-topmost positions p of t such that $t|_p$ is a $\mathcal{B}$-term reducible by $\mathcal{S}$. Note that the innermost rewriting strategy of the rewrite sequence D' on t implies an innermost rewriting strategy on $t|_p$ for every $p \in P$. Therefore, for every $p \in P$ we can extract a rewrite sequence of the form $t|_p \curvearrowright^!_{\overline{\mathcal{S}}} t' \cdot \gamma \rightsquigarrow^!_{E_{\mathcal{B}}} t'' \cdot \delta$ from D'. Then, by Lemma 5.3.37, we have that $t''\delta \equiv t|_p$ or $t''\delta \to t|_p \in E_{\mathcal{B}}$. Now observe that, in either case, once $t'' \cdot \delta$ has been derived from $t|_p$ in D', then no further step is applied strictly below p, since $t'' \cdot \delta$ is irreducible w.r.t. $\rightsquigarrow_{E_{\mathcal{B}}}$. Moreover, if $t''\delta \equiv t|_p$ then no further step is applied at position p in D' since, by Lemma 5.3.12, we have that $t|_p \notin \mathit{Left}(E_{\mathcal{B}})$. If, otherwise, $t''\delta \to t|_p \in E_{\mathcal{B}}$, then this rule must have been applied at position p in D' in some step before $t_n \cdot \alpha$ is obtained, since the rewrite sequence is innermost, $p \in P_{\mathcal{B}^{\triangleleft}}(t)$, and, by Lemma 5.3.13, there are no two different rules in $E_{\mathcal{B}}$ with the same left-hand side. After this step, no further step can be applied at position p since, as stated before, $t|_p \notin \mathit{Left}(E_{\mathcal{B}})$. Furthermore, no rule is applied strictly above p in D' before $t_n \cdot \alpha$ is obtained. Since this occurs for every $p \in P$, we conclude that $t_n\alpha \equiv t$. □

For the following lemma, recall from Definition 5.2.1 that $\mathit{arg\_abs}(t,R)$ denotes the result of abstracting as much as possible the arguments of a term t w.r.t. a TRS R, which implies that all proper $\mathcal{B}$-subterms of t are in the substitution part and, hence, blocked.

Lemma 5.3.23 *Let t be a term in $\mathcal{B}$ and $\mathcal{S}$ be a non-overlapping subset of $E_{\mathcal{B}}$. Then for every rewrite sequence D of the form $t \curvearrowright^{!}_{\mathcal{S}} t_1 \cdot \rho \rightsquigarrow^{!}_{E_{\mathcal{B}}} t_m \cdot \beta$ there is a rewrite sequence D' of the form $arg\_abs(t,\mathcal{R}) \curvearrowright^{*}_{E_{\mathcal{B}}} t'_m \cdot \beta'$, with $t'_m\beta' \equiv t_m\beta$, consisting of the steps in $D(\curvearrowright_{E_{\mathcal{B}}})$ which are applied at positions not in $P_{\mathcal{B}^{\triangleleft}}(t)$.*

Proof. By Lemma 5.3.22, we have that there is a rewrite sequence D'' of the form $t_n \cdot \alpha \curvearrowright^{*}_{E_{\mathcal{B}}} t_m \cdot \beta$ consisting of the steps in $D(\curvearrowright_{E_{\mathcal{B}}})$ which are applied at positions not in $P_{\mathcal{B}^{\triangleleft}}(t)$, and where $t_n\alpha \equiv t$. Now let D' be the rewrite sequence consisting of the same steps as in D'' but beginning with $arg\_abs(t,\mathcal{R})$ instead of $t_n \cdot \alpha$. This is a rewrite sequence with $\curvearrowright_{E_{\mathcal{B}}}$, since no steps are applied at positions in $P_{\mathcal{B}^{\triangleleft}}(t)$ and hence the blocked positions in $arg\_abs(t,\mathcal{R})$ do not forbid any step. Moreover, we clearly have that the last abstracted term $t'_m \cdot \beta'$ of D' fulfills $t'_m\beta' \equiv t_m\beta$. $\square$

Lemma 5.3.37 establishes the Return Property of $\mathcal{B}$-terms for rewrite sequences using $E_{\mathcal{B}}$. Its proof, although based on the relatively simple intuition given at the beginning of this subsection, is quite long and technical. This is due mainly to the parallel rewriting issue, combined with the need of explicitly constructing a fair and complete return path to the initial term, while avoiding tautologies.

We must point out that, although our aim is to prove the Return Property for rewrite sequences using $\mathcal{R}$ (Theorem 5.3.41), this weaker lemma is necessary since sometimes "illegal" steps (that is, steps on smaller $\mathcal{B}$-subterms) can only be "repaired" (that is, we can come back to the initial $\mathcal{B}$-subterm) by using rules in $E_{\mathcal{B}} \setminus \mathcal{R}$. The following are two auxiliary lemmas used in subsequent proofs.

Lemma 5.3.24 *Let t be a term in $T(\mathcal{F})$ and R be a ground TRS such that $l \not\equiv r$ for all rules $l \to r$ in R. Then, for every rewrite sequence of the form $t \curvearrowright^{+}_{R} u \cdot \delta$, we have that $t|_p \not\equiv u\delta|_p$ for every position p in $P_X(u)$.*

Proof. Since $l \not\equiv r$ for all rules $l \to r$ in R and, moreover, in a rewrite sequence with $\curvearrowright^{+}_{R}$ there are no overlapped rewriting steps, it follows that $t|_p \not\equiv u\delta|_p$ for any rewritten position p. $\square$

For the following lemma, we take $\succ_E$ as the total ordering which is used in the inductive definition of $\mathcal{R}$ (see Definitions 5.3.4 and 5.3.5), and we define $\succeq_E$ as the quasi-ordering $\succ_E \cup \equiv^{mul}$.

Lemma 5.3.25 *Let t be a term in $\mathcal{B}$ and $s \to t$ be the minimal rule w.r.t. $\succ_E$ in $\mathcal{R}$ with t as right-hand side. If a rule $l \to r$ in $E_{\mathcal{B}}$ reduces t at some position not in $P_{\mathcal{B}^{\triangleleft}}(t)$, then $l \to r \succeq_E s \to t$.*

Proof. Let $l \to r$ be a rule in $E_{\mathcal{B}}$ which reduces t at some position not in $P_{\mathcal{B}^{\triangleleft}}(t)$. Since $l \to r \in E_{\mathcal{B}}$, then, by Lemma 5.3.12, $l \notin \mathcal{B}$ and, hence, λ is a non-$\mathcal{B}$-position of l. Then, by Lemma 5.3.14, there is some rule $l_1 \to r_1$ in $\mathcal{R}$ which reduces l at some non-$\mathcal{B}$-position. Thus, since $l \to r$ reduces t at some position not in $P_{\mathcal{B}^{\triangleleft}}(t)$, then also does $l_1 \to r_1$.
Now, since $e = s \to t$ is the minimal rule w.r.t. $\succ_E$ in $\mathcal{R}$ with t as right-hand side, then $t \notin \mathcal{B}^e$ and, hence, $P_{\mathcal{B}^e}(t) \subseteq P_{\mathcal{B}^{\triangleleft}}(t)$. Therefore, $l_1 \to r_1$ reduces t at some position not in $P_{\mathcal{B}^e}(t)$. Now assume that $s \to t \succ_E l_1 \to r_1$. Then $l_1 \to r_1 \in \mathcal{R}^e$, contradicting condition 2 to generate a rule in $\mathcal{R}$ for $s \to t$. Hence we necessarily have $l_1 \to r_1 \succeq_E s \to t$, which implies $l \to r \succeq_E s \to t$ (recall that, by Lemma 5.3.13, there are no two different rules in $E_{\mathcal{B}}$ with the same left-hand side, and hence either $l \succ_{\triangleright} l_1$ or $\{l,r\} \equiv^{mul} \{l_1,r_1\}$). $\square$

During the proof of the Return Property, some rewriting steps on abstracted terms need to be simulated by a corresponding paramodulation step at the non-ground level. The problematic steps are the ones which would take place under a variable position at the non-ground level: since we do not allow paramodulation into variables[1], such an step is simulated by a substitution replacement; then the proof becomes involved, since substitution replacements induce non-local changes on non-linear equations; non-local changes at the non-ground level are the reason why we use parallel rewriting at the ground level, and what obliges to "repair" (by subsequent returning steps) some "illegal" steps induced by the non-local modifications.

The following two definitions somehow classify (occurrences of) variables as *green* (not dangerous of non-local effects within our proof), *yellow* (possibly dangerous) and *red* (definitely dangerous). The utility of these definitions will become clearer in the context of the proof.

Definition 5.3.26 (Green and yellow variables) *Let s and t be two unifiable terms in $T(\mathcal{F},\mathcal{X})$.*

1. *By $green\_vars(s,t)$ we denote the set of variables x such that x occurs at some position of s which is not strictly above a variable position of t.*

2. *By $yellow\_vars(s,t)$ we denote the set of variables x such that x occurs at some position of s which is strictly above a variable position of t.*

Note that we have $green\_vars(s,t) \cup yellow\_vars(s,t) = vars(s)$ and, moreover, $green\_vars(s,t) \cap yellow\_vars(s,t)$ may be non-empty.

Definition 5.3.27 (Red variables) *Let R be a ground TRS and let $\curvearrowright$ denote a binary relation on abstracted terms, such that $\curvearrowright \subseteq \curvearrowright_R$. Let s be a term in $T(\mathcal{F},\mathcal{X})$ and $t \cdot \delta$ be an abstracted term w.r.t. R, such that s and t are unifiable. Then by $red\_vars(s, t \cdot \delta, \curvearrowright)$ we denote the set of variables x in $yellow\_vars(s,t)$ such that x occurs at some position of s which is above a position in $red\_pos(t \cdot \delta, \curvearrowright)$.*

The so-called *self-returning* terms are, roughly, terms for which, under certain positions, the Return Property holds in just one step.

Definition 5.3.28 (Self-returning term) *Let t be a term in $\mathcal{B}$. We say that t is* self-returning for a rule $l \to r$ in $E_{\mathcal{B}}$ *iff*

1. *t is of the form $C[l]_p$ for some position p not in $P_{\mathcal{B}^{\triangleleft}}(t)$,*
2. *the minimal rule w.r.t. $\succ_E$ in $\mathcal{R}$ with $C[l]_p$ as right-hand side is of the form $C[r]_p \to C[l]_p$, and*
3. *there is some equation $s' \simeq t'$ in E such that p is a non-variable position of t' and, for some substitution σ, we have that $s'\sigma \equiv C[r]_p$ and $t'\sigma \equiv C[l]_p$.*

Here follows the most technical part of the proof of the Return Property. Lemmas 5.3.29 to 5.3.37, together with Lemmas 5.3.22 and 5.3.23, constitute an split proof of the Return Property for rewrite sequences using $E_{\mathcal{B}}$.

Lemma 5.3.29 *Let t be a term in $\mathcal{B}$, and $\mathcal{S}$ be a non-empty and non-overlapping subset of $E_{\mathcal{B}}$ such that each rule in $\mathcal{S}$ reduces t at some position not in $P_{\mathcal{B}^{\triangleleft}}(t)$. Then either t is self-returning for some rule in $\mathcal{S}$, or there is an equation $u \simeq v$ in E such that, for some substitution φ, we have*

1. *$u\varphi \equiv t$,*

[1] Except for a very special case; see Definition 5.3.1.

2. $v\varphi \equiv t_0\delta$, *where* $t \curvearrowright^{\pm}_{\mathcal{S}} t_0 \cdot \delta$,

3. $\forall q \in red\_pos(t_0 \cdot \delta, \curvearrowright_{\overline{\mathcal{S}}})$: *q is a non-variable position of v, and*

4. $yellow\_vars(v,t_0) \cap (vars(u) \cup green\_vars(v,t_0)) = \emptyset$.

Proof. Assume that t is non-self-returning for any rule in $\mathcal{S}$, and let $s \to t$ be the minimal rule w.r.t. $\succ_E$ in $\mathcal{R}$ with t as right-hand side. Then there is some equation $s' \simeq t'$ in E such that $s'\sigma \equiv s$ and $t'\sigma \equiv t$ for some substitution σ. Let V denote the set of variables x of t' such that $x\sigma$ is reducible by $\mathcal{S}$. Several cases have to be considered:

1. $V = \emptyset$. In this case, let $l \to r$ be a rule in $\mathcal{S}$ (recall that $\mathcal{S}$ is not empty) reducing t at some position p not in $P_{\mathcal{B}^{\triangleleft}}(t)$, and $l' \simeq r'$ be an equation in E such that, w.l.o.g., $l'\sigma \equiv l$ and $r'\sigma \equiv r$. Then there is an inference by ordered paramodulation,

$$\frac{l' \simeq r' \qquad s' \simeq t'}{(s' \simeq t'[r']_p)\theta},$$

where $\theta = mgu(t'|_p, l')$, and whose conclusion has an instance of the form $s \simeq t[r]_p$ under the substitution σ. Note that, since $V = \emptyset$, p is a non-variable position of t'. And we have $t \curvearrowright_{\overline{\mathcal{S}}} t_0 \cdot \delta$, where $t_0 = t[x]_p$ and $\delta = \{x \mapsto r\}$ for some variable x. Moreover, $s \not\equiv t[r]_p$, since otherwise t would be self-returning for $l \to r$ (see Definition 5.3.28). Hence, the conclusion of the inference is not a tautology of the form $w \simeq w$. We have the following possibilities:

   (a) $s'\theta = y \in \mathcal{X}$ and $y \notin vars(t'[r']_p\theta)$. Then let φ be a substitution such that $y\varphi \equiv t$ and $z\varphi \equiv z\sigma$ for all other variables z. And let u denote $s'\theta$ and v denote $t'[r']_p\theta$. Then we have that $u \simeq v$ fulfills the required conditions under the substitution φ:

   - $u\varphi \equiv t$, since $u = y$.
   - $v\varphi \equiv t_0\delta$, where $t \curvearrowright_{\overline{\mathcal{S}}} t_0 \cdot \delta$. This holds since $y \notin vars(v)$ and, hence, $v\varphi \equiv v\sigma \equiv t[r]_p$.
   - $\forall q \in red\_pos(t_0 \cdot \delta, \curvearrowright_{\overline{\mathcal{S}}})$: q is a non-variable position of v. This holds since $V = \emptyset$, $v[]_p$ is a particularization of $t'[]_p$, and $p \notin red\_pos(t_0 \cdot \delta, \curvearrowright_{\overline{\mathcal{S}}})$.
   - $yellow\_vars(v,t_0) \cap (vars(u) \cup green\_vars(v,t_0)) = \emptyset$. On the one hand, we have that t_0 is of the form $t[x]_p$, where x is a variable and t is ground, that is, p is the only variable position of t_0. On the other hand, p is a non-variable position of t', and v denotes $t'[r']_p\theta$. Therefore, no position of t' strictly above p can be a variable position and, hence, $yellow\_vars(v,t_0) = \emptyset$.

   (b) $s'\theta \notin \mathcal{X}$ or $s'\theta = y \in vars(t'[r']_p\theta)$. Then there is an inference by ordered paramodulation,

$$\frac{s' \simeq t' \qquad s'\theta \simeq t'[r']_p\theta}{(t' \simeq t'[r']_p\theta)\gamma},$$

   where $\gamma = mgu(s', s'\theta)$, and whose conclusion has an instance of the form $t \simeq t[r]_p$ under the substitution σ, which is clearly not a tautology of the form $w \simeq w$. Note that this is a correct inference since, if $s'\theta$ is a variable y, then $y \in vars(t'[r']_p\theta)$ and s' is also a variable (see Definition 5.3.1 for details). Now we choose $t'\gamma$ as u, $t'[r']_p\theta\gamma$ as v and σ as φ, and we have $u\varphi \equiv t$, $v\varphi \equiv t_0\delta$ with $t \curvearrowright_{\overline{\mathcal{S}}} t_0 \cdot \delta$, and the remaining conditions are fulfilled for the same reasons as in case 1a.

2. $V \neq \emptyset$. In this case, first of all, we will show that we have $s' \not\triangleright x$ (but possibly $s' = x$) for all variables x in V:

   For every variable x in V, we have that $t \equiv t'\sigma \trianglerighteq x\sigma \trianglerighteq l$ for some rule $l \to r$ in $\mathcal{S}$ (and hence in $E_{\mathcal{B}}$). Since we are assuming that $s \to t$ is the minimal rule w.r.t. $\succ_E$ in $\mathcal{R}$ and each rule in $\mathcal{S}$ reduces t at some position not in $P_{\mathcal{B}^{\triangleleft}}(t)$, then, by Lemma 5.3.25, we have that $l \to r \succeq_E s \to t$ and, therefore, $s \not\triangleright l$, which implies $s' \not\triangleright x$.

   Now let σ' be a substitution such that $x\sigma' \equiv w\rho$, where $x\sigma \curvearrowright^{!}_{\overline{\mathcal{S}}} w \cdot \rho$, for all variables x in V, and $y\sigma' \equiv y\sigma$ for all other variables y. Note that, since $\mathcal{S}$ is assumed to be non-overlapping, then the normal form w.r.t. $\curvearrowright_{\overline{\mathcal{S}}}$ of a term is unique by Lemma 5.3.20. Two cases have to be considered:

   (a) $s'\sigma' \not\equiv s$. Then we necessarily have that $s' = z$ for some variable z in V and hence in $vars(t')$. Now consider any rule $l \to r$ in $\mathcal{S}$ which reduces $z\sigma$. As seen before, we have that $l \to r \succeq_E s \to t$. Then, since $s \trianglerighteq l$, we necessarily have $l \equiv s$. But then, by Lemma 5.3.13, we have that $l \to r$ and $s \to t$ are the same rule. So we have that $s \to t \in \mathcal{S}$ and, therefore, $s'\sigma' \equiv t$. In this case $s'\sigma' \simeq t'\sigma'$ is not a tautology of the form $w \simeq w$: $s' \simeq t'$ cannot be of the form $z \simeq z$ (otherwise $s \to t \notin \mathcal{R}$) and, hence, we have that $t' \triangleright z$, which implies $t'\sigma' \not\equiv t$.

   Note that we have $t \curvearrowright^{+}_{\overline{\mathcal{S}}} t_0 \cdot \delta$ (recall that $\mathcal{S}$ is not empty), where $t_0\delta \equiv t'\sigma'$, i.e., $t_0 \cdot \delta$ is of the form $t[x_1]_{p_1} \ldots [x_n]_{p_n} \cdot \{x_1 \mapsto r_1, \ldots, x_n \mapsto r_n\}$ for some variables $x_1, \ldots, x_n$, where $\forall i \in \{1 \ldots n\} : t|_{p_i} \to r_i \in S$. We choose s' as u, t' as v and σ' as φ. Then the required conditions are fulfilled:

   - $u\varphi \equiv t$.
   - $v\varphi \equiv t_0\delta$, where $t \curvearrowright^{+}_{\overline{\mathcal{S}}} t_0 \cdot \delta$.
   - $\forall q \in red\_pos(t_0 \cdot \delta, \curvearrowright_{\overline{\mathcal{S}}}) : q$ is a non-variable position of v. By definition of σ', all subterms occurring in t at variable positions of t' (and hence of v) have been normalized w.r.t. $\curvearrowright_{\overline{\mathcal{S}}}$.
   - $yellow\_vars(v,t_0) \cap (vars(u) \cup green\_vars(v,t_0)) = \emptyset$. On the one hand, we have that $yellow\_vars(v,t_0) \cap vars(u) = \emptyset$: we have that $u = z$; so it suffices so see that $z \notin yellow\_vars(v,t_0)$, which holds since $z\sigma \equiv s$ and $z\sigma' \equiv t$, i.e., $s \curvearrowright_{\overline{\mathcal{S}}} x \cdot \{x \mapsto t\}$ for some variable x, and hence no occurrence of z in v is strictly above a variable position of t_0. On the other hand, we have that $yellow\_vars(v,t_0) \cap green\_vars(v,t_0) = \emptyset$: for every variable x in $vars(v)$ and positions q_1 and q_2 such that $v|_{q_1} \equiv x$ and $v|_{q_2} \equiv x$, we have that $t_0|_{q_1} \cdot \delta \equiv t_0|_{q_2} \cdot \delta$ and, hence, x is either in $yellow\_vars(v,t_0)$ or in $green\_vars(v,t_0)$, but not in both.

   (b) $s'\sigma' \equiv s$. Then we consider the following situations:

      i. $s' = z \in \mathcal{X}$ and $z \notin vars(t')$. Then let $\varphi = \sigma' \cup \{z \mapsto t\}$. We choose s' as u and t' as v. The conditions are fulfilled for the same reasons as in the previous case, except for $yellow\_vars(v,t_0) \cap vars(u) = \emptyset$, which is now due to the fact that $z \notin vars(v)$.

      ii. $s' \notin \mathcal{X}$ or $s' = z \in vars(t')$. Then there is an inference by ordered paramodulation,

$$\frac{s'' \simeq t'' \qquad s' \simeq t'}{(t'' \simeq t')\gamma},$$

      where the left premise $s'' \simeq t''$ is a variable renamed version of the right premise $s' \simeq t'$ and $\gamma = mgu(s'', s')$. Note that this is a correct inference since, if s' is a variable

z, then it occurs in t' and s'' is also a variable. Note also that, since γ is simply a renaming substitution, we can assume, w.l.o.g., that $s''\gamma \equiv s'$. Then $vars(t''\gamma) \subseteq vars(t'') \cup vars(s')$ and $t'\gamma \equiv t'$. Now, since $s'' \simeq t''$ is a variable renamed version of $s' \simeq t'$, we have that $vars(t'') \cap vars(t') = \emptyset$ and, hence, $vars(t'') \cap V = \emptyset$. And we also have $vars(s') \cap V = \emptyset$, since $s'\sigma' \equiv s$. Therefore, $vars(t''\gamma) \cap V = \emptyset$, and hence $t''\gamma \simeq t'$ has an instance of the form $t \simeq t'\sigma'$ under some substitution φ. This instance is clearly not a tautology of the form $w \simeq w$, and hence $t''\gamma \simeq t'$ is in E.
We choose $t''\gamma$ as u and t' as v. As before, we have $t \curvearrowright^{+}_{\overline{\mathcal{S}}} t_0 \cdot \delta$, where $t_0\delta \equiv t'\sigma'$, and the required conditions are fulfilled for the same reasons as in case 2a, except for $yellow\_vars(v, t_0) \cap vars(u) = \emptyset$, that is, $yellow\_vars(t', t_0) \cap vars(t''\gamma) = \emptyset$, which is now due to the fact that $yellow\_vars(t', t_0) \subseteq V$ and, as seen before, $vars(t''\gamma) \cap V = \emptyset$.
□

Lemma 5.3.30 *Let $u \simeq v$ be an equation in E and $\mathcal{S}$ a non-overlapping subset of $E_{\mathcal{B}}$. If, for some substitution φ, we have*

1. $v\varphi \equiv t_0\delta$, *where* $u\varphi \curvearrowright^{+}_{\overline{\mathcal{S}}} t_0 \cdot \delta$,
2. $\forall q \in red\_pos(t_0 \cdot \delta, \curvearrowright_{\overline{\mathcal{S}}}) : q$ *is a non-variable position of* v, *and*
3. $yellow\_vars(v, t_0) \cap (vars(u) \cup green\_vars(v, t_0)) = \emptyset$,

then there is an equation $u' \simeq v'$ in E such that, for some ground substitution ψ, we have

1. $u'\psi \equiv u\varphi$,
2. $v'\psi \equiv t_1\rho$, *where* $t_0 \cdot \delta \curvearrowright^{!}_{\overline{\mathcal{S}}} t_1 \cdot \rho$, *and*
3. $yellow\_vars(v', t_1) \cap (vars(u') \cup green\_vars(v', t_1)) = \emptyset$.

Proof. We proceed by induction on $\curvearrowright^{+}_{\overline{\mathcal{S}}}$. If $t_0 \cdot \delta$ is irreducible w.r.t. $\curvearrowright_{\overline{\mathcal{S}}}$ then we are done: we choose u as u', v as v', φ as ψ and $t_0 \cdot \delta$ as $t_1 \cdot \rho$. Otherwise, let p be a position in $red\_pos(t_0 \cdot \delta, \curvearrowright_{\overline{\mathcal{S}}})$. Then $v\varphi|_p \equiv l$ for some rule $l \to r$ in $\mathcal{S}$, and there is some equation $l' \simeq r'$ in E such that, w.l.o.g., $l'\varphi \equiv l$ and $r'\varphi \equiv r$. Note that, by condition 2, p is a non-variable position of v. Then there is an inference by ordered paramodulation,

$$\frac{l' \simeq r' \qquad u \simeq v}{(u \simeq v[r']_p)\theta},$$

where $\theta = mgu(v|_p, l')$, and whose conclusion has an instance of the form $u\varphi \simeq v\varphi[r]_p$ under some substitution ψ.
Note that we have $u\varphi \curvearrowright^{+}_{\overline{\mathcal{S}}} t_0 \cdot \delta \curvearrowright_{\overline{\mathcal{S}}} t_0[x]_p \cdot (\delta \cup \{x \mapsto r\})$ for some variable x. Let t_0' denote $t_0[x]_p$ and δ' denote $\delta \cup \{x \mapsto r\}$. We have that $v\varphi \equiv t_0\delta$ and $v\varphi[r]_p \equiv t_0'\delta'$. Neither $u\varphi \simeq v\varphi$ nor $u\varphi \simeq v\varphi[r]_p$ are tautologies of the form $w \simeq w$, by Lemma 5.3.24. Hence, the conclusion of the inference is in E. Now we show that the remaining conditions are preserved for the conclusion of the inference.
We have that $\forall q \in red\_pos(t_0' \cdot \delta', \curvearrowright_{\overline{\mathcal{S}}}) : q$ is a non-variable position of $v[r']_p\theta$, since $v\theta[]_p$ is a particularization of $v[]_p$, and $p \notin red\_pos(t_0' \cdot \delta', \curvearrowright_{\overline{\mathcal{S}}})$.
W.r.t. $yellow\_vars(v[r']_p\theta, t_0') \cap (vars(u\theta) \cup green\_vars(v[r']_p\theta, t_0')) = \emptyset$:

- Since $p \in red\_pos(t_0 \cdot \delta, \curvearrowright_{\overline{\mathcal{S}}})$, it holds that $vars(t_0|_p) = \emptyset$. Then $vars(v|_p) \subseteq green\_vars(v, t_0)$ and, since $yellow\_vars(v, t_0) \cap green\_vars(v, t_0) = \emptyset$, we have $vars(v|_p) \cap yellow\_vars(v, t_0) = \emptyset$.

- Since we can assume w.l.o.g. that variables in each equation are distinct, then we have that $vars(l') \cap vars(v) = \emptyset$, and hence $vars(l') \cap yellow\_vars(v,t_0) = \emptyset$.

The two previous points give us $(vars(v|_p) \cup vars(l')) \cap yellow\_vars(v,t_0) = \emptyset$. Then:

- Since $Dom(\theta) \subseteq vars(v|_p) \cup vars(l')$, we have that $Dom(\theta) \cap yellow\_vars(v,t_0) = \emptyset$. This fact, together with $t_0|_p \notin \mathcal{X}$ and $t_0' \equiv t_0[x]_p$, gives us $yellow\_vars(v[r']_p\theta, t_0') = yellow\_vars(v,t_0)$.
- Since $vars(Ran(\theta)) \subseteq vars(v|_p) \cup vars(l')$, we have that $vars(Ran(\theta)) \cap yellow\_vars(v,t_0) = \emptyset$. Therefore:
  - Since, by condition 3, we have that $yellow\_vars(v,t_0) \cap vars(u) = \emptyset$, then we also have $yellow\_vars(v,t_0) \cap vars(u\theta) = \emptyset$.
  - Since, w.l.o.g., $vars(r') \cap vars(v) = \emptyset$, then we have that $vars(r') \cap yellow\_vars(v,t_0) = \emptyset$. Moreover, by condition 3, we have that $yellow\_vars(v,t_0) \cap green\_vars(v,t_0) = \emptyset$. Hence $yellow\_vars(v,t_0) \cap green\_vars(v[r']_p\theta, t_0) = \emptyset$. Now, since $t_0' \equiv t_0[x]_p$ and p is a valid position of v, we have that $green\_vars(v[r']_p\theta, t_0') = green\_vars(v[r']_p\theta, t_0)$ and, therefore, $yellow\_vars(v,t_0) \cap green\_vars(v[r']_p\theta, t_0') = \emptyset$.

So we have that $yellow\_vars(v,t_0) \cap (vars(u\theta) \cup green\_vars(v[r']_p\theta, t_0')) = \emptyset$ and, since, as we have seen before, $yellow\_vars(v[r']_p\theta, t_0') = yellow\_vars(v,t_0)$, then also $yellow\_vars(v[r']_p\theta, t_0') \cap (vars(u\theta) \cup green\_vars(v[r']_p\theta, t_0')) = \emptyset$.
Once seen that the conditions are preserved, we conclude by the induction hypothesis, since $t_0 \cdot \delta \curvearrowright_{\overline{S}} t_0' \cdot \delta'$ and $\curvearrowright_{\overline{S}} \subseteq \curvearrowright_S$. □

Lemma 5.3.31 *Let t be a term in $\mathcal{B}$. If t is self-returning for some rule $l \to r$ in $E_{\mathcal{B}}$, then all positions in $red\_pos(arg\_abs(t,\mathcal{R}), \curvearrowright_{E_{\mathcal{B}}})$ are overlapped, and l occurs only once in t.*

Proof. We proceed by contradiction. Let $s \to t$ be the minimal rule w.r.t. $\succ_E$ in $\mathcal{R}$ with t as right-hand side. Since t is self-returning (see Definition 5.3.28) for $l \to r$, we know that s is of the form $C[r]_p$ and t is of the form $C[l]_p$ for some position p not in $P_{\mathcal{B}^\triangleleft}(t)$. Now observe that all positions in $red\_pos(arg\_abs(t,\mathcal{R}), \curvearrowright_{E_{\mathcal{B}}})$ do not belong to $P_{\mathcal{B}^\triangleleft}(t)$. If a rule $l' \to r'$ (possibly $l \to r$) in $E_{\mathcal{B}}$ reduces t at some position p' not in $P_{\mathcal{B}^\triangleleft}(t)$ which is not overlapped with p, then, since s is of the form $t[r]_p$, we have that $s \rhd l'$, which implies $s \to t \succ_E l' \to r'$, and hence a contradiction with Lemma 5.3.25. Therefore all positions in $red\_pos(arg\_abs(t,\mathcal{R}), \curvearrowright_{E_{\mathcal{B}}})$ must be overlapped. Now observe that if l occurs more than once in t (possibly at some position in $P_{\mathcal{B}^\triangleleft}(t)$), then also $s \rhd l$ and hence $s \to t \succ_E l \to r$, contradicting Lemma 5.3.25 as well since $l \to r$ reduces t at position p, which is not in $P_{\mathcal{B}^\triangleleft}(t)$. Therefore l must occur only once in t. □

Lemma 5.3.32 *Let t be a term in $\mathcal{B}$ and $s \to t$ be the minimal rule w.r.t. $\succ_E$ in $\mathcal{R}$ with t as right-hand side. If $|s| < |t|$ and, for some position p not in $P_{\mathcal{B}^\triangleleft}(t)$ and rule $l \to r$ in $E_{\mathcal{B}}$, s is of the form $C[r]_p$ and t is of the form $C[l]_p$, then $C[x]_p \cdot \{x \mapsto r\} \rightsquigarrow^!_{E_{\mathcal{B}}} w \cdot \gamma$ for some abstracted term $w \cdot \gamma$ such that $w\gamma \equiv C[r]_p$.*

Proof. We assume that Lemma 5.3.37 holds for terms whose size is smaller than $|t|$. If $C[x]_p \cdot \{x \mapsto r\}$ is irreducible w.r.t. $\rightsquigarrow_{E_{\mathcal{B}}}$, then we are done. Otherwise, let D denote a rewrite sequence of the form

$$C[x]_p \cdot \{x \mapsto r\} \rightsquigarrow^!_{E_{\mathcal{B}}} w \cdot \gamma.$$

Then D is of the form

$$t_1 \cdot \alpha_1 \curvearrowright^{\|}_{\underline{\{l_1 \to r_1\}}} t_2 \cdot \alpha_2 \cdots t_{n-1} \cdot \alpha_{n-1} \curvearrowright^{\|}_{\underline{\{l_{n-1} \to r_{n-1}\}}} t_n \cdot \alpha_n,$$

where $n > 1$ and for all i in $1 \dots n-1$ we have that $l_i \to r_i \in E_{\mathcal{B}}$ and there is no rule $l'_i \to r'_i$ in $E_{\mathcal{B}}$ with $l_i \rhd l'_i$ reducing $t_i \cdot \alpha_i$ w.r.t. $\curvearrowright_{\underline{\{l'_i \to r'_i\}}}$.
Recall from Lemma 5.3.21 that this rewrite sequence D is unique up to permutation of non-overlapped steps. Moreover, since $t_1 \cdot \alpha_1 \equiv C[x]_p \cdot \{x \mapsto r\}$ and $C[x]_p$ has only one variable, then necessarily any rewrite subsequence

$$t_i \cdot \alpha_i \curvearrowright^{\|}_{\underline{\{l_i \to r_i\}}} t_{i+1} \cdot \alpha_{i+1}$$

has only one step. Then the rewrite sequence

$$D_E = t_1 \cdot \alpha_1 \curvearrowright_{\underline{\{l_1 \to r_1\}}} t_2 \cdot \alpha_2 \cdots t_{n-1} \cdot \alpha_{n-1} \curvearrowright_{\underline{\{l_{n-1} \to r_{n-1}\}}} t_n \cdot \alpha_n$$

is equivalent to D. Now we put our attention to the first (and possibly only) step in D_E.

- We have that $t_1 \equiv C[x]_p$ and $t_2 \equiv C[y]_{p'}$ for some variable y and position p' strictly above p. (Note that $p' \not\equiv p$ since, by definition of $\curvearrowright_{E_{\mathcal{B}}}$, p' must be a non-variable position of t_1.)
- By definition of $\rightsquigarrow_{E_{\mathcal{B}}}$ we have that $t_2 \notin \mathcal{X}$, and hence $p' \neq \lambda$. Now recall that $s \equiv C[r]_p$, and hence $s|_{p'} \equiv l_1$. Moreover, $s \to t \in \mathcal{R}$ and $l_1 \to r_1 \in E_{\mathcal{B}}$. Then, by Lemma 5.3.15, we have that p' is a $\mathcal{B}$-position of s.

Now let p'_b be the highest position of s above p' such that $s|_{p'_b}$ is a $\mathcal{B}$-term. Then $s|_{p'_b}$ is the biggest (in size) $\mathcal{B}$-term in s above p'. Let w denote this $\mathcal{B}$-term $s|_{p'_b}$. Since $|s| < |t|$, we have that $|w| < |t|$. Moreover,

- $p'_b \neq \lambda$, since otherwise we would have $s \in \mathcal{B}$ and $s \in \mathit{Left}(\mathcal{R})$, contradicting Lemma 5.3.11, and
- for every position $q' \neq p'$ below p'_b in s such that $s|_{q'} \equiv l_1$, there is a position q'_b above q' which is not overlapped with p' and such that $s|_{q'_b}$ is a $\mathcal{B}$-term:

  Let q' be any position below p'_b in s such that $q' \neq p'$ and $s|_{q'} \equiv l_1$. Then, since $s \equiv t[l_1]_{p'}$, we have that q' is not overlapped with p' and $t|_{q'} \equiv l_1$.

  Then we must have that $t|_{q'_b}$ is a $\mathcal{B}$-term for some position $q'_b \neq \lambda$ above q': assume the contrary (i.e., let t be the only $\mathcal{B}$-term above q' in t); then $l_1 \to r_1$ reduces t at some position q' not in $P_{\mathcal{B}^{\lhd}}(t)$ and, since $l_1 \to r_1 \in E_{\mathcal{B}}$ and $s \to t$ is the minimal rule w.r.t. $\succ_E$ in $\mathcal{R}$ with t as right-hand side, then, by Lemma 5.3.25, we have that $l_1 \to r_1 \succeq_E s \to t$, which is a contradiction with $s \rhd l_1$ (which holds since $p' \neq \lambda$ and $q' \neq \lambda$).

  Finally observe that this position q'_b is clearly not below p', since q' is not overlapped with p' and q'_b is above q'. But q'_b can neither be above p', since $q'_b \neq \lambda$ and p, which is below p', is not in $P_{\mathcal{B}^{\lhd}}(t)$. Then, since $s \equiv t[l_1]_{p'}$, we have that $s|_{q'_b} \equiv t|_{q'_b}$.

Now let $\mathcal{S}_w$ denote $\{l_1 \to r_1\}$ and D^w denote a rewrite sequence of the form

$$w \curvearrowright^{!}_{\underline{\mathcal{S}_w}} w_1 \cdot \beta_1 \rightsquigarrow^{!}_{E_{\mathcal{B}}} w_{n'} \cdot \beta_{n'}.$$

Since, as seen before, $l_1 \to r_1 \in E_{\mathcal{B}}$ and $|w| < |t|$, then we can apply Lemma 5.3.37 choosing w as t and $\mathcal{S}_w$ as $\mathcal{S}$, and we have $w_{n'}\beta_{n'} \equiv w$ or $w_{n'}\beta_{n'} \to w \in E_{\mathcal{B}}$. Now consider the rewrite sequence $D^w(\curvearrowright_{E_{\mathcal{B}}})$. Since $s \equiv C[w]_{p'_b}$, then the rewrite sequence D'_E consisting of the same steps of $D^w(\curvearrowright_{E_{\mathcal{B}}})$ but where the context $C[\,]_{p'_b}$ is added to each term is a rewrite sequence with $\curvearrowright_{E_{\mathcal{B}}}$ whose first term corresponds to s and whose last term corresponds to $C[w_{n'}]_{p'_b} \cdot \beta_{n'}$. Note that, by definition of $\rightsquigarrow_{E_{\mathcal{B}}}$, we have that $w_{n'} \notin \mathcal{X}$.

- If $w_{n'}\beta_{n'} \equiv w$, then $C[w_{n'}]_{p'_b} \cdot \beta_{n'}$ is irreducible w.r.t. $\rightsquigarrow_{E_{\mathcal{B}}}$, since $w_{n'} \cdot \beta_{n'}$ is irreducible w.r.t. $\rightsquigarrow_{E_{\mathcal{B}}}$, $w \notin \mathit{Left}(E_{\mathcal{B}})$ by Lemma 5.3.12, and positions higher than p'_b are also irreducible w.r.t. $\rightsquigarrow_{E_{\mathcal{B}}}$ by Lemma 5.3.15, given that p'_b is the highest position of s above p' such that $s|_{p'_b}$ is a $\mathcal{B}$-term (and hence higher positions are not $\mathcal{B}$-positions), $C[w]_{p'_b} \equiv s$ and $s \to t \in \mathcal{R}$.
- If $w_{n'}\beta_{n'} \to w \in E_{\mathcal{B}}$, then we have not only $C[w_{n'}]_{p'_b} \cdot \beta_{n'} \curvearrowright_{E_{\mathcal{B}}} C[z]_{p'_b} \cdot \{z \mapsto w\}$, since $w_{n'} \notin \mathcal{X}$, but also $C[w_{n'}]_{p'_b} \cdot \beta_{n'} \rightsquigarrow_{E_{\mathcal{B}}} C[z]_{p'_b} \cdot \{z \mapsto w\}$, since $p'_b \neq \lambda$ and hence $C[z]_{p'_b} \notin \mathcal{X}$. And $C[z]_{p'_b} \cdot \{z \mapsto w\}$ is irreducible w.r.t. $\rightsquigarrow_{E_{\mathcal{B}}}$ by Lemma 5.3.15 for the same reasons as in the previous case.

Then in either case there is a rewrite sequence

$$D''_E = t'_1 \cdot \alpha'_1 \curvearrowright_{\{l'_1 \to r'_1\}} t'_2 \cdot \alpha'_2 \cdots t'_{m-1} \cdot \alpha'_{m-1} \curvearrowright_{\{l'_{m-1} \to r'_{m-1}\}} t'_m \cdot \alpha'_m,$$

where $t'_1 \cdot \alpha'_1 \equiv C[r]_p \cdot \{\}$, consisting of the steps of D'_E plus a possibly added last step so that $t'_m\alpha'_m \equiv C[w]_{p'_b}$ and $t'_m \cdot \alpha'_m$ is irreducible w.r.t. $\rightsquigarrow_{E_{\mathcal{B}}}$.
Now let $i_1 \ldots i_k$ denote the set of indices i in $1 \ldots m-1$ such that the *i-th* step in D''_E is applied at a position above p (note that, although no steps are applied strictly below p in D'_E, some steps may be applied at positions which are not overlapped with p, and hence not all steps are necessarily applied at a position above p). We will prove that for every step of the form

$$t'_{i_j} \cdot \alpha'_{i_j} \curvearrowright_{\{l'_{i_j} \to r'_{i_j}\}} t'_{i_j}[x'_{i_j}]_{p'_{i_j}} \cdot \alpha'_{i_j+1}$$

in D''_E, there is a step of the form

$$t_j \cdot \alpha_j \curvearrowright_{\underline{\{l'_{i_j} \to r'_{i_j}\}}} t_j[x_j]_{p'_{i_j}} \cdot \alpha_{j+1}$$

in D_E, i.e., there is a one-to-one correspondence between the steps in D''_E and the first k steps in D_E. We proceed by induction on j:

- If $j = 1$, then no previous step to the i_j*-th* in D''_E has been applied at any position above p. Therefore, although possibly $i_j > 1$, the fact that $t'_1 \cdot \alpha'_1 \equiv C[r]_p \cdot \{\}$ and $C[r]_p|_{p'} \equiv l_1$ gives us $p'_{i_j} = p'$ and $l'_{i_j} \to r'_{i_j} = l_1 \to r_1$. And, since $t_1 \cdot \alpha_1 \equiv C[x]_p \cdot \{x \mapsto r\}$, the step

  $$t_1 \cdot \alpha_1 \curvearrowright_{\underline{\{l_1 \to r_1\}}} t_1[x_1]_{p'} \cdot \{x_1 \mapsto r_1\}$$

  is in D_E.
- Now we prove the induction step, namely that if

  $$t_j \cdot \alpha_j \curvearrowright_{\underline{\{l'_{i_j} \to r'_{i_j}\}}} t_j[x_j]_{p'_{i_j}} \cdot \alpha_{j+1}$$

  and $j < k$, then

  $$t_{j+1} \cdot \alpha_{j+1} \curvearrowright_{\underline{\{l'_{i_{j+1}} \to r'_{i_{j+1}}\}}} t_{j+1}[x_{j+1}]_{p'_{i_{j+1}}} \cdot \alpha_{j+2}.$$

  For this, consider the i_{j+1}*-th* step in D''_E, which is of the form

  $$t'_{i_{j+1}} \cdot \alpha'_{i_{j+1}} \curvearrowright_{\{l'_{i_{j+1}} \to r'_{i_{j+1}}\}} t'_{i_{j+1}}[x'_{i_{j+1}}]_{p'_{i_{j+1}}} \cdot \alpha'_{i_{j+1}+1}.$$

  First of all, we will prove that $t'_{i_{j+1}}\alpha'_{i_{j+1}}|_{p'_{i_j}} \equiv r'_{i_j}$, $t_{j+1}\alpha_{j+1}|_{p'_{i_j}} \equiv r'_{i_j}$ and $t'_{i_{j+1}}\alpha'_{i_{j+1}}[\,]_{p'_{i_j}}|_{p'_{i_{j+1}}} \equiv t_{j+1}\alpha_{j+1}[\,]_{p'_{i_j}}|_{p'_{i_{j+1}}}$.

- W.r.t. $t'_{i_{j+1}}\alpha'_{i_{j+1}}|_{p'_{i_j}} \equiv r'_{i_j}$, it holds since the i_{j+1}-*th* step in D''_E is the first one after the i_j-*th* which is applied at a position above p'_{i_j}.
- W.r.t. $t_{j+1}\alpha_{j+1}|_{p'_{i_j}} \equiv r'_{i_j}$, it holds since $t_{j+1}\cdot\alpha_{j+1} \equiv t_j[x_j]_{p'_{i_j}}\cdot\{x_j \mapsto r'_{i_j}\}$.
- W.r.t. $t'_{i_{j+1}}\alpha'_{i_{j+1}}[\,]_{p'_{i_j}}|_{p'_{i_{j+1}}} \equiv t_{j+1}\alpha_{j+1}[\,]_{p'_{i_j}}|_{p'_{i_{j+1}}}$, we proceed by contradiction.

  Let us suppose that there is some position q'' below $p'_{i_{j+1}}$ which is not overlapped with p'_{i_j} and such that $t'_{i_{j+1}}\alpha'_{i_{j+1}}|_{q''} \not\equiv t_{j+1}\alpha_{j+1}|_{q''}$. Then q'' must be a prefix of some position q' in s such that $s|_{q'} \equiv l_1$: otherwise, by the form of the two rewrite sequences D''_E and D_E, we would have $t'_{i_{j+1}}\alpha'_{i_{j+1}}|_{q''} \equiv t'_1\alpha'_1|_{q''} \equiv C[r]_p|_{q''} \equiv t_1\alpha_1|_{q''}$, and $t_{j+1}\alpha_{j+1}|_{q''} \equiv t_1\alpha_1|_{q''}$ since q'' is not overlapped with p'_{i_j}.

  In this case, as stated before, we have that $s|_{q'_b}$ is a $\mathcal{B}$-term for some position q'_b above q' which is not overlapped with p', and hence neither with p'_{i_j}. Therefore we have that $w \equiv s|_{p'_b} \rhd s|_{q'_b} \equiv w'$, where w and w' are $\mathcal{B}$-terms. W.l.o.g., we assume that q'_b is the highest $\mathcal{B}$-position above q' in s which is not overlapped with p.

  Now note that the innermost rewriting strategy of the rewrite sequence D^w on w implies an innermost rewriting strategy on its subterm w'. And, by Lemma 5.3.37, we have that if $w' \curvearrowright^{!}_{S_w} w'_1\cdot\rho' \rightsquigarrow^{!}_{E_{\mathcal{B}}} w'_{m'}\cdot\beta'$, then $w'_{m'}\beta' \equiv w'$ or $w'_{m'}\beta' \to w' \in E_{\mathcal{B}}$. Therefore, some term of the form $t'_{k'}\cdot\alpha'_{k'}$ with $t'_{k'}\alpha'_{k'}|_{q'_b} \equiv w'$ is obtained in D''_E before the i_{j+1}-*th* step (observe that $p'_{i_{j+1}}$ is strictly above q'_b, since $p'_{i_{j+1}}$ is overlapped with p', q'_b is not overlapped with p', q'_b is above q' and q' is below $p'_{i_{j+1}}$; therefore, in the case that $w'_{m'}\beta' \not\equiv w'$, then the rule $w'_{m'}\beta' \to w'$ in $E_{\mathcal{B}}$ would be applied before the i_{j+1}-*th* step in D''_E given that, by Lemma 5.3.13, there are no two different rules in $E_{\mathcal{B}}$ with the same left-hand side).

  Now recall that $t_{j+1}\alpha_{j+1}|_{q''} \equiv C[r]_p|_{q''}$, since q'' is not overlapped with p'_{i_j}. Therefore, since the i_{j+1}-*th* step in D''_E is the first one after the i_j-*th* which is applied at a position above p'_{i_j}, if $t'_{i_{j+1}}\alpha'_{i_{j+1}}|_{q''} \not\equiv t_{j+1}\alpha_{j+1}|_{q''}$ then necessarily some step in D''_E before the i_{j+1}-*th* must have been applied at some position strictly above q'_b which is not overlapped with p'_{i_j}, and hence neither with p.

  Then there is a rule of the form $C[r]_p|_{q''_b} \to u$ for some u in $E_{\mathcal{B}}$, where q''_b is strictly above q'_b but not overlapped with p, and hence $s \rhd s|_{q''_b} \equiv C[r]_p|_{q''_b}$. And, since t is of the form $C[l]_p$, we have that $t|_{q''_b} \equiv s|_{q''_b}$. Now observe that q'_b is the highest $\mathcal{B}$-position above q' in $P_{\mathcal{B}^{\lhd}}(t)$, since p is not in $P_{\mathcal{B}^{\lhd}}(t)$ and q'_b is the highest $\mathcal{B}$-position above q' in s which is not overlapped with p. Thus, q''_b is a position not in $P_{\mathcal{B}^{\lhd}}(t)$. Then, by Lemma 5.3.25, we have that $C[r]_p|_{q''_b} \to u \succeq_E s \to t$, contradicting $s \rhd s|_{q''_b}$.

So we have that $t'_{i_{j+1}}\alpha'_{i_{j+1}}|_{p'_{i_{j+1}}} \equiv t_{j+1}\alpha_{j+1}|_{p'_{i_{j+1}}}$.

To conclude with the induction step recall that, by definition of $\rightsquigarrow_{E_{\mathcal{B}}}$, there is no rule $l' \to r'$ in $E_{\mathcal{B}}$ with $l'_{i_{j+1}} \rhd l'$ reducing $t'_{i_{j+1}}\alpha'_{i_{j+1}}$ at a position above p'_{i_j}. And there is neither no rule $l'' \to r''$ in $E_{\mathcal{B}}$ with $l_{j+1} \rhd l''$ reducing $t_{j+1}\alpha_{j+1}$ at a position above p'_{i_j}. Then, by Lemma 5.3.13, we have that $l_{j+1} \to r_{j+1} = l'_{i_{j+1}} \to r'_{i_{j+1}}$, and hence

$$t_{j+1}\cdot\alpha_{j+1} \curvearrowright_{\underline{\{l'_{i_{j+1}} \to r'_{i_{j+1}}\}}} t_{j+1}[x_{j+1}]_{p'_{i_{j+1}}}\cdot\alpha_{j+2}.$$

Moreover, we have that $t'_m\alpha'_m|_{p'_{i_k}} \equiv r'_{i_k}$ and $t'_m\alpha'_m[\,]_{p'_{i_k}}|_{p'_b} \equiv t_{k+1}\alpha_{k+1}[\,]_{p'_{i_k}}|_{p'_b}$:

- W.r.t. $t'_m\alpha'_m|_{p'_{i_k}} \equiv r'_{i_k}$, it holds since no steps after the i_k-*th* in D''_E are applied above p.
- W.r.t. $t'_m\alpha'_m[]_{p'_{i_k}}|_{p'_b} \equiv t_{k+1}\alpha_{k+1}[]_{p'_{i_k}}|_{p'_b}$, it can be proved with a reasoning identical to the one used before to prove that $t'_{i_{j+1}}\alpha'_{i_{j+1}}[]_{p'_{i_j}}|_{p'_{i_{j+1}}} \equiv t_{j+1}\alpha_{j+1}[]_{p'_{i_j}}|_{p'_{i_{j+1}}}$, where m plays the role of i_{j+1}, k the role of j, and p'_b the role of $p'_{i_{j+1}}$.

So we have that $t'_m\alpha'_m|_{p'_b} \equiv t_{k+1}\alpha_{k+1}|_{p'_b}$, since $t_{k+1}\cdot\alpha_{k+1} \equiv t_k[x_k]_{p'_{i_k}}\cdot\{x_k \mapsto r'_{i_k}\}$.
Moreover, $t'_m\alpha'_m[]_{p'_b} \equiv C[]_{p'_b}$ and $t_{k+1}\alpha_{k+1}[]_{p'_b} \equiv C[]_{p'_b}$, since there are no steps at positions above p'_b in D''_E (by definition), and the first k steps in D_E correspond to the steps in D''_E. So we have that $t'_m\alpha'_m \equiv t_{k+1}\alpha_{k+1}$.
Now recall that $t'_m\cdot\alpha'_m$ is irreducible w.r.t. $\rightsquigarrow_{E_\mathcal{B}}$. Then also must be $t_{k+1}\cdot\alpha_{k+1}$, since its only blocked position is p'_{i_k}, that is, $P_\mathcal{X}(t_{k+1}) = \{p'_{i_k}\}$, and p'_{i_k} is also a variable position of $t'_m\cdot\alpha'_m$, since p'_{i_k} was blocked in the i_k-*th* step of D''_E, which is the last step applied at a position above p. Finally, the irreducibility of $t_{k+1}\cdot\alpha_{k+1}$ w.r.t. $\rightsquigarrow_{E_\mathcal{B}}$ gives us $k+1 \equiv n$ and, since $t_{k+1}\alpha_{k+1} \equiv t'_m\alpha'_m \equiv C[w]_{p'_b} \equiv s \equiv C[r]_p$, then we are done. □

Lemma 5.3.33 *Let t be a term in $\mathcal{B}$, and $\mathcal{S}$ be a non-empty and non-overlapping subset of $E_\mathcal{B}$ such that each rule in $\mathcal{S}$ reduces t at some position not in $P_{\mathcal{B}^\triangleleft}(t)$. Then there is an equation $u \simeq v \in E$ such that, for some substitution σ, we have*

1. *$u\sigma \equiv t$,*
2. *$v\sigma \equiv t_n\alpha$, where $t \curvearrowright^!_{\overline{\mathcal{S}}} t_1\cdot\rho \rightsquigarrow^*_{E_\mathcal{B}} t_n\cdot\alpha$, and*
   (a) *$t_1\cdot\rho \rightsquigarrow^!_{E_\mathcal{B}} t_n\cdot\alpha$ and $t_n\alpha \to t \in \mathcal{R}$, or*
   (b) *$red\_vars(v, t_n\cdot\alpha, \curvearrowright_{\underline{E_\mathcal{B}}}) \cap (vars(u) \cup green\_vars(v, t_n)) = \emptyset$.*

Proof. First of all note that, since $\mathcal{S}$ is non-overlapping then, by Lemma 5.3.20, t has exactly one normal form $t_1\cdot\rho$ w.r.t. $\curvearrowright_{\overline{\mathcal{S}}}$. And, by Lemma 5.3.21, the rewrite sequence $t_1\cdot\rho \rightsquigarrow^*_{E_\mathcal{B}} t_n\cdot\alpha$ is unique up to permutation of non-overlapped steps.
If t is non-self-returning for any rule in $\mathcal{S}$ (see Definition 5.3.28) then, by Lemma 5.3.29, there is an equation $u \simeq v \in E$ such that, for some substitution φ, we have

1. $u\varphi \equiv t$,
2. $v\varphi \equiv t_0\delta$, where $t \curvearrowright^+_{\overline{\mathcal{S}}} t_0\cdot\delta$,
3. $\forall q \in red\_pos(t_0\cdot\delta, \curvearrowright_{\overline{\mathcal{S}}})$: q is a valid non-variable position in v and
4. $yellow\_vars(v, t_0) \cap (vars(u) \cup green\_vars(v, t_0)) = \emptyset$.

Then, by Lemma 5.3.30, there is an equation $u' \simeq v'$ in E such that, for some ground substitution ψ, we have

1. $u'\psi \equiv u\varphi$,
2. $v'\psi \equiv t_1\rho$, where $t_0\cdot\delta \curvearrowright^!_{\overline{\mathcal{S}}} t_1\cdot\rho$, and
3. $yellow\_vars(v', t_1) \cap (vars(u') \cup green\_vars(v', t_1)) = \emptyset$.

In this case we can conclude by choosing u' as u, v' as v, ψ as σ and $t_1 \cdot \rho$ as $t_n \cdot \alpha$ since, by definition, it holds that $red\_vars(v, t_1 \cdot \rho, \curvearrowright_{\underline{E_{\mathcal{B}}}}) \subseteq yellow\_vars(v, t_1)$.

If, otherwise, t is self-returning for some rule $l \to r$ in $\mathcal{S}$ then, by Lemma 5.3.31, $l \to r$ must be the only rule in $\mathcal{S}$, and $l \to r$ must reduce t at one single position p not in $P_{\mathcal{B}^{\lhd}}(t)$. Now let $s' \simeq t'$ be an equation in E such that $s'\varphi \equiv s$ and $t'\varphi \equiv t$ for some substitution φ, where $s \to t$ is the minimal rule w.r.t. $\succ_E$ in $\mathcal{R}$ with t as right-hand side. Note that, since t is self-returning for $l \to r$, we have that $s \equiv t[r]_p$, and we can assume w.l.o.g. that p is a non-variable position of t'. We denote $t[x]_p$ by t_1 and $\{x \mapsto r\}$ by ρ. Since $l \to r$ reduces t only at position p, we clearly have that $t \curvearrowright^{!}_{\mathcal{S}} t_1 \cdot \rho$. The following cases have to be considered:

1. $t_1 \cdot \rho$ is irreducible w.r.t. $\rightsquigarrow_{E_{\mathcal{B}}}$. Then we choose t' as u, s' as v, φ as σ and $t_1 \cdot \rho$ as $t_n \cdot \alpha$, and we have

   - $u\sigma \equiv t$, since $t'\varphi \equiv t$,
   - $v\sigma \equiv t_n\alpha$, where $t \curvearrowright^{!}_{\mathcal{S}} t_1 \cdot \rho \rightsquigarrow^{*}_{E_{\mathcal{B}}} t_n \cdot \alpha$ since, as seen before, we have $t \curvearrowright^{!}_{\mathcal{S}} t_1 \cdot \rho$ and $v\sigma \equiv s'\varphi \equiv s \equiv t[r]_p \equiv t_1\rho \equiv t_n\alpha$, and
   - $t_1 \cdot \rho \rightsquigarrow^{!}_{E_{\mathcal{B}}} t_n \cdot \alpha$, since $t_n \cdot \alpha \equiv t_1 \cdot \rho$ and $t_1 \cdot \rho$ is irreducible w.r.t. $\rightsquigarrow_{E_{\mathcal{B}}}$, and $t_n\alpha \to t \in \mathcal{R}$ since $t_n\alpha \equiv s$ and $s \to t \in \mathcal{R}$.

   Thus we are done.

2. $t_1 \cdot \rho$ is reducible w.r.t. $\rightsquigarrow_{E_{\mathcal{B}}}$. In this case, we have that $t_1 \cdot \rho \rightsquigarrow_{E_{\mathcal{B}}} t_1' \cdot \rho'$ for some abstracted term $t_1' \cdot \rho'$. Then, for some rule $l_1 \to r_1$ in $E_{\mathcal{B}}$, we have

   $$t_1 \cdot \rho \curvearrowright^{\parallel}_{\underline{\{l_1 \to r_1\}}} t_1' \cdot \rho'.$$

   Note that, since t_1 has only one variable, then necessarily the rewrite sequence has only one step, that is, we have $red\_pos(t_1 \cdot \rho, \rightsquigarrow_{E_{\mathcal{B}}}) = \{p'\}$ for some prefix p' of p and

   $$t_1 \cdot \rho \equiv t[x]_p \cdot \{x \mapsto r\} \curvearrowright_{\underline{\{l_1 \to r_1\}}} t[y]_{p'} \cdot \{y \mapsto r_1\} \equiv t_1' \cdot \rho'.$$

   Moreover, $p' \not\equiv p$ since, by definition of $\curvearrowright_{\underline{E_{\mathcal{B}}}}$, p' must be a non-variable position of t_1. We have the following possibilities:

   (a) p' is a non-variable position of s'. In this case we choose t' as u, s' as v, φ as σ and $t_1 \cdot \rho$ as $t_n \cdot \alpha$ as in case 1. Conditions 1 and 2 are fulfilled as in case 1. But in this case condition 2b is fulfilled instead of condition 2a: we have that $t_n \cdot \alpha$ is reducible by $\curvearrowright_{\underline{E_{\mathcal{B}}}}$ only at positions above p', which is a non-variable position of s'; hence, it follows that $red\_vars(v, t_n \cdot \alpha, \curvearrowright_{\underline{E_{\mathcal{B}}}}) = \emptyset$.

   (b) $s'|_{p''}$ is a variable z for some prefix p'' of p'.

      i. $t|_{p'} \equiv l_1$. This is not possible, since $s \equiv t[l_1]_{p'}$ and then we would have $s \equiv t$, contradicting $s \succ t$.

      ii. $t|_{p'} \rhd l_1$. In this case, since $s \equiv t[l_1]_{p'}$, we have that $|s| < |t|$. Now recall that $s \equiv t[r]_p$ and $t|_p \equiv l$, that is, s is of the form $C[r]_p$ and t is of the form $C[l]_p$. Then, by Lemma 5.3.32, we have that $C[x]_p \cdot \{x \mapsto r\} \rightsquigarrow^{!}_{E_{\mathcal{B}}} w \cdot \gamma$, where $w\gamma \equiv C[r]_p$. Therefore $t_1 \cdot \rho \rightsquigarrow^{!}_{E_{\mathcal{B}}} w \cdot \gamma$, since t_1 denotes $t[x]_p$ and ρ denotes $\{x \mapsto r\}$. And moreover $w\gamma \equiv s$ and $s \to t \in \mathcal{R}$. Then we choose t' as u, s' as v and $w \cdot \gamma$ as $t_n \cdot \alpha$, and conditions 1, 2 and 2a are fulfilled.

iii. $t|_{p'} \not\trianglerighteq l_1$. In this case, recall that p is a non-variable position of t', and hence so is p'. Moreover, since $s \equiv t[r]_p \equiv t[l_1]_{p'}$ and $s|_{p''}$ is a variable z for some prefix p'' of p', we have that $z\varphi \trianglerighteq l_1$. Therefore, since $t|_{p'} \not\trianglerighteq l_1$, in this case we must have that $t'|_{p'} \not\trianglerighteq z$, and we have the following possibilities:

A. s' is a variable. Then necessarily $s' = z$. In this case $z \notin vars(t')$ since, on the one hand, we have that $t'|_{p'} \not\trianglerighteq z$ and hence $t'|_p \not\trianglerighteq z$ and, on the other hand, $t'|_q \not\equiv z$ for any position q which is not overlapped with p, since otherwise, given that $s' = z$, we would have $s \not\equiv t[r]_p$. Then we choose t' as u, s' as v, φ as σ and $t_1 \cdot \rho$ as $t_n \cdot \alpha$ as in case 1. Conditions 1 and 2 are fulfilled for the same reasons. And we have that $red\_vars(v, t_n \cdot \alpha, \curvearrowright_{\underline{E_{\mathcal{B}}}}) \cap (vars(u) \cup green\_vars(v, t_n)) = \emptyset$, i.e., condition 2b is fulfilled for the following reasons:

- $red\_vars(v, t_n \cdot \alpha, \curvearrowright_{\underline{E_{\mathcal{B}}}}) \subseteq vars(v) = \{z\}$,
- $z \notin vars(u)$ since, as we have seen, $z \notin vars(t')$, and
- $green\_vars(v, t_n) = \emptyset$, since $v|_\lambda \equiv z$ and λ is strictly above p (we have that $p \neq \lambda$ since $t \notin Left(E_{\mathcal{B}})$ by Lemma 5.3.12), which is a variable position of t_n.

B. s' is not a variable. Since p is a non-variable position of t', there is an inference by ordered paramodulation,

$$\frac{l' \simeq r' \qquad s' \simeq t'}{(s' \simeq t'[r']_p)\theta},$$

where $\theta = mgu(t'|_p, l')$ and $l' \simeq r'$ is an equation in E such that, w.l.o.g., $l'\varphi \equiv l$ and $r'\varphi \equiv r$. Note that, although $(s' \simeq t'[r']_p)\theta\varphi$ is a tautology of the form $w \simeq w$ (since $s'\theta\varphi \equiv s \equiv t[r]_p \equiv t'[r']_p\theta\varphi$), we can prove that there is another substitution φ' such that $(s' \simeq t'[r']_p)\theta\varphi'$ is not a tautology: recall that $s'|_{p''}$ is a variable z for some prefix p'' (of p' and) of p, and $s'\varphi \equiv s \equiv t[l]_p$; hence, $p = p'' \cdot q$ for some position q of $z\varphi$; let φ' be a substitution such that $z\varphi' \equiv z\varphi[l]_q$ and $z'\varphi' \equiv z'\varphi$ for all other variables z; then we have that $s'\theta\varphi'|_p \equiv l$ and $t'[r']_p\theta\varphi'|_p \equiv r$ since $t'|_{p'} \not\trianglerighteq z$.

Since $(s' \simeq t'[r']_p)\theta\varphi'$ is not a tautology, we can assume that $(s' \simeq t'[r']_p)\theta$ is in E. Moreover, we have that $t'[r']_p\theta\varphi'|_{p'} \equiv t[r]_p|_{p'} \equiv l_1$. Now let $l'_1 \simeq r'_1$ be an equation in E such that, w.l.o.g., $l'_1\varphi \equiv l_1$ and $r'_1\varphi \equiv r_1$. Then there is an inference by ordered paramodulation,

$$\frac{l'_1 \simeq r'_1 \qquad s'\theta \simeq t'[r']_p\theta}{(s'\theta \simeq t'\theta[r'_1]_{p'})\theta'},$$

where $\theta' = mgu(t'[r']_p\theta|_{p'}, l'_1)$. Note that this is a correct inference: p' is a non-variable position of $t'[r']_p\theta$ since it already was a non-variable position of t'. Moreover, we have that $s'\theta\theta'\varphi \equiv s \equiv t[l_1]_{p'}$ and $t'\theta[r'_1]_{p'}\theta'\varphi \equiv t[r_1]_{p'}$. Therefore, $(s'\theta \simeq t'\theta[r'_1]_{p'})\theta'\varphi$ is not a tautology, and hence we can assume that $(s'\theta \simeq t'\theta[r'_1]_{p'})\theta'$ is in E. Moreover, since s' is not a variable, then $s'\theta\theta'$ is not a variable either. Then there is an inference by ordered paramodulation,

$$\frac{s' \simeq t' \qquad s'\theta\theta' \simeq t'\theta[r'_1]_{p'}\theta'}{(t' \simeq t'\theta[r'_1]_{p'}\theta')\theta''},$$

where $\theta'' = mgu(s'\theta\theta', s')$, whose conclusion has an instance of the form $t \simeq t[r_1]_{p'}$ under the substitution φ. Now recall that we have that $t \curvearrowright^{!}_{S} t_1 \cdot \rho \rightsquigarrow_{E_{\mathcal{B}}}$

$t_1' \cdot \rho'$ with $t_1'\rho' \equiv t[r_1]_{p'}$. Then $t \not\equiv t[r_1]_{p'}$ by Lemma 5.3.19, since $\curvearrowright_{\underline{S}} \subseteq \curvearrowright_{E_{\mathcal{B}}}$, $\rightsquigarrow_{E_{\mathcal{B}}} \subseteq \curvearrowright_{E_{\mathcal{B}}}$, and $p' \neq \lambda$ by definition of $\rightsquigarrow_{E_{\mathcal{B}}}$.
Therefore we can assume that $(t' \simeq t'\theta[r_1']_{p'}\theta')\theta''$ is in E. Then we choose $t'\theta''$ as u, $t'\theta[r_1']_{p'}\theta'\theta''$ as v, φ as σ and $t_1' \cdot \rho'$ as $t_n \cdot \alpha$. Conditions 1 and 2 are clearly fulfilled, and also condition 2b: on the one hand, p' is the only variable position of t_n; on the other hand, since p' is a non-variable position of $t'[r']_p\theta$, there is no variable strictly above p' in $t'\theta[r_1']_{p'}\theta'\theta''$; hence, $red\_vars(v, t_n \cdot \alpha, \curvearrowright_{\underline{E_{\mathcal{B}}}}) = \emptyset$.

□

Lemma 5.3.34 *If there is an equation $u \simeq v$ in E such that, for some substitution σ, we have*

1. $u\sigma \curvearrowright_{E_{\mathcal{B}}} u\sigma[x]_{p_0} \cdot \varepsilon \curvearrowright^*_{E_{\mathcal{B}}} t_0 \cdot \alpha$ *for some position p_0 not in $P_{\mathcal{B}\triangleleft u\sigma}(u\sigma)$,*
2. $v\sigma \equiv t_2\gamma$, *where* $t_0 \cdot \alpha \curvearrowright^*_{E_{\mathcal{B}}} t_1 \cdot \beta \curvearrowright^*_{\underline{\{l \to r\}}} t_2 \cdot \gamma$,
3. $\forall q \in red\_pos(t_1 \cdot \beta, \curvearrowright_{\underline{\{l \to r\}}}) : t_2|_q \in \mathcal{X}$ *or* $t_2|_q \equiv t_1|_q$, *and*
4. $red\_vars(v, t_2 \cdot \gamma, \curvearrowright_{\underline{E_{\mathcal{B}}}}) \cap (vars(u) \cup green\_vars(v, t_2)) = \emptyset$,

where $l \to r$ is a rule in $E_{\mathcal{B}}$ such that $l \not\equiv t_1\beta$ and there is no rule $s \to t$ in $E_{\mathcal{B}}$ with $l \rhd s$ reducing $t_1 \cdot \beta$ w.r.t. $\curvearrowright_{\underline{\{s \to t\}}}$, then there is an equation $u' \simeq v'$ in E such that, for some substitution σ', we have

1. $u'\sigma' \equiv u\sigma$,
2. $v'\sigma' \equiv t_3\delta$, *where* $t_1 \cdot \beta \curvearrowright^*_{\underline{\{l \to r\}}} t_3 \cdot \delta$,
3. $\forall q \in red\_pos(t_1 \cdot \beta, \curvearrowright_{\underline{\{l \to r\}}}) : t_3|_q \in \mathcal{X}$, *and*
4. $red\_vars(v', t_3 \cdot \delta, \curvearrowright_{\underline{E_{\mathcal{B}}}}) \cap (vars(u') \cup green\_vars(v', t_3)) = \emptyset$.

Proof. Let
$$Q = \{q \in red\_pos(t_1 \cdot \beta, \curvearrowright_{\underline{\{l \to r\}}}) \mid t_2|_q \equiv t_1|_q\}.$$
We proceed by induction on $(u \simeq v)\sigma$ w.r.t. $\#Q$. If $\#Q = 0$, then we are done: we choose u as u', v as v' and σ as σ' (and hence $t_2 \cdot \gamma$ as $t_3 \cdot \delta$). Otherwise, let p be a position in Q. Now, from

$$t_1 \cdot \beta \curvearrowright^*_{\underline{\{l \to r\}}} t_2 \cdot \gamma$$

and $p \in Q$, it follows that
$$p \in red\_pos(t_2 \cdot \gamma, \curvearrowright_{\underline{\{l \to r\}}}).$$
Moreover, note that $p \neq \lambda$, since $p \in red\_pos(t_1 \cdot \beta, \curvearrowright_{\underline{\{l \to r\}}})$ and $l \not\equiv t_1\beta$.
In the rest of this proof, by $P_{\mathcal{X}}(v, \sigma)$ we denote the set of positions of $v\sigma$ which are below a variable position of v. Two cases have to be considered:

1. $p \notin P_{\mathcal{X}}(v, \sigma)$. In this case, first of all note that $u\sigma \simeq v\sigma$ is not a tautology of the form $w \simeq w$: we have that $u\sigma \curvearrowright_{E_{\mathcal{B}}} u\sigma[x]_{p_0} \cdot \varepsilon \curvearrowright^*_{E_{\mathcal{B}}} t_2 \cdot \gamma$ with $p_0 \notin P_{\mathcal{B}\triangleleft u\sigma}(u\sigma)$ and $t_2 \notin \mathcal{X}$ (since $t_2 \cdot \gamma$ is reducible at position p); then, by Lemma 5.3.19, we have that $t_2\gamma \not\equiv u\sigma$.

   Therefore, there is an inference by ordered paramodulation,
$$\frac{l' \simeq r' \qquad u \simeq v}{(u \simeq v[r']_p)\theta},$$

where $l' \simeq r'$ is an equation in E such that, w.l.o.g., $l'\sigma \equiv l$ and $r'\sigma \equiv r$, where $\theta = mgu(v|_p, l')$, and whose conclusion has an instance of the form $u\sigma \simeq v\sigma[r]_p$. Moreover, we have that

$$t_2 \cdot \gamma \curvearrowright_{\underline{\{l \to r\}}} t_2[y]_p \cdot (\gamma \cup \{y \mapsto r\})$$

for some variable y. Since $p \neq \lambda$, we have that $u\sigma \simeq v\sigma[r]_p$ is not a tautology of the form $w \simeq w$ by Lemma 5.3.19.

Therefore $(u \simeq v[r']_p)\theta$ is in E. Now we prove that the required conditions for $u \simeq v$ are also fulfilled by $(u \simeq v[r']_p)\theta$. For this, let t_2' denote $t_2[y]_p$ and γ' denote $\gamma \cup \{y \mapsto r\}$. Note that $t_2'\gamma' \equiv v\sigma[r]_p$. Then the two fist conditions are clearly fulfilled. We detail the proof of the other two conditions:

- $\forall q \in red\_pos(t_1 \cdot \beta, \curvearrowright_{\underline{\{l \to r\}}}) : t_2'|_q \in \mathcal{X}$ or $t_2'|_q \equiv t_1|_q$.

  This holds since $\forall q \in red\_pos(t_1 \cdot \beta, \curvearrowright_{\underline{\{l \to r\}}}) : t_2|_q \in \mathcal{X}$ or $t_2|_q \equiv t_1|_q$, and we have that $t_2' \equiv t_2[y]_p$ and $p \in red\_pos(t_1 \cdot \beta, \curvearrowright_{\underline{\{l \to r\}}})$.
- $red\_vars(v[r']_p\theta, t_2' \cdot \gamma', \curvearrowright_{\underline{E_{\mathcal{B}}}}) \cap (vars(u\theta) \cup green\_vars(v[r']_p\theta, t_2')) = \emptyset$.

  In order to prove this, two properties will be useful.

  – The first one being that, since $p \notin P_{\mathcal{X}}(v, \sigma)$, then for every variable z in v, if $z\theta$ is still a variable in $red\_vars(v[r']_p\theta, t_2' \cdot \gamma', \curvearrowright_{\underline{E_{\mathcal{B}}}})$ then $z \in red\_vars(v, t_2 \cdot \gamma, \curvearrowright_{\underline{E_{\mathcal{B}}}})$.

  – The second one being that, for every variable z in $vars(v|_p)$, we have that $z\theta \notin red\_vars(v[r']_p\theta, t_2' \cdot \gamma', \curvearrowright_{\underline{E_{\mathcal{B}}}})$:
    Otherwise, by the first property, we would have that $z \in red\_vars(v, t_2 \cdot \gamma, \curvearrowright_{\underline{E_{\mathcal{B}}}})$ and, since $z \in vars(v|_p)$ and $p \notin P_{\mathcal{X}}(v, \sigma)$, this would contradict there is no rule $\underline{s \to t}$ in $E_{\mathcal{B}}$ with $l \rhd s$ reducing $t_1 \cdot \beta$ w.r.t. $\curvearrowright_{\underline{\{s \to t\}}}$.

  Now let z' be a variable in $vars(u\theta) \cup green\_vars(v[r']_p\theta, t_2')$. The following two cases have to be considered:

  (a) $z' \in vars(u\theta)$. Then we have that $z' \equiv z\theta$ for some variable z in $vars(u)$. Again, we consider two situations.

    i. $z \notin vars(v|_p)$. Then $z\theta \equiv z$. In this case $z\theta \notin red\_vars(v[r']_p\theta, t_2' \cdot \gamma', \curvearrowright_{\underline{E_{\mathcal{B}}}})$ since, otherwise, by the first property we would have that $z \in red\_vars(v, t_2 \cdot \gamma, \curvearrowright_{\underline{E_{\mathcal{B}}}})$, contradicting $red\_vars(v, t_2 \cdot \gamma, \curvearrowright_{\underline{E_{\mathcal{B}}}}) \cap vars(u) = \emptyset$.

    ii. $z \in vars(v|_p)$. Then $z\theta \notin red\_vars(v[r']_p\theta, t_2' \cdot \gamma', \curvearrowright_{\underline{E_{\mathcal{B}}}})$ by the second property.

    Therefore $red\_vars(v[r']_p\theta, t_2' \cdot \gamma', \curvearrowright_{\underline{E_{\mathcal{B}}}}) \cap vars(u\theta) = \emptyset$.

  (b) $z' \in green\_vars(v[r']_p\theta, t_2')$.

    i. z' occurs in $v[\,]_p\theta$. Then $z' \equiv z\theta$ for some variable z in $green\_vars(v, t_2)$, and we can conclude that $z\theta \notin red\_vars(v[r']_p\theta, t_2' \cdot \gamma', \curvearrowright_{\underline{E_{\mathcal{B}}}})$ analogously to before:

      A. $z \notin vars(v|_p)$. Then $z\theta \equiv z$. In this case $z\theta \notin red\_vars(v[r']_p\theta, t_2' \cdot \gamma', \curvearrowright_{\underline{E_{\mathcal{B}}}})$ since, otherwise, by the first property we would have that $z \in red\_vars(v, t_2 \cdot \gamma, \curvearrowright_{\underline{E_{\mathcal{B}}}})$, contradicting $red\_vars(v, t_2 \cdot \gamma, \curvearrowright_{\underline{E_{\mathcal{B}}}}) \cap green\_vars(v, t_2) = \emptyset$.

      B. $z \in vars(v|_p)$. Then $z\theta \notin red\_vars(v[r']_p\theta, t_2' \cdot \gamma', \curvearrowright_{\underline{E_{\mathcal{B}}}})$ by the second property.

    ii. z' occurs in $r'\theta$.

A. $z' \equiv z\theta$ for some variable z in $vars(v|_p)$. As before, we conclude that $z\theta \notin red\_vars(v[r']_p\theta, t_2' \cdot \gamma', \underline{\curvearrowright_{E_{\mathcal{B}}}})$ by the second property.

B. $z' \not\equiv z\theta$ for all variables z in $vars(v|_p)$. Then z' does not occur in $v[\,]_p\theta$ and hence $z' \notin red\_vars(v[r']_p\theta, t_2' \cdot \gamma', \underline{\curvearrowright_{E_{\mathcal{B}}}})$.

Therefore $red\_vars(v[r']_p\theta, t_2' \cdot \gamma', \underline{\curvearrowright_{E_{\mathcal{B}}}}) \cap green\_vars(v[r']_p\theta, t_2') = \emptyset$.

2. $p \in P_X(v,\sigma)$. In this case $v|_{p'}$ is a variable z for some prefix p' of p, i.e., $p = p' \cdot p''$ for some p'' and $z\sigma|_{p''}$ is l. Now let σ' be the ground substitution with the same domain as σ but where $z\sigma' = z\sigma[r]_{p''}$ and $z'\sigma' = z'\sigma$ for all other variables z'. We have that $v\sigma' \equiv t_2'\gamma'$, where t_2' is of the form $t_2[x_1]_{p_1} \dots [x_n]_{p_n}$ and γ' is of the form $\gamma \cup \{x_1 \mapsto r, \dots, x_n \mapsto r\}$ for some set of variables $x_1, \dots, x_n$ and positions $p_1, \dots, p_n$.

   Let us see that $u \simeq v$ also fulfills the required conditions under this new substitution σ'. Here the key is to notice that $z \in red\_vars(v, t_2 \cdot \gamma, \underline{\curvearrowright_{E_{\mathcal{B}}}})$.

   - Condition 1 is preserved since, by condition 4, we have that $red\_vars(v, t_2 \cdot \gamma, \underline{\curvearrowright_{E_{\mathcal{B}}}}) \cap vars(u) = \emptyset$, and hence $z \cap vars(u) = \emptyset$, which gives us $u\sigma' \equiv u\sigma$.
   - Condition 2 is preserved since, by condition 4, we have that $red\_vars(v, t_2 \cdot \gamma, \underline{\curvearrowright_{E_{\mathcal{B}}}}) \cap green\_vars(v, t_2) = \emptyset$, and hence $z \cap green\_vars(v, t_2) = \emptyset$, which gives us

     $$\{p_1, \dots, p_n\} \subseteq red\_pos(t_2 \cdot \gamma, \underline{\curvearrowright_{\{l \to r\}}}),$$

     and therefore $t_2 \cdot \gamma \underline{\curvearrowright^{+}_{\{l \to r\}}} t_2' \cdot \gamma'$.
   - W.r.t. condition 3, since

     $$\forall q \in red\_pos(t_1 \cdot \beta, \underline{\curvearrowright_{\{l \to r\}}}) : t_2|_q \in \mathcal{X} \text{ or } t_2|_q \equiv t_1|_q$$

     and

     $$\{p_1, \dots, p_n\} \subseteq red\_pos(t_2 \cdot \gamma, \underline{\curvearrowright_{\{l \to r\}}}) \subseteq red\_pos(t_1 \cdot \beta, \underline{\curvearrowright_{\{l \to r\}}}),$$

     then it follows that $\forall q \in red\_pos(t_1 \cdot \beta, \underline{\curvearrowright_{\{l \to r\}}}) : t_2'|_q \in \mathcal{X}$ or $t_2'|_q \equiv t_1|_q$.
   - Finally, condition 4 is preserved since, if $p'' \neq \lambda$, then

     $$red\_vars(v, t_2' \cdot \gamma', \underline{\curvearrowright_{E_{\mathcal{B}}}}) = red\_vars(v, t_2 \cdot \gamma, \underline{\curvearrowright_{E_{\mathcal{B}}}})$$

     and

     $$green\_vars(v, t_2') = green\_vars(v, t_2)$$

     and if, otherwise, $p'' = \lambda$, then

     $$red\_vars(v, t_2' \cdot \gamma', \underline{\curvearrowright_{E_{\mathcal{B}}}}) = red\_vars(v, t_2 \cdot \gamma, \underline{\curvearrowright_{E_{\mathcal{B}}}}) \setminus \{z\}$$

     and

     $$green\_vars(v, t_2') = green\_vars(v, t_2) \cup \{z\}.$$

Once seen that the conditions are preserved, we conclude by the induction hypothesis, since

$$\#\{q \in red\_pos(t_1 \cdot \beta, \underline{\curvearrowright_{\{l \to r\}}}) \mid t_2|_q \equiv t_1|_q\} > \#\{q \in red\_pos(t_1 \cdot \beta, \underline{\curvearrowright_{\{l \to r\}}}) \mid t_2'|_q \equiv t_1|_q\}.$$

$\square$

Lemma 5.3.35 *If there is an equation $u \simeq v$ in E such that, for some substitution σ, we have*

1. $u\sigma \curvearrowright_{E_{\mathcal{B}}} u\sigma[x]_{p_0} \cdot \varepsilon \curvearrowright^{*}_{E_{\mathcal{B}}} t_1 \cdot \rho$ *for some position p_0 not in $P_{\mathcal{B}^{\lhd u\sigma}}(u\sigma)$,*
2. $v\sigma \equiv t_n\alpha$*, where* $t_1 \cdot \rho \rightsquigarrow^{*}_{E_{\mathcal{B}}} t_n \cdot \alpha$*, and*
3. $red\_vars(v, t_n \cdot \alpha, \curvearrowright_{\underline{E_{\mathcal{B}}}}) \cap (vars(u) \cup green\_vars(v, t_n)) = \emptyset$*,*

then there is an equation $u' \simeq v'$ in E such that, for some substitution σ', we have

1. $u'\sigma' \equiv u\sigma$*,*
2. $v'\sigma' \equiv t_m\beta$*, where* $t_n \cdot \alpha \rightsquigarrow^{!}_{E_{\mathcal{B}}} t_m \cdot \beta$*, and*
3. $red\_vars(v', t_m \cdot \beta, \curvearrowright_{\underline{E_{\mathcal{B}}}}) \cap (vars(u') \cup green\_vars(v', t_m)) = \emptyset$*.*

Proof. We proceed by induction on $\rightsquigarrow^{+}_{E_{\mathcal{B}}}$. If $t_n \cdot \alpha$ is irreducible w.r.t. $\rightsquigarrow_{E_{\mathcal{B}}}$, then we are done: we choose u as u', v as v' and σ as σ'. If, otherwise, $t_n \cdot \alpha$ is reducible w.r.t. $\rightsquigarrow_{E_{\mathcal{B}}}$, then there is a rule $l \to r$ in $E_{\mathcal{B}}$ reducing $t_n \cdot \alpha$ w.r.t. $\curvearrowright_{\underline{\{l \to r\}}}$, and such that $l \not\equiv t_n\alpha$. We choose such a rule with l minimal w.r.t. $\rhd$.
Therefore we can apply Lemma 5.3.34 taking $t_1 \cdot \rho$ as $t_0 \cdot \alpha$, and taking $t_n \cdot \alpha$ as $t_1 \cdot \beta$ and also as $t_2 \cdot \gamma$, and we obtain an equation $u' \simeq v'$ in E such that, under some substitution σ', fulfills the same conditions required for $u \simeq v$ under the substitution σ: note that we have that $v'\sigma' \equiv t_3\delta$, where $t_n \cdot \alpha \rightsquigarrow_{E_{\mathcal{B}}} t_3 \cdot \delta$, since

$$\forall q \in red\_pos(t_n \cdot \alpha, \curvearrowright_{\underline{\{l \to r\}}}) : t_3|_q \in \mathcal{X}$$

implies

$$t_n \cdot \alpha \curvearrowright^{\parallel}_{\underline{\{l \to r\}}} t_3 \cdot \delta.$$

Then we conclude by the induction hypothesis. □

Lemma 5.3.36 *Let t be a term in $\mathcal{B}$, and $\mathcal{S}$ be a non-empty and non-overlapping subset of $E_{\mathcal{B}}$ such that each rule in $\mathcal{S}$ reduces t at some position not in $P_{\mathcal{B}^{\lhd}}(t)$. If there is an equation $u' \simeq v'$ in E such that, for some substitution σ', we have*

1. $u'\sigma' \equiv t$*,*
2. $v'\sigma' \equiv t_m\beta$*, where* $t \curvearrowright^{!}_{\underline{\mathcal{S}}} t_1 \cdot \rho \rightsquigarrow^{!}_{E_{\mathcal{B}}} t_m \cdot \beta$*, and*
3. $red\_vars(v', t_m \cdot \beta, \curvearrowright_{\underline{E_{\mathcal{B}}}}) \cap vars(u') = \emptyset$*,*

then $t_m\beta \succ t$ (and hence $t_m\beta \to t \in E_{\mathcal{B}}$).

Proof. We proceed by contradiction. Suppose that $t \succ t_m\beta$ (observe that, by Lemma 5.3.19, we have that $t \not\equiv t_m\beta$). Since $t \in \mathcal{B}$, there is a rule $e = s \to t$ in $\mathcal{R}$. W.l.o.g. we choose such an e with t as right-hand side to be minimal w.r.t. $\succ_E$. Then we have that $t \notin \mathcal{B}^e$. Moreover s is irreducible by $E_{\mathcal{B}} \setminus \{s \to t\}$ at topmost position by Lemma 5.3.13. We also have that $s \not\rhd t$, since otherwise we would have $s \to t \succ_E l \to r$ for every rule $l \to r$ in $\mathcal{S}$, contradicting Lemma 5.3.25.
Now let $s' \simeq t'$ be an equation such that, w.l.o.g., $s'\sigma' \equiv s$ and $t'\sigma' \equiv t$. Note that, whatever terms are s' and t', if t' is a variable x then x cannot occur in s': on the one hand we clearly have that $s' \not\equiv x$, since $s \succ t$; on the other hand we have that $s' \not\rhd x$, since otherwise we would have $s \rhd t$. Then the following cases have to be considered:

1. t' is not a variable. Then there is an inference by ordered paramodulation,

$$\frac{u' \simeq v' \qquad s' \simeq t'}{(s' \simeq v')\theta},$$

where $\theta = mgu(t', u')$, whose conclusion has an instance of the form $s \simeq t_m\beta$. Note that $s \simeq t_m\beta$ is not a tautology of the form $w \simeq w$, since $s \succ t \succ t_m\beta$.

Then $e' = s \to t_m\beta \in \vec{E} \setminus \mathcal{R}$: if $s \to t_m\beta \in \mathcal{R}$ then, since $\mathcal{R} \subseteq E_{\mathcal{B}}$ and $t_m\beta \not\equiv t$, s would be reducible by $E_{\mathcal{B}} \setminus \{s \to t\}$ at topmost position, contradicting Lemma 5.3.13.

We now analyze the possible reasons why e' has not generated a rule in $\mathcal{R}$. If e' failed condition 1 to generate a rule then, by Lemma 5.3.9, also would have failed e, since $e = s \to t \succ_E s \to t_m\beta = e'$. Therefore e' must fail condition 2 to generate a rule, i.e., there must be some rule $l \to r$ in $\mathcal{R}^{e'}$ reducing $t_m\beta$ at some position not in $P_{\mathcal{B}^{e'}}(t_m\beta)$. Then, by Lemma 5.3.9, $l \to r$ also reduces $t_m\beta$ at some position p not in $P_{\mathcal{B}}(t_m\beta)$. Two possibilities have to be considered:

   (a) $p \neq \lambda$. In this case, recall that $t \curvearrowright^{!}_{S} t_1 \cdot \rho \rightsquigarrow^{!}_{E_{\mathcal{B}}} t_m \cdot \beta$. Then, by Lemma 5.3.23, there is a rewrite sequence of the form $arg\_abs(t, \mathcal{R}) \curvearrowright^{*}_{E_{\mathcal{B}}} t'_m \cdot \beta'$ with $t'_m\beta' \equiv t_m\beta$. Now, since $p \notin P_{\mathcal{B}}(t_m\beta)$, we have that $p \notin P_{\mathcal{B}}(t'_m\beta')$ which, by the form of the rewrite sequence, implies that $p \notin P_{\mathcal{B}^{\lhd}}(t)$.

   Now observe that p is a non-variable position of t_m since, on the one hand, for all variable positions q of t_m we have that $t_m\beta|_q \in \mathcal{B}$ and, on the other hand, $l \notin B$ by Lemma 5.3.11 and $t_m\beta|_p \equiv l$. Moreover, $vars(t_m|_p) = \emptyset$ since, otherwise, given that $t_m\beta|_p \equiv l$, $l \to r \in E_{\mathcal{B}}$ and $p \neq \lambda$, $t_m \cdot \beta$ would be reducible w.r.t. $\rightsquigarrow_{E_{\mathcal{B}}}$, which is not the case. Therefore, $t_m\beta|_p \equiv t|_p$. Finally note that not only $p \notin P_{\mathcal{B}^{\lhd}}(t)$, but also $p \notin P_{\mathcal{B}^e}(t)$ since, as stated before, t is not in $\mathcal{B}^e$. But this contradicts $e \in \mathcal{R}$, since $l \to r$ is in R^e (since $e \succ e'$) and it reduces t at a position not in $P_{\mathcal{B}^e}(t)$.

   (b) $p = \lambda$. Again, we have two possibilities:

      i. $v'\theta$ is not a variable. Then there is an inference by ordered paramodulation,

      $$\frac{l' \simeq r' \qquad s'\theta \simeq v'\theta}{(s'\theta \simeq r')\theta'},$$

      where $l' \simeq r'$ is an equation in E such that, w.l.o.g., $l'\sigma \equiv l$ and $r'\sigma \equiv r$, where $\theta' = mgu(v'\theta, l')$, and whose conclusion has an instance of the form $s \simeq r$. Note that $s \simeq r$ is not a tautology of the form $w \simeq w$, since $s \succ t \succ t_m\beta \equiv l \succ r$. Then we have $s \to r \in E_{\mathcal{B}}$, since $s \to r \in \vec{E}$ and $r \in \mathcal{B}$. But this contradicts $s \to t \in E_{\mathcal{B}}$ by Lemma 5.3.13.

      ii. $v'\theta$ is a variable. In this case v' must be a variable x, too. Now, given that S is a non-empty subset of $E_{\mathcal{B}}$ and $t \curvearrowright^{!}_{S} t_1 \cdot \rho \rightsquigarrow^{!}_{E_{\mathcal{B}}} t_m \cdot \beta$, we have that $vars(t_m) \neq \emptyset$ and, hence, $x \in red\_vars(v', t_m \cdot \beta, \curvearrowright_{\underline{E_{\mathcal{B}}}})$.

      Then, by condition 3, we have that $x \notin vars(u')$ and, therefore, $v'\theta \equiv x$ and $x \notin vars(s'\theta)$. Then let σ'' be a substitution such that $x\sigma'' \equiv r$ and $y\sigma'' \equiv y\sigma'$ for all other variables y. We have that $(s' \simeq v')\theta\sigma'' \equiv s \simeq r \in gnd(E)$, and we conclude as in the previous case.

2. t' is a variable x. In this case, as stated before, x does not occur in s'. Then let σ'' be a substitution such that $x\sigma'' \equiv t_m\beta$ and $y\sigma'' \equiv y\sigma'$ for all other variables y. We have that $(s' \simeq t')\sigma'' \equiv s \simeq t_m\beta$.

Then we proceed analogously to case 1 except for subcase 1b, where $s \simeq r$ is obtained again from $s' \simeq t'$ but under a different substitution.

□

Lemma 5.3.37 *Let t be a term in $\mathcal{B}$, and $\mathcal{S}$ be a non-empty and non-overlapping subset of $E_{\mathcal{B}}$. If $t \curvearrowright^{!}_{\mathcal{S}} t_1 \cdot \rho \rightsquigarrow^{!}_{E_{\mathcal{B}}} t_m \cdot \beta$, then $t_m\beta \equiv t$ or $t_m\beta \to t \in E_{\mathcal{B}}$.*

Proof. Let

$$D_1 = t \curvearrowright^{!}_{\mathcal{S}} t_1 \cdot \rho \rightsquigarrow^{!}_{E_{\mathcal{B}}} t_m \cdot \beta.$$

Recall that, by Lemmas 5.3.20 and 5.3.21, this rewrite sequence is unique up to permutation of non-overlapped steps. Then, by Lemma 5.3.22, there is a rewrite sequence

$$D_1' = t \curvearrowright^{*}_{E_{\mathcal{B}}} t_{n_1} \cdot \rho_{n_1} \curvearrowright^{*}_{E_{\mathcal{B}}} t_m \cdot \beta,$$

which is equal to $D_1(\curvearrowright_{E_{\mathcal{B}}})$ up to permutation of non-overlapped steps, and such that

$$t_{n_1}\rho_{n_1} \equiv t$$

and $t_{n_1} \cdot \rho_{n_1} \curvearrowright^{*}_{E_{\mathcal{B}}} t_m \cdot \beta$ consists of the steps in $D_1(\curvearrowright_{E_{\mathcal{B}}})$ which are applied at positions not in $P_{\mathcal{B}^{\triangleleft}}(t)$.
Then $t \curvearrowright^{*}_{E_{\mathcal{B}}} t_{n_1} \cdot \rho_{n_1}$ consists of the steps in $D_1(\curvearrowright_{E_{\mathcal{B}}})$ which are applied at positions in $P_{\mathcal{B}^{\triangleleft}}(t)$.
Now, on the one hand, let $\mathcal{S}_n$ denote the subset of rules in $\mathcal{S}$ which reduce t at some position not in $P_{\mathcal{B}^{\triangleleft}}(t)$ —which can possibly also reduce t at some position in $P_{\mathcal{B}^{\triangleleft}}(t)$—. Observe that, since $t_{n_1}\rho_{n_1} \equiv t$ and $\mathcal{S}$ is non-overlapping, for all positions p in $red\_pos(t_{n_1} \cdot \rho_{n_1}, \curvearrowright_{\mathcal{S}_n})$ we have that $vars(t_{n_1}|_p) = \emptyset$.
Then the fragment

$$t_{n_1} \cdot \rho_{n_1} \curvearrowright^{*}_{E_{\mathcal{B}}} t_m \cdot \beta$$

is equivalent to a rewrite sequence of the form

$$t_{n_1} \cdot \rho_{n_1} \curvearrowright^{!}_{\mathcal{S}_n} t_{n_1'} \cdot \rho_{n_1'} \rightsquigarrow^{!}_{E_{\mathcal{B}}} t_m \cdot \beta.$$

On the other hand, let $\mathcal{S}_n'$ denote the set of rules in $E_{\mathcal{B}}$ which reduce $arg\_abs(t, \mathcal{R})$ w.r.t. $\rightsquigarrow_{E_{\mathcal{B}}}$ and whose left-hand side is reducible by $\mathcal{S}$ but not by $\mathcal{S}_n$. And let $\mathcal{S}' = \mathcal{S}_n \cup \mathcal{S}_n'$.
If $\mathcal{S}'$ is empty, then $t_{n_1} \cdot \rho_{n_1} \equiv t_m \cdot \beta$ and, since $t_{n_1}\rho_{n_1} \equiv t$, we are done. If, otherwise, $\mathcal{S}'$ is not empty, note that $\mathcal{S}'$ is a non-overlapping set of rules such that each rule in $\mathcal{S}'$ reduces t at some position not in $P_{\mathcal{B}^{\triangleleft}}(t)$. Then, by Lemmas 5.3.33, 5.3.35 and 5.3.36 we have there is a rewrite sequence

$$D_2 = t \curvearrowright^{!}_{\mathcal{S}'} t_{n_2} \cdot \rho_{n_2} \rightsquigarrow^{!}_{E_{\mathcal{B}}} t_{m_2} \cdot \rho_{m_2}$$

with $t_{m_2}\rho_{m_2} \to t \in E_{\mathcal{B}}$.
Then, by Lemma 5.3.22, there is a rewrite sequence

$$D_2' = t \curvearrowright^{*}_{E_{\mathcal{B}}} t_{n_2'} \cdot \rho_{n_2'} \curvearrowright^{*}_{E_{\mathcal{B}}} t_{m_2} \cdot \rho_{m_2},$$

which is equal to $D_2(\curvearrowright_{E_{\mathcal{B}}})$ up to permutation of non-overlapped steps, and such that

$$t_{n_2'}\rho_{n_2'} \equiv t$$

and $t_{n_2'} \cdot \rho_{n_2'} \curvearrowright^{*}_{E_{\mathcal{B}}} t_{m_2} \cdot \rho_{m_2}$ consists of the steps in $D_2(\curvearrowright_{E_{\mathcal{B}}})$ which are applied at positions not in $P_{\mathcal{B}^{\triangleleft}}(t)$.

Observe that, since $t_{n'_2}\rho_{n'_2} \equiv t$ and $\mathcal{S}'$ is non-overlapping, for all positions p in $red\_pos(t_{n'_2} \cdot \rho_{n'_2}, \curvearrowright_{\mathcal{S}'})$ we have that $vars(t_{n'_2}|_p) = \emptyset$. Then the fragment

$$t_{n'_2} \cdot \rho_{n'_2} \curvearrowright^*_{E_\mathcal{B}} t_{m_2} \cdot \rho_{m_2}$$

is equivalent to a rewrite sequence of the form

$$t_{n'_2} \cdot \rho_{n'_2} \curvearrowright^!_{\mathcal{S}_n} t_{n''_2} \cdot \rho_{n''_2} \curvearrowright^!_{\mathcal{S}'_n} t_{n'''_2} \cdot \rho_{n'''_2} \rightsquigarrow^!_{E_\mathcal{B}} t_{m_2} \cdot \rho_{m_2}.$$

Finally observe that $t_{n'_1}\rho_{n'_1} \equiv t_{n''_2}\rho_{n''_2}$, since $t_{n_1}\rho_{n_1} \equiv t_{n'_2}\rho_{n'_2} \equiv t$. Then, by definition of $\mathcal{S}'_n$, we have that the fragments

$$t_{n'_1} \cdot \rho_{n'_1} \rightsquigarrow^!_{E_\mathcal{B}} t_m \cdot \beta$$

and

$$t_{n''_2} \cdot \rho_{n''_2} \curvearrowright^!_{\mathcal{S}'_n} t_{n'''_2} \cdot \rho_{n'''_2} \rightsquigarrow^!_{E_\mathcal{B}} t_{m_2} \cdot \rho_{m_2}$$

fulfill $t_m \cdot \beta \equiv t_{m_2} \cdot \rho_{m_2}$. Then, since $t_{m_2}\rho_{m_2} \to t \in E_\mathcal{B}$, we are done. □

Now, roughly, we have to go from the Return Property with $E_\mathcal{B}$ to the Return Property with $\mathcal{R}$. To this end we show that for some particular abstracted terms, the so-called *innermost reduced terms*, innermost rewrite sequences with $\curvearrowright_{E_\mathcal{B}}$ are in fact rewrite sequences with $\curvearrowright_\mathcal{R}$ and preserve the innermost reduction property of the terms (Lemma 5.3.40). The proof of this lemma is based on Lemma 5.3.15.

Definition 5.3.38 (Innermost reduced term) *Let $s \cdot \gamma$ be an abstracted term w.r.t. $\mathcal{R}$. We say that $s \cdot \gamma$ is* innermost reduced *if there is no position p in $red\_pos(s \cdot \gamma, \curvearrowright_{E_\mathcal{B}})$ such that $vars(s|_p) = \emptyset$ (i.e., there is no redex in $s \cdot \gamma$ which has never been reduced inside).*

Lemma 5.3.39 *Let $s \cdot \gamma$ be an abstracted term w.r.t. $\mathcal{R}$ which is innermost reduced. If $s \cdot \gamma \curvearrowright_{E_\mathcal{B}} t \cdot \delta$, then $t \cdot \delta$ is innermost reduced.*

Proof. Assume that $t \cdot \delta$ is of the form $s[x]_p \cdot (\gamma \cup \{x \mapsto r\})$. Then, on the one hand, we have that $vars(s|_{p'}) \neq \emptyset$ for every prefix p' of p. And, on the other hand, for every position q which is not overlapped with p we have that $vars(s|_q) = vars(s[x]_p|_q)$, since $s|_q = s[x]_p|_q$. Therefore $t \cdot \delta$ is innermost reduced, since $s \cdot \gamma$ is innermost reduced and, by definition of $\curvearrowright_{E_\mathcal{B}}$, only positions strictly above p or positions which are not overlapped with p can belong to $red\_pos(t \cdot \delta, \curvearrowright_{E_\mathcal{B}})$. □

Lemma 5.3.40 *Let $s \cdot \gamma$ be an abstracted term w.r.t. $\mathcal{R}$ which is innermost reduced, and let D be an innermost rewrite sequence of the form $s \cdot \gamma \curvearrowright^+_{E_\mathcal{B}} t \cdot \delta$ by non-abstraction steps. Then D is a rewrite sequence with $\curvearrowright_\mathcal{R}$ and $t \cdot \delta$ is innermost reduced.*

Proof. We proceed by induction on the length of D. We have that D is of the form $s \cdot \gamma \curvearrowright_{E_\mathcal{B}} s[x]_p \cdot (\gamma \cup \{x \mapsto r\}) \curvearrowright^*_{E_\mathcal{B}} t \cdot \delta$, where $s\gamma|_p \to r \in E_\mathcal{B}$. Since the rewrite sequence is innermost, $s\gamma|_p$ can be reducible by $E_\mathcal{B}$ at most at positions below $P_\mathcal{X}(s|_p)$, apart from λ. Now note that positions in $P_\mathcal{X}(s|_p)$ are $\mathcal{B}$-positions of $s\gamma|_p$, since $s \cdot \gamma$ is an abstracted term w.r.t. $\mathcal{R}$ and $\mathcal{B} = Right(\mathcal{R})$. So we have that $s\gamma|_p$ is irreducible by $E_\mathcal{B}$ at every non-topmost non-$\mathcal{B}$-position, and therefore $s\gamma|_p \to r \in \mathcal{R}$ by Lemma 5.3.15. And by Lemma 5.3.39 we have that $s[x]_p \cdot (\gamma \cup \{x \mapsto r\})$ is innermost reduced. Now, if $s[x]_p \cdot (\gamma \cup \{x \mapsto r\}) \equiv t \cdot \delta$ then we are done. Otherwise, we have $s[x]_p \cdot (\gamma \cup \{x \mapsto r\}) \curvearrowright^+_{E_\mathcal{B}} t \cdot \delta$ and we conclude by the induction hypothesis, since $s[x]_p \cdot (\gamma \cup \{x \mapsto r\})$ is an abstracted term w.r.t. $\mathcal{R}$ which is innermost reduced, and $s[x]_p \cdot (\gamma \cup \{x \mapsto r\}) \curvearrowright^+_{E_\mathcal{B}} t \cdot \delta$ is an innermost rewrite sequence by non-abstraction steps whose length is smaller than the length of D. □

Now we are ready to obtain the Return Property of $\mathcal{B}$-terms for rewrite sequences using $\mathcal{R}$, by combining Lemmas 5.3.23 and 5.3.37: by carefully choosing the subset $\mathcal{S}$ of $E_{\mathcal{B}}$ (and, in fact, of $\mathcal{R}$) for these two lemmas, we obtain an abstracted term $t_1 \cdot \rho$ which is innermost reduced, and then we can show, by Lemma 5.3.40, that we have a return rewrite sequence using only $\mathcal{R}$. Finally, we show that some steps of the resulting sequence can be removed, in order to obtain a sequence with $\mathcal{R}$ from $arg\_abs(t,\mathcal{R})$ (with the required initial step) to an abstraction of t again.

Theorem 5.3.41 (Return Property) *Let t be a term in $\mathcal{B}$. If $arg\_abs(t,\mathcal{R}) \curvearrowright_{\mathcal{R}} u\cdot\gamma$, then $u\cdot\gamma \curvearrowright^{!}_{\mathcal{R}} v\cdot\rho$ with $v\rho \equiv t$.*

Proof. Let $t'\cdot\delta$ denote $arg\_abs(t,\mathcal{R})$. If the step

$$arg\_abs(t,\mathcal{R}) \curvearrowright_{\mathcal{R}} u\cdot\gamma$$

is an abstraction step (see Definition 5.2.2) then $u\cdot\gamma$ is of the form $t'[x]_p \cdot (\delta \cup \{x \mapsto t'\delta|_p\})$ for some position p not in $P_{\mathcal{B}^{\triangleleft}}(t)$, and $t'\delta|_p \in Right(\mathcal{R})$. Now observe that $t'\delta \equiv t$. Then $t|_p \in Right(\mathcal{R})$. But, since $Right(\mathcal{R}) = \mathcal{B}$ (see Definition 5.3.5) and moreover $t \in \mathcal{B}$ and p not in $P_{\mathcal{B}^{\triangleleft}}(t)$, we must have that $p = \lambda$. So in this case $u\cdot\gamma$ is of the form $x\cdot\{x \mapsto t\}$, and hence we are done.
If, otherwise, we have a non-abstraction step, then $u\cdot\gamma$ is of the form $t'[x]_p \cdot (\delta \cup \{x \mapsto r\})$ for some position p not in $P_{\mathcal{B}^{\triangleleft}}(t)$, $t'\delta|_p \equiv l$ and $l \to r \in \mathcal{R}$. In this case, let

$$\mathcal{S}_n = \{l' \to r' \in E_{\mathcal{B}} \mid t \rhd l'\},$$

and let

$$\mathcal{S} = \{l' \to r' \in \mathcal{S}_n \mid l' \text{ is minimal w.r.t. } \rhd \text{ in } Left(\mathcal{S}_n)\}.$$

We have that $\mathcal{S}$ is non-overlapping since, by Lemma 5.3.13, for every rule $l' \to r'$ in $E_{\mathcal{B}}$ we have that l' is irreducible by $E_{\mathcal{B}} \setminus \{l' \to r'\}$ at topmost position. Moreover, $\mathcal{S} \subseteq \mathcal{R}$ by Lemma 5.3.15. And $\mathcal{S}$ is not empty, since $l \to r \in \mathcal{S}_n$ (note that $t \rhd l$ by Lemma 5.3.11).
Now let

$$D = t \curvearrowright^{!}_{\mathcal{S}} t_1 \cdot \rho \rightsquigarrow^{!}_{E_{\mathcal{B}}} t_m \cdot \beta.$$

Recall that, by Lemmas 5.3.20 and 5.3.21, this rewrite sequence is unique up to permutation of non-overlapped steps. We divide D in the two subsequences

$$D_1 = t \curvearrowright^{!}_{\mathcal{S}} t_1 \cdot \rho$$

and

$$D_2 = t_1 \cdot \rho \rightsquigarrow^{!}_{E_{\mathcal{B}}} t_m \cdot \beta.$$

We have that $D(\curvearrowright_{E_{\mathcal{B}}})$ is a rewrite sequence with $\curvearrowright_{\mathcal{R}}$ and its last term is innermost reduced: since $\mathcal{S} \subseteq \mathcal{R}$, then D_1 is a rewrite sequence with $\curvearrowright_{\mathcal{R}}$ and $t_1 \cdot \rho$ is an abstracted term w.r.t. $\mathcal{R}$; moreover, $t_1 \cdot \rho$ is innermost reduced (see Definition 5.3.38) by definition of $\mathcal{S}$ and, hence, by Lemma 5.3.40, $D_2(\curvearrowright_{E_{\mathcal{B}}})$ is also a rewrite sequence with $\curvearrowright_{\mathcal{R}}$ and its last term is innermost reduced.
Moreover, by Lemma 5.3.37, we have that $t_m\beta \equiv t$ or $t_m\beta \to t \in E_{\mathcal{B}}$. And, by Lemma 5.3.23, there is a rewrite sequence

$$D' = arg\_abs(t,\mathcal{R}) \curvearrowright^{*}_{E_{\mathcal{B}}} t'_m \cdot \beta',$$

with $t'_m\beta' \equiv t_m\beta$, consisting of the steps in $D(\curvearrowright_{E_{\mathcal{B}}})$ which are applied at positions not in $P_{\mathcal{B}^{\triangleleft}}(t)$. Then D' is also a rewrite sequence with $\curvearrowright_{\mathcal{R}}$.

Now, since $t_m \cdot \beta$ is innermost reduced and moreover it is irreducible w.r.t. $\rightsquigarrow_{E_{\mathcal{B}}}$, if $t_m\beta \to t \in E_{\mathcal{B}}$ then the rewrite sequence

$$t_m \cdot \beta \curvearrowright_{\{t_m\beta \to t\}} x \cdot \{x \mapsto t\}$$

is a rewrite sequence with $\curvearrowright_{\mathcal{R}}$ by Lemma 5.3.40, and hence $t_m\beta \to t \in \mathcal{R}$.
Finally let

$$D'' = \begin{cases} D' & \text{if } t_m\beta \equiv t \text{, and} \\ D' \curvearrowright_{\{t_m\beta \to t\}} x \cdot \{x \mapsto t\} & \text{otherwise.} \end{cases}$$

Note that in either case the last term in D'' is irreducible w.r.t. $\curvearrowright_{\mathcal{R}}$. This is clearly the case for $x \cdot \{x \mapsto t\}$ and, for the case that $t_m\beta \equiv t$, recall that $t_m \cdot \beta$ is irreducible w.r.t. $\rightsquigarrow_{E_{\mathcal{B}}}$, and hence it might only be reduced at topmost position, but this is not possible by Lemma 5.3.11.
To conclude, we show that (i) a non-abstraction step with $l \to r$ at position p is present in D'', and (ii) D'' can be rearranged to make this be the first step. Observe that $D(\curvearrowright_{E_{\mathcal{B}}})$ does not have abstraction steps, since

- D_1 is a rewrite sequence with $\curvearrowright_{\overline{\mathcal{R}}}$, and
- $D_2(\curvearrowright_{E_{\mathcal{B}}})$ is a rewrite sequence with $\curvearrowright_{\underline{\mathcal{R}}}$.

Therefore neither D' nor D'' have abstraction steps. Now recall that D' consists of the steps in $D(\curvearrowright_{E_{\mathcal{B}}})$ which are applied at positions not in $P_{\mathcal{B}^{\lhd}}(t)$. Two cases have to be considered.

- $l \to r \in \mathcal{S}$. Then a non-abstraction step with $l \to r$ at position p is clearly present in D' and hence in D''.

- $l \to r \notin \mathcal{S}$. Then l must be reducible by some rule in $\mathcal{S}$ at non-topmost position. But, since $l \to r \in \mathcal{R}$, then by Lemma 5.3.15 we have that rules in $\mathcal{S}$ can only reduce l at $\mathcal{B}$-positions. Now recall that $t \rhd l$. So, on the one hand, for every $\mathcal{B}$-position p' of l which is reducible by $\mathcal{S}$ we have that $p \cdot p'$ is in $P_{\mathcal{B}^{\lhd}}(t)$. On the other hand, by Lemma 5.3.22, we have that there is a rewrite sequence

  $$D''' = t \curvearrowright^*_{E_{\mathcal{B}}} t_n \cdot \alpha \curvearrowright^*_{E_{\mathcal{B}}} t_m \cdot \beta,$$

  which is equal to $D(\curvearrowright_{E_{\mathcal{B}}})$ up to permutation of non-overlapped steps, and such that $t_n\alpha \equiv t$ and $t_n \cdot \alpha \curvearrowright^*_{E_{\mathcal{B}}} t_m \cdot \beta$ consists of the steps in $D(\curvearrowright_{E_{\mathcal{B}}})$ which are applied at positions not in $P_{\mathcal{B}^{\lhd}}(t)$. Therefore we have that p is a valid position of t_n and $vars(t_n|_p) \neq \emptyset$, since p is a position not in $P_{\mathcal{B}^{\lhd}}(t)$ and $t|_p$ is reducible at $\mathcal{B}$-positions by $\mathcal{S}$. So $l \to r$ reduces $t_n \cdot \alpha$ at position p w.r.t. $\curvearrowright_{\underline{E_{\mathcal{B}}}}$.

  Therefore, in the innermost rewrite sequence with $\curvearrowright_{E_{\mathcal{B}}}$ from $t_n \cdot \alpha$ to $t_m \cdot \beta$ there must be a step with $l \to r$ at position p, since l is irreducible by $E_{\mathcal{B}}$ at any non-$\mathcal{B}$-position apart from λ by Lemma 5.3.15, and there are no two rules with the same left-hand side in $E_{\mathcal{B}}$ by Lemma 5.3.13. So a step with $l \to r$ occurs in $D_2(\curvearrowright_{E_{\mathcal{B}}})$ at position p, and hence also in D' and D''.

Moreover, D'' can be rearranged to make this be the first step, since $t|_p \equiv l$ and D'' is innermost. This gives us a rewrite sequence of the form $arg\_abs(t, \mathcal{R}) \curvearrowright_{\mathcal{R}} u \cdot \gamma \curvearrowright^!_{\mathcal{R}} v \cdot \rho$ with $v\rho \equiv t$, and hence we are done. □

5.3.3 Rewrite proofs for the equational case

In order to prove that we have $\mathcal{B}$-rewrite proofs for all consequences of the closed set of equations E, we first need to prove the confluence of the rewrite relation $\curvearrowright^{+}_{\mathcal{R}}$ on abstracted terms.

The Return Property is the cornerstone for proving local confluence. By construction of $\mathcal{R}$,

- left-hand sides of $\mathcal{R}$ can only overlap under terms that correspond to right-hand sides of $\mathcal{R}$ (Lemma 5.3.15), and
- right-hand sides of $\mathcal{R}$ fulfill the Return Property (Theorem 5.3.41).

Hence, using the Return Property we can show that, if we reduce an abstraction of a left-hand side of a rule in $\mathcal{R}$ (not at top position), then there is a $\mathcal{B}$-rewrite sequence going back to (another abstraction of) the left-hand side again. So the diamond closes with a final step with the same rule.

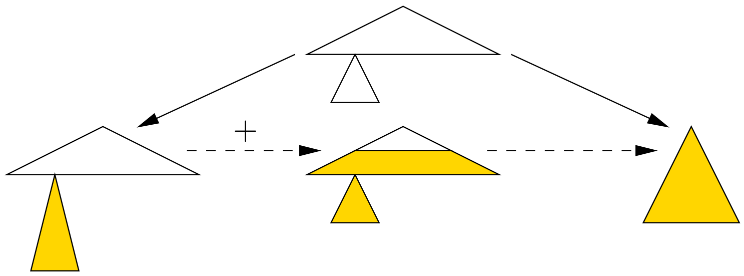


Figure 5.4: Graphical proof of local confluence.

Lemma 5.3.42 *Let $s \cdot \gamma$ be an abstracted term w.r.t. $\mathcal{R}$ such that $s\gamma \in \mathcal{B}$. If there is a position $p \neq \lambda$ such that $s|_p\gamma \equiv l$ for some rule $l \to r$ in $\mathcal{R}$, then $s[x]_p \cdot (\gamma \cup \{x \mapsto r\}) \curvearrowright^{+}_{\mathcal{R}} s' \cdot \gamma'$ for some $s' \cdot \gamma'$ such that $s'\gamma' \equiv s\gamma$.*

Proof. Let q be the deepest position of s such that $s|_q\gamma$ is a $\mathcal{B}$-term and q is a prefix of p, that is, $q \cdot p' = p$ for some p'. Note that $p' \neq \lambda$ since, by Lemma 5.3.11, left-hand sides of rules in $\mathcal{R}$ are not $\mathcal{B}$-terms. Then, by some abstraction steps we can get $arg\_abs(s|_q\gamma, \mathcal{R})$ from $s|_q \cdot \gamma$, and $arg\_abs(s|_q\gamma, \mathcal{R})$ is still reducible, i.e.,

$$s|_q \cdot \gamma \curvearrowright^{*}_{\mathcal{R}} arg\_abs(s|_q\gamma, \mathcal{R}) \equiv t \cdot \delta \curvearrowright_{\mathcal{R}} t[x]_{p'} \cdot (\delta \cup \{x \mapsto r\}).$$

Then, since the steps clearly commute, we also have that

$$s|_q \cdot \gamma \curvearrowright_{\mathcal{R}} s|_q[x]_{p'} \cdot (\gamma \cup \{x \mapsto r\}) \curvearrowright^{*}_{\mathcal{R}} t[x]_{p'} \cdot (\delta \cup \{x \mapsto r\}).$$

Now, by Theorem 5.3.41, we have that

$$t[x]_{p'} \cdot (\delta \cup \{x \mapsto r\}) \curvearrowright^{!}_{\mathcal{R}} v \cdot \rho$$

with $v\rho \equiv s|_q\gamma$, which implies

$$s[x]_p \cdot (\gamma \cup \{x \mapsto r\}) \curvearrowright^{+}_{\mathcal{R}} s[v]_q \cdot (\gamma \cup \rho)$$

and $s[v]_q(\gamma \cup \rho) \equiv s\gamma$. Hence, with $s' = s[v]_q$ and $\gamma' = (\gamma \cup \rho)$, the lemma holds. $\square$

Lemma 5.3.43 *Let $s \cdot \gamma$ be an abstracted term w.r.t. $\mathcal{R}$ such that $s\gamma \in Left(\mathcal{R})$. If there is a position $p \neq \lambda$ such that $s|_p\gamma \equiv l$ for some rule $l \to r$ in $\mathcal{R}$, then $s[x]_p \cdot (\gamma \cup \{x \mapsto r\}) \curvearrowright^+_{\mathcal{R}} s' \cdot \gamma'$, for some $s' \cdot \gamma'$ such that s' is not a variable and $s'\gamma' \equiv s\gamma$.*

Proof. Since $s\gamma$ is in $Left(\mathcal{R})$, by Lemma 5.3.15, $s\gamma$ is irreducible at non-$\mathcal{B}$-positions different from λ. Therefore, since $p \neq \lambda$, there is a position q with $q \cdot p' = p$ for some p', such that $s|_q\gamma$ is a $\mathcal{B}$-term. Note that $p' \neq \lambda$ since, by Lemma 5.3.11, left-hand sides of rules in $\mathcal{R}$ are not $\mathcal{B}$-terms. Then, by Lemma 5.3.42, we have that

$$s|_q[x]_{p'} \cdot (\gamma \cup \{x \mapsto r\}) \curvearrowright^+_{\mathcal{R}} v \cdot \rho$$

for some $v \cdot \rho$ such that $v\rho \equiv s|_q\gamma$, which implies

$$s[x]_p \cdot (\gamma \cup \{x \mapsto r\}) \curvearrowright^+_{\mathcal{R}} s[v]_q \cdot (\gamma \cup \rho)$$

and $s[v]_q(\gamma \cup \rho) \equiv s\gamma$. Finally, observe that $q \neq \lambda$, since $s\gamma \in Left(\mathcal{R})$ and $s|_q\gamma \in \mathcal{B}$. Hence, $s[v]_q$ is not a variable and, with $s' = s[v]_q$ and $\gamma' = (\gamma \cup \rho)$, the lemma holds. □

From the two previous lemmas, we know that we can close the only potential "peaks" that we may have with $\curvearrowright_{\mathcal{R}}$. Therefore, local confluence follows.

Lemma 5.3.44 (Local confluence) *Let $s \cdot \gamma$ be an abstracted term. If $t \cdot \gamma_1 \curvearrowleft_{\mathcal{R}} s \cdot \gamma \curvearrowright_{\mathcal{R}} u \cdot \gamma_2$ then $t \cdot \gamma_1 \curvearrowright^*_{\mathcal{R}} v \cdot \delta \curvearrowleft^*_{\mathcal{R}} u \cdot \gamma_2$, for some abstracted term $v \cdot \delta$.*

Proof. If $t \cdot \gamma_1 \curvearrowleft_{\mathcal{R}} s \cdot \gamma \curvearrowright_{\mathcal{R}} u \cdot \gamma_2$, we have that $t \equiv s[x]_{p_1}$ for some fresh variable x and position p_1, and $\gamma_1 = \gamma \cup \{x \mapsto r_1\}$, where $s\gamma|_{p_1} \to r_1 \in \mathcal{R}$ or $s\gamma|_{p_1} \equiv r_1 \in \mathcal{B}$; and $u \equiv s[y]_{p_2}$ for some fresh variable y and position p_2, and $\gamma_2 = \gamma \cup \{y \mapsto r_2\}$, where $s\gamma|_{p_2} \to r_2 \in \mathcal{R}$ or $s\gamma|_{p_2} \equiv r_2 \in \mathcal{B}$. The following cases have to be considered:

1. There is no overlapping between p_1 and p_2. In this case, we can easily close the diamond by applying either $s\gamma|_{p_2} \to r_2$ or an abstraction step at position p_2 in $t \cdot \gamma_1$, and either $s\gamma|_{p_1} \to r_1$ or an abstraction step at position p_1 in $u \cdot \gamma_2$, obtaining $v = s[x]_{p_1}[y]_{p_2}$ and $\delta = \gamma \cup \{x \mapsto r_1\} \cup \{y \mapsto r_2\}$.

2. There is an overlapping between p_1 and p_2. In this case, assume w.l.o.g. that p_2 is below p_1, and hence $p_2 = p_1 \cdot p_3$ for some p_3. Since, by Lemmas 5.3.11 and 5.3.13, left-hand sides of rules in $\mathcal{R}$ are not $\mathcal{B}$-terms and moreover they are unique, we assume that $p_3 \neq \lambda$ (otherwise we have either two abstraction steps which are the same or two steps with the same rule at the same position, and hence the diamond closes trivially).

   Then, since the relation $\curvearrowright_{\mathcal{R}}$ is monotonic, it is enough to show that we can close the diamond starting from $s|_{p_1} \cdot \gamma$, i.e., we have to show that there is some abstracted term $v' \cdot \delta'$ such that $x \cdot \{x \mapsto r_1\} \curvearrowright^*_{\mathcal{R}} v' \cdot \delta' \curvearrowleft^*_{\mathcal{R}} s|_{p_1}[y]_{p_3} \cdot (\gamma \cup \{y \mapsto r_2\})$. The following cases have to be considered:

   (a) Both steps are abstraction steps, i.e., $s\gamma|_{p_1} \equiv r_1 \in \mathcal{B}$ and $s\gamma|_{p_2} \equiv r_2 \in \mathcal{B}$. Then we can apply an abstraction step at topmost position in $s|_{p_1}[y]_{p_3} \cdot (\gamma \cup \{y \mapsto r_2\})$ obtaining $x \cdot \{x \mapsto r_1\}$; hence, taking $v' = x$ and $\delta' = \{x \mapsto r_1\}$ it holds.

   (b) $s\gamma|_{p_1} \to r_1 \in \mathcal{R}$ and $s\gamma|_{p_2} \equiv r_2 \in \mathcal{B}$. In this case, we can apply $s\gamma|_{p_1} \to r_1 \in \mathcal{R}$ on $s|_{p_1}[y]_{p_3} \cdot (\gamma \cup \{y \mapsto r_2\})$, and we conclude as in the previous case.

(c) $s\gamma|_{p_1} \equiv r_1 \in \mathcal{B}$ and $s\gamma|_{p_2} \to r_2 \in \mathcal{R}$. Then, by Lemma 5.3.42, we have that

$$s|_{p_1}[y]_{p_3} \cdot (\gamma \cup \{y \mapsto r_2\}) \curvearrowright^{+}_{\mathcal{R}} s' \cdot \gamma',$$

for some $s' \cdot \gamma'$ such that $s'\gamma' \equiv s\gamma|_{p_1}$. Therefore, either $s' \cdot \gamma' \equiv x \cdot \{x \mapsto r_1\}$ or we can apply an abstraction step at topmost position in $s' \cdot \gamma'$, and we conclude analogously to case 2a.

(d) $s\gamma|_{p_1} \to r_1 \in \mathcal{R}$ and $s\gamma|_{p_2} \to r_2 \in \mathcal{R}$. Then, by Lemma 5.3.43, we have that

$$s|_{p_1}[y]_{p_3} \cdot (\gamma \cup \{y \mapsto r_2\}) \curvearrowright^{+}_{\mathcal{R}} s' \cdot \gamma',$$

for some $s' \cdot \gamma'$ such that s' is not a variable and $s'\gamma' \equiv s\gamma|_{p_1}$. Therefore, applying finally $s\gamma|_{p_1} \to r_1$ on $s' \cdot \gamma'$ we have that

$$x \cdot \{x \mapsto r_1\} \curvearrowleft^{*}_{\mathcal{R}} s|_{p_1}[y]_{p_3} \cdot (\gamma \cup \{y \mapsto r_2\}),$$

which implies that taking $v' = x$ and $\delta' = \{x \mapsto r_1\}$ it holds.

□

Since $\curvearrowright_{\mathcal{R}}$ is terminating and locally confluent, it is confluent.

Theorem 5.3.45 (Confluence) *$\curvearrowright_{\mathcal{R}}$ is confluent.*

Now we prove that all equations in E have a $\mathcal{B}$-rewrite proof using $\mathcal{R}$, which also implies that $\mathcal{R}^*$ is a model for E: first, we show that there is a $\mathcal{B}$-rewrite proof for all instances which are *irreducible* (at non-$\mathcal{B}$-positions) in the substitution part; second, by first normalizing the substitution w.r.t. $\curvearrowright_{\mathcal{R}}$, which gives us an irreducible instance, we show that there are $\mathcal{B}$-rewrite proofs for all instances.

This result, together with confluence of $\curvearrowright_{\mathcal{R}}$, gives us that all consequences of E have a $\mathcal{B}$-rewrite proof using $\mathcal{R}$.

Definition 5.3.46 (Irreducible substitution) *A substitution σ is* irreducible by $\mathcal{R}$ at non-$\mathcal{B}$-positions *in a term t if, for all variables $x \in Dom(\sigma)$, either*

1. *$x\sigma$ is irreducible by $\mathcal{R}$ at non-$\mathcal{B}$-positions, or*
2. *x occurs in t only at $\mathcal{B}$-positions of $t\sigma$.*

A substitution σ is irreducible by $\mathcal{R}$ at non-$\mathcal{B}$-positions *in an equation $s \simeq t$ if σ is irreducible by $\mathcal{R}$ at non-$\mathcal{B}$-positions in s and t.*

This irreducibility property is roughly inherited by the conclusion in paramodulation inferences with rules in $\mathcal{R}$.

Here we show that there is a $\mathcal{B}$-rewrite proof for all instances which are irreducible (at non-$\mathcal{B}$-positions) in the substitution part.

Lemma 5.3.47 *Let $s \simeq t$ be an equation in E and σ be a substitution such that $(s \simeq t)\sigma$ is ground and σ is irreducible at non-$\mathcal{B}$-positions in $s \simeq t$. If $u_1 \cdot \gamma_1$ is an abstraction of $s\sigma$ w.r.t. $\mathcal{R}$ and $u_2 \cdot \gamma_2$ is an abstraction of $t\sigma$ w.r.t. $\mathcal{R}$, then $u_1 \cdot \gamma_1 \curvearrowright^{*}_{\mathcal{R}} u \cdot \gamma \curvearrowleft^{*}_{\mathcal{R}} u_2 \cdot \gamma_2$ for some abstracted term $u \cdot \gamma$.*

Proof. We proceed by induction on $|u_1|_v + |u_2|_v$, where $|t|_v$ denotes the size of a term t without counting the variables.

1. If either $u_1\gamma_1$ or $u_2\gamma_2$ has a $\mathcal{B}$-term at a non-variable position in u_1 or u_2 (respectively) then we abstract in this position. We show it for u_1 (for u_2 it is analogous). Let p be a non-variable position of u_1 such that $u_1|_p\gamma_1 \in Right(\mathcal{R})$. Then we have that $u_1 \cdot \gamma_1 \curvearrowright_{\mathcal{R}} u_1[x]_p \cdot \gamma_1'$, where $\gamma_1' = (\gamma_1 \cup \{x \mapsto u_1|_p\gamma_1\})$, and, by the induction hypothesis, it holds that $u_1[x]_p \cdot \gamma_1' \curvearrowright^*_{\mathcal{R}} u \cdot \gamma \curvearrowleft^*_{\mathcal{R}} u_2 \cdot \gamma_2$ for some abstracted term $u \cdot \gamma$.

2. If neither $u_1\gamma_1$ nor $u_2\gamma_2$ has a $\mathcal{B}$-term at a non-variable position of u_1 or u_2 (respectively), then there are two possibilities. If $u_1 \cdot \gamma_1 \equiv u_2 \cdot \gamma_2$, then we are done. Otherwise, since $u_1 \cdot \gamma_1$ is a top abstraction of $s\sigma$ (i.e., it cannot be further abstracted) and $u_2 \cdot \gamma_2$ is a top abstraction of $t\sigma$, and they are not equivalent, we have that $s\sigma \not\equiv t\sigma$. Assume that $s\sigma \succ t\sigma$.

   If $s\sigma \to t\sigma$ is in $\mathcal{R}$, then $t\sigma \in \mathcal{B}$, and hence u_2 is a variable x and γ_2 is $\{x \mapsto t\sigma\}$, and $u_1 \cdot \gamma_1 \curvearrowright_{\mathcal{R}} x \cdot \{x \mapsto t\sigma\}$. Therefore, taking $u = x$ and $\gamma = \gamma_2$ we are done.

   Otherwise, if $s\sigma \to t\sigma$ is not in $\mathcal{R}$, then, by Lemma 5.3.9 and the definition of $\mathcal{R}$, either $s\sigma$ or $t\sigma$ should be reducible by $\mathcal{R}$ at positions not in $P_{\mathcal{B}}(t)$, and hence at a non-variable position of u_1 or u_2 (respectively). We consider the case in which $s\sigma$ is reducible (the case for $t\sigma$ is analogous). Let p be a non-variable position of u_1 such that there is a rule $l \to r$ in $\mathcal{R}$ with $u_1|_p\gamma_1 \equiv l$. Then, we have that $u_1 \cdot \gamma_1 \curvearrowright_{\mathcal{R}} u_1' \cdot \gamma_1'$ where $u_1' = u_1[x]_p$ and $\gamma_1' = \gamma_1 \cup \{x \mapsto r\}$. On the other hand, since σ is irreducible at non-$\mathcal{B}$-positions in $s \simeq t$, p is a non-variable position of s and if $l' \simeq r'$ is an equation of E such that $l'\sigma' \equiv l$ and $r'\sigma' \equiv r$ for some substitution σ', then we have an inference by ordered paramodulation,

   $$\frac{l' \simeq r' \qquad s \simeq t}{(s[r']_p \simeq t)\theta},$$

   where $\theta = mgu(s|_p, l')$ and $\sigma'' = \sigma \cup \sigma'$ is a substitution such that $(s[r']_p \simeq t)\theta\sigma''$ is ground and σ'' is irreducible at non-$\mathcal{B}$-positions in $(s[r']_p \simeq t)\theta$, since $r'\theta\sigma'' \equiv r \in \mathcal{B}$ and σ is irreducible at non-$\mathcal{B}$-positions in $s \simeq t$. Moreover, since $s[r']_p\theta\sigma'' \equiv s\sigma[r]_p$ and $t\theta\sigma'' \equiv t\sigma$, we have that $u_1' \cdot \gamma_1'$ is an abstraction of $s[r']_p\theta\sigma''$ and $u_2 \cdot \gamma_2$ is an abstraction of $t\theta\sigma''$. Finally, either $u_1' \cdot \gamma_1' \equiv u_2 \cdot \gamma_2$ and hence we are done, or else the conclusion of the inference is not a tautology and hence is in E and, since σ'' is irreducible at non-$\mathcal{B}$-positions in the conclusion and $|u_1|_v > |u_1'|_v$ then, by the induction hypothesis, we have that $u_1' \cdot \gamma_1' \curvearrowright^*_{\mathcal{R}} u \cdot \gamma \curvearrowleft^*_{\mathcal{R}} u_2 \cdot \gamma_2$ for some abstracted term $u \cdot \gamma$. □

Now, as said before, we show that we can obtain a $\mathcal{B}$-rewrite proof using $\mathcal{R}$ for every instance of an equation in E, by first normalizing the substitution w.r.t. $\curvearrowright_{\mathcal{R}}$.

Lemma 5.3.48 *Let $t \cdot \gamma$ be an abstracted term w.r.t. $\mathcal{R}$. If $t \cdot \gamma$ is irreducible by $\curvearrowright_{\mathcal{R}}$, then $t\gamma$ is irreducible by $\mathcal{R}$ at non-$\mathcal{B}$-positions.*

Proof. It holds trivially since the range of γ only includes right-hand side terms of $\mathcal{R}$, i.e., $\mathcal{B}$-terms, and hence since $t \cdot \gamma$ is irreducible, $t\gamma$ can only be reducible in γ, i.e., in a $\mathcal{B}$-position. □

Lemma 5.3.49 *Let $s \simeq t$ be an equation in $gnd(E)$. Then $s \curvearrowright^*_{\mathcal{R}} u \cdot \gamma \curvearrowleft^*_{\mathcal{R}} t$ for some abstracted term $u \cdot \gamma$.*

Proof. Let $s \simeq t$ be an instance of an equation $s' \simeq t'$ in E, such that $s \equiv s'\sigma$ and $t \equiv t'\sigma$ for some substitution σ. First of all we normalize σ, i.e., we define a new substitution σ' such that $x\sigma' \equiv w\gamma_w$ if $x\sigma \curvearrowright^!_{\mathcal{R}} w \cdot \gamma_w$. Then there is an equation in $gnd(E)$ of the form $s'\sigma' \simeq t'\sigma'$, where σ' is irreducible at

non-$\mathcal{B}$-positions in $s' \simeq t'$ by Lemma 5.3.48. Moreover, we have that $s \curvearrowright^*_{\mathcal{R}} u_1 \cdot \gamma_1$ and $t \curvearrowright^*_{\mathcal{R}} u_2 \cdot \gamma_2$ for some $u_1 \cdot \gamma_1$ and $u_2 \cdot \gamma_2$ abstracted terms w.r.t. $\mathcal{R}$ of $s'\sigma'$ and $t'\sigma'$ respectively (these abstracted terms are easily obtained from the normalization of the substitution, i.e., abstracting each $w\gamma_w$).
Finally, by Lemma 5.3.47, we have that $u_1 \cdot \gamma_1 \curvearrowright^*_{\mathcal{R}} u \cdot \gamma \curvearrowleft^*_{\mathcal{R}} u_2 \cdot \gamma_2$, which allows us to conclude by composing the rewriting steps. □

The following theorem follows directly from the previous lemma. Note that rewriting with $\curvearrowright_{\mathcal{R}}$ provides also a rewrite sequence with $\mathcal{R}$.

Theorem 5.3.50 $\mathcal{R}^* \models E$.

Now, together with confluence of $\curvearrowright_{\mathcal{R}}$, we can show that all consequences of E have a $\mathcal{B}$-rewrite proof.

Lemma 5.3.51 *If $E \models s \simeq t$ for some ground terms s and t, then $s \curvearrowright^*_{\mathcal{R}} u \cdot \gamma \curvearrowleft^*_{\mathcal{R}} t$ for some abstracted term $u \cdot \gamma$.*

Proof. Since $E \models s \simeq t$, we can go from s to t using the equations of $gnd(E)$. Then there is a sequence of terms $s_0, s_1 \ldots, s_n$ such that $s_0 = s$ and $s_n = t$, and for every i in $0 \ldots n-1$ we have $s_i = C[e_i]$ and $s_{i+1} = C[e_{i+1}]$ for some context C and $e_i \simeq e_{i+1} \in gnd(E)$. By Lemma 5.3.49 and monotonicity of $\curvearrowright_{\mathcal{R}}$, for every i in $0 \ldots n-1$ we have $s_i \curvearrowright^*_{\mathcal{R}} u_i \cdot \gamma_i \curvearrowleft^*_{\mathcal{R}} s_{i+1}$ for some abstracted term $u_i \cdot \gamma_i$. Finally, by confluence of $\curvearrowright_{\mathcal{R}}$, we have that $s = s_0 \curvearrowright^*_{\mathcal{R}} u \cdot \gamma \curvearrowleft^*_{\mathcal{R}} s_{n+1} = t$ for some abstracted term $u \cdot \gamma$. □

This result implies that we have produced a Knuth-Bendix completion procedure for $\mathcal{B}$-rewriting, i.e., for ordered rewriting with well-founded orderings.

5.4 The Horn case

In this section we generalize the results of the previous section to Horn clauses. As in Section 3.4, in the following inference system it is assumed that in each clause with a non-empty antecedent one of the negative equations, the one that is written underlined, has been *selected*.

Definition 5.4.1 *The inference system $\mathcal{H}'$ for Horn clauses with respect to the well-founded ordering $\succ$ is defined as follows:*

paramodulation right:

$$\frac{\rightarrow l \simeq r \qquad \rightarrow s \simeq t}{\rightarrow (s[r]_p \simeq t)\sigma} \qquad \textit{where } \sigma = mgu(l, s|_p)$$

paramodulation left:

$$\frac{\rightarrow l \simeq r \qquad \Gamma, \underline{s \simeq t} \rightarrow \Delta}{(\Gamma, s[r]_p \simeq t \rightarrow \Delta)\sigma} \qquad \textit{where } \sigma = mgu(l, s|_p)$$

equality resolution:

$$\frac{\Gamma, \underline{s \simeq t} \rightarrow \Delta}{(\Gamma \rightarrow \Delta)\sigma} \qquad \textit{where } \sigma = mgu(s, t)$$

where, moreover, in both paramodulation rules l is maximal in its premise (that is, for some ground substitution θ, it holds that $l\sigma\theta \succ r\sigma\theta$), and $s|_p$ is not a variable except in the paramodulation right inference when $s = x \in vars(t)$, $p = \lambda$ and l is also a variable (as it happens in the equational case).

Definition 5.4.2 *Let S be a set of Horn clauses. By E_S we denote the set of equations occurring as positive unit clauses in S, that is, $E_S = \{s \simeq t \mid \to s \simeq t \in S\}$, and by $\mathcal{R}_S$ we denote the set of rules generated by E_S as in Definition 5.3.5.*

First it is shown that, if S does not contain the empty clause, then $\mathcal{R}_S^*$ is a model for all irreducible (at non-$\mathcal{B}$-positions) instances of clauses in S.

A substitution σ is irreducible by $\mathcal{R}_S$ at non-$\mathcal{B}$-positions in a clause $\Gamma \to \Delta$ if σ is irreducible at non-$\mathcal{B}$-positions in all equations $s \simeq t$ (see Definition 5.3.46) occurring in Γ or Δ.

Lemma 5.4.3 *Let S be a set of Horn clauses closed under $\mathcal{H}'$. Then $\mathcal{R}_S^* \models C\sigma$ for all clauses C in S and ground substitutions σ that are irreducible by $\mathcal{R}_S$ at non-$\mathcal{B}$-positions in C.*

Proof. We refer to an equation between abstracted terms as an abstracted equation, and we write $(s \simeq t)\cdot\gamma$ for $s\cdot\gamma \simeq t\cdot\gamma$. A clause consisting of abstracted equations is called an abstracted clause, and it is analogously denoted by $C\cdot\gamma$, where C is a clause. We must point out that we only use abstractions at the proof level, i.e., they do not occur in the inference system.
Given a clause C, by $|C|_v$ we denote the sum of the size of the terms occurring in C as a side of an equation, without counting the variables.
We proceed by minimal counterexample. Let $C'\cdot\gamma$ be the abstracted clause w.r.t. $\mathcal{R}_S$ with minimal $|C'|_v$ such that $C'\cdot\gamma$ is an abstraction of a clause $C\sigma$ with $C \in S$, σ irreducible by $\mathcal{R}_S$ at non-$\mathcal{B}$-positions in C, and $\mathcal{R}_S^* \not\models C\sigma$.
Since E_S is closed under ordered paramodulation then, by Theorem 5.3.50, we have that $\mathcal{R}_S^* \models E_S$, and therefore $C\sigma$ cannot be a positive unit clause. Then C is of the form $\Gamma, \underline{s \simeq t} \to \Delta$ and C' is of the form $\Gamma', \underline{s' \simeq t'} \to \Delta'$. Two cases have to be considered:

1. If $s\sigma \equiv t\sigma$, then there is an inference by equality resolution,

$$\frac{\Gamma, \underline{s \simeq t} \to \Delta}{(\Gamma \to \Delta)\theta},$$

   whose conclusion D has an instance $D\sigma$ such that $(\Gamma' \to \Delta')\cdot\gamma$ is an abstraction of $D\sigma$ with $|C'|_v > |(\Gamma' \to \Delta')|_v$ and, moreover, σ is irreducible by $\mathcal{R}_S$ at non-$\mathcal{B}$-positions in D and $\mathcal{R}_S^* \not\models D\sigma$, which is a contradiction.

2. If $s\sigma \not\equiv t\sigma$, since $\mathcal{R}_S^* \not\models C\sigma$, we have that $\mathcal{R}_S^* \models s\sigma \simeq t\sigma$. Then, since $E_S \models \mathcal{R}_S^*$, by Lemma 5.3.51 there must be a $\mathcal{B}$-rewrite proof of $s\sigma \simeq t\sigma$ with $\mathcal{R}_S$. Moreover, since $s\sigma \curvearrowright^*_{\mathcal{R}_S} s'\cdot\gamma$ and $t\sigma \curvearrowright^*_{\mathcal{R}_S} t'\cdot\gamma$ (using abstraction steps), by confluence of $\curvearrowright_{\mathcal{R}_S}$, there is some abstracted term $u\cdot\delta$ such that $s'\cdot\gamma \curvearrowright^*_{\mathcal{R}_S} u\cdot\delta \curvearrowleft^*_{\mathcal{R}_S} t'\cdot\gamma$. Since $s'\gamma \not\equiv t'\gamma$, either $s'\cdot\gamma$ or $t'\cdot\gamma$ is reducible by $\curvearrowright_{\mathcal{R}_S}$ with a non-abstraction step. Assume that $s'\cdot\gamma$ is reducible (the other case is analogous) at some non-variable position p of s'. By irreducibility of the substitution σ at non-$\mathcal{B}$-positions of C and minimality of $|C'|_v$, p must be a non-variable position of s, and hence there is an inference by paramodulation left,

$$\frac{\to l \simeq r \qquad \Gamma, \underline{s \simeq t} \to \Delta}{(\Gamma, s[r]_p \simeq t \to \Delta)\theta},$$

   whose conclusion D has an instance $D\sigma$ such that $(\Gamma', s'[x]_p \simeq t' \to \Delta')\cdot(\gamma \cup \{x \mapsto r\sigma\})$ is an abstraction of $D\sigma$ with $|C'|_v > |(\Gamma', s'[x]_p \simeq t' \to \Delta')|_v$ and, moreover, σ is irreducible by $\mathcal{R}_S$ at non-$\mathcal{B}$-positions in D (note that $r\sigma$ is a $\mathcal{B}$-term) and $\mathcal{R}_S^* \not\models D\sigma$, which is a contradiction.

$\square$

Now, as in the equational case, by first normalizing the substitution, we can prove the main theorem.

Theorem 5.4.4 (Refutation completeness of $\mathcal{H}'$ for Horn clauses) *Let S be a set of Horn clauses closed under $\mathcal{H}'$. Then $\Box \in S$ if, and only if, S is unsatisfiable.*

Proof. The left-to-right implication is trivial. For the other one, let S be a set of clauses closed under $\mathcal{H}'$ such that $\Box \notin S$. We prove that $\mathcal{R}_S^*$ is a model of S, i.e., $\mathcal{R}_S^* \models C\sigma$ for all clauses C in S and ground substitutions σ. Let σ' be a substitution such that $x\sigma' \equiv w\gamma_w$ if $x\sigma \curvearrowright^!_{\mathcal{R}_S} w \cdot \gamma_w$. Then $\mathcal{R}_S \cup \{C\sigma'\} \models C\sigma$ and, since σ' is irreducible by $\mathcal{R}_S$ at non-$\mathcal{B}$-positions in C, by Lemma 5.4.3, we have that $\mathcal{R}_S^* \models C\sigma'$, which implies that $\mathcal{R}_S^* \models C\sigma$. $\Box$

5.5 Conclusions

In this chapter we have improved on the result of Chapter 3, by showing that well-foundedness of the term ordering is sufficient for completeness of (positive) ordered paramodulation for Horn clauses with equality. The difference from Chapter 3, where orderings need not be monotone but have to be compatible with the subterm ordering, is that now even equations may be ordered from subterms to superterms as in $a \to f(a)$.

This is a non-trivial extension of the previous one, since rules of the form $a \to f(a)$ necessarily cause non-termination of standard rewriting. In the completeness proof we have shown confluence of a non-terminating limit rewrite system where left-hand sides may overlap, and induction has been based in part on a restricted and terminating form of rewriting, called *blocked rewriting* ($\mathcal{B}$-rewriting for short). Interestingly, a by-product of our results is a Knuth-Bendix completion procedure for $\mathcal{B}$-rewriting, and its completeness.

We have pointed out two potential applications of our result:

- The first one is in reference to deduction modulo built-in equational theories E, since there is always a well-founded E-compatible ordering and, hence, requiring only well-foundedness allows us to consider any theory E.

- The second one concerns goal-oriented proofs. We have realized that, sometimes, a goal-oriented (ordered) paramodulation proof can only be obtained if the ordering contradicts the subterm property (see Example 5.1.1 of the introduction)

Although these potential applications are interesting, one should also provide additional techniques for redundancy elimination, like simplification by rewriting, to make this calculus useful in practice. Note that from our proof only tautology deletion is clearly possible. Subsumption seems to be possible, but it may interfere with the proof of the so-called Return Property, which is the hardest and most technical part of our result. W.r.t. demodulation, we think that it may only be applicable in very restricted cases: observe that, in order to fulfill the Return Property, one may need to include pairs of rules like

$$a \to f(a)$$
$$f(f(a)) \to f(a)$$

in the limit rewrite system which defines the model and, in that case, e.g., simplification with $a \to f(a)$ should clearly be avoided. (See Chapter 7 for a result on compatibility of paramodulation with non-monotonic orderings with demodulation, for the case where the subterm property is fulfilled.)

$\mathcal{B}$-rewriting is closely related to the use of *basic* strategies (see Section 1.2.4 and [BGLS95, NR95]), which we believe can be compatible with our results (although it may somehow interfere with the proof of the Return Property). The basicness results should be worked out using the notion of *closures* defined for clauses in [BGLS95], since it coincides, at the level of clauses, with our abstracted terms (which we have used only in the proofs).

Finally we have to study how to extend our results to full first-order clauses. Here the problem comes from the fact that, within the model generation proof technique, the completeness for general clauses is usually based on the fact that at most one positive equation is satisfied in the generated (rewrite) model. Since, in our case, the ordering can contradict the subterm relation, we do not know how to obtain such a minimal model. The source of the problem is that some equation $l \simeq r$ which has been discarded to generate a rule in the limit rewrite system $\mathcal{R}$ which defines the model (because, although $l \simeq r$ is not satisfied by the model generated so far, it is reducible by some smaller rule at some non-$\mathcal{B}$-position), can follow from other (greater) rules and hence be satisfied in the final model. Let us illustrate this with a simple example.

Example 5.5.1 *Let $\succ$ be a (total) well-founded ordering on ground terms such that*

$$g(f(a)) \succ b \succ g(a) \succ f(f(a)) \succ a \succ f(a)$$

and E be a set of equations closed under ordered paramodulation, containing

$$\begin{aligned} a &\simeq f(a), \\ f(f(a)) &\simeq f(a), \\ b &\simeq g(a), \\ g(f(a)) &\simeq g(a), \quad \textit{and} \\ g(f(a)) &\simeq b \end{aligned}$$

among other (greater) equations. Then the generated $\mathcal{R}$ would be

$$\begin{aligned} a &\to f(a) \\ f(f(a)) &\to f(a) \\ g(f(a)) &\to b, \end{aligned}$$

since both $b \to g(a)$ and $g(f(a)) \to g(a)$ are reducible by $a \to f(a)$ at a non-$\mathcal{B}$-position of their right-hand side and, hence, they do not generate a rule in $\mathcal{R}$. However, $b \simeq g(a)$ is finally fulfilled by the model, since it follows from $a \simeq f(a)$ and $g(f(a)) \simeq b$, where the later is a greater equation.

In the case of full first-order clauses, this could imply that a non-unit positive clause with some of its equations having generated rule in $\mathcal{R}$, had at last more than one equation satisfied by the model.

Also, even if ordered paramodulation with well-founded orderings remained complete for the case of full first-order clauses, it is not clear from our proof of the Return Property how tautologies of the form $\Gamma, C \to \Delta, C$ could be avoided.

Chapter 6

Modular redundancy

When dealing with total reduction orderings, simplification and deletion techniques that are compatible with refutation completeness can be uniformly covered, in the model generation proof technique of [BG94b], by abstract notions of redundancy for inferences and clauses (cf. Section 1.2.2). Within paramodulation and Knuth-Bendix completion with west orderings, practical redundancy notions like tautology deletion or subsumption still preserve completeness (cf. Chapters 3 and 4), but there exist examples (cf. Chapter 7) showing how refutation completeness is lost if simplification by rewriting is fully applied. And, within paramodulation with well-founded orderings, it is even not obvious whether subsumption preserves completeness (cf. Chapter 5). Moreover, there exist examples showing the incompatibility of arbitrary selection techniques with tautology deletion. For example, refutation completeness is lost in logic programming with equality and arbitrary selection rules [Lyn97], and also in strict superposition [BG98b], if certain classes of tautologies are deleted.

Since, apart from the situations where it is known that the standard redundancy techniques cannot be used, there is also a large number of other cases where for several reasons the compatibility with redundancy is hard to prove or is unknown, our aim is to provide here modular notions of redundancy that can be used in a general setting.

6.1 Introduction

Let us recall from Section 1.2.2 that the main idea of saturation is that a theorem proving procedure does not need to compute the closure of a set of formulae w.r.t. a given inference system, but only the closure *up to redundancy*. Roughly, a saturation procedure adds conclusions of non-redundant inferences and removes redundant formulae, obtaining in the limit a saturated set.

Bachmair and Ganzinger [BG94b] gave general abstract notions of redundancy for inferences and formulae where, roughly, something is redundant if it is a logical consequence of smaller logical formulae (w.r.t. a given ordering on formulae). In provers based on ordered resolution, paramodulation or superposition w.r.t. a total reduction ordering on ground terms $\succ$, these abstract redundancy notions cover a large number of practical techniques, like removing tautologies, subsumption, or demodulation w.r.t. $\succ$.

However, as said before, there are contexts where refutation completeness is lost if such redundancy elimination methods are applied, or where it is unclear how to extend the completeness proof techniques to include redundancy, or where the theorem proving method is simply not based on the closure of a set of formulae under a set of inference rules. For all these cases our aim is to provide here modular notions of redundancy.

Assume we have an unsatisfiable initial set of formulae S_0, and we want to prove its unsatisfiability by some refutation complete prover $\mathcal{P}$ that is a “black box”, i.e., apart from its refutation completeness nothing is known about it. In particular, nothing is known about its compatibility with redundancy elimination methods. In principle, the prover $\mathcal{P}$ may be based on any technique, but in our context it will be useful that $\mathcal{P}$ soundly generates new formulae that are consequences from its input (this happens, for instance, if $\mathcal{P}$ is based on a sound inference system).

Now the problem we address is: how, and up to which degree, can we still exploit redundancy elimination methods in this general context?

One instance of our ideas roughly amounts to the following. Apart from the prover $\mathcal{P}$, there is a *redundancy module* $\mathcal{R}$ that works with a given arbitrary well-founded ordering $>$ on formulae (this ordering $>$ may be completely different from the one $\mathcal{P}$ uses, if $\mathcal{P}$ is based on some ordered inference system). The redundancy module $\mathcal{R}$ keeps track of a set of formulae U that is assumed to be unsatisfiable. The procedure is the following:

- Initially, U is S_0.
- Each time a new formula is generated by $\mathcal{P}$, this is communicated to $\mathcal{R}$, who uses this information to try to prove the redundancy of some formula F of U, i.e., it checks whether F follows from smaller (w.r.t. $>$) formulae $G_1, \ldots, G_n$ that are in U or have been generated by $\mathcal{P}$ (or are logical consequences of these).
- Each time this is the case, a new set U is obtained by replacing F by $G_1, \ldots, G_n$. This fact is then communicated to $\mathcal{P}$, whose aim becomes to refute the new U instead of the previous one.

In a naive setting, $\mathcal{P}$ could simply restart with the new U, but of course one can do better. Instead, $\mathcal{P}$ can apply several techniques (that may already be implemented in $\mathcal{P}$) in order to exploit the new situation. For example, $\mathcal{P}$ can apply *orphan murder* techniques: the descendants of F that have already been computed by $\mathcal{P}$ can be eliminated (although some of them can be kept if we want to completely avoid the repeated computation of formulae; for this, some additional bookkeeping is required). In this procedure, U is a finite set of formulae that decreases w.r.t. the well-founded multiset extension $\gg$ of $>$ each time it is modified. Hence at some point of the procedure U will not be modified any more and its limit U_∞ is obtained. If $\mathcal{P}$ computes all inferences from U_∞, then the refutation completeness of the general procedure follows.

Our techniques have been conceived with first-order clausal logic in mind. However, note that we do not rely on any particular logic.

This chapter is structured as follows. In Section 6.2 we give a formal description of prover modules, redundancy modules, their combination and some first results. In Section 6.3 we explain how a redundancy module with the right properties can be built. In Section 6.4 we explain for three cases how to build a prover module in terms of an existing prover: first, this is done for an (almost) arbitrary prover, in a rather naive way where not much power of the redundancy module is exploited; second, it is done for a prover based on the closure with respect to a complete inference system; and third, it is done for saturation-based provers (which is the case in most applications, like the ones of [Lyn97], [BG98b], and the ones of Chapters 3, 4 and 5). Finally in Section 6.5 we give some conclusions.

6.2 Modular Redundancy

We now give a formal description of prover modules, redundancy modules, their combination and some first results.

Definition 6.2.1 (Prover module) *A* prover module *is a procedure satisfying:*

- *It takes as input a set S_0 of formulae.*
- *At any point of its execution, it may receive messages of the form "replace F by $G_1,\ldots,G_n$", where F and $G_1,\ldots,G_n$ are formulae.*
- *At any point of its execution, it may produce messages of the form "F has been deduced", where F is a formula.*
- *It may halt with output "unsatisfiable".*

Definition 6.2.2 *Let $\mathcal{P}$ be a prover module, and consider a given execution of $\mathcal{P}$. By S_0 we denote the input set of formulae and, for each $k > 0$, by S_k we denote the set $S_k = S_{k-1} \cup \{G_1,\ldots,G_n\}$, where "replace F by $G_1,\ldots,G_n$" is the k-th message it receives.*

Furthermore, by U_0 we denote the input set of formulae and, for each $k > 0$, by U_k we denote the set $U_k = (U_{k-1} \setminus \{F\}) \cup \{G_1,\ldots,G_n\}$, where "replace F by $G_1,\ldots,G_n$" is the k-th message it receives.

Definition 6.2.3 *A prover module $\mathcal{P}$ is* correct *w.r.t. the given logic if:*

- *$S_k \models F$ whenever $\mathcal{P}$ emits an output message "F has been deduced" after receiving its k-th message.*
- *If $\mathcal{P}$ halts with output "unsatisfiable" after receiving its k-th message, then S_k is unsatisfiable.*

Definition 6.2.4 *A prover module $\mathcal{P}$ is* refutation complete *if, for every execution where it receives exactly k messages, it halts with output "unsatisfiable" whenever U_k is unsatisfiable.*

Definition 6.2.5 (Redundancy module) *A* redundancy module *is a procedure that satisfies:*

- *It takes as input a set S_0 of formulae.*
- *At any point of its execution, it may produce messages of the form "replace F by $G_1,\ldots,G_n$", where F and $G_1,\ldots G_n$ are formulae.*
- *At any point of its execution, it may receive messages of the form "F has been deduced", where F is a formula.*

Definition 6.2.6 *Let $\mathcal{R}$ be a redundancy module, and consider a given execution of $\mathcal{R}$. By R_0 we denote the input set of formulae and, for each $k > 0$, by R_k we denote the set $R_k = R_{k-1} \cup \{F\}$, where "F has been deduced" is the k-th received message.*

Definition 6.2.7 *A redundancy module is* correct *if $R_k \models \{G_1,\ldots,G_n\}$ and $G_1, \ldots, G_n \models F$ whenever it outputs a message "replace F by $G_1,\ldots,G_n$" after receiving its k-th message.*

Definition 6.2.8 *A redundancy module is* finitary *if for every execution the number of output messages produced is finite.*

A prover module $\mathcal{P}$ can be combined with a redundancy module $\mathcal{R}$ in the expected way to obtain what we will call a theorem prover: both modules receive the same input set of formulae, and the output messages of each one of them are the received messages of the other one. The theorem prover halts if and only if $\mathcal{P}$ halts. Let us remark that it is of course not our aim to discuss priority or synchronization issues or the like between both modules; one can simply assume that $\mathcal{R}$ uses a finite amount of CPU time initially and after each received message; when this time is finished $\mathcal{R}$ (possibly after sending a message) transfers control to $\mathcal{P}$ until $\mathcal{P}$ emits the next message.

Theorem 6.2.9 *A correct and complete prover module in combination with a correct and finitary redundancy module is a correct and refutation complete theorem prover, that is, a theorem prover that halts with output "unsatisfiable" if, and only if, the input set of formulae S_0 is unsatisfiable.*

Proof. From the correctness of both modules it follows that for all i, all sets S_i, U_i, and R_i are logically equivalent to S_0 (by induction on the subindex i: the new formulae coming from a message that is sent are a logical consequence of these sets with smaller index). Since $\mathcal{R}$ is finitary it sends only k messages, for some k. If S_0 is unsatisfiable, the corresponding set U_k is unsatisfiable, and by completeness of $\mathcal{P}$ the prover will halt with output "unsatisfiable". Conversely, if $\mathcal{P}$ halts with output "unsatisfiable" after receiving its j-th message, then, by correctness of $\mathcal{P}$, the set S_j and hence S_0 is unsatisfiable. $\square$

6.3 Finitary redundancy modules

Here we show how to create a finitary redundancy module, given a well-founded ordering $>$ on formulae. In order to abstract from concrete algorithms and strategies that can be used in a redundancy module, i.e., to be as general as possible, we first model the execution of a redundancy module by the notion of a *redundancy derivation*, a sequence of pairs of sets of formulae (U_i, R_i). The sets R_i correspond to what we defined in the previous section. The U_i will play the role of sets of formulae that are known to be unsatisfiable as the set U mentioned in the introduction (hence the name U, for unsatisfiable).

Definition 6.3.1 *We model the execution of a redundancy module by the notion of a* redundancy derivation*, a sequence $(U_0,R_0),(U_1,R_1),\ldots$ where all U_i and R_i are sets of formulae, and where the transition from a pair (U_i,R_i) to a pair (U_{i+1},R_{i+1}) takes place whenever a message is received or sent:*

- *$U_0 = R_0 = S_0$, where S_0 is the input set of formulae.*
- *If a message "F has been deduced" is received, then $U_{i+1} = U_i$ and $R_{i+1} = R_i \cup \{F\}$*
- *If a message "replace F by $G_1 \ldots G_n$" is sent, then $U_{i+1} = U_i \setminus \{F\} \cup \{G_1 \ldots G_n\}$ and $R_{i+1} = R_i$.*

We now impose conditions ensuring that the redundancy module satisfies the correctness and finiteness requirements.

Definition 6.3.2 *Let $>$ be a well-founded ordering on formulae. We say that a formula F is* redundant *w.r.t. $>$ by $G_1 \ldots G_n$ if $\{G_1 \ldots G_n\} \models F$ and $F > G_i$ for all i in $1 \ldots n$.*

By $\mathcal{R}_>$ we denote a redundancy module modeled by a redundancy derivation $(U_0,R_0),(U_1,R_1),\ldots$ such that if $U_{i+1} = U_i \setminus \{F\} \cup \{G_1 \ldots G_n\}$, i.e., a message "replace F by $G_1 \ldots G_n$" is sent, then

- *$R_i \models \{G_1 \ldots G_n\}$ and*

- *F is a formula in U_i that is redundant by $G_1 \dots G_n$.*

A practical redundancy module implementing $\mathcal{R}_>$ can be a procedure that updates the set R whenever it receives a message, and keeps looking for formulae $G_1 \dots G_n$ that can be used for proving the redundancy of some F in the set U, by means of (probably) well-known redundancy elimination techniques, and then replacing F by these $G_1 \dots G_n$. For example, one can simplify F by rewriting (demodulating) applying some equational formula G_1 of U, and replace F by its simplified version G_2. In practice it will be useful to apply formulae G_k that are in U, since, on the one hand, this helps keeping the set U small, which will be useful, and on the other hand, by construction all formulae in each U_i are logical consequences of R_i, so this does not have to be checked.

Theorem 6.3.3 *$\mathcal{R}_>$ is a finitary redundancy module.*

Proof. Let $\gg$ denote the multiset extension of $>$, that is, the smallest ordering such that $S \cup \{F\} \gg S \cup \{F_1, \dots, F_n\}$ whenever $F > F_i$ for all i in $1 \dots n$. The ordering $\gg$ is a well-founded ordering on finite (multi)sets of formulae whenever $>$ is a well-founded ordering on formulae (see Section 2.11). Indeed every U_i is a finite set of formulae (by induction on i: initially it is, and at each step only a finite number of formulae are added). Moreover U decreases w.r.t. $\gg$ at each message sent. Hence the number of output messages must be finite. □

Theorem 6.3.4 *$\mathcal{R}_>$ is a correct redundancy module.*

Proof. From the definition of $\mathcal{R}_>$, whenever a message "replace F by $G_1 \dots G_n$" is sent we have that $R_i \models G_1, \dots, G_n$ and $G_1, \dots, G_n \models F$. This implies correctness. □

6.4 How do we obtain a prover module from a given prover

6.4.1 A prover module from any prover

It is easy to see that, in a naive way, a complete prover module can be obtained from any correct and refutation complete theorem prover A (with the name A for any): given an unsatisfiable set of formulae S as input, A halts with output "unsatisfiable" after finite time. The main idea is simply to restart A with the new set U_k every time a message "replace F by $G_1, \dots, G_n$" is received.

Instead of formally defining the communications between A and the "layer" around it, that is, the remaining part of the prover module that uses A, we assume, without going into these details, that the layer knows whether A has inferred a formula, and that it is possible to tell A by means of the operation *Restart*(S) that it has to restart from scratch with input the set of formulae S.

We now define the prover module in terms of the prover A. In this prover module there will be a set of formulae U that will play the same role as before (see Definition 6.2.2) and as the set U in the redundancy module. It will be maintained by this prover module according to this invariant. Hence, initially U is the input set S_0 and each time a received message "replace F by $G_1 \dots G_n$" is treated, the prover module will update U by doing $U := (U \setminus \{F\}) \cup \{G_1 \dots G_n\}$.

Summarizing, each time a message "replace F by $G_1 \dots G_n$" is received we do: $U := (U \setminus \{F\}) \cup \{G_1 \dots G_n\}$, and *Restart*$(U)$. Each time the prover A infers F, a message "F has been deduced" is sent.

Definition 6.4.1 *Let $\mathcal{P}_A$ denote the prover module as defined above.*

Theorem 6.4.2 *If A is a correct and refutation complete prover, then $\mathcal{P}_A$ is a correct and complete prover module.*

Proof. We clearly have $U = U_k$ after the k-th received message in $\mathcal{P}_A$. And moreover $S_k \models U_k$. Then, correctness of $\mathcal{P}_A$ follows from correctness of A and from the fact that A is restarted with input U_k after every received message. For completeness, suppose $\mathcal{P}_A$ receives exactly k messages. We have $U = U_k$ after the k-th message. Then, completeness of $\mathcal{P}_A$ follows from completeness of A, given that $\mathcal{P}_A$ just restarts A with input U_k after the k-th received message. □

It is doubtful that this naive use of A will lead to any practical usefulness, since restarting A from scratch is a very drastic measure that produces large amounts of repeated work. A possible alternative would be not to restart A after each received message, but restart it each n units of time for an arbitrary fixed n. Obviously this does not affect completeness. Another better solution is given in the next subsection, where no work is repeated due to restarts, and an important amount of unnecessary work is avoided.

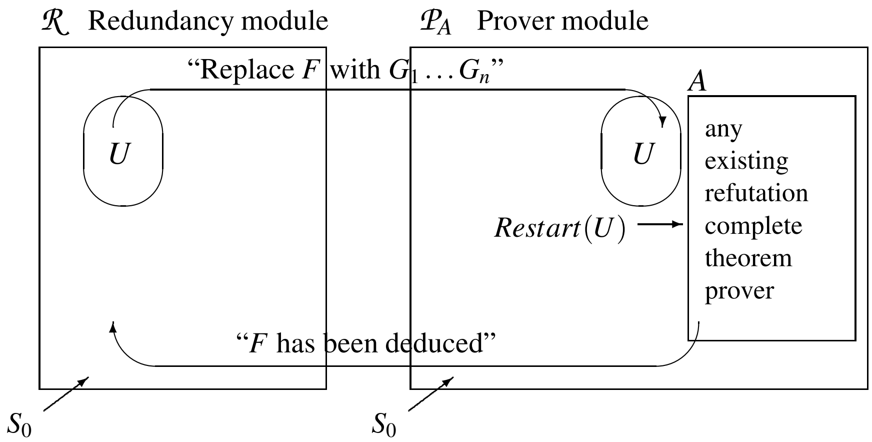


Figure 6.1: Schematic representation of a prover module built from any prover.

6.4.2 A prover module from a closure prover

Here we consider a given prover C (with the name C for closure) that is simply based on some correct and refutation complete inference system, and computes the closure w.r.t. this inference system using some strategy. Furthermore, it is assumed that this is done in such a way that the result is a complete prover: given an unsatisfiable set of formulae S as input, it halts with output "unsatisfiable", since it finds a contradictory formula, denoted by $\Box$.

In the following, this kind of prover will be called a *closure prover*. We will show that from C we can build a correct and complete prover module if C fulfills some minimal requirements needed in order to interact with it. The resulting prover module, in combination with a redundancy module, can reasonably well exploit our modular redundancy notions by means of *orphan murder*.

We now define the prover module in terms of the prover C.

In this prover module there will be two disjoint sets of formulae, U and P, that will be maintained by this prover module with the following invariants:

- U will play the same role as before (see Definition 6.2.2), and
- P is the set of remaining formulae stored by the prover C.

Hence, initially U is the input set S_0, and P is empty, and each time a received message "replace F by $G_1 \ldots G_n$" is treated, the prover module will update P and U by doing $P := P \setminus \{G_1 \ldots G_n\}$ and $U := (U \setminus \{F\}) \cup \{G_1 \ldots G_n\}$ (note that this may cause F to disappear from $U \cup P$). Similarly, each time the prover C infers a new formula F, the prover module will update P by doing $P := P \cup \{F\}$.

Furthermore, we associate to each F in $U \cup P$ a set of sets of formulae, denoted by $Parents(F)$, with the invariant that the sets $\{F_1, \ldots, F_n\}$ in $Parents(F)$ correspond, one to one, to the inference steps with premises $F_1, \ldots, F_n$ and conclusion F that have taken place in the prover C and none of $F_1, \ldots, F_n$ have been removed after the inference step. In order to maintain these invariants, each time such an inference takes place, $\{F_1, \ldots, F_n\}$ will be added to $Parents(F)$ if F already existed in $U \cup P$, and $Parents(F)$ will be $\{\{F_1, \ldots, F_n\}\}$ if F is new. Similarly, since we require that all formulae occurring in $Parents(F)$ have not been removed afterwards from $U \cup P$, if $\{F_1, \ldots, F_n\} \in Parents(F)$ and some F_i is removed from $U \cup P$, then the set $\{F_1, \ldots, F_n\}$ is removed from $Parents(F)$.

In this context, the idea of orphan murder is the following. If, for some formula F in P (and hence F not in U) we have $Parents(F) = \emptyset$, then F appeared because it was inferred, in one or more ways, by the prover C, but, for each one of these ways, the premises do not exist any more. Hence F itself is not needed for the completeness of the prover C and can be removed as well. This triggers new eliminations of other formulae G of P if F occurs in all sets of $Parents(G)$, and so on. We assume that each time a formula is eliminated in this way, this fact is communicated to the prover C by an operation $Remove(F)$, which has the effect that C simply removes F from its data structures. This operation has its counterpart $Add(S)$, which tells C to consider the formulae of a set of formulae S as additional input formulae.

Expressing this in an algorithmic way, we obtain the following. Initially U is the input set S_0, and P is empty. Each time a message "replace F by $G_1 \ldots G_n$" is received we do:

{ Precondition: $F \in U$ **and** $Parents(F) = \emptyset$ }
$Remove(F)$
$Add(\{G_1, \ldots, G_n\} \setminus (U \cup P))$
$P := P \setminus \{G_1 \ldots G_n\}$
$U := (U \setminus \{F\}) \cup \{G_1 \ldots G_n\}$
While $\exists F' \in (U \cup P)$ **and** $\exists S' \in Parents(F')$ **and** $S' \not\subseteq (U \cup P)$ **Do**
    $Parents(F') := Parents(F') \setminus \{S'\}$
    If $Parents(F') = \emptyset$ **and** $F' \in P$ **Then**
        $P := P \setminus \{F'\}$
        $Remove(F')$
    EndIf
EndWhile

As a precondition we assume that $F \in U$ and $Parents(F) = \emptyset$ since otherwise completeness may be lost. This precondition can be ensured in practice by an additional communication mechanism between both modules, for example, by making the redundancy module ask permission before it sends a message.

On the other hand, each time the prover C infers F from premises $G_1 \ldots G_n$, a message "F has been deduced" is sent, and we do:

If $F \in (U \cup P)$ **Then**
    $Parents(F) := Parents(F) \cup \{\{G_1, \ldots, G_n\}\}$
Else
    $P := P \cup \{F\}$
    $Parents(F) := \{\{G_1, \ldots, G_n\}\}$
EndIf

Note that, instead of formally defining the interactions between C and the "layer" around it, that is, the remaining part of the prover module that uses C, we have assumed, without going into these details, that the layer knows whether C has inferred a formula from certain premises, and that it is possible to tell C by means of the operations $Remove(F)$ and $Add(S)$ that a certain formulae can be removed or have to be added.

Definition 6.4.3 *Let $\mathcal{P}_C$ denote the prover module as defined above.*

We will now prove that the prover module $\mathcal{P}_C$ is correct and complete. For this purpose we need a reasonable additional assumption on the prover C, namely a notion of fairness:

Definition 6.4.4 *A formula F is* persistent *in C if it is stored by C at some point and not removed by the operation $Remove(F)$ later on.*

The prover C is called fair *if, whenever $G_1, \ldots, G_n$ become persistent formulae at some point of the execution, and there is a possible inference with them, then this inference is performed after that point.*

Lemma 6.4.5 *Given an execution of $\mathcal{P}_C$, if $G_1, \ldots, G_n$ are persistent formulae in C (and hence in $U \cup P$) and there is a possible inference with them producing F, then F is a persistent formula in C.*

Proof. By fairness, this inference is done at some point of the execution after $G_1, \ldots, G_n$ have become persistent. Furthermore $\{G_1, \ldots, G_n\} \in Parents(F)$ at that point. Moreover $\{G_1, \ldots, G_n\}$ will remain in $Parents(F)$. Then F will not be removed since $Parents(F)$ will never be empty. □

Theorem 6.4.6 *$\mathcal{P}_C$ is a correct prover module.*

Proof. Let S_k be defined as in Definition 6.2.2. We will show that $S_k \models (U \cup P)$ after the k-th received message and before a new message is received, and the correctness of the procedure will follow trivially. It suffices to show that this property is satisfied initially, and it is preserved during the execution. Initially $U \cup P$ is S_0. When an inference is done, its conclusion F, which is added to P, is a consequence of $(U \cup P)$ and, therefore, of the corresponding S_k. When a message is received, $S_k = S_{k-1} \cup \{G_1, \ldots, G_n\}$ and nothing else but $\{G_1, \ldots, G_n\}$ is added to $(U \cup P)$. □

Lemma 6.4.7 *Let U_k be defined as in Definition 6.2.2. Then $U_k = U$ after the k-th received message in $\mathcal{P}_C$.*

Theorem 6.4.8 *$\mathcal{P}_C$ is a complete prover module.*

Proof. We have to show that, in an execution of our prover module with exactly k input messages, if U_k is unsatisfiable then the prover module halts with output "unsatisfiable". By the previous lemma, $U_k = U$ immediately after the k-th received message. Furthermore, by fairness all inferences with premises in U are performed after the point their premises become persistent. By Lemma 6.4.5 the corresponding conclusions are persistent as well. All these descendants will be persistent (by iterating this, i.e., by induction on the number of inference steps required to obtain a descendant F from U). Hence all formulae in the closure of U w.r.t. the given inference system are indeed retained by C, which implies that $\square$ will be inferred since the inference system of C is refutation complete. $\square$

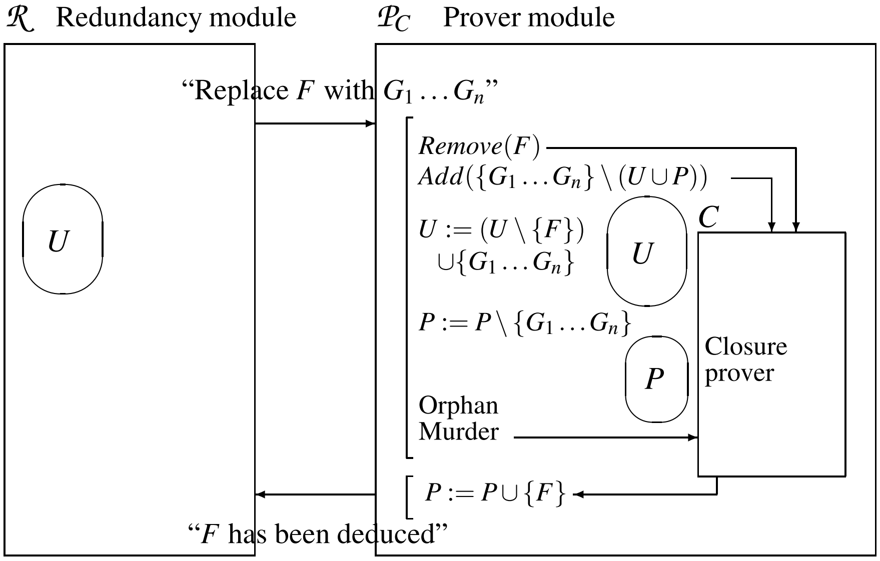


Figure 6.2: Schematic representation of a prover module built from a closure prover.

In this subsection we have developed the prover module in terms of a prover that fulfills a reasonable fairness requirement. We could have followed an alternative approach, without this fairness requirement. Instead of explaining this alternative here for closure provers, we develop it for saturation provers in the next subsection. It will become clear that its application to closure provers will be a particular case.

6.4.3 A prover module from a saturation prover

Here we consider a standard saturation-based prover D (with the name D for prover with *deletion* of redundant formulae). Without loss of generality, such a prover D can be assumed to consist of some *strategy* controlling the application of only two basic operations:

1. adding a new formula inferred by some correct deduction system, and
2. removing a formula F whose redundancy follows from other formulae $F_1,\ldots,F_n$ w.r.t. some *internal* redundancy criterion, with the minimal requirement that $F_1,\ldots,F_n \models F$ and a kind of transitivity property, namely if F is redundant by the set of formulae $F_1,\ldots,F_n$ and F_1 is redundant by the set of formulae $G_1,\ldots,G_m$, then F is redundant by the set $G_1,\ldots,G_m,F_2,\ldots,F_n$.

Note that redundancy techniques like demodulation of F into F' can be expressed by a combination of these two steps: first add F' (which is a logical consequence), then remove F (which has become redundant).

As usual, a prover like D can be modeled by a derivation, i.e., a sequence of sets of formulae, of the form $D_0, D_1, \ldots$ where at each step the strategy decides by which concrete instance of one of the two basic operations D_{i+1} is obtained from D_i. Of course, this is done in such a way that the result is a complete prover. For technical reasons that will become clear soon, here we will make a reasonable additional assumption, namely that if, for a finite number of basic steps of the derivation (i.e., for a finite number of decisions about which basic step is performed next), the strategy is overruled by an external event, like user interaction, the prover D remains complete. Note that the usual internal fairness requirements of saturation-based provers are not affected by such a finite number of "wrong" decisions.

As in the previous subsection, here we assume that the prover D also provides the operations *Add* and *Remove*, and provides information each time it infers a formula F from a set of premises $F_1, \ldots, F_r$. In addition, here we will assume that each time the prover removes a formula F that is redundant by a set $F_1, \ldots, F_r$, this is communicated as well.

In a practical prover, its strategy can be based on many different types of information (stored in data structures, like priority queues based on weighting functions) and that may depend on the current state as well as on historical information about previous states. Hence it has to be discussed how to deal with the external operations *Add* and *Remove* that were not foreseen when the strategy was developed. For instance, in a practical prover where a new formula F is added with the operation $Add(F)$, this F has to be inserted in the different data structures on which the strategy is based.

In the most general setting the strategy can be based on the whole *history* of the current derivation. Then the situation is as follows. Suppose that an operation *Add* or *Remove* takes place after step n of a derivation $D_0, \ldots, D_n$, and this operation converts D_n into another set S. Of course the strategy of the prover cannot be expected to behave well any more if it continues based on the history of the derivation $D_0, \ldots, D_n$. But fortunately, our prover module will only perform operations *Add* and *Remove* in such a way that another very similar derivation $D'_0, \ldots, D'_m$ exists where D'_m is S. Hence it is reasonable to expect that the prover D will behave well if it continues with the strategy it would have followed if it had reached S by this alternative similar derivation.

Therefore, we will assume that it is possible to update the internal state of the prover (on which its strategy is based) into the state it would be in after m basic steps of the alternative derivation $D'_0, \ldots, D'_m$. This is how we will proceed.

However, note that, although this alternative sequence consists of j correct steps, it may not be a derivation with the strategy. But since we know that the prover D is complete even if, for a finite number of decisions, the strategy is overruled by an external event, intuitively, the completeness of the prover module follows, since there will only be a finite number of such external events.

We now define the prover module in terms of the prover D. Again, there will be sets U and P and $Parents(F)$ for each F in $U \cup P$. As before, $U \cup P$ contains the formulae kept by the prover D but, due to the internal redundancy, U may not be U_k but a subset of it. In addition, each formula F in $U \cup P$ can also be *blocked* for external removals, i.e., it will never be removed from D by a $Remove(F)$ operation, and we denote by $Blocked(U \cup P)$ the set of blocked formulae in $U \cup P$, that is, the set of blocked formulae kept by the prover D. In this case we only need the following invariant: $U \cup Blocked(U \cup P) \models U_k$. No assumption is needed about the set $Parents(F)$ in order to prove correctness and completeness of the prover module. Nevertheless, as we will see at the end of this section, the way we deal with these sets in combination with the way we construct the alternative derivations will allow us in most of the cases to avoid repeated work.

Expressing the previous ideas algorithmically, we obtain the following. Initially U is the input set S_0, and P is empty. Each time a message "replace F by $G_1 \ldots G_r$" is received we do:

{ Precondition: $F \in U$ **and** $Parents(F) = \emptyset$ **and** F is not blocked }
$Remove(F)$
$Add(\{G_1, \ldots, G_r\} \setminus (U \cup P))$
$P := P \setminus \{G_1 \ldots G_r\}$
$U := (U \setminus \{F\}) \cup \{G_1 \ldots G_r\}$
While $\exists F' \in (U \cup P)$ **and** $\exists S' \in Parents(F')$ **and** $S' \not\subseteq (U \cup P)$ **Do**
  $Parents(F') := Parents(F') \setminus \{S'\}$
  If $Parents(F') = \emptyset$ **and** $F' \in P$ **and** F' is not blocked **Then**
    $P := P \setminus \{F'\}$
    $Remove(F')$
  EndIf
EndWhile

The precondition is ensured as in the previous subsection. On the other hand, as said at the beginning of this subsection, after each $Remove(F)$ or $Add(F)$ operation, we have to update the internal state of the prover into a new state which can be reached by a correct derivation with the prover D ending with the same final set of formulae. At the end of this subsection, such an alternative derivation is given for each one of these operations. The prover is then updated into the state it would be in after producing the alternative derivation.

The alternative derivation can be built in several different ways. As a general rule, for provers keeping more historical information it pays off to build alternative derivations that mimic the past computations quite accurately in order to avoid repeated computations; for others, similar results will be obtained by shorter derivations leading to the current clause set. In any case, for the main results, it suffices for the moment to assume that the alternative derivation is defined correctly, in the sense that its final state corresponds to the current set of formulae stored in the prover.

Each time the prover D infers F from premises $G_1 \ldots G_r$, a message "F has been deduced" is sent, and we do:

If $F \in (U \cup P)$ **Then**
  $Parents(F) := Parents(F) \cup \{\{G_1, \ldots, G_r\}\}$
Else
  $P := P \cup \{F\}$
  $Parents(F) := \{\{G_1, \ldots, G_r\}\}$
EndIf

Finally, each time the prover D removes a formula F due to its redundancy by a set $F_1 \ldots F_r$, we do:

Foreach F_i in $\{F_1 \ldots F_r\}$ **Do**
  $Block(F_i)$
EndForeach
$U := U \setminus F$
$P := P \setminus F$

Note that, although at this point we do not apply any orphan murder, it will be applied only if, later on, another "replace" message is received by the redundancy module, which triggers the actual *Remove* operations. This is done because we can ensure completeness only if no more than a finite number of *Remove* operations occur, and there may be an infinite number of internal redundancy steps.

Definition 6.4.9 *Let $\mathcal{P}_D$ denote the prover module as defined above.*

Theorem 6.4.10 *$\mathcal{P}_D$ is a correct prover module.*

Proof. Analogous to the correctness proof of the previous subsection. □

Theorem 6.4.11 *$\mathcal{P}_D$ is a complete prover module.*

Proof. We have to show that, in an execution of our prover module with exactly k input messages, if U_k is unsatisfiable then the prover module halts with output "unsatisfiable".
Since there are finitely many input messages, there are finitely many, say n, operations *Add* or *Remove*. Let $D_0,\dots,D_m$ be the n-th alternative derivation. By the invariant of the process we have $U \cup Blocked(U \cup P) \models U_k$, and therefore $U \cup P$ is unsatisfiable.
Finally, since there are only a finite number of steps (m) contradicting the strategy of the prover D, it remains complete, and hence the prover module halts with output "unsatisfiable". □

Now it remains to define the alternative derivations. Before it was mentioned that for some provers it pays off to build alternative derivations that mimic the past computations quite accurately in order to avoid repeated computations and for others perhaps not. Here we explain a way that attempts to preserve as much as possible the previous history.

Suppose $D_1,\dots,D_n$ is the current derivation. The following additional invariant, which is necessary to prove the correctness of the alternative derivation, is always satisfied:

All formulae in $(\bigcup_{i\leq n} D_i) \setminus (U \cup P)$ are internally redundant by $Blocked(U \cup P)$.

This is how we construct the alternative derivation:

- **An operation $Add(F)$ takes place.** Then, a new derivation, with $D_n \cup F$ as final set, is constructed as follows. There are two cases to be considered. If $F \notin D_i$ for all $i \leq n$, then we consider the derivation $D_0 \cup \{F\},\dots,D_n \cup \{F\}$. Otherwise, let D_i be the set with maximal i such that $F \in D_i$. Note that $i < n$. Hence D_{i+1} is obtained by an internal redundancy of F. Then the alternative derivation is $D_0,\dots,D_i,D_{i+2} \cup \{F\},\dots,D_n \cup \{F\}$.

- **An operation $Remove(F)$ takes place.** Then, a new derivation, with $D_n \setminus \{F\}$ as final set, can be constructed. The new derivation $D'_0,\dots,D'_m$ consists of two parts $D'_0,\dots,D'_k$ containing only inference steps and $D'_k,\dots,D'_m$ containing only internal redundancy steps.

  1. We take D'_0 as $(D_0 \setminus \{F\}) \cup Ch(F)$, where $Ch(F)$ denotes the set of children of F in inferences along the sequence $D_0,\dots,D_n$. Let $D_{i_1},\dots,D_{i_k}$ be the sets of the derivation obtained by an inference step where F is neither a premise nor a conclusion. Then $D'_0,\dots,D'_k$ is the derivation where D'_j is obtained from D'_{j-1} by the same inference step between D_{i_j-1} and D_{i_j}. Note that all steps are correct since we have added $Ch(F)$ to the initial set. Now, we have $D'_k = (\bigcup_{i\leq n} D_i) \setminus \{F\}$, and hence $D_n \setminus \{F\} \subseteq D'_k$.
  2. The second part of the new derivation consists of eliminating the formulae in $D'_k \setminus D_n$ by internal redundancies. This can be done since $D_n \setminus \{F\}$ contains all blocked formulae (F is not blocked since a $Remove(F)$ is done) and, by the invariant above, all formulae in $D'_k \setminus D_n$ are redundant by the blocked formulae.

Regarding the aim of avoiding repeated work, a last invariant, that is also satisfied, shows the reason for maintaining the sets *Parents*(F): for all inference steps in the derivation with premises $F_1,\ldots,F_r$ and conclusion F, such that none of $F_1,\ldots,F_r,F$ are redundant by *Blocked*$(U \cup P)$, we have $\{F_1,\ldots,F_r\} \in$ *Parents*(F). This invariant ensures that the only missing information in the sets *Parents*(F) (about performed inferences) involves either a premise or a conclusion that is redundant by the blocked formulae. Therefore, although it is clear that some repeated work might be done, the prover could avoid it by means of internal redundancy steps.

6.5 Conclusions

We have introduced a notion of modular redundancy for theorem proving that can be used to exploit redundancy elimination techniques in combination with arbitrary existing theorem provers, in a refutation complete way, even if these provers are not (or not known to be) complete in combination with the redundancy techniques when applied in the usual sense.

We have explained for three cases how to build a prover module in terms of an existing prover: for arbitrary provers, closure provers and saturation-based provers. The latter case seems to be the most general and useful one, since there are several settings where the standard general redundancy notions cannot be used, like [Lyn97], [BG98b], or the calculus presented in the previous chapters.

Specially in the case of the calculus presented in the previous chapters, where the completeness of paramodulation w.r.t. non-monotonic orderings is proved, we believe that our modular redundancy results can be a reasonable option to deal with redundancy. The reason is that standard redundancy notions should be applied w.r.t. an ordering that is built (at the completeness proof level) based on semantic information, and hence it appears that cannot be approximated in a strong enough way during the saturation process.

Chapter 7

Redundancy in paramodulation with west orderings

Simplification of formulae and elimination of redundant formulae are essential components of automated theorem provers. In fact, in most successful automated theorem provers, simplification is the primary mode of computation, whereas prolific deduction rules are used only sparingly. Therefore, although modular notions of redundancy like the ones presented in the previous chapter can be applied, it is always better to have ad hoc redundancy techniques. To this end, in this chapter we present a calculus which strictly improves the one of Chapter 3 in that it is compatible with powerful redundancy notions, which include simplification by rewriting (i.e., demodulation).

As a side effect, like in Chapter 4, we can apply our results to obtain a Knuth-Bendix completion procedure, but in this case compatible with simplification techniques. This procedure can be used for finding, whenever it exists, a convergent TRS for a given set of equations and a (possibly non-totalizable) reduction ordering.

7.1 Introduction

Here we present a calculus which strictly improves the one in Chapter 3. On the one hand, regarding the amount of inferences needed to be performed, our inference system is essentially the same as the one in Chapter 3 but, on the other hand, our calculus is compatible with powerful redundancy notions which include demodulation (although with some limitations).

In our calculus it is assumed that there is a known reduction ordering $\succ_r$, which is included in the (possibly non-monotonic) ordering $\succ$ w.r.t. which equations are oriented. Then we show that redundancy notions w.r.t. the reduction ordering $\succ_r$ can be applied while keeping refutation completeness. Roughly, a formula is redundant if it is a logical consequence of smaller (w.r.t. a well-founded ordering including $\succ_r$) formulae. Then demodulation w.r.t. $\succ_r$ fits in this abstract notion of redundancy since, in particular, a formula which has been simplified by rewriting follows from its smaller w.r.t. $\succ_r$ (due to monotonicity of $\succ_r$) "simplified version" and the (also smaller) equations used to simplify it. As a particular case, if all equations involved in the saturation process turn out to be orientable with $\succ_r$, then demodulation can be fully applied. The following well-known simple challenge example illustrates this situation:

Example 7.1.1 *Under our inference system,*

$$f(f(x)) \simeq f(g(f(x)))$$

is saturated if $f(f(x)) \succ_r f(g(f(x)))$, and the ordering $\succ$ w.r.t. which ordered paramodulation is applied is an extension of $\succ_r$ (which cannot be totalized without losing monotonicity, since $g(f(x))$ and $f(x)$ should be incomparable in any monotonic extension). As shown below, for every possible inference, the conclusion can be simplified into a tautology. By an ordered paramodulation with

$$f(f(x)) \simeq f(g(f(x)))$$

on itself at the underlined position of

$$f(\underline{f(y)}) \simeq f(g(f(y))),$$

where the most general unifier of $f(f(x))$ and $f(y)$ is $\{y \mapsto f(x)\}$, we get

$$f(f(g(f(x)))) \simeq f(g(f(f(x)))).$$

Now, by previously renaming the variables, both sides of the conclusion

$$f(f(g(f(z)))) \simeq f(g(f(f(z))))$$

can be rewritten in one step with $f(f(x)) \simeq f(g(f(x)))$ (oriented with the reduction ordering) into $f(g(f(g(f(z)))))$, i.e., the conclusion of the inference can be simplified into a tautology. On the other hand, by an ordered paramodulation with

$$f(f(x)) \simeq f(g(f(x)))$$

on itself at the underlined position of

$$f(f(y)) \simeq f(g(\underline{f(y)})),$$

we get

$$f(f(f(x))) \simeq f(g(f(g(f(x))))),$$

i.e., $f(f(f(z))) \simeq f(g(f(g(f(z)))))$. In this case, $f(f(f(z)))$ can be rewritten in two steps with $f(f(x)) \simeq f(g(f(x)))$ (oriented with the reduction ordering) into $f(g(f(g(f(z)))))$, and hence the conclusion of the inference can be simplified into a tautology as before. Therefore, the system is saturated. It is worth noting that, without considering demodulation, the closure of $f(f(x)) \simeq f(g(f(x)))$ under ordered paramodulation w.r.t. $\succ$ diverges. $\square$

We must point out that, although demodulation w.r.t. $\succ_r$ is allowed in our inference system, not every clause can be demodulated: during our saturation process, some subterms of clauses become “blocked” for demodulation. Roughly, we block for demodulation those subterms that occur as the small side (w.r.t. $\succ$) of an equation that cannot be handled with $\succ_r$. Later inferences will inherit these blockings. Note that, if paramodulation is applied w.r.t. a non-monotonic ordering $\succ$ and we use full demodulation w.r.t. $\succ_r$, we can lose completeness as shown in Example 7.4.8 of Section 7.4.2.

The reason for adding these blockings comes from our completeness proof, which is based on the model generation technique of [BG94b] and its variant used in Chapter 3. In the latter, in contrast to the former, the ordering used for orienting the equations in the inference system and the ordering used for induction in the completeness proof do not coincide. Concretely, equations are oriented w.r.t. a non-monotonic ordering $\succ$, whereas completeness is proved by induction w.r.t. $\xrightarrow{+}_R$, where R is the limit ground rewrite system that defines the model. This is the key for the completeness proof of ordered paramodulation with non-monotonic orderings given in Chapter 3. Since completeness is proved by induction w.r.t. $\xrightarrow{+}_R$, the redundancy notions should be defined w.r.t. $\xrightarrow{+}_R$ as well, instead of w.r.t. $\succ$, which is what would be desirable in principle. Unfortunately, $\xrightarrow{+}_R$ is unknown during the saturation process, and it is not clear how it can be approximated sufficiently. This is why in Chapter 3 it was left open to what extent demodulation could be applied (although some practical redundancy notions like tautology deletion and subsumption were shown to preserve completeness).

Here our aim is to add for instance demodulation w.r.t. the reduction ordering $\succ_r$, which is included in the (possibly) non-monotonic extension $\succ$. The idea for doing that is, roughly, to find a well-founded ordering $\succ_i$, which combines as much as possible $\succ_r$ with $\xrightarrow{+}_R$, and then prove completeness by induction w.r.t. $\succ_i$. The problem is that, although $\succ_r$ is a well-founded and monotonic ordering and R is a terminating TRS whose rules are included in a well-founded extension $\succ$ of $\succ_r$, the relation $\rightarrow_R \cup \succ_r$ is not well-founded in general and, in fact, $\xrightarrow{+}_R$ can even contradict $\succ_r$. Take, e.g., $f(b) \succ_r f(a)$ and a well-founded extension $\succ$ of $\succ_r$ such that $f(b) \succ f(a) \succ a \succ b$. Then possibly $a \rightarrow b \in R$ (since the rules in R are oriented w.r.t. $\succ$) and hence the infinite sequence $f(a) \rightarrow_R f(b) \succ_r f(a) \cdots$ can be built.

The idea to circumvent this non-well-foundedness problem is to block the "bad" right-hand sides, i.e., the ones that can be introduced by a rewriting step with R which is not included in $\succ_r$. Then we can combine $\succ_r$ with $\rightarrow_R$ if the comparisons with $\succ_r$ cannot take into account the blocked terms. For instance, in the previous example, b should be blocked, and then $f(b)$ with b blocked is no longer greater than $f(a)$.

A convenient way to represent terms with blocked positions is by means of superindexed subterms, also called *marked terms*. For example $f(b^x)$, where x is a variable, denotes the term $f(b)$ where the subterm b is blocked. Then, although for performing inferences $f(b^x)$ still corresponds to the term $f(b)$, for the redundancy notions it will be seen as $f(x)$ with a blocked term b in x. In this context $f(b^x)$ cannot be simplified with $f(b) \rightarrow f(a)$, but it could still be simplified with $f(x) \rightarrow g(x)$ into $g(b^x)$, if $f(x) \succ_r g(x)$. Marked terms resemble the *abstracted terms* (or *term closures*, in the terminology of [BGLS95]) of Chapter 5. However, their semantics is fairly different, since here the blockings have only effect on the redundancy notions.

Detection of terms that have to be marked and introduction of marks in the clauses will be part of our inference system. The terms to be marked come from the equations that cannot be oriented with $\succ_r$ (since we do not know in advance which equations will be part of R). These terms are kept in a set, which is used for introducing marks in the clauses. Hence, if there are many equations that cannot be handled by the reduction ordering then many terms will become blocked and we will have almost no redundancy, but the same amount of inferences by paramodulation as in Chapter 3. On the other hand, if many equations can be handled by the reduction ordering, then few terms will be blocked and we will have as much redundancy as in the paramodulation calculi for total reduction orderings [HR91, BDH86, BD94, BG94b].

The chapter is structured as follows. Marked terms and orderings on marked terms are defined in Section 7.2. In Section 7.3, our calculus, including the notions of redundancy for inferences, is presented for deduction with sets of equations. It is extended to Horn clauses in Section 7.4. Saturation derivations including redundancy of clauses are considered in Section 7.4.1. In Section 7.4.2 some practical notions of redundancy for Horn clauses are presented. In Section 7.5 we show, as in Chapter 4, how from our results a Knuth-Bendix completion procedure for finding convergent TRSs can be obtained. In Section7.6 we extend our results to the case of general first order clauses. Finally in Section 7.7 we give some conclusions.

7.2 Marked terms

As said in the introduction, marked terms resemble the *abstracted terms* (or *term closures*, in the terminology of [BGLS95]) of Chapter 5. A term closure, following [BGLS95], is a pair $s \cdot \gamma$ consisting of a term s (the *skeleton*) and a substitution γ from variables to terms. For example, $f(x, g(y)) \cdot \{x \mapsto a, y \mapsto h(z)\}$ is a closure with skeleton $f(x, g(y))$ and substitution $\{x \mapsto a, y \mapsto h(z)\}$. We generalize closures in that the substitution can map a variable not only to a term but also to a closure. For example, $f(x, g(y)) \cdot \{x \mapsto a, y \mapsto h(z) \cdot \{z \mapsto b\}\}$ is a generalized term closure, called a *marked term*. Moreover, closures are used in [BGLS95] to express that certain positions are blocked for inferences, whereas in our case they are blocked only for redundancy purposes, i.e., the ordered paramodulation inferences are applied at both blocked and unblocked positions. And, while in Chapter 5 abstracted terms appeared only at the proof level, here marked terms are already present in the equations. In order to ease the reading, sometimes we will denote marked terms by superindexing the "marked" subterms with variables. For example,

$$f(x, g(y)) \cdot \{x \mapsto a, y \mapsto h(z) \cdot \{z \mapsto b\}\}$$

will be written as

$$f(a^x, g(h(b^z)^y)).$$

Marked terms are extended in the natural way to equations and clauses. An equation between marked terms is called a *marked equation*, and a clause on marked equations is called a *marked clause*. Given a substitution γ of the form $\{x_1 \mapsto t_1 \cdot \gamma_1, \ldots, x_n \mapsto t_n \cdot \gamma_n\}$, the domain of γ, denoted by $Dom(\gamma)$, is defined as the set of variables $\{x_1, \ldots, x_n\}$, and the range of γ, denoted by $Ran(\gamma)$, is defined as the set of marked terms $\{t_1 \cdot \gamma_1, \ldots, t_n \cdot \gamma_n\}$. The variables occurring in $Dom(\gamma)$ are called *marking variables*. We will restrict marked terms to those which fulfill that marking variables occur at most once in the skeleton. For example $f(x, y) \cdot \{x \mapsto a, y \mapsto a\}$ is a valid marked term, but $f(z, z) \cdot \{z \mapsto a\}$ is not. Therefore, although we generalize closures in one sense, in another sense we restrict them.[1]

The congruence $\equiv$ on marked terms is defined as the monotonic, reflexive, symmetric and transitive closure including (i) $t^{xy} \equiv t^x$, and (ii) $s[t^x]_p \equiv s[t^y]_p$ if neither x nor y occur in $s[t]_p$.[2] Note that if $s \cdot \gamma \equiv t \cdot \delta$, then $s \doteq t$, i.e., s is a variant of t.

Let $s \cdot \{x_1 \mapsto t_1 \cdot \gamma_1, \ldots, x_n \mapsto t_n \cdot \gamma_n\}$ be a marked term and σ be a substitution of the form $\{y_1 \mapsto u_1 \cdot \delta_1, \ldots, y_m \mapsto u_m \cdot \delta_m\}$, such that the variables in the domains of all substitutions are distinct. Then $(s \cdot \gamma)\sigma$ is defined as

$$s\{y_1 \mapsto u_1, \ldots, y_m \mapsto u_m\} \cdot (\{x_1 \mapsto (t_1 \cdot \gamma_1)\sigma, \ldots, x_n \mapsto (t_n \cdot \gamma_n)\sigma\} \cup \delta_1 \cup \cdots \cup \delta_m),$$

where the marking variables of the resulting skeleton are renamed if necessary to keep linearity. For example,

$$f(x, g(x)^z)\{x \mapsto a^y\} = f(a^y, g(a^{y'})^z),$$

and

$$f(x, x)\{x \mapsto a^y\} = f(a^y, a^z).$$

By *Forget*$(s \cdot \gamma)$ we denote the term in $T(\mathcal{F}, \mathcal{X})$ obtained from $s \cdot \gamma$ by repeatedly applying all the substitutions. For example,

$$Forget(f(a^y, g(a^{y'})^z)) = f(a, g(a)).$$

[1] Both the generalization and the restriction are necessary for our orderings on marked terms being well-founded.

[2] There is no condition that prevents terms like t^{xy} from appearing. For example, a marked variable z in, e.g., z^y may be instantiated with a marked term t^x, giving t^{xy}. However, we will assume that these variable-variable bindings are instantiated whenever they arise, i.e., that terms are kept in its simpler form (t^x in the example).

A marked term $s\cdot\gamma$ is said to be ground if $Forget(s\cdot\gamma)$ is a ground term. If $(s\cdot\gamma)\sigma$ is a ground marked term, then σ is called a *ground substitution* for $s\cdot\gamma$, and $(s\cdot\gamma)\sigma$ is called a *ground instance* of $s\cdot\gamma$. This notation is extended to sets of terms, equations and clauses in the usual way.

The usual concepts of (and notation for) position, subterm and subterm replacement can be easily extended for marked terms. For example,

$$f(x,g(x)^z)|_2 = g(x)^z,$$

and

$$f(x,g(x)^z)[a^y]_{2\cdot 1} = f(x,g(a^y)^z).$$

We write $u \trianglerighteq t\cdot\delta$ if $t\cdot\delta$ occurs in u, i.e., $u|_p \equiv t\cdot\delta$ for some position p, and we write $u \triangleright t\cdot\delta$ if $p \neq \lambda$. From these definitions, the notion of rewriting for terms can be extended to marked terms in the natural way.

Recall that, given two terms s and s' in $T(\mathcal{F},\mathcal{X})$, we write $s \doteq s'$ if s' is a variant (that is, a variable renamed version) of s. This extends to marked terms by writing $s\cdot\gamma \doteq s'\cdot\gamma'$ if there are some substitutions σ and σ' such that $(s\cdot\gamma)\sigma \equiv s'\cdot\gamma'$ and $s\cdot\gamma \equiv (s'\cdot\gamma')\sigma'$. For example, if x, y, z and z' are variables and a is not a variable, then $h(g(z)^x) \doteq h(g(z')^y)$, whereas $h(g(a^z)^x) \not\doteq h(g(a)^y)$.

The concept of unifier between terms is extended to marked terms as follows: a unifier of two marked terms $s\cdot\gamma$ and $t\cdot\delta$ is a substitution σ such that $(s\cdot\gamma)\sigma \equiv (t\cdot\delta)\sigma$. There exists a unique (up to renaming of variables) most general unifier between marked terms, which can be obtained by straightforwardly adapting any known unification algorithm for terms.

7.2.1 Orderings on marked terms

Let us recall from Chapter 2 that (i) the *subsumption* relation, denoted by $\geq$, is a quasi-ordering defined as $s \geq t$ if s is an instance of t, i.e., $s \equiv t\sigma$ for some σ, (ii) the equivalence relation associated with $\geq$ is denoted by $\doteq$, and (iii) the *encompassment* relation, denoted by $\trianglerighteq$, is a quasi-ordering defined as $s \trianglerighteq t$ if a subterm of s is an instance of t, i.e., $s \trianglerighteq t\sigma$ for some σ; therefore, encompassment is the composition of the subterm and the subsumption relations.

Let $\succ_r$ be a reduction ordering, $\succ$ be a west ordering including $\succ_r$, and R be a terminating ground TRS. It is folk knowledge that the union of any terminating rewrite relation with the (proper) encompassment ordering $\triangleright$ is also well-founded (see Section 2.11.2). Therefore, $\triangleright \cup \succ_r$ and $\triangleright \cup \xrightarrow{+}_R$ are well-founded.

In this section we define the two main orderings on marked terms, namely $\succ_{red}$, which will be used in the redundancy notions and will be closely related to $\succ_r$, and $\succ_i$, which will be used for induction in the completeness proof and will combine as much as possible $\succ_{red}$ with $\xrightarrow{+}_R$. For the reasons explained in the introduction, it is crucial for the completeness proof that $\succ_{red} \subseteq \succ_i$.

We first define the marking ordering $\succ_m$. It only applies to terms that just differ in the markings and it is similar to the subsumption ordering for terms, in the sense that a term is smaller if it has higher marks.

Definition 7.2.1 *Let $s\cdot\gamma$ and $t\cdot\delta$ be two marked terms. Then $s\cdot\gamma \succ_m t\cdot\delta$ iff $Forget(s\cdot\gamma) \equiv Forget(t\cdot\delta)$, $s\cdot\gamma \not\equiv t\cdot\delta$ and (i) $s > t$ or (ii) $s \doteq t$ and $Ran(\gamma) \succ_m^{mul} Ran(\delta)$.*

Given a marked term $s\cdot\gamma$, if $(s\cdot\gamma)|_p$ is of the form $(u\cdot\rho)^x$ then p is called a *marked position* of $s\cdot\gamma$. For example, 1 and $1\cdot 1$ are marked positions of $g(f(a^x,b)^y)$. Observe that, if $Forget(s\cdot\gamma) \equiv Forget(t\cdot\delta)$ and $s\cdot\gamma \not\equiv t\cdot\delta$, then $s\cdot\gamma$ and $t\cdot\gamma$ differ only in the marked positions. Then, if $s\cdot\gamma \succ_m t\cdot\delta$ by case (i), the skeleton t is the same as s except that some additional marking variables have been added. However, the number of marked positions of the whole marked term do not necessarily increase. For example, by case (i) we have

$$g(f(a,b)) \succ_m g(f(a^x,b)) \succ_m g(f(a^x,b^y)) \succ_m g(f(a,b))^z.$$

And by case (ii) we have, for example, $g(f(a,b))^z \succ_m g(f(a^x,b))^z$. Note also that, although $a \succ_m a^x$ by case (i), we do not have $a^x \succ_m a^{xy}$ by case (ii), since $a^x \equiv a^{xy}$.

Lemma 7.2.2 *$\succ_m$ is a well-founded ordering.*

Proof. It suffices to show that $\succ_m$ is a well-founded transitive relation. Given a marked term $s \cdot \gamma$, we define the interpretation $I(s \cdot \gamma)$ as the multiset of terms $Forget((s \cdot \gamma)|_p)$ for all non-marked positions p of $s \cdot \gamma$. It is not difficult to see that, if $s \cdot \gamma \succ_m t \cdot \delta$, then $I(s \cdot \gamma) \rhd^{mul} I(t \cdot \delta)$. Therefore well-foundedness of $\succ_m$ follows from well-foundedness of $\rhd^{mul}$. Transitivity of $\succ_m$ follows from transitivity of $\gtrdot$ and $\doteq$, and compatibility of $\gtrdot$ with $\doteq$. □

Since $\succ_{red}$ will be used in the redundancy notions, all its ingredients are known while performing paramodulation steps. Given two marked terms, $\succ_{red}$ roughly compares the skeletons w.r.t. the reduction ordering and the encompassment relation and, if the skeletons are the same and the terms only differ in the markings, then it compares the level of the marks in the substitution part.

Definition 7.2.3 *Let $s \cdot \gamma$ and $t \cdot \delta$ be two marked terms. Then $s \cdot \gamma \succ_{red} t \cdot \delta$ iff (i) $s \rhd \cup \succ_r t$ or (ii) $s \doteq t$ and $Ran(\gamma) \succ_m^{mul} Ran(\delta)$.*

Since $\succ_i$ will be used for induction in the completeness proof, it extends $\succ_{red}$ with an additional case which uses as ingredient a terminating rewrite system R that will be known only in the proof.

Definition 7.2.4 *Let $s \cdot \gamma$ and $t \cdot \delta$ be two marked terms. Then $s \cdot \gamma \succ_i t \cdot \delta$ iff (i) $s \rhd \cup \succ_r t$ or (ii) $s \doteq t$ and $Ran(\gamma) \succ_m^{mul} Ran(\delta)$ or (iii) $s \doteq t$ and $Forget(Ran(\gamma)) \xrightarrow{+}_R Forget(Ran(\delta))$.*

In what follows, we will consider $\succ_i$ and $\succ_{red}$ as the transitive closures of the orderings defined above (composed with the congruence $\equiv$ on marked terms).

Lemma 7.2.5 *$\succ_i$ is well-founded.*

Proof. First of all observe that, in a sequence $s_1 \cdot \gamma_1 \succ_i s_2 \cdot \gamma_2 \cdots$, there can only be finitely many steps by case (i) since the relation $(\rhd \cup \succ_r) \circ \doteq$ is well-founded (see Section 2.11.2 and [DJ90]). Hence, from some point on, there are only steps by case (ii) or (iii). Now observe that, if $s_k \cdot \gamma_k \succ_i s_m \cdot \gamma_m \succ_i s_n \cdot \gamma_n$, where the first step is by case (ii) and the second step is by case (iii), then $s_k \cdot \gamma_k \succ_i s_n \cdot \gamma_n$ by case (iii), since $Forget(s_k \cdot \gamma_k) \equiv Forget(s_m \cdot \gamma_m)$ by definition of $\succ_m$. Then, since R is terminating, there can only be finitely many steps by case (iii), i.e., from some point on all steps must be by case (ii). Finally, since $\succ_m$ is well-founded by Lemma 7.2.2, there can only be finitely many steps by case (ii), which implies that $\succ_i$ is well-founded. □

Example 7.2.6 *Here there are some examples of applications of $\succ_i$.*

1. *If $f(x) \succ_r g(x)$ for all x, then $f(a) \succ_i f(b^x) \succ_i g(b^x) \succ_i b^x$ by case (i) of the definition of $\succ_i$.*
2. *We have $h(g(f(a^x,a))^y) \succ_i h(g(f(a^x,a^z))^y) \succ_i h(g(f(a,a)^x)^y)$ by case (ii) of the definition of $\succ_i$.*

7.3 Paramodulation with equations

In this section we present an ordered paramodulation calculus for sets of equations. The calculus will work on both a set of marked equations E and a set of terms T, which contains the set of terms that are going to be marked.

In the following, let $\succ_r$ be a given reduction ordering and $\succ$ be a given west ordering including $\succ_r$. We assume that the non-marking variables of all equations are disjoint (if necessary, the variables can be renamed). On the other hand, to ease the reading, the marked terms of each equation are considered to have the same substitution (if necessary, the substitution can be extended and the marking variables of one of the two skeletons can be renamed).

Definition 7.3.1 *Let E be a set of marked equations and T be a set of terms, which satisfy the following condition: for every marked subterm $(s\cdot\gamma)^x$ occurring in E, we have that $Forget(s\cdot\gamma)$ is an instance of a term in T. The inference system $\mathcal{E}_2$ consists of the following two inference rules and a mark detection rule:*

Paramodulation:

$$\frac{l\cdot\delta \simeq r\cdot\delta \qquad s\cdot\gamma \simeq t\cdot\gamma}{(s\cdot\gamma[r\cdot\delta]_p \simeq t\cdot\gamma)\sigma} \qquad \textit{if}$$

1. *$\sigma = mgu(l\cdot\delta, s\cdot\gamma|_p)$, the most general unifier of $l\cdot\delta$ and $s\cdot\gamma|_p$,*
2. *$Forget(s\cdot\gamma|_p)$ is not a variable,*
3. *$l \succ_r r$ or $r\cdot\delta$ is marked on top, and*
4. *for some ground substitution θ, we have $Forget((l\cdot\delta)\sigma\theta) \succ Forget((r\cdot\delta)\sigma\theta)$ and, if $p=\lambda$, then we also have $Forget((s\cdot\gamma)\sigma\theta) \succ Forget((t\cdot\gamma)\sigma\theta)$.*

Marking:

$$\frac{s\cdot\gamma[u\cdot\delta]_p \simeq t\cdot\gamma}{(s\cdot\gamma[(u\cdot\delta)^x]_p \simeq t\cdot\gamma)\sigma} \qquad \textit{if}$$

1. *$\sigma = mgu(Forget(u\cdot\delta), u')$ for some term u' in T,*
2. *u is not a variable, or u' is a variable and $p=\lambda$, and*
3. *x is a fresh variable.*

Mark detection:

$$T \Longrightarrow T \cup \{Forget(r\cdot\delta)\} \qquad \textit{if}$$

1. *$l\cdot\delta \simeq r\cdot\delta \in E$,*
2. *$l \not\succ_r r$ and $r\cdot\delta$ is not marked on top, and*
3. *for some ground substitution θ, we have that $Forget((l\cdot\delta)\theta) \succ Forget((r\cdot\delta)\theta)$,*

meaning that $Forget(r\cdot\delta)$ is added to the set T.

Some examples of applications of these rules can be found in Example 7.4.8 of Section 7.4.2.

7.3.1 Redundancy notions: a static view

Here we define some abstract redundancy notions for inferences, like the ones in [BG94b] and [NR01]. Some practical redundancy notions fitting in these abstract notions are discussed afterwards, in Section 7.4.2.

Given a multiset of marked terms S, by $Marks(S)$ we denote the multiset of marked subterms of the form $(s \cdot \gamma)^x$ that occur in some marked term of S. Given a set of terms M and a substitution θ, we say that θ is a substitution *with marks in M* if every term t in $Forget(Marks(Ran(\theta)))$ is an instance of a term in M.

By $gnd(E)$ we denote the set of all ground instances of equations in E (where a substitution is a mapping from variables to marked terms). Given a set of terms M, by $gnd_M(E)$ we denote the set of ground instances $e\theta$ of equations e in E such that θ is a substitution with marks in M.

Let $>$ be a well-founded ordering on marked equations. By $E^{<e}$ we denote the set of all d in E such that $e > d$. An analogous notation is used for quasi-orderings $\geq$. We define $\succeq_{red}$ as $\succ_{red} \cup \equiv$, where $\equiv$ denotes the equivalence of marked terms as defined in Section 7.2. Note that, since $\succ_{red}$ is compatible with $\equiv$, $\succeq_{red}$ is a quasi ordering whose associated strict partial ordering is $\succ_{red}$ (see Section 2.11 of Chapter 7). Now we define redundancy of inferences and mark detection rules. Some explanations are given below.

Definition 7.3.2 *Let π be an inference by $\mathcal{E}_2$ with premises $e_1, \ldots e_n$ and conclusion d. Then a* ground instance $\pi\theta$ of the inference π *is an inference by $\mathcal{E}_2$ with premises $e_1\theta, \ldots e_n\theta$ and conclusion $d\theta$ for some ground substitution θ.*

Let G be a set of ground marked equations. A ground inference by $\mathcal{E}_2$ with premises $e_1, \ldots e_n$ and conclusion d is redundant in G *if*

$$Forget(G^{\prec_{red}^{mul} e_n}) \cup Forget(G^{\preceq_{red}^{mul} d}) \models Forget(d).$$

Let E be a set of marked equations. An inference by $\mathcal{E}_2$ with premises $e_1, \ldots e_n$ and conclusion d is redundant in E *if, for all sets of terms M and ground substitutions θ with marks in M, $\pi\theta$ is redundant in $gnd_M(E)$.*

Let T be a set of terms. A mark detection rule $T \Longrightarrow T \cup \{t\}$ is redundant in T if t is an instance of a term in T.

Note that the set of terms M is universally quantified in the previous definition of redundancy for inferences. This is due to the fact that the set of terms T to be marked will change during the computation, and it is crucial to have a notion of redundancy which is preserved along the computation. Moreover, the set $G^{\preceq_{red}^{mul} d}$ may not be included in $G^{\prec_{red}^{mul} e_n}$ since, in general, the conclusion of an inference does not need to be smaller w.r.t. $\succ_{red}^{mul}$ than the rightmost premise.

Redundant inferences and mark detection rules are unnecessary, and therefore we are interested in computing the closure of the set of equations with respect to $\mathcal{E}_2$ up to redundancy.

Definition 7.3.3 *Let E be a set of equations and T be a set of terms. The pair $\langle E, T\rangle$ is* saturated *with respect to $\mathcal{E}_2$ if every inference by $\mathcal{E}_2$ is redundant in E and every mark detection rule is redundant in T.*

7.3.2 Model generation

Given a set of marked equations E and a set of terms T, here we show how to build an equational model of E defined by means of a set of rules coming from ground instances of E and T. Since the model is defined inductively, we need to treat E and T in a uniform way. To this end, a new constant symbol $\perp$ is introduced and the west ordering is extended with $\perp$ as minimal element.

Let T_{E} denote the set of equations $t \simeq \perp$ for all t in T, and T_{G} denote the set of mark free ground instances of equations in T_{E}. The model will be defined inductively on the equations in T_{G} and $gnd_T(E)$ with respect to $\succ_{mod}$, a total well-founded ordering including: $(s \simeq t) \succ_{mod} (u \simeq v)$ if

1. $Forget(s \simeq t) \succ^{mul} Forget(u \simeq v)$.
2. $Forget(s \simeq t) \equiv^{mul} Forget(u \simeq v)$ and $(s \simeq t) \succ^{mul}_m (u \simeq v)$, which means that $u \simeq v$ has marks at higher positions than $s \simeq t$.

Now we define a rewrite system R_E from $gnd_T(E)$, which requires to define at the same time a rewrite system R_T from T_{G}.

Definition 7.3.4 *The rewrite system R_{ET} is the union of R_E and R_T. Let e be a ground (marked) equation in $gnd_T(E) \cup T_{\mathrm{G}}$.*

1. *If e is an instance $(s\cdot\gamma)\theta \simeq (t\cdot\gamma)\theta$ in $gnd_T(E)$ of some equation $s\cdot\gamma \simeq t\cdot\gamma$ in E, then it generates the rule $(s\cdot\gamma)\theta \rightarrow (t\cdot\gamma)\theta$ in R_E if*
   - *(a) $s \succ_r t$ or $t\cdot\gamma$ is marked on top,*
   - *(b) $Forget((s\cdot\gamma)\theta) \succ Forget((t\cdot\gamma)\theta)$,*
   - *(c) $(s\cdot\gamma)\theta$ is irreducible by $R^e_E \cup R^e_T$, and*
   - *(d) $(t\cdot\gamma)\theta$ is irreducible by $R^e_E \cup R^e_T$ at non-topmost positions.*
2. *If e is an instance $t\theta \simeq \perp$ in T_{G} of an equation $t \simeq \perp$ in T_{E}, then it generates the rule $u\cdot\delta \rightarrow (u\cdot\delta)^x$ in R_T if $u\cdot\delta$ is the normal form of $t\theta$ w.r.t. R^e_T and $u\cdot\delta$ is irreducible by R^e_E,*

where R^e_E and R^e_T denote the set of rules in R_E and R_T respectively, generated by equations d such that $e \succ_{mod} d$.

The following two lemmas hold easily by construction.

Lemma 7.3.5 *Let $l\cdot\delta \rightarrow r\cdot\delta$ be a rule in R_T. Then $r\cdot\delta \equiv (l\cdot\delta)^x$, $l\cdot\delta$ is not marked on top, and the rule has been generated by an equation of the form $t\theta \simeq \perp$, where $t \in T$, $t\theta \in T(\mathcal{F})$, and $Forget(l\cdot\delta) \equiv t\theta$.*

Proof. We proceed by induction on the equation which has generated the rule w.r.t. $\succ_{mod}$. Observe that all equations e in T_{G} are of the form $t\theta \simeq \perp$, where $t \in T$ and $t\theta$ is a mark free term. Therefore $t\theta$ is not marked on top, and obviously $Forget(t\theta) \equiv t\theta$. Thus $t\theta \rightarrow (t\theta)^x$ fulfills the property.
Now let $u\cdot\rho$ denote the normal form of $t\theta$ w.r.t. R^e_T. We will show that the property is preserved in $u\cdot\rho \rightarrow (u\cdot\rho)^x$. By the induction hypothesis, all rules $l'\cdot\delta' \rightarrow r'\cdot\delta'$ in R^e_T fulfill $Forget(l'\cdot\delta') \equiv Forget(r'\cdot\delta')$. Therefore, we necessarily have that $Forget(u\cdot\rho) \equiv t\theta$. We conclude by showing that in the normalization of $t\cdot\theta$ w.r.t. R^e_T, no step can be at topmost position, and therefore $u\cdot\rho$ is not marked on top: assume the contrary and let $l'\cdot\delta' \rightarrow r'\cdot\delta'$ be the rule in R^e_T, generated by an equation $d = (s \simeq \perp)$, which has been applied at topmost position; then $Forget(l'\cdot\delta') \equiv t\theta$, contradicting $e \succ_{mod} d$ since, by the induction hypothesis again, $Forget(l'\cdot\delta') \equiv s$. □

Lemma 7.3.6 *For all rules $l\cdot\delta \rightarrow r\cdot\delta$ in R_{ET} we have that*

1. *$r\cdot\delta$ is irreducible by R_{ET} at non-topmost positions, and*

2. $l \cdot \delta$ *is irreducible by* $R_{ET} \setminus \{l \cdot \delta \to r \cdot \delta\}$.

Proof. For property 1, by construction, if an instance e generates $l \cdot \delta \to r \cdot \delta$, then the marked term $r \cdot \delta$ is irreducible by $R^e_E \cup R^e_T$ at non-topmost positions: this is clearly the case if $l \cdot \delta \to r \cdot \delta \in R_E$, since irreducibility w.r.t. $R^e_E \cup R^e_T$ of $r \cdot \delta$ at non-topmost positions is required, and also if $l \cdot \delta \to r \cdot \delta \in R_T$, since in this case $r \cdot \delta$ is irreducible by R^e_T and irreducibility w.r.t. R^e_E is required. We also have that $l \cdot \delta \to r \cdot \delta$ itself cannot reduce $r \cdot \delta$: if $l \cdot \delta \to r \cdot \delta \in R_E$ then it is not possible since $Forget(l \cdot \delta) \succ Forget(r \cdot \delta)$ and $\succ$ is irreflexive and fulfills the subterm property; if $l \cdot \delta \to r \cdot \delta \in R_T$ then, from Lemma 7.3.5, it follows that the marks in $l \cdot \delta$ and $r \cdot \delta$ do not coincide, and hence the reduction is neither possible. Finally, for every rule $l' \cdot \delta' \to r' \cdot \delta'$ generated by an instance d with $d \succ_{mod} e$, we must have that $Forget(l' \cdot \delta') \succeq Forget(l \cdot \delta)$. Now, if $l \cdot \delta \to r \cdot \delta \in R_E$ then, since $Forget(l \cdot \delta) \succ Forget(r \cdot \delta)$, we also have that $Forget(l' \cdot \delta') \succ Forget(r \cdot \delta)$, which implies that $l' \cdot \delta'$ cannot occur in $r \cdot \delta$. If $l \cdot \delta \to r \cdot \delta \in R_T$ then, by Lemma 7.3.5, $Forget(l \cdot \delta) \equiv Forget(r \cdot \delta)$ and therefore, if $Forget(l' \cdot \delta') \succ Forget(l \cdot \delta)$, we have that $l' \cdot \delta'$ cannot occur in $r \cdot \delta$ either and, if $Forget(l' \cdot \delta') \equiv Forget(l \cdot \delta)$, then $l' \cdot \delta'$ can only occur at topmost position in $r \cdot \delta$.
For property 2, by construction, if an instance e generates $l \cdot \delta \to r \cdot \delta$, then either irreducibility of $l \cdot \delta$ w.r.t. $R^e_E \cup R^e_T$ is required, or $l \cdot \delta$ is irreducible w.r.t. R^e_T and irreducibility w.r.t. R^e_E is required. Now, as before, for every rule $l' \cdot \delta' \to r' \cdot \delta'$ generated by an instance d with $d \succ_{mod} e$, we must have that $Forget(l' \cdot \delta') \succeq Forget(l \cdot \delta)$. If $Forget(l' \cdot \delta') \succ Forget(l \cdot \delta)$, then $l' \cdot \delta'$ cannot occur in $l \cdot \delta$. If, otherwise, $Forget(l' \cdot \delta') \equiv Forget(l \cdot \delta)$ then, since $l' \cdot \delta'$ must be irreducible by $R^d_E \cup R^d_T$, which contains $l \cdot \delta \to r \cdot \delta$, we have that $l' \cdot \delta' \not\equiv l \cdot \delta$, and hence $l' \cdot \delta'$ does not occur in $l \cdot \delta$ either. □

From the previous lemmas, convergence of R_{ET} follows.

Lemma 7.3.7 R_{ET} *is convergent.*

Proof. Recall that if $l \cdot \delta \to r \cdot \delta \in R_E$, then $Forget(l \cdot \delta) \succ Forget(r \cdot \delta)$ and, if $l \cdot \delta \to r \cdot \delta \in R_T$, then $l \cdot \delta \succ_m r \cdot \delta$ by Lemma 7.3.5. Now let $\succ_F$ denote the well-founded ordering defined as $l \cdot \delta \succ_F r \cdot \delta$ if, and only if, $Forget(l \cdot \delta) \succ Forget(r \cdot \delta)$, and let $\equiv_F$ denote the equivalence relation defined as $l \cdot \delta \equiv_F r \cdot \delta$ if, and only if, $Forget(l \cdot \delta) \equiv Forget(r \cdot \delta)$.
We will show that $\succ_F \cup \succ_m$ is a well-founded ordering: in every sequence $s_1 \succ_F \cup \succ_m s_2 \cdots$ there can only be finitely steps with $\succ_F$, since $s_i \cdot \gamma_i \succ_m s_j \cdot \gamma_j$ implies $s_i \cdot \gamma_i \equiv_F s_j \cdot \gamma_j$, and $\succ_F$ is well-founded and compatible with $\equiv_F$; then, from some point on, all steps must be with $\succ_m$ but, since $\succ_m$ is well-founded by Lemma 7.2.2, the sequence must be finite.
Therefore, $\succ_F \cup \succ_m$ is well-founded and, since rules in R_{ET} are oriented by $\succ_F \cup \succ_m$ and, moreover, by Lemma 7.3.6 their right-hand sides are irreducible by R_{ET} at non-topmost positions, termination follows from Lemma 3.2.4 of Chapter 3. For confluence, since R_{ET} is terminating, by Newman's lemma we only need to show that R_{ET} is locally confluent, which holds by the second property of Lemma 7.3.6. □

The same irreducibility and convergence results of R_{ET} hold as well for $Forget(R_E)$. In order to prove these, some auxiliary lemmas are given before.

Lemma 7.3.8 *Let* $s \cdot \gamma$ *be a marked subterm occurring in* R_{ET}. *If* $s \cdot \gamma$ *is marked on top, then* $Forget(s \cdot \gamma)$ *is an instance of a term in* T.

Proof. For all marked terms $(s \cdot \gamma)^x$ occurring in R_T, by construction, we have that $(s \cdot \gamma)^x$ occurs as a right-hand side of some rule in R_T. So in this case the property follows from Lemma 7.3.5. W.r.t.

the marked terms occurring in R_E, the property follows from the facts that (i) the rules in R_E are generated by equations in $gnd_T(E)$ and (ii) for every marked subterm $(s\cdot\gamma)^x$ occurring in E, we have that $Forget(s\cdot\gamma)$ is an instance of a term in T. □

Lemma 7.3.9 *Let $s\cdot\gamma$ be a marked subterm occurring in R_{ET}. If $Forget(s\cdot\gamma)$ is an instance of a term in T, then there is a rule of the form $t\cdot\theta \rightarrow (t\cdot\theta)^x$ in R_T, where $t\cdot\theta$ is not marked on top, and $s\cdot\gamma \equiv t\cdot\theta$ or $s\cdot\gamma \equiv (t\cdot\theta)^x$.*

Proof. We proceed by induction on $s\cdot\gamma$ w.r.t. $\rhd$. Let $l\cdot\delta \rightarrow r\cdot\delta$ be a rule in R_{ET} such that $l\cdot\delta \unrhd s\cdot\gamma$ or $r\cdot\delta \unrhd s\cdot\gamma$, and $Forget(s\cdot\gamma)$ is an instance of term t in T. Then there is an equation e of the form $t\theta \simeq \perp$ in T_G, for some ground substitution θ such that $t\theta \equiv Forget(s\cdot\gamma)$. Now, let $t'\cdot\theta'$ denote the normal form of $t\theta$ w.r.t. R^e_T. Recall that, by Lemma 7.3.5, we have that $Forget(s\cdot\gamma) \equiv Forget(t'\cdot\theta')$. Moreover, $t'\cdot\theta'$ is not marked on top since, by definition of $\succ_{mod}$ and Lemma 7.3.5, no equation smaller than e can generate a rule $l'\cdot\delta' \rightarrow r'\cdot\delta'$ in R_T with $Forget(l'\cdot\delta') \equiv t\theta$.
Now we show that either $s\cdot\gamma \equiv (t'\cdot\theta')^x$ or $s\cdot\gamma \equiv t'\cdot\theta'$. Assume the contrary. Then, since $Forget(s\cdot\gamma) \equiv Forget(t'\cdot\theta')$, there must be some position $p \neq \lambda$ such that either p is a marked position of $s\cdot\gamma$ and not of $t'\cdot\theta'$, or p is a marked position of $t'\cdot\theta'$ and not of $s\cdot\gamma$. Assume w.l.o.g. that p is such an innermost position.

1. If p is a marked position of $t'\cdot\theta'$ then, by construction of R_T and Lemma 7.3.5, we have that there is an equation of the form $t''\cdot\theta'' \rightarrow (t''\cdot\theta'')^x$ in R_T, where $t''\cdot\theta''$ is not marked on top and $t'\cdot\theta'|_p \equiv (t''\cdot\theta'')^x$. Now, if p is a non-marked position of $s\cdot\gamma$, this is a contradiction with Lemma 7.3.6 for $l\cdot\delta \rightarrow r\cdot\delta$.

2. If p is a marked position of $s\cdot\gamma$ then, by Lemma 7.3.8 and the induction hypothesis, there is an equation of the form $s'\cdot\gamma' \rightarrow (s'\cdot\gamma')^x$ in R_T, where $s'\cdot\gamma'$ is not marked on top and $s\cdot\gamma|_p \equiv (s'\cdot\gamma')^x$. Now, if p is a non-marked position of $t'\cdot\theta'$, then $t'\cdot\theta'|_p \equiv s'\cdot\gamma'$. But this contradicts that $t'\cdot\theta'$ is in normal form w.r.t. R^e_T.

Finally, we show that $t'\cdot\theta' \rightarrow (t'\cdot\theta')^x \in R_T$. If $t'\cdot\theta' \rightarrow (t'\cdot\theta')^x \notin R_T$, then it must be because $t'\cdot\theta'$ is reducible by R^e_E. This reduction must take place necessarily at some position $p \neq \lambda$ since, by definition of $\succ_{mod}$, no equation generating a rule in R_E and containing $t'\cdot\theta'$ can be smaller than e. But, since $s\cdot\gamma \equiv (t'\cdot\theta')^x$ or $s\cdot\gamma \equiv t'\cdot\theta'$, this is a contradiction with Lemma 7.3.6 for $l\cdot\delta \rightarrow r\cdot\delta$. □

Lemma 7.3.10 *Let $s\cdot\gamma$ be a marked subterm occurring, not as right-hand side, in some rule of R_E. Then $s\cdot\gamma$ is marked on top if, and only if, $Forget(s\cdot\gamma)$ is an instance of a term in T.*

Proof. Let $s\cdot\gamma$ be a marked term such that $l\cdot\delta \unrhd s\cdot\gamma$ or $r\cdot\delta \rhd s\cdot\gamma$ for some rule $l\cdot\delta \rightarrow r\cdot\delta$ in R_E. The left-to-right implication holds by Lemma 7.3.8. For the right-to-left implication, if $Forget(s\cdot\gamma)$ is an instance of a term in T then, by Lemma 7.3.9, there is a rule of the form $t\cdot\theta \rightarrow (t\cdot\theta)^x$ in R_T, where $t\cdot\theta$ is not marked on top and $s\cdot\gamma \equiv t\cdot\theta$ or $s\cdot\gamma \equiv (t\cdot\theta)^x$. Therefore, we necessarily have that $s\cdot\gamma \equiv (t\cdot\theta)^x$, since $s\cdot\gamma \equiv t\cdot\theta$ is a contradiction with Lemma 7.3.6 for $l\cdot\delta \rightarrow r\cdot\delta$. □

Then, from Lemmas 7.3.6 and 7.3.10, we obtain the same irreducibility result for $Forget(R_E)$ as for R_{ET}, which easily implies the convergence of $Forget(R_E)$.

Lemma 7.3.11 *For all rules $l \rightarrow r$ in $Forget(R_E)$ we have that*

1. *r is irreducible by $Forget(R_E)$ at non-topmost positions, and*

2. *l is irreducible by $Forget(R_E) \setminus \{l \to r\}$.*

Lemma 7.3.12 *$Forget(R_E)$ is convergent.*

Proof. For all rules $l \cdot \delta \to r \cdot \delta$ in R_E we have that $Forget(l \cdot \delta) \succ Forget(r \cdot \delta)$. Then, analogously to Lemma 7.3.7, we can conclude by Lemmas 7.3.11 and 3.2.4 (of Chapter 3). $\square$

Finally, we show that $Forget(R_E)^*$ is a model of $Forget(E)$ whenever the pair $\langle E, T\rangle$ is saturated with respect to $\mathcal{I}_2$. Once we have that $Forget(R_E)$ is convergent, the proof is an adaptation of the one of Theorem 3.3.5 of Chapter 3, but where the induction is done w.r.t. $\succ_i^{mul}$, taking $R = Forget(R_E)$ (see Section 7.2.1 for the definition of $\succ_i$). A contradiction is derived from the existence of a minimal w.r.t. $\succ_i^{mul}$ ground instance $e\theta$ in $gnd_T(E)$ of an equation e in E, such that $R^* \not\models Forget(e\theta)$. As usual, we have that $e\theta$ has not generated a rule in R_E. If this is because $e\theta$ is reducible by R_E, then we have an inference by paramodulation and, if $e\theta$ is reducible by R_T, then we have an inference by marking and, in both cases, a contradiction is obtained from minimality of $e\theta$ and saturatedness of $\langle E, T\rangle$.

Theorem 7.3.13 *If a pair $\langle E, T\rangle$ is saturated with respect to $\mathcal{I}_2$, then $Forget(R_E)^* \models Forget(E)$.*

Proof. We proceed by induction w.r.t. $\succ_i^{mul}$, taking $R = Forget(R_E)$. A contradiction is derived from the existence of a minimal w.r.t. $\succ_i^{mul}$ ground instance e in $gnd_T(E)$ of the form $(s \cdot \gamma)\theta \simeq (t \cdot \gamma)\theta$ of an equation $s \cdot \gamma \simeq t \cdot \gamma$ in E, such that $R^* \not\models Forget(e)$.
Since $R^* \not\models Forget(e)$, then necessarily $Forget((s \cdot \gamma)\theta) \not\equiv Forget((t \cdot \gamma)\theta)$, and hence we can assume w.l.o.g. that $Forget((s \cdot \gamma)\theta) \succ Forget((t \cdot \gamma)\theta)$. Moreover, again since $R^* \not\models Forget(e)$, the equation e has not generated any rule of R_E. We have the following possibilities:

1. There is an instance of an equation $l \cdot \delta \simeq r \cdot \delta$ in $E \cup T_{\mathrm{E}}$ that has generated a rule $l' \cdot \delta' \to r' \cdot \delta'$ in R_{ET}, which reduces $(s \cdot \gamma)\theta$, or $(t \cdot \gamma)\theta$ at a non-topmost position.

   (a) $(s \cdot \gamma)\theta|_p \equiv l' \cdot \delta'$ for some position p.

      i. p is a non-variable position of $Forget(s \cdot \gamma)$.

         A. $l \cdot \delta \simeq r \cdot \delta \in T_{\mathrm{E}}$. Then $l\delta \equiv l \in T$ and $r\delta \equiv r \equiv \bot$. Moreover $l' \cdot \delta' \to r' \cdot \delta' \in R_T$ and, by Lemma 7.3.5, we have that $r' \cdot \delta' \equiv (l' \cdot \delta')^x$, $l' \cdot \delta'$ is not marked on top, and $Forget(l' \cdot \delta')$ is a ground instance of l. Since $l' \cdot \delta'$ is not marked on top and $(s \cdot \gamma)\theta|_p \equiv l' \cdot \delta'$, then p is a non-marked position of $(s \cdot \gamma)\theta$, and hence of $s \cdot \gamma$. Therefore there is an inference by *marking*,

$$\frac{s \cdot \gamma \simeq t \cdot \gamma}{(s \cdot \gamma[(s \cdot \gamma|_p)^x]_p \simeq t \cdot \gamma)\sigma},$$

where $\sigma = mgu(Forget(s \cdot \gamma|_p), l)$, whose conclusion has an instance d of the form $(s \cdot \gamma[(s \cdot \gamma|_p)^x]_p \simeq t \cdot \gamma)\theta$. Moreover, since p is a non-marked position of $s \cdot \gamma$, we have that either $s \gg s'$, or $s \doteq s'$ and $Ran(\gamma) \succ_m^{mul} Ran(\gamma')$, where $s' \cdot \gamma'$ denotes $s \cdot \gamma[(s \cdot \gamma|_p)^x]_p$. Therefore, $s \cdot \gamma \succ_i s \cdot \gamma[(s \cdot \gamma|_p)^x]_p$ either by case (i) or (ii), and hence $e \succ_i^{mul} d$. Now, since $\langle E, T\rangle$ is saturated w.r.t. $\mathcal{I}_2$, this inference is redundant in E. Therefore, since $e \in gnd_T(E)$, we have that $Forget(gnd_T(E)^{\prec_{red}^{mul} e}) \cup Forget(gnd_T(E)^{\preceq_{red}^{mul} d}) \models Forget(d)$. Moreover, as seen before, we have $e \succ_i^{mul} d$, and $\succ_{red} \subseteq \succ_i$. Therefore since, by minimality of e, $R^* \models Forget(gnd_T(E)^{\prec_i^{mul} e})$, then also $R^* \models Forget(gnd_T(E)^{\prec_{red}^{mul} e}) \cup Forget(gnd_T(E)^{\preceq_{red}^{mul} d})$ and hence $R^* \models Forget(d)$, contradicting $R^* \not\models Forget(e)$.

B. $l\cdot\delta \simeq r\cdot\delta \in E$. Then $l'\cdot\delta' \to r'\cdot\delta' \in R_E$, with $l'\cdot\delta' \equiv (l\cdot\delta)\theta$, $r'\cdot\delta' \equiv (r\cdot\delta)\theta$, $Forget((l\cdot\delta)\theta) \succ Forget((r\cdot\delta)\theta)$, and moreover $l \succ_r r$ or $r\cdot\delta$ is marked on top. Therefore there is an inference by *paramodulation*,

$$\frac{l\cdot\delta \simeq r\cdot\delta \qquad s\cdot\gamma \simeq t\cdot\gamma}{(s\cdot\gamma[r\cdot\delta]_p \simeq t\cdot\gamma)\sigma},$$

where $\sigma = mgu(s\cdot\gamma|_p, l\cdot\delta)$, whose conclusion has an instance d of the form $(s\cdot\gamma[r\cdot\delta]_p \simeq t\cdot\gamma)\theta$. As before, we have that $e \succ_i^{mul} d$:

- If $r\cdot\delta$ is marked on top, then either p is a marked position of $s\cdot\gamma$ or else, as seen in the previous case, $(s\cdot\gamma) \succ_i (s\cdot\gamma[(s\cdot\gamma|_p)^x]_p)$. Therefore, in this particular case, $(s\cdot\gamma)\theta \succeq_i (s\cdot\gamma[(s\cdot\gamma|_p)^x]_p)\theta$. Moreover, since $(s\cdot\gamma)\theta|_p \equiv (l\cdot\delta)\theta$, we have $(s\cdot\gamma[(s\cdot\gamma|_p)^x]_p)\theta \equiv (s\cdot\gamma[(l\cdot\delta)^x]_p)\theta$. And, since $Forget((l\cdot\delta)\theta) \to_R Forget((r\cdot\delta)\theta)$, we have $(s\cdot\gamma[(l\cdot\delta)^x]_p)\theta \succ_i (s\cdot\gamma[(r\cdot\delta)^x]_p)\theta$ by case (iii). Therefore, $(s\cdot\gamma)\theta \succeq_i (s\cdot\gamma[(l\cdot\delta)^x]_p)\theta \succ_i (s\cdot\gamma[(r\cdot\delta)^x]_p)\theta$, and thus $e \succ_i^{mul} d$.
- If $l \succ_r r$, then either there is no prefix p' of p such that p' is a marked position of $s\cdot\gamma$, and then, by monotonicity and stability under substitutions of $\succ_r$, $(s\cdot\gamma)\theta \succ_i (s\cdot\gamma[r\cdot\delta]_p)\theta$ by case (i), or else $(s\cdot\gamma)\theta \succ_i (s\cdot\gamma[r\cdot\delta]_p)\theta$ by case (iii), and hence $e \succ_i^{mul} d$.

Now, since $\langle E,T\rangle$ is saturated w.r.t. $\mathcal{E}_2$, this inference is redundant in E. Then, as in case 1(a)iA, we can prove that $R^* \models Forget(d)$ which, together with the fact that $R^* \models Forget((l\cdot\delta \simeq r\cdot\delta)\theta)$, contradicts $R^* \not\models Forget(e)$.

ii. Lifting: $Forget(s\cdot\gamma)|_{p'}$ is a variable x for some prefix p' of p. Then $p = p'\cdot p''$ for some position p'' such that $x\theta|_{p''} \equiv l'\cdot\delta'$. Now let θ' be a ground substitution with the same domain as θ but where $x\theta' \equiv x\theta[r'\cdot\delta']_{p''}$ and $y\theta' \equiv y\theta$ for all other variables y. Then, by the same arguments as in case 1(a)i, we can argue that $e \succ_i^{mul} (s\cdot\gamma \simeq t\cdot\gamma)\theta'$. Now, since $l'\cdot\delta' \to r'\cdot\delta' \in R_{ET}$, we have that either $l'\cdot\delta' \to r'\cdot\delta' \in R_E$, or else $l'\cdot\delta' \to r'\cdot\delta' \in R_T$ and, by Lemma 7.3.5, $Forget(l'\cdot\delta') \equiv Forget(r'\cdot\delta')$. Therefore, since $R = Forget(R_E)$, we have that $R^* \models Forget(l'\cdot\delta' \simeq r'\cdot\delta')$. And, since $R^* \not\models Forget(e)$, we necessarily have that $R^* \not\models Forget((s\cdot\gamma \simeq t\cdot\gamma)\theta')$. We also have that $(s\cdot\gamma \simeq t\cdot\gamma)\theta' \in gnd_T(E)$, since $e \in gnd_T(E)$ and, moreover, for every marked subterm of the form $(w\cdot\varphi)^z$ in $r'\cdot\delta'$ we have that $Forget(w\cdot\varphi)$ is an instance of a term in T by Lemma 7.3.8. Therefore $(s\cdot\gamma \simeq t\cdot\gamma)\theta'$ contradicts the minimality of e.

(b) $(t\cdot\gamma)\theta|_p \equiv l'\cdot\delta'$ for some position $p \neq \lambda$.

i. p is a non-variable position of $Forget(t\cdot\gamma)$. We proceed analogously to case 1(a)i, with the only difference that the inferences do not take place at topmost position.

ii. $Forget(t\cdot\gamma)|_{p'}$ is a variable x for some prefix p' of p. We proceed analogously to case 1(a)ii.

2. $s \not\succ_r t$ and $t\cdot\gamma$ is not marked on top. In this case we can apply the *mark detection* rule

$$T \Longrightarrow T \cup \{Forget(t\cdot\gamma)\}.$$

Now, since $\langle E,T\rangle$ is saturated w.r.t. $\mathcal{E}_2$, this inference is redundant in T. Therefore $Forget(t\cdot\gamma)$ is an instance of a term t' in T. But then there is an inference by *marking*,

$$\frac{s\cdot\gamma \simeq t\cdot\gamma}{s\cdot\gamma \simeq (t\cdot\gamma)^x}.$$

Note that, if t is a variable, then t' is also a variable. Now, since $t\cdot\gamma$ is not marked on top, we clearly have that $t\cdot\gamma \succ_i (t\cdot\gamma)^x$ by case (i). From this, a contradiction is derived as in case 1(a)iA. □

7.4 Horn clauses

In this section we generalize the results of the previous section to Horn clauses. As in Chapter 3, in the inference system it is assumed that in each clause with a non-empty antecedent one of these negative equations, the one that is written underlined, has been selected.

Definition 7.4.1 *Let S be a set of Horn clauses and T be a set of terms, which satisfy the following condition: for every marked subterm $(s\cdot\gamma)^x$ occurring in S, we have that Forget$(s\cdot\gamma)$ is an instance of a term in T. Our inference system $\mathcal{H}_2$ consists of the following five inference rules and a mark detection rule:*

Paramodulation right*:*

$$\frac{\to l\cdot\delta \simeq r\cdot\delta \qquad \to s\cdot\gamma \simeq t\cdot\gamma}{(\to s\cdot\gamma[r\cdot\delta]_p \simeq t\cdot\gamma)\sigma} \qquad \textit{if}$$

1. *$\sigma = mgu(l\cdot\delta, s\cdot\gamma|_p)$, the most general unifier of $l\cdot\delta$ and $s\cdot\gamma|_p$,*
2. *Forget$(s\cdot\gamma|_p)$ is not a variable,*
3. *$l \succ_r r$ or $r\cdot\delta$ is marked on top, and*
4. *for some ground substitution θ, we have Forget$((l\cdot\delta)\sigma\theta) \succ$ Forget$((r\cdot\delta)\sigma\theta)$ and, if $p=\lambda$, then we also have Forget$((s\cdot\gamma)\sigma\theta) \succ$ Forget$((t\cdot\gamma)\sigma\theta)$.*

Paramodulation left:

$$\frac{\to l\cdot\delta \simeq r\cdot\delta \qquad \Gamma, \underline{s\cdot\gamma \simeq t\cdot\gamma} \to \Delta}{(\Gamma, s\cdot\gamma[r\cdot\delta]_p \simeq t\cdot\gamma \to \Delta)\sigma} \qquad \textit{if}$$

1. *$\sigma = mgu(l\cdot\delta, s\cdot\gamma|_p)$,*
2. *Forget$(s\cdot\gamma|_p)$ is not a variable,*
3. *$l \succ_r r$ or $r\cdot\delta$ is marked on top, and*
4. *for some ground substitution θ, we have Forget$((l\cdot\delta)\sigma\theta) \succ$ Forget$((r\cdot\delta)\sigma\theta)$ and, if $p=\lambda$, then we also have Forget$((s\cdot\gamma)\sigma\theta) \succ$ Forget$((t\cdot\gamma)\sigma\theta)$.*

Marking right:

$$\frac{\to s\cdot\gamma[u\cdot\delta]_p \simeq t\cdot\gamma}{(\to s\cdot\gamma[(u\cdot\delta)^x]_p \simeq t\cdot\gamma)\sigma} \qquad \textit{if}$$

1. *$\sigma = mgu(\textit{Forget}(u\cdot\delta), u')$ for some term u' in T,*
2. *u is not a variable, or u' is a variable and $p=\lambda$, and*
3. *x is a fresh variable.*

Marking left:

$$\frac{\Gamma, \underline{\underline{s\cdot\gamma[u\cdot\delta]_p \simeq t\cdot\gamma}} \to \Delta}{(\Gamma, s\cdot\gamma[(u\cdot\delta)^x]_p \simeq t\cdot\gamma \to \Delta)\sigma} \qquad \textit{if}$$

1. $\sigma = mgu(Forget(u\cdot\delta), u')$ *for some term* u' *in* T,
2. u *is not a variable,*
3. x *is a fresh variable, and*
4. *if* $p = \lambda$ *then, for some substitution* θ, *we have that* $Forget((s\cdot\gamma)\sigma\theta) \succ Forget((t\cdot\gamma)\sigma\theta)$.

Equality resolution:

$$\frac{\Gamma, \underline{\underline{s\cdot\gamma \simeq t\cdot\gamma}} \to \Delta}{(\Gamma \to \Delta)\sigma} \qquad \textit{if} \quad \sigma = mgu(s\cdot\gamma, t\cdot\gamma).$$

Mark detection:

$$T \Longrightarrow T \cup \{Forget(r\cdot\delta)\} \qquad \textit{if}$$

1. $\to l\cdot\delta \simeq r\cdot\delta \in S$,
2. $l \not\succ_r r$ *and* $r\cdot\delta$ *is not marked on top, and*
3. *for some ground substitution* θ, *we have that* $Forget((l\cdot\delta)\theta) \succ Forget((r\cdot\delta)\theta)$.

Definition 7.4.2 *Let S be a set of Horn clauses. By E_S we denote the set of equations occurring as positive unit clauses in S, that is, $E_S = \{s\cdot\gamma \simeq t\cdot\gamma \mid \to s\cdot\gamma \simeq t\cdot\gamma \in S\}$, and we denote by R_{ST} the set of rules generated by E_S and T as in Definition 7.3.4 of Section 7.3.2 (i.e., $R_{ST} = R_{E_ST}$). Also, following the notation of Definition 7.3.4, we denote by R_S and R_T the set of rules generated by all equations in $gnd_T(E_S)$ and T_{G}, respectively.*

Let us recall from Chapter 3 the following multiset extensions that lift orderings on terms to orderings on equations and clauses: Let C be a clause and let $emul(s\cdot\gamma \simeq t\cdot\gamma)$ be $\{s\cdot\gamma, t\cdot\gamma\}$ if $s\cdot\gamma \simeq t\cdot\gamma$ is a positive equation in C, and $\{s\cdot\gamma, s\cdot\gamma, t\cdot\gamma, t\cdot\gamma\}$ if it is negative. Then, if $\succ$ is an ordering on marked terms, we define the ordering $\succ^e$ on marked equations by $e \succ^e e'$ if $emul(e) \succ^{mul} emul(e')$; similarly, $\succ^c$ on clauses is defined by $C \succ^c D$ if $mse(C)\ (\succ^{mul})^{mul}\ mse(D)$, where $mse(C)$ is the multiset of all $emul(e)$ for occurrences e of equations in C. (We analogously lift every equivalence relation $\sim$ on marked terms to an equivalence relation $\sim^e$ on equations and $\sim^c$ on clauses.)

Then the notions of redundancy and saturation are directly obtained from the ones given in Section 7.3.1 by using Horn clauses instead of equations and the inference system $\mathcal{H}_2$ instead of $\mathcal{E}_2$, and lifting the orderings adequately.

Using Lemmas 7.3.10, 7.3.11 and 7.3.12, and Theorem 7.3.13 applied to the rewrite model obtained from the positive unit clauses, the following completeness theorem is obtained in the usual way.

Theorem 7.4.3 (Refutation completeness of $\mathcal{H}_2$ for Horn clauses) *Let S be a set of Horn clauses and T a set of terms such that the pair $\langle S, T\rangle$ is saturated with respect to $\mathcal{H}_2$. Then $\square \in S$ if, and only if, $Forget(S)$ is unsatisfiable.*

Proof. The left-to-right implication is trivial. For the other one, as in Theorem 7.3.13, we proceed by induction w.r.t. $\succ_i^c$, taking $R = Forget(R_S)$.
Again it is proved that R^* is a model of $Forget(S)$ by deriving a contradiction from the existence of a minimal w.r.t. $\succ_i^c$ ground instance $C\theta$ in $gnd_T(S)$ of a clause C in S such that $R^* \not\models Forget(C\theta)$. Since $\langle S,T\rangle$ is saturated w.r.t. $\mathcal{H}_2$, then $\langle E_S,T\rangle$ is saturated w.r.t. $\mathcal{E}_2$ and, by Theorem 7.3.13, we have that $R^* \models Forget(E_S)$. Therefore C cannot be a positive unit clause. Two cases have to be considered:

1. C is a clause $\Gamma, \underline{s\cdot\gamma \simeq t\cdot\gamma} \rightarrow \Delta$, where $Forget((s\cdot\gamma)\theta) \equiv Forget((t\cdot\gamma)\theta)$. Then there is an inference by *equality resolution*,

$$\frac{\Gamma, \underline{s\cdot\gamma \simeq t\cdot\gamma} \rightarrow \Delta}{(\Gamma \rightarrow \Delta)\sigma},$$

   where $\sigma = mgu(s\cdot\gamma, t\cdot\gamma)$. Let D denote the clause $(\Gamma \rightarrow \Delta)\sigma$. Now observe that $C\theta \succ_i^c D\theta$. From this fact and from the saturatedness assumption, a contradiction is derived as in Theorem 7.3.13: Since $\langle S,T\rangle$ is saturated w.r.t. $\mathcal{H}_2$, this inference is redundant in S. Therefore, since $C\theta \in gnd_T(S)$, we have that $Forget(gnd_T(S)^{\prec_{red}^{mul} C\theta}) \cup Forget(gnd_T(S)^{\preceq_{red}^{c} D\theta}) \models Forget(D\theta)$. Moreover, by minimality of $C\theta$, we have that $R^* \models Forget(gnd_T(S)^{\prec_i^c C\theta})$. But then, since $C\theta \succ_i^c D\theta$ and $\succ_{red}^c \subseteq \succ_i^c$, we have that $R^* \models Forget(D\theta)$, contradicting $R^* \not\models Forget(C\theta)$.

2. C is a clause $\Gamma, \underline{s\cdot\gamma \simeq t\cdot\gamma} \rightarrow \Delta$ where, w.l.o.g., $Forget((s\cdot\gamma)\theta) \succ Forget((t\cdot\gamma)\theta)$. Let s' denote $Forget((s\cdot\gamma)\theta)$ and t' denote $Forget((t\cdot\gamma)\theta)$. Then, since $R^* \not\models Forget(C\theta)$, we have that $R^* \models s' \simeq t'$ and, since R is convergent by Lemma 7.3.12, there must be a rewrite proof of $s' \simeq t'$ with R, that is, s' and t' must rewrite into the same normal form with R. This implies that either t' is reducible at a non-topmost position or else s' is reducible. (Note that it cannot be the case that the only possible reduction step on $s' \simeq t'$ is at the topmost position of t'. By such a step, a new term t'' is obtained with $t' \succ t''$ and t'' again irreducible at non-topmost positions by Lemma 7.3.11; since $s' \succ t'$, such a sequence of topmost steps on t' can never produce s'). We consider the case where s' is reducible. (The other one is analogous.) Then there is an instance of an equation $l\cdot\delta \simeq r\cdot\delta$ in E_S that has generated a rule $(l\cdot\delta)\theta \rightarrow (r\cdot\delta)\theta$ in R_S, such that $Forget((s\cdot\gamma)\theta)|_p \equiv Forget((l\cdot\delta)\theta)$ for some position p. We have the following possibilities:

   (a) $(s\cdot\gamma)\theta|_p \equiv (l\cdot\delta)\theta$. Then either the lifting argument applies like in Theorem 7.3.13, or else there is an inference by *paramodulation left*,

$$\frac{\rightarrow l\cdot\delta \simeq r\cdot\delta \qquad \Gamma, \underline{s\cdot\gamma \simeq t\cdot\gamma} \rightarrow \Delta}{(\Gamma, s\cdot\gamma[r\cdot\delta]_p \simeq t\cdot\gamma \rightarrow \Delta)\sigma},$$

   which leads to a contradiction analogously to case 1(a)iB of Theorem 7.3.13.

   (b) $(s\cdot\gamma)\theta|_p \not\equiv (l\cdot\delta)\theta$. Then, since $Forget((s\cdot\gamma)\theta)|_p \equiv Forget((l\cdot\delta)\theta)$, there must be some positions q and p' such that $q = p\cdot p'$ and $Forget((s\cdot\gamma)\theta)|_q \equiv Forget((l\cdot\delta)\theta)|_{p'}$, and either q is a marked position of $(s\cdot\gamma)\theta$ and p' is a non-marked position of $(l\cdot\delta)\theta$, or q is a non-marked position of $(s\cdot\gamma)\theta$ and p' is a marked position of $(l\cdot\delta)\theta$.

      i. If q is a marked position of $(s\cdot\gamma)\theta$ then, since $C\theta \in gnd_T(S)$ and, moreover, for every marked term of the form $(l'\cdot\delta')^x$ in occurring in S we have that $Forget(l'\cdot\delta')$ is an instance of a term in T, we must have that $Forget((s\cdot\gamma)\theta|_q)$ is an instance of a term t in T. But then also $Forget((l\cdot\delta)\theta|_{p'})$ is an instance of t and, therefore, by Lemma 7.3.10 we have that $(l\cdot\delta)\theta|_{p'}$ is marked on top, contradicting that p' is a non-marked position of $(l\cdot\delta)\theta$.

ii. If q is a non-marked position of $(s \cdot \gamma)\theta$ and p' is a marked position of $(l \cdot \delta)\theta$ then,

A. if $Forget(s \cdot \gamma)|_{q'}$ is a variable x for some prefix q' of q, we have that $q = q' \cdot q''$ for some position q'' such that q'' is a non-marked position of $x\theta$ and $Forget(x\theta)|_{q''} \equiv Forget((l \cdot \delta)\theta)|_{p'}$. Now let θ' be a ground substitution with the same domain as θ but where $x\theta' \equiv x\theta[(x\theta|_{q''})^z]_{q''}$ and $y\theta' \equiv y\theta$ for all other variables y. We have that $C\theta \succ_i^c C\theta'$ and $R^* \not\models Forget(C\theta')$. Moreover, we have that $C\theta' \in gnd_T(S)$, since $C\theta \in gnd_T(S)$ and $Forget((l \cdot \delta)\theta|_{p'})$ is an instance of a term in T by Lemma 7.3.8. Therefore $C\theta'$ contradicts the minimality of $C\theta$.

B. Otherwise, there is an inference by *marking left*,

$$\frac{\Gamma, \underline{s \cdot \gamma \simeq t \cdot \gamma} \rightarrow \Delta}{(\Gamma, s \cdot \gamma[(s \cdot \gamma|_q)^x]_p \simeq t \cdot \gamma \rightarrow \Delta)\sigma},$$

where $\sigma = mgu(Forget(s \cdot \gamma|_q), t)$, for the term t in T such that $Forget((l \cdot \delta)\theta|_{p'})$ is an instance of t, which leads to a contradiction analogously to case 1(a)iA of Theorem 7.3.13.

□

7.4.1 Derivations and redundancy of clauses

We have showed that if an unsatisfiable set of (Horn) clauses is saturated, then it must contain the empty clause. Saturatedness has been defined in terms of redundancy of inferences. Here we address the problem of how to compute such saturated sets. Usually, a saturation procedure is modeled by means of a *derivation*, a sequence of sets of clauses where each set can be obtained from the previous either by adding a logical consequence or by removing some redundant clause (see Section 1.2.2 of this thesis and [NR01]). Here, a ground clause C is defined to be *redundant* in a set of ground clauses G if $Forget(G^{\prec_{red}^c C}) \models Forget(C)$. Then a clause C is *redundant* in a set of clauses S if, for all sets of terms M and ground substitutions θ with marks in M, $C\theta$ is redundant in $gnd_M(S)$.

Since we are dealing with a set of clauses S and a set of terms T, the notion of derivation is adapted as follows: a *derivation* is a (possibly infinite) sequence of pairs of sets of clauses and terms $\langle S_0, T_0\rangle, \langle S_1, T_1\rangle, \ldots$ where for each $\langle S_{i+1}, T_{i+1}\rangle$ we have either

1. $T_{i+1} = T_i$ and
   (a) $S_{i+1} = S_i \cup \{C\}$, for some C such that $Forget(S_i) \models Forget(C)$, or
   (b) $S_{i+1} = S_i \setminus \{C\}$, for some C which is redundant in S_i, or
2. $S_{i+1} = S_i$ and
   (a) $T_{i+1} = T_i \cup \{t\}$ for some term t, or
   (b) $T_{i+1} = T_i \setminus \{t\}$ where t is an instance of a term in $T_i \setminus \{t\}$.

The abstract redundancy notions for inferences presented in Section 7.3.1 cover the so-called *forward* redundancy elimination techniques in which, for example, the conclusion of an inference needs not be stored if it follows from smaller clauses. On the other hand, redundancy for clauses is intended to cover the so-called *backward* redundancy elimination techniques in which, for example, an existing clause can be deleted if it follows from smaller ones, including newer clauses that have been generated later on. It is well-known that simplification by rewriting fits into the notion of derivation [NR01]. As shown in the following section, we can add the simplified clause and then remove the original one, which has become redundant.

Clauses belonging, from some i on, to all S_k with $k > i$, are called *persistent*. The set of persistent clauses S_∞ is formally defined as $S_\infty = \cup_i \cap_{k>i} S_k$. Analogously, the set of persistent terms is defined as $T_\infty = \cup_i \cap_{k>i} T_k$. The usual property that states that all non-persistent clauses occurring in a derivation are redundant w.r.t. the persistent ones also holds here.

Lemma 7.4.4 *Let $\langle S_0, T_0\rangle, \langle S_1, T_1\rangle, \ldots$ be a derivation. Then, for every term t in $\cup_i T_i$, we have that t is an instance of a term in T_∞.*

Proof. Let t be a term in some T_i. By induction on the derivation, it follows that, for all T_j with $j > i$, we have that t is an instance of a term in T_j. Therefore t is also an instance of a term in T_∞. □

Lemma 7.4.5 *Let $\langle S_0, T_0\rangle, \langle S_1, T_1\rangle, \ldots$ be a derivation and let C be a clause in $(\cup_i S_i) \setminus S_\infty$. Then C is redundant in S_∞.*

Proof. We need to prove that, for all sets of terms M and ground substitutions θ with marks in M, $C\theta$ is redundant in $gnd_M(S_\infty)$. For this, we assume the contrary for some set of terms N and some substitution σ with marks in N, and derive a contradiction.
Since $C \in (\cup_i S_i) \setminus S_\infty$, there is some S_j such that C is redundant in S_j. Then, for all sets of terms M and substitutions θ with marks in M, $C\theta$ is redundant in $gnd_M(S_j)$, and hence in $gnd_M(\cup_i S_i)$. Therefore, in particular, $C\sigma$ is redundant in $gnd_N(\cup_i S_i)$.
Now let G be the minimal (w.r.t. $(\succ^c_{red})^{mul}$) subset of $gnd_N(\cup_i S_i)$ such that $C\sigma$ is redundant in G. Since $gnd_N(S_\infty) \not\subseteq G$ (because $C\sigma$ is not redundant in $gnd_N(S_\infty)$), there is some instance of the form $D\sigma'$ in G for some clause D which is not persistent, and hence redundant in some S_k. But then $Forget(G') \models Forget(D\sigma')$ for some $G' \subseteq gnd_N(S_k)^{\prec^c_{red} D\sigma'}$. Therefore, since $gnd_N(S_k)^{\prec^c_{red} D\sigma'} \subseteq gnd_N(\cup_i S_i)^{\prec^c_{red} D\sigma'}$, $C\sigma$ is redundant as well in the smaller subset $G' \cup G \setminus \{D\sigma'\}$, contradicting the minimality of G. □

In order to obtain a saturated set in the limit of a derivation, some notion of *fairness* of the derivation is required. We will use the following: a derivation $\langle S_0, T_0\rangle, \langle S_1, T_1\rangle, \ldots$ is *fair* with respect to an inference system *Inf* if (i) every inference using clauses in S_∞ and terms in T_∞ is redundant in some S_j and (ii) every mark detection rule using clauses in S_∞ is redundant in some T_j.

Lemma 7.4.6 *If $\langle S_0, T_0\rangle, \langle S_1, T_1\rangle, \ldots$ is a fair derivation with respect to an inference system Inf, then $\langle S_\infty, T_\infty\rangle$ is saturated with respect to Inf.*

Proof. By fairness, every inference π using clauses in S_∞ and terms in T_∞ is redundant in some S_j. Then, by Lemma 7.4.5, π is also redundant in S_∞. By fairness again, every mark detection rule using clauses in S_∞ is redundant in some T_j and hence, by Lemma 7.4.4, so is in T_∞. Therefore $\langle S_\infty, T_\infty\rangle$ is saturated. □

Then, from Lemma 7.4.6 and Theorem 7.4.3 it follows the refutation completeness of any theorem proving procedure which computes a fair derivation w.r.t. the inference system $\mathcal{H}_2$. Note that, in order to ensure fairness in practice, in the inference systems presented in this chapter it suffices that no inference (or mark detection rule) with persistent premises is postponed infinitely many times, since adding the conclusion always makes the inference (or mark detection rule) redundant. For example, this could be achieved by periodically considering all inferences with the clause whose *size* (in number of symbols) is smallest. If a certain clause persists then it will eventually be considered, since there are only finitely many clauses with smaller size.

In principle, it would be desirable to postpone the use of *marking* inference rules as much as possible, since marking of terms generally will not allow much demodulation. However, our redundancy notions do not preserve refutation completeness if blockings are not introduced at all, as shown in Example 7.4.8 of the following section.

7.4.2 Practical notions of redundancy

In this section we provide sufficient conditions for proving redundancy of inferences, which include simplification by rewriting for those rules included in the reduction ordering $\succ_r$.

Let S be a set of Horn clauses. By S_r we denote the set of rules $l\cdot\delta \rightarrow r\cdot\delta$ such that the clause $\rightarrow l\cdot\delta \simeq r\cdot\delta$ is in S and $l \succ_r r$. An inference with premises $C_1,\dots,C_n$ and conclusion D can be shown redundant in S in two steps:

1. First, D is simplified by rewriting it into D' using marked rules $l\cdot\delta \rightarrow r\cdot\delta$ in S_r. In order to ensure that we are only using smaller (w.r.t. $\succ^c_{red}$) instances of clauses in S, we have the following sufficient conditions. If D is of the form $\Gamma \rightarrow s \simeq t$, then (i) we can apply any rewriting step on Γ, but only at positions of the skeletons; (ii) we can apply any rewriting step at a non-top position of the skeleton of s (resp. t) obtaining s' (resp. t'), and after that we can apply any rewriting step at any position of the skeleton of s' (resp. t'); (iii) we can apply a rewriting step at top position of s (resp. t) obtaining s' (resp. t') if the instance of the rule applied is smaller (w.r.t. $\succ^c_{red}$) than C_n or smaller than or equal to D, and after that we can apply any rewriting step at any position of the skeleton of s' (resp. t'). Note that in all these cases we can show that for any ground instance of the inference, the ground instances of the applied rules are either smaller (w.r.t. $\succ^c_{red}$) than the instance of C_n or smaller than or equal to the instance of D.

2. Second, we check whether $D_1,\dots,D_n \models D'$ for some instances $D_1,\dots,D_n$ of clauses in S such that $D'(\trianglerighteq_{sk} \cup \succ_m)^c D_i$ for all i in $1\dots n$, where $s\cdot\gamma \trianglerighteq_{sk} t\cdot\delta$ if $t\cdot\delta$ occurs in $s\cdot\gamma$ at some position of its skeleton. This includes, among other cases, the case where D' is in S, the case where there is a version of D' with additional marks in S, and the case where D' is a tautology. If this situation does not apply, then we can add D' to S to make the inference redundant.

Example 7.4.7 *In Example 7.1.1 of the introduction we showed how, by an ordered paramodulation inference with* $\underline{f(f(x))} \simeq f(g(f(x)))$ *on* $f(f(y)) \simeq f(g(\underline{f(y)}))$*, we got the equation* $f(f(f(x))) \simeq f(g(f(g(f(x)))))$*, i.e.,* $f(f(f(z))) \simeq f(g(f(g(f(z)))))$*. In this case,* $f(f(f(z)))$ *could be rewritten in two steps with* $f(f(x)) \rightarrow f(g(f(x)))$ *into* $f(g(f(g(f(z)))))$*, and hence the conclusion of the inference could be simplified into a tautology. Observe that the first step is applied at a non-top position, obtaining* $f(f(g(f(z)))$*, and then the second step can be applied on top obtaining* $f(g(f(g(f(z)))))$*.*

All previous results can be adapted in the usual way for showing redundancy of a clause C by using only smaller clauses w.r.t. $\prec^c_{red}$ than C.

In the following example, we show how demodulation w.r.t. $\succ_r$ and redundancy with smaller equations w.r.t. the subterm relation can cause incompleteness if no blockings are introduced at all. This thus shows the need for some control or, in other words, that our blockings are not merely a technicality to prove completeness.

Example 7.4.8 *Let $\succ_r$ be a reduction ordering such that*

$$f(x,x) \succ_r f(a,b) \text{ and}$$
$$f(x,x) \succ_r f(b,a).$$

Then necessarily

$$a \not\succ_r b \text{ and}$$
$$b \not\succ_r a.$$

Let S denote the following inconsistent set of Horn clauses:

$$\begin{array}{ll}
1) & \rightarrow a \simeq b \\
2) & \rightarrow f(x,x) \simeq f(a,b) \\
3) & \rightarrow f(x,x) \simeq f(b,a) \\
4) & \rightarrow f(x,x) \simeq f(b,b) \\
5) & \rightarrow g(f(a,b)) \simeq g(g(f(b,a))) \\
6) & x \simeq g(x) \rightarrow
\end{array}$$

First of all, we will show how the empty clause can be derived under our inference system (which introduces the convenient blockings). Let $\succ$ be a west ordering which is an extension of $\succ_r$ and that includes

$$g(g(f(a,b))) \succ g(f(a,b)) \succ g(g(f(b,a))) \succ g(f(b,a))$$
$$\succ f(a,a) \succ f(b,b) \succ f(a,b) \succ f(b,a) \succ a \succ b.$$

Initially, the set T of terms that are going to be marked is empty. Then, since $a \succ b$ and $a \not\succ_r b$, the mark detection *rule gives us $T = \{b\}$. After this, an inference by* marking right *into 1 gives us*

$$7) \quad \rightarrow a \simeq b^{x_0}$$

Now, an inference by paramodulation right *with 7 into 5 is possible (recall that an inference by paramodulation right with 1 into 5 is not possible since $a \simeq b$ does not fulfill $a \succ_r b$ and b is not marked on top), giving*

$$8) \quad \rightarrow g(f(b^{x_1},b)) \simeq g(g(f(b,a)))$$

A new inference by paramodulation right *with 7 into 8 gives us*

$$9) \quad \rightarrow g(f(b^{x_1},b)) \simeq g(g(f(b,b^{x_4})))$$

And, with two more inferences by marking right*, we get*

$$\begin{array}{ll}
10) & \rightarrow g(f(b^{x_1},b^{x_2})) \simeq g(g(f(b,b^{x_4}))) \\
11) & \rightarrow g(f(b^{x_1},b^{x_2})) \simeq g(g(f(b^{x_3},b^{x_4})))
\end{array}$$

It is worth noting that clause 1 follows from the smaller clause 7, clauses 9 and 10 follow from the smaller clause 11, and clause 8 follows from the smaller clauses 7 and 11. Hence, using redundancy of clauses *(c.f. Section 7.4.1), clauses 1, 8, 9 and 10 could be deleted.*

Now recall that $\succ$ fulfills the subterm property. Therefore we have that $Forget(g(g(f(b^{x_3},b^{x_4})))) \succ Forget(g(f(b^{x_1},b^{x_2})))$, and hence an inference by paramodulation left *with 11 into 6 gives us*

$$12) \quad g(f(b^{x_1},b^{x_2})) \simeq g(f(b^{x_1},b^{x_2})) \rightarrow$$

And finally, the empty clause is obtained with an inference by equality resolution.

Now let us show that, if the blockings were not introduced (in other words, if inferences by paramodulation right *with 1 instead of 7 were allowed), then refutation completeness would be lost. If an inference by* paramodulation right *with 1 into 5 was possible, then it would give either*

$$\rightarrow g(f(b,b)) \simeq g(g(f(b,a)))$$

or

$$\rightarrow g(f(a,b)) \simeq g(g(f(b,b))),$$

which could be simplified back into 5 respectively with 2 and 3, which are included in the reduction ordering. Moreover, inferences by paramodulation right *with 1 into 2 and 3 give us 4. And inferences by* paramodulation right *between 2, 3 and 4 give us already existing clauses or clauses like*

$$\rightarrow f(a,b) \simeq f(b,a),$$

which follow from the smaller (w.r.t. the subterm relation) clause 1. Finally, since $g(f(a,b)) \succ g(g(f(b,a)))$, an inference by paramodulation left *with 5 into 6 gives us*

$$7) \quad f(a,b) \simeq g(g(f(b,a))) \rightarrow$$

but, since $f(a,b) \simeq g(g(f(b,a)))$ does not follow from the positive equations, further inferences on 7 will not lead to the empty clause. Therefore the set S can be saturated without the empty clause being derived. □

It is not difficult to see that useful methods like *subsumption* of clauses can be covered as usual, for instance, by more general notions of redundancy for clauses like *non-strict redundancy*, together with the notion of *ground fairness* [NR01] (see also [BG94b] for an alternative way of capturing subsumption).

7.5 Knuth-Bendix completion

In this section, as a side effect of our results, like in Chapter 4 we obtain a Knuth-Bendix completion procedure, but in this case compatible with simplification techniques. This procedure can be used for finding, whenever it exists, a convergent TRS for a given set of equations and a (possibly non-totalizable) reduction ordering.

Let us recall from Chapter 4 that if E denotes a set of equations and $\succ_r$ denotes a (possibly non-totalizable) reduction ordering on $T(\mathcal{F},\mathcal{X})$, then a *convergent TRS R for E and $\succ_r$* is a convergent TRS, logically equivalent to E, and such that $l \succ_r r$ for all its rules $l \rightarrow r$. The problem we deal with is finding a convergent TRS for the given E and $\succ_r$ whenever it exists, and find it in finite time if it is finite. Note that being a reduction ordering is the minimal possible requirement on $\succ_r$, since a TRS terminates if, and only if, it is contained in a reduction ordering.

As mentioned in Chapter 4, it is not difficult to devise a procedure of theoretical nature for doing this by enumerating all equational consequences of E. However, finding practically useful procedures remained as an open problem for a long time. Devie showed that for left- and right *linear* E (i.e., no variable occurs more than once in a side of an equation) standard Knuth-Bendix completion finds R [Dev90]. But the first procedure for the general problem, not relying on the enumeration of all equational consequences, is the one we presented in Chapter 4.

Here we improve that method by making it compatible with powerful redundancy notions, like simplification by rewriting. The proof is a straightforward adaptation of the one in Chapter 4 since, within the present results, an interreduced convergent TRS is generated in the limit and, hence, the same proof techniques can be applied.

In the following, R denotes the canonical TRS for E and $\succ_r$ (we assume that R exists), R_g denotes the canonical[3] TRS for $gnd(R)$, the set of all ground instances of rules of R, and $\succ$ denotes a west ordering including $\succ_r$. We will use the same inference system as in Definition 7.3.1, but where the *paramodulation* inference is relaxed by changing condition 4 as follows:

4. for some ground substitution θ, we have $Forget((l\cdot\delta)\sigma\theta) \succ Forget((r\cdot\delta)\sigma\theta)$.

With this change, now the ordered paramodulation rule is also applied on top of the small sides. This is necessary in order to obtain an interreduced ground TRS for the model. We denote the resulting inference system by $\mathcal{K}_2$.

By R_E, R_T and R_{ET} we denote the TRSs generated from a set of marked equations E and a set of terms T as in Definition 7.3.4, but where in case 1d total irreducibility is required instead of just irreducibility at non-topmost positions. Then it follows that $Forget(R_E)$ is interreduced and included in $\succ$.

Lemma 7.5.1 *For all rules $l \to r$ in $Forget(R_E)$ we have that*

1. *r is irreducible by $Forget(R_E)$, and*
2. *l is irreducible by $Forget(R_E) \setminus \{l \to r\}$.*

Moreover, since an equation $s \simeq t$ can be identified with a clause $\to s \simeq t$, the dynamic redundancy notions and lemmas for clauses of Section 7.4.1 hold for equations as well.

Theorem 7.5.2 *Let $\langle E_0, T_0\rangle, \langle E_1, T_1\rangle, \ldots$ be a fair derivation with respect to $\mathcal{K}_2$, where E_0 is the given set of equations E, and let R_{E_∞} denote the TRS R_E obtained taking $E = E_\infty$ and $T = T_\infty$. Then $Forget(R_{E_\infty})^* \models Forget(gnd(E_\infty))$.*

Proof. By Lemma 7.4.6 we have that $\langle E_\infty, T_\infty\rangle$ is saturated with respect to $\mathcal{K}_2$. Then we proceed analogously to the proof of Theorem 7.3.13, with the only difference that inferences at topmost positions of small sides of equations also are considered. □

Lemma 7.5.3 *Let $\langle E_0, T_0\rangle, \langle E_1, T_1\rangle, \ldots$ be a fair derivation with respect to $\mathcal{K}_2$, where E_0 is the given set of equations E. Then $R_g = Forget(R_{E_\infty})$.*

Proof. By Theorem 7.5.2, we have $Forget(R_{E_\infty})^* \models Forget(gnd(E_\infty))$, and hence $Forget(R_{E_\infty})^* \models Forget(E_\infty)$. Then, by Lemma 7.4.5, we have $Forget(R_{E_\infty})^* \models E$. Therefore $E^* = Forget(R_{E_\infty})^*$, since all rules in $Forget(R_{E_\infty})$ are logical consequences of E. Furthermore, we clearly have $E^* = R^* = R_g^*$, and hence $R_g^* = Forget(R_{E_\infty})^*$. We also have $R_g \subseteq \succ_r \subseteq \succ$, and $Forget(R_{E_\infty}) \subseteq \succ$. Therefore, since both are interreduced we conclude by Lemma 4.3.1 of Chapter 4 that $R_g = Forget(R_{E_\infty})$. □

[3] Recall from Chapter 4 that for every convergent TRS its unique *canonical* version can be obtained by *interreducing* it by (i) normalizing all right-hand sides and (ii) removing all rules whose left-hand sides are reducible by other rules.

Then, by the same arguments as in Chapter 4 we get the following result.

Theorem 7.5.4 *Let $\langle E_0, T_0\rangle, \langle E_1, T_1\rangle, \ldots$ be a fair derivation with respect to $\mathcal{K}_2$, where E_0 is the given set of equations E. Then $Forget(E_\infty) \supseteq R$.*

Proof. W.l.o.g. assume there are sufficiently many new constants in $\mathcal{F}$ that do not occur in E or R. Let $l \to r$ be an arbitrary rule in R. We prove that $l \simeq r \in Forget(E_\infty)$. Let σ be the ground substitution replacing each variable with a distinct new constant. Then $l\sigma \to r\sigma$ is in $gnd(R)$ and also in R_g: since R is interreduced, r is irreducible by R (and hence $r\sigma$ by $gnd(R)$), and l is irreducible by $R \setminus \{l \to r\}$ (and hence $l\sigma$ by $gnd(R) \setminus \{l\sigma \to r\sigma\}$). If $l\sigma \to r\sigma$ is in R_g and by the previous lemma $R_g = Forget(R_{E_\infty})$, then $l\sigma \to r\sigma$ is in $Forget(R_{E_\infty})$. Since the new constants do not occur in E, if $l\sigma \to r\sigma \in Forget(R_{E_\infty})$, then some $l' \simeq r'$ is in $Forget(E_\infty)$, and $l\sigma \equiv l'\theta\sigma$ and $r\sigma \equiv r'\theta\sigma$ for some θ. We conclude by showing that θ is the identity substitution: there is no equation $l' \simeq r'$ in $Forget(E_\infty)$ that strictly subsumes a rule in R; otherwise, the rewrite proof by R of $l' \simeq r'$ would apply some rule different from $l \to r$, which would then also reduce $l \to r$, contradicting $l \to r \in R$. □

Therefore, if R is finite, it will be contained in $Forget(E_j)$ for some E_j of the derivation.

7.6 General clauses

The extension of the presented calculus to general first order clauses is not straightforward since, roughly, the orderings must be carefully extended or the inference system be adapted so that, in the completeness proof, the conclusion of an inference be always smaller than the maximal premise (even in the case of a positive strategy, paramodulation with non-unit clauses is clearly necessary and, hence, some equations can be added to the conclusion of an inference, which moreover need be compared with the rightmost premise). The approach we present here consists in using almost the same ordering $\succ_i^c$ as in the Horn case, and roughly marking all terms which are "introduced" in the conclusion and are not smaller w.r.t. $\succ_{red}$ than some term in the rightmost premise. This is in some sense the natural extension, to the case of general clauses, of the strategy used under the Horn case, but, when applying paramodulation with non-unit clauses, it tends to generate many marks. At present we do not know whether these additional marks (w.r.t. the ones we already added in the Horn case) are necessary to preserve completeness.

Here we will deal, as in the Horn case, with a set of clauses S and a set of terms T, which contains the set of terms that are going to be marked. We will analogously have a *mark detection* rule which adds terms to the set T but, additionally, some terms will be added to the set T when performing inferences. An alternative to this approach could be to have additional mark detection rules (and not let inference rules add terms to T), but this would not allow us to keep the set T as reduced as possible.[4]

We extend the definition of $Marks(S)$ for a multiset of marked terms S given in Section 7.3.1, to multisets of multisets of marked terms as follows: if S is a multiset $\{S_1, \ldots, S_n\}$ of multisets of marked terms, then $Marks(S) = \bigcup_{i \in 1 \ldots n} Marks(S_i)$.

As for the Horn case, in the following inference system we consider that in each clause with a non-empty antecedent one of its negative equations, the one that is written underlined, has been selected. We also assume that all terms in the premises of an inference share a unique substitution.

[4] Technically, although now the inference rules can also add terms to the set T, a mark detection rule is still necessary for the lifting argument.

Definition 7.6.1 *Let S be a set of general first order clauses and T be a set of terms, which satisfy the following condition: for every marked subterm $(s\cdot\gamma)^x$ occurring in S, we have that $Forget(s\cdot\gamma)$ is an instance of a term in T. Our inference system I_2 consists of the following six inference rules and a mark detection rule:*

Paramodulation right:

$$\frac{\rightarrow l\cdot\gamma\simeq r\cdot\gamma,\Delta_1 \qquad \rightarrow s\cdot\gamma\simeq t\cdot\gamma,\Delta_2}{(\rightarrow s'\cdot\gamma'\simeq t\cdot\gamma,\Delta_1',\Delta_2)\sigma} \qquad \textit{if}$$

1. $\sigma = mgu(l\cdot\gamma, s\cdot\gamma|_p)$ *for some position* p,
2. $Forget(s\cdot\gamma|_p)$ *is not a variable,*
3. $Forget(s'\cdot\gamma')\equiv Forget(s\cdot\gamma[r\cdot\gamma]_p)$, *and* $Forget(\Delta_1')$ *is the same as* $Forget(\Delta_1)$,
4. *for some ground substitution* θ, *we have* $Forget((l\cdot\gamma)\sigma\theta)\succ Forget((r\cdot\gamma)\sigma\theta)$ *and* $Forget((l\cdot\gamma\simeq r\cdot\gamma)\sigma\theta)\succ^{mul} Forget(e\sigma\theta)$ *for all equations* e *in* Δ_1 *and, if* $p=\lambda$, *then* $Forget((s\cdot\gamma)\sigma\theta)\succ Forget((t\cdot\gamma)\sigma\theta)$ *and* $Forget((s\cdot\gamma\simeq t\cdot\gamma)\sigma\theta)\succ^{mul} Forget(e\sigma\theta)$ *for all equations* e *in* Δ_2,
5. $(s\cdot\gamma)\sigma\succ_{red}(s'\cdot\gamma')\sigma$, *or* $s\doteq s'$ *and* $s|_q$ *is a variable for some prefix of* q *of* p, *and*
6. *for every equation* e *in* Δ_1', $\{s\cdot\gamma, t\cdot\gamma\}\sigma\succ_{red}^{mul} e\sigma$ *or both sides of* e *are marked on top,*

where, moreover, the set of terms

$$\{Forget(u\cdot\rho)\sigma \mid u\cdot\rho\in Marks(\{s'\cdot\gamma'\}\cup\Delta_1') \text{ and } Forget(u\cdot\rho) \text{ is not an instance of a term in } T\}$$

is added to T whenever the conclusion is added to S.

Observe that, for some parts of the conclusion, it is not precisely specified what positions should be marked. For instance, one strategy satisfying condition 5 could be to take $s\cdot\gamma[r\cdot\gamma]_p$ as $s'\cdot\gamma'$ whenever $(s\cdot\gamma)\sigma\succ_{red}(s\cdot\gamma[r\cdot\gamma]_p)\sigma$ or p is below a marked position of $s\cdot\gamma$, and take $s\cdot\gamma[(r\cdot\gamma)^x]_p$, where x is a fresh variable, otherwise (which makes $(s\cdot\gamma)\sigma\succ_{red}(s'\cdot\gamma')\sigma$, since $s\gtrdot s'$); this is the strategy we took in the equational and Horn cases, for clarity reasons. One simple strategy satisfying condition 6 could be to take Δ_1' as Δ_1, but where every side of an equation which is not smaller w.r.t. $\succ_{red}$, under the substitution σ, than $(s\cdot\gamma)\sigma$ or $(t\cdot\gamma)\sigma$, has been marked on top. However, more sophisticated strategies can be considered, including the possibility of forgetting *some marks in the conclusion: as an example, if the right premise has no marks and $(s\simeq t)\sigma\succ_r^e Forget(e)\sigma$ for every equation e in Δ_1, then we can take $Forget(\Delta_1)$ as Δ_1'.*

Paramodulation left:

$$\frac{\rightarrow l\cdot\gamma\simeq r\cdot\gamma,\Delta_1 \qquad \Gamma,\underline{s\cdot\gamma\simeq t\cdot\gamma}\rightarrow\Delta_2}{(\Gamma,s'\cdot\gamma'\simeq t\cdot\gamma\rightarrow\Delta_1',\Delta_2)\sigma} \qquad \textit{if}$$

1. $\sigma = mgu(l\cdot\gamma, s\cdot\gamma|_p)$ *for some position* p,
2. $Forget(s\cdot\gamma|_p)$ *is not a variable,*
3. $Forget(s'\cdot\gamma')\equiv Forget(s\cdot\gamma[r\cdot\gamma]_p)$, *and* $Forget(\Delta_1')$ *is the same as* $Forget(\Delta_1)$,
4. *for some ground substitution* θ, *we have* $Forget((l\cdot\gamma)\sigma\theta)\succ Forget((r\cdot\gamma)\sigma\theta)$ *and* $Forget((l\cdot\gamma\simeq r\cdot\gamma)\sigma\theta)\succ^{mul} Forget(e\sigma\theta)$ *for all equations* e *in* Δ_1 *and, if* $p=\lambda$, *then* $Forget((s\cdot\gamma)\sigma\theta)\succ Forget((t\cdot\gamma)\sigma\theta)$.

5. $(s \cdot \gamma)\sigma \succ_{red} (s' \cdot \gamma')\sigma$, *or* $s \doteq s'$ *and* $s|_q$ *is a variable for some prefix of* q *of* p, *and*
6. *for every equation* e *in* Δ'_1, $\{s \cdot \gamma, s \cdot \gamma, t \cdot \gamma, t \cdot \gamma\}\sigma \succ^{mul}_{red} e\sigma$ *or both sides of* e *are marked on top,*

where, moreover, the set of terms

$$\{Forget(u \cdot \rho)\sigma \mid u \cdot \rho \in Marks(\{s' \cdot \gamma'\} \cup \Delta'_1) \text{ and } Forget(u \cdot \rho) \text{ is not an instance of a term in } T\}$$

is added to T *whenever the conclusion is added to* S.

Marking right:

$$\frac{\rightarrow s \cdot \gamma[u \cdot \delta]_p \simeq t \cdot \gamma, \Delta}{(\rightarrow s \cdot \gamma[(u \cdot \delta)^x]_p \simeq t \cdot \gamma, \Delta)\sigma} \qquad \textit{if}$$

1. $\sigma = mgu(Forget(u \cdot \delta), u')$ *for some term* u' *in* T,
2. u *is not a variable, or* u' *is a variable and* $p = \lambda$, *and*
3. x *is a fresh variable.*

Marking left:

$$\frac{\Gamma, \underline{s \cdot \gamma[u \cdot \delta]_p \simeq t \cdot \gamma} \rightarrow \Delta}{(\Gamma, s \cdot \gamma[(u \cdot \delta)^x]_p \simeq t \cdot \gamma \rightarrow \Delta)\sigma} \qquad \textit{if}$$

1. $\sigma = mgu(Forget(u \cdot \delta), u')$ *for some term* u' *in* T,
2. u *is not a variable,*
3. x *is a fresh variable, and*
4. *if* $p = \lambda$ *then, for some substitution* θ, *we have that* $Forget((s \cdot \gamma)\sigma\theta) \succ Forget((t \cdot \gamma)\sigma\theta)$.

Equality resolution:

$$\frac{\Gamma, \underline{s \cdot \gamma \simeq t \cdot \gamma} \rightarrow \Delta}{(\Gamma \rightarrow \Delta)\sigma} \qquad \textit{if} \quad \sigma = mgu(s \cdot \gamma, t \cdot \gamma).$$

Equality factoring:

$$\frac{\rightarrow l \cdot \gamma \simeq r \cdot \gamma, s \cdot \gamma \simeq t \cdot \gamma, \Delta}{(r' \cdot \gamma' \simeq t' \cdot \gamma' \rightarrow l \cdot \gamma \simeq r \cdot \gamma, \Delta)\sigma} \qquad \textit{if}$$

1. $\sigma = mgu(l \cdot \gamma, s \cdot \gamma)$,
2. $Forget(r' \cdot \gamma') \equiv Forget(r \cdot \gamma)$ *and* $Forget(t' \cdot \gamma') \equiv Forget(t \cdot \gamma)$,
3. *for some ground substitution* θ, *we have* $Forget((l \cdot \gamma)\sigma\theta) \succ Forget((r \cdot \gamma)\sigma\theta)$ *and* $Forget((l \cdot \gamma \simeq r \cdot \gamma)\sigma\theta) \succeq^{mul} Forget(e\sigma\theta)$ *for all equations* e *in* $\{s \cdot \gamma \simeq t \cdot \gamma\} \cup \Delta$, *and*
4. $(s \cdot \gamma)\sigma \succ_{red} (r' \cdot \gamma')\sigma$ *or* $r' \cdot \gamma'$ *is marked on top, and* $(s \cdot \gamma)\sigma \succ_{red} (t' \cdot \gamma')\sigma$ *or* $t' \cdot \gamma'$ *is marked on top,*

where, moreover, the set of terms

$$\{Forget(u \cdot \rho)\sigma \mid u \cdot \rho \in Marks(\{r' \cdot \gamma', t' \cdot \gamma'\}) \text{ and } Forget(u \cdot \rho) \text{ is not an instance of a term in } T\}$$

is added to T whenever the conclusion is added to S.

Mark detection:

$$T \Longrightarrow T \cup \{Forget(r \cdot \gamma)\} \qquad \text{if}$$

1. $\rightarrow l \cdot \gamma \simeq r \cdot \gamma, \Delta \in S$,
2. $l \not\succ_r r$ *and* $r \cdot \gamma$ *is not marked on top, and*
3. *for some ground substitution* θ, *we have* $Forget((l \cdot \gamma)\theta) \succ Forget((r \cdot \gamma)\theta)$ *and* $Forget((l \cdot \gamma \simeq r \cdot \gamma)\theta) \succ^{mul} Forget(e\theta)$ *for all equations* e *in* Δ.

Observe that the previous inference system preserves the invariant that for every marked subterm $(s \cdot \gamma)^x$ occurring in S, $Forget(s \cdot \gamma)$ is an instance of a term in T.

The following is a toy example to illustrate how the previous inference system works.

Example 7.6.2 *Let $\succ_r$ be a reduction ordering including*

$$g(a) \succ_r g(f(a)) \succ_r h(f(a)) \succ_r h(a),$$
$$h(a) \succ_r f(a), \text{ and } h(a) \succ_r a.$$

Observe that, on the one hand, $f(a)$ and a must be incomparable in $\succ_r$ and, on the other hand, we must have $f(a) \succ a$ in any west ordering $\succ$ including $\succ_r$.

Let S denote the following inconsistent set of general first order clauses,

$$\begin{array}{ll} 1) & \rightarrow g(f(a)) \simeq h(a), g(a) \simeq h(f(a)) \\ 2) & g(x) \simeq h(y) \rightarrow x \simeq y \\ 3) & g(x) \simeq h(x) \rightarrow \end{array}$$

and let $T = \emptyset$.

We illustrate how the empty clause can be derived from the pair $\langle S, T \rangle$ with the inference system I_2. In order to ease the reading, we sometimes underline the terms that are unified in an inference.

First of all, an inference by paramodulation left *with 1 into 2, that is with $\rightarrow g(f(a)) \simeq h(a), \underline{g(a)} \simeq h(f(a))$ into $\underline{g(x)} \simeq h(y) \rightarrow x \simeq y$ is possible, giving*

$$4) \quad h(f(a)) \simeq h(y) \rightarrow a \simeq y, g(f(a)) \simeq h(a)$$

Observe that this is a valid inference, where $\sigma = \{x \mapsto a\}$, since $g(a)$ is strictly maximal w.r.t. $\succ$ in the left premise (where $\succ$ is the west ordering including $\succ_r$), and $g(x)\sigma\theta \succ h(y)\sigma\theta$ for some ground substitution θ (take, e.g., $\theta = \{y \mapsto a\}$). Moreover, no terms needs to be marked in the conclusion, since $g(a) \succ_r h(f(a))$ (and hence $g(a) \succ_{red} h(f(a))$), and $\{g(a), h(y)\} \succ_r^{mul} \{g(f(a)), h(a)\}$ (and hence $\{g(a), h(y)\} \succ_{red}^{mul} \{g(f(a)), h(a)\}$).

Now, an inference by equality resolution *on 4 is possible, giving*

$$5) \quad \rightarrow a \simeq f(a), g(f(a)) \simeq h(a)$$

Then an inference by paramodulation left *with 5 into 2, that is with* $\rightarrow a \simeq f(a), \underline{g(f(a))} \simeq h(a)$ *into* $\underline{g(x)} \simeq h(y) \rightarrow x \simeq y$ *is possible, giving*

$$6) \quad h(a) \simeq h(y) \rightarrow f(a) \simeq y, a \simeq f(a)$$

Again, no mark needs to be added to the conclusion, since $g(f(a)) \succ_r h(a)$ *and* $\{g(f(a)), h(y)\} \succ_r^{mul} \{a, f(a)\}$.

Now, an inference by equality resolution *on 6 gives us*

$$7) \quad f(a) \simeq a, a \simeq f(a)$$

And with an inference by equality factoring *on 7 we obtain*

$$8) \quad a \simeq a \rightarrow f(a) \simeq a$$

Note that this is a valid inference, since $f(a) \succ a$ *and* $\{f(a), a\} \succeq^{mul} \{a, f(a)\}$*. Moreover, no mark is added to the conclusion since, although* $f(a) \not\succ_r a$*, we have that* $f(a) \rhd a$ *and hence* $f(a) \succ_{red} a$.

Now with an inference by equality resolution *on 8 we get*

$$9) \quad \rightarrow f(a) \simeq a$$

Then there is an inference by paramodulation right *with 9 into 1, that is with* $\rightarrow \underline{f(a)} \simeq a$ *into* $\rightarrow g(\underline{f(a)}) \simeq h(a), g(a) \simeq h(f(a))$*, giving*

$$10) \quad \rightarrow g(a^{x_1}) \simeq h(a), g(a) \simeq h(f(a))$$

Here a mark is necessary at the inference position, since neither $g(f(a)) \succ_r g(a)$ *nor* $g(f(a)) \rhd g(a)$ *(in fact, we have* $g(a) \succ_r g(f(a))$*); by introducing the mark, we obtain* $g(f(a)) \succ_{red} g(a^{x_1})$ *(since* $g(f(a)) \rhd g(x_1)$*). Recall that after this, the term a is added to the set T of terms that are going to be marked.*

A new inference by paramodulation right *with 9 into 10, that is with* $\rightarrow \underline{f(a)} \simeq a$ *into* $\rightarrow g(a^{x_1}) \simeq h(a), g(a) \simeq h(\underline{f(a)})$*, gives us*

$$11) \quad \rightarrow g(a^{x_1}) \simeq h(a), g(a) \simeq h(a)$$

In this case a mark is not necessary at the inference position, since $h(f(a)) \succ_r h(a)$.

Now, since $a \in T$*, an inference by* marking right *on 11 is possible, and we obtain*

$$12) \quad \rightarrow g(a^{x_1}) \simeq h(a), g(a^{x_2}) \simeq h(a)$$

After this, $g(a^{x_1})$ *and* $g(a^{x_2})$ *unify, and hence an inference by* equality factoring *on 12 gives us*

$$13) \quad h(a) \simeq h(a) \rightarrow g(a^{x_1}) \simeq h(a)$$

Then, with an inference by equality resolution *on 13 we get*

$$14) \quad \rightarrow g(a^{x_1}) \simeq h(a)$$

And an inference by paramodulation left *with 14 into 3, that is with* $\rightarrow \underline{g(a^{x_1})} \simeq h(a)$ *into* $g(x) \simeq h(x) \rightarrow$*, gives us*

$$15) \quad h(a)^{x_2} \simeq h(a^{x_1}) \rightarrow$$

Observe that a mark is introduced at the inference position in the conclusion. This is because we are assuming that $g(x_1) \not\succ_r h(a)$ for all x_1. However, in the case that, for example, we had $g(x) \succ_r h(x)$ for all x, $h(a^{x_1}) \simeq h(a^{x_1}) \rightarrow$ would also be a valid conclusion. In the present case, since we mark the whole term, $h(a)$ is added to T, and hence after this inference $T = \{a, h(a)\}$.

Then, with an inference by marking left *on 15 we get*

$$16) \quad h(a^{x_1})^{x_2} \simeq h(a^{x_1}) \rightarrow$$

And with another inference by marking left, *we get*

$$17) \quad h(a^{x_1})^{x_2} \simeq h(a^{x_1})^{x_2} \rightarrow$$

Finally, with an inference by equality resolution *on 17 the empty clause is obtained.* □

7.6.1 Redundancy notions

Here, as for the case of Horn clauses, the notions of redundancy and saturation are obtained from the ones given in Section 7.3.1, by using clauses instead of equations and the inference system I_2 instead of $\mathcal{E}_2$, and lifting the orderings adequately. Moreover, since the inference rules are non-deterministic, an inference is defined to be redundant also when some of the possible conclusions are included in the set of clauses S. Since now inference rules can also add terms to the set T, we refer to the inferences as using a set of terms T as well as some clauses as premises. However, only for the case of *marking* inferences the set T determines whether the inference can be performed.

Definition 7.6.3 *Let π be an inference by I_2, using a set of terms T, with premises $C_1, \dots C_n$ and conclusion D. Then a* ground instance $\pi\theta$ of the inference π *is an inference by I_2, using T, with premises $C_1\theta, \dots C_n\theta$ and conclusion $D\theta$ for some ground substitution θ.*

Let G be a set of ground marked clauses. A ground inference by I_2, using some set of terms T, with premises $C_1, \dots C_n$ and conclusion D is redundant in G *if*

$$Forget(G^{\prec^c_{red} C_n}) \cup Forget(G^{\prec^c_{red} D}) \models Forget(D)$$

or there is a clause D' in G such that D' is a valid conclusion for the same premises $C_1, \dots, C_n$ and the same set of terms T.

Let S be a set of marked clauses. An inference by I_2, using some set of terms T, with premises $C_1, \dots C_n$ and conclusion D is redundant in S *if, for all sets of terms M and ground substitutions θ with marks in M, $\pi\theta$ is redundant in $gnd_M(S)$.*

Let T be a set of terms. A mark detection rule $T \Longrightarrow T \cup \{t\}$ is redundant in T if t is an instance of a term in T.

As usual, from redundancy of inferences, saturation with respect to the inference system can be defined.

Definition 7.6.4 *Let S be a set of clauses and T be a set of terms. The pair $\langle S, T\rangle$ is* saturated *with respect to I_2 if every inference by I_2, using T and with premises in S, is redundant in S, and every mark detection rule is redundant in T.*

We take the same notion of redundancy for clauses as in Section 7.4.1, and also the same notion of *derivation*, but with an additional possibility for removing terms from the set T, based on the fact that now it is likely that some marks can disappear from S when deleting non-unit clauses (e.g., when a clause like $\rightarrow l\cdot\gamma \simeq r\cdot\gamma, \Delta$ is deleted in the presence of $\rightarrow \Delta$ and the marks in $l\cdot\gamma \simeq r\cdot\gamma$ do not occur in any other clause). Then case 2b becomes:

2b. $T_{i+1} = T_i \setminus \{t\}$ where t is an instance of a term in $T_i \setminus \{t\}$, or t has not been obtained by a mark detection rule and there is no marked subterm of the form $(s \cdot \gamma)^x$ occurring in S such that $Forget(s \cdot \gamma)$ is an instance of t.

From the notion of redundancy for clauses, the following lemma holds as well.

Lemma 7.6.5 *Let $\langle S_0, T_0\rangle, \langle S_1, T_1\rangle, \ldots$ be a derivation and let C be a clause in $(\cup_i S_i) \setminus S_\infty$. Then C is redundant in S_∞.*

The notion of fairness is the same as in Section 7.4.1. Then Lemma 7.4.6, applied to I_2, holds as well:

Lemma 7.6.6 *If $\langle S_0, T_0\rangle, \langle S_1, T_1\rangle, \ldots$ is a fair derivation with respect to I_2, then $\langle S_\infty, T_\infty\rangle$ is saturated with respect to I_2.*

7.6.2 Model generation

Analogously to Section 7.3.2, we denote by T_{S} the set of positive unit clauses $\rightarrow t \simeq \bot$ for all t in T, and we denote by T_{G} the set of mark free ground instances of clauses in T_{S}. The model will be defined inductively on the clauses in T_{G} and $gnd_T(S)$ with respect to $\succ_{mod}$, a total well-founded ordering including: $C \succ_{mod} D$ if

1. $Forget(C) \succ^c Forget(D)$.
2. $Forget(C) \equiv^c Forget(D)$ and $C \succ^c_m D$, which means that D has marks at higher positions than C.

Now we define a rewrite system R_S from $gnd_T(S)$, which requires us to define at the same time a rewrite system R_T from T_{G}.

Definition 7.6.7 *The rewrite system R_{ST} is the union of R_S and R_T. Let C be a ground (marked) clause in $gnd_T(S) \cup T_{\mathrm{G}}$.*

1. *If C is an instance $\rightarrow (s \cdot \gamma \simeq t \cdot \gamma)\theta, \Delta\theta$ in $gnd_T(S)$ of some clause $\rightarrow s \cdot \gamma \simeq t \cdot \gamma, \Delta$ in S, then it generates the rule $(s \cdot \gamma)\theta \rightarrow (t \cdot \gamma)\theta$ in R_S if*
   - *(a) $Forget(R^C_S)^* \not\models Forget(C)$,*
   - *(b) $s \succ_r t$ or $t \cdot \gamma$ is marked on top,*
   - *(c) $Forget((s \cdot \gamma)\theta) \succ Forget((t \cdot \gamma)\theta)$ and $Forget((s \cdot \gamma \simeq t \cdot \gamma)\theta) \succ^{mul} Forget(e\theta)$ for all equations e in Δ,*
   - *(d) $(s \cdot \gamma)\theta$ is irreducible by $R^C_S \cup R^C_T$, and*
   - *(e) $(t \cdot \gamma)\theta$ and $\Delta\theta$ are irreducible by $R^C_S \cup R^C_T$ at non-topmost positions.*
   - *(f) $Forget(R^C_S)^* \models Forget((r \cdot \gamma \simeq t \cdot \gamma)\theta)$ for no equation $l \cdot \gamma \simeq t \cdot \gamma$ in Δ,*
2. *If C is an instance $\rightarrow t\theta \simeq \bot$ in T_{G} of a clause $\rightarrow t \simeq \bot$ in T_{S}, then it generates the rule $u \cdot \delta \rightarrow (u \cdot \delta)^x$ in R_T if $u \cdot \delta$ is the normal form of $t\theta$ w.r.t. R^C_T and $u \cdot \delta$ is irreducible by R^C_S,*

where R^C_S and R^C_T denote the set of rules in R_S and R_T respectively, generated by clauses D such that $C \succ_{mod} D$.

By construction, as in Section 7.3.2, we obtain the following analogous results:

Lemma 7.6.8 *Let $l \cdot \delta \to r \cdot \delta$ be a rule in R_T. Then $r \cdot \delta \equiv (l \cdot \delta)^x$, $l \cdot \delta$ is not marked on top, and the rule has been generated by a clause of the form $\to t\theta \simeq \bot$, where $t \in T$, $t\theta \in T(\mathcal{F})$, and $Forget(l \cdot \delta) \equiv t\theta$.*

Lemma 7.6.9 *Let $s \cdot \gamma$ be a marked subterm occurring in R_{ST}. If $s \cdot \gamma$ is marked on top, then $Forget(s \cdot \gamma)$ is an instance of a term in T.*

Lemma 7.6.10 *Let $s \cdot \gamma$ be a marked subterm occurring, not as right-hand side, in some rule of R_S. Then $s \cdot \gamma$ is marked on top if, and only if, $Forget(s \cdot \gamma)$ is an instance of a term in T.*

Lemma 7.6.11 *For all rules $l \to r$ in $Forget(R_S)$ we have that*

1. *r is irreducible by $Forget(R_S)$ at non-topmost positions, and*
2. *l is irreducible by $Forget(R_S) \setminus \{l \to r\}$.*

Lemma 7.6.12 *$Forget(R_S)$ is convergent.*

By the same arguments as for irreducibility of right-hand sides of rules in $Forget(R_S)$, and by the same arguments as in Lemma 3.5.5 of Chapter 3, we obtain the following:

Lemma 7.6.13 *If a clause $\to l \cdot \delta \simeq r \cdot \delta, \Delta$ in $gnd_T(S)$ generates the rule $l \cdot \delta \to r \cdot \delta$ in R_S, then $Forget(\Delta)$ is irreducible by $Forget(R_S)$ at non-topmost positions and $Forget(R_S)^* \not\models Forget(\Delta)$.*

7.6.3 Completeness

We now introduce the well-founded ordering $\succ_j$ that will be used in the proof of Theorem 7.6.17. It extends $\succ_i$ (see Definition 7.2.4) with an additional case:

Definition 7.6.14 *Let $s \cdot \gamma$ and $t \cdot \delta$ be two marked terms, and let R denote $Forget(R_S)$. Then $s \cdot \gamma \succ_j t \cdot \delta$ iff (i) $s \rhd \cup \succ_r t$ or (ii) $s \doteq t$ and $Ran(\gamma) \succ_m^{mul} Ran(\delta)$ or (iii) $s \doteq t$ and $Forget(Ran(\gamma)) \xrightarrow{+}_R Forget(Ran(\delta))$ or (iv) $s \doteq t$, all terms in $Forget(Ran(\delta))$ are irreducible by R at non-topmost positions and $Forget(Ran(\gamma)) \succ^{mul} Forget(Ran(\delta))$.*

In what follows, we will consider $\succ_j$ as the transitive closure of the ordering defined above (composed with the congruence $\equiv$ on marked terms).

Property 7.6.15 *Let R denote $Forget(R_S)$. Then for all ground terms s and t such that s is irreducible by R at non-topmost positions and $s \to_R^+ t$, we have*

1. *$s \succ t$ and*
2. *t is irreducible by R_S at non-topmost positions.*

Proof. If $s \to_R s_1 \to_R \cdots \to_R s_n \to_R t$ then, since s is irreducible at non-topmost positions and, by Lemma 7.6.11, all right-hand sides of rules in R are irreducible at non-topmost positions, all steps in the sequence are at topmost position, which implies on the one hand that t is irreducible at non-topmost positions and, on the other hand, by definition of R_S, that $s \succ s_1 \succ \cdots \succ s_n \succ t$. □

Lemma 7.6.16 *$\succ_j$ is well-founded.*

Proof. Since the relation $(\rhd \cup \succ_r)\circ \doteq$ is well-founded (see Section 2.11.2 and [DJ90]), in a sequence $s_1 \cdot \gamma_1 \succ_j s_2 \cdot \gamma_2 \cdots$ there can only be finitely many steps by case (i). Hence, from some point on, there are only steps by case (ii), (iii) or (iv). Now observe that, if $s_k \cdot \gamma_k \succ_j s_m \cdot \gamma_m \succ_j s_n \cdot \gamma_n$, where the first step is by case (ii) and the second step is by case (iii) or (iv), then $s_k \cdot \gamma_k \succ_j s_n \cdot \gamma_n$ by case (iii) or (iv), since $Forget(s_k \cdot \gamma_k) \equiv Forget(s_m \cdot \gamma_m)$ by definition of $\succ_m$. But there must be only finitely many steps by (iii) or (iv), since R is terminating, $\succ$ is well-founded and, by Property 7.6.15, after a step by case (iv), all steps by case (iii) are also steps by case (iv). Finally, since $\succ_m$ is well-founded by Lemma 7.2.2, there can only be finitely many steps by case (ii), which implies that $\succ_j$ is well-founded. $\square$

Theorem 7.6.17 (Refutation completeness of I_2 for general clauses) *Let S be a set of clauses and T a set of terms such that the pair $\langle S,T\rangle$ is saturated with respect to I_2. Then $\square \in S$ if, and only if, $Forget(S)$ is unsatisfiable.*

Proof. Let R denote $Forget(R_S)$. We prove that R^* is a model of $Forget(S)$ by induction on $\succ_j^c$. We derive a contradiction from the existence of a minimal w.r.t. $\succ_j^c$ ground instance $C\theta$ in $gnd_T(S)$ of a clause C in S such that $R^* \not\models Forget(C\theta)$.

1. We first consider the case where C is a positive clause $\rightarrow s\cdot\gamma \simeq t\cdot\gamma, \Delta$, where $Forget((s\cdot\gamma \simeq t\cdot\gamma)\theta)$ is strictly maximal with respect to $\succ^{mul}$ in $Forget(C\theta)$ and w.l.o.g. $Forget((s\cdot\gamma)\theta) \succ Forget((t\cdot\gamma)\theta)$. Since $R^* \not\models Forget(C\theta)$, we know that $C\theta$ has not generated any rule due to one of the following reasons:

   (a) $s \not\succ_r t$ and $t\cdot\gamma$ is not marked on top. In this case we can apply the *mark detection* rule

   $$T \Longrightarrow T \cup \{Forget(t\cdot\gamma)\}.$$

   Now, since $\langle S,T\rangle$ is saturated w.r.t. I_2, this inference is redundant in T. Thus, $Forget(t\cdot\gamma)$ is an instance of a term t' in T. But then there is an inference π by *marking right*,

   $$\frac{\rightarrow s\cdot\gamma \simeq t\cdot\gamma, \Delta}{\rightarrow s\cdot\gamma \simeq (t\cdot\gamma)^x, \Delta}.$$

   Note that, if t is a variable, then t' is also a variable. Since $t\cdot\gamma$ is not marked on top, we clearly have that $t\cdot\gamma \succ_j (t\cdot\gamma)^x$ by case (i), and hence $C\theta \succ_j D\theta$, where D denotes the conclusion of the previous inference. Now, since $\langle S,T\rangle$ is saturated w.r.t. I_2, this inference is redundant in S. Therefore, since $C\theta \in gnd_T(S)$, we have that (I) $Forget(gnd_T(S)^{\prec_{red}^c C\theta}) \cup Forget(gnd_T(S)^{\prec_{red}^c D\theta}) \models Forget(D\theta)$ or (II) there is a clause $D'\theta$ in $gnd_T(S)$ of a clause D' in S such that $D'\theta$ is a valid conclusion for the same premises as $\pi\sigma$ and the same set of terms T. If we are in case (I), since $C\theta \succ_j^c D\theta$, $\succ_{red} \subseteq \succ_j$ and, by minimality of $C\theta$, $R^* \models Forget(gnd_T(S)^{\prec_j^c C\theta})$, then we also have that $R^* \models Forget(gnd_T(S)^{\prec_{red}^c C\theta}) \cup Forget(gnd_T(S)^{\prec_{red}^c D\theta})$, and hence $R^* \models Forget(D\theta)$, contradicting $R^* \not\models Forget(C\theta)$. If we are in case (II), then $D'\theta \equiv D\theta$ and hence $D'\theta$ is smaller than $C\theta$ w.r.t. $\succ_j^c$ as well, which implies that $R^* \models Forget(D'\theta)$, contradicting $R^* \not\models Forget(C\theta)$ as before.

   (b) $(s\cdot\gamma)\theta$ is reducible by $R_S^C \cup R_T^C$. Then there exists an instance $C'\theta$ of a clause C' of the form $\rightarrow l\cdot\gamma \simeq r\cdot\gamma, \Delta_0$ in $S \cup T_S$, that has generated a rule $l'\cdot\gamma' \rightarrow r'\cdot\gamma'$ reducing $(s\cdot\gamma)\theta$ at some position p. There are the following possibilities:

i. Lifting: $Forget(s\cdot\gamma)|_{p'}$ is a variable x for some prefix p' of p.
Then $p = p'\cdot p''$ for some position p'' such that $x\theta|_{p''} \equiv l'\cdot\gamma'$. Now let θ' be a ground substitution with the same domain as θ but where $x\theta' \equiv x\theta[r'\cdot\gamma']_{p''}$ and $y\theta' \equiv y\theta$ for all other variables y.

   A. $l'\cdot\gamma' \to r'\cdot\gamma' \in R_S$. Then $l'\cdot\gamma' \equiv (l\cdot\gamma)\theta$, $r'\cdot\gamma' \equiv (r\cdot\gamma)\theta$, and either $l \succ_r r$ or $r\cdot\gamma$ is marked on top, which gives us that $C\theta \succ_j^c C\theta'$ by cases (i) and (iii). Moreover, since $R = Forget(R_S)$, we have that $R^* \models Forget(l'\cdot\gamma' \simeq r'\cdot\gamma')$. Then, since $R^* \not\models Forget(C\theta)$, we necessarily have that $R^* \not\models Forget(C\theta')$. We also have that $C\theta' \in gnd_T(S)$, since $C\theta \in gnd_T(S)$ and, moreover, for every marked subterm of the form $(u\cdot\rho)^z$ in $r'\cdot\gamma'$ we have that $Forget(u\cdot\rho)$ is an instance of a term in T by Lemma 7.6.9. Therefore $C\theta'$ contradicts the minimality of $C\theta$.

   B. $l'\cdot\gamma' \to r'\cdot\gamma' \in R_T$. In this case, by Lemma 7.6.8, $l'\cdot\gamma'$ is not marked on top, $r'\cdot\gamma'$ is marked on top, and $Forget(l'\cdot\gamma') \equiv Forget(r'\cdot\gamma')$. On the one hand, this gives us that $C\theta \succ_j^c C\theta'$ by cases (i) and (ii). On the other hand, we obviously have $R^* \models Forget(l'\cdot\gamma' \simeq r'\cdot\gamma')$. Then a contradiction is derived as in the previous case.

ii. An inference: p is a non-variable position of $Forget(s\cdot\gamma)$.

   A. $l'\cdot\gamma' \to r'\cdot\gamma' \in R_S$. Then $l'\cdot\gamma' \equiv (l\cdot\gamma)\theta$, $r'\cdot\gamma' \equiv (r\cdot\gamma)\theta$, and $Forget((l\cdot\gamma)\theta) \succ Forget((r\cdot\gamma)\theta)$. Therefore there exists an inference π by *paramodulation right*,

$$\frac{\to l\cdot\gamma \simeq r\cdot\gamma, \Delta_0 \qquad \to s\cdot\gamma \simeq t\cdot\gamma, \Delta}{(\to s'\cdot\gamma' \simeq t\cdot\gamma, \Delta_0', \Delta)\sigma},$$

where $\sigma = mgu(l\cdot\gamma, s\cdot\gamma|_p)$, $Forget(s'\cdot\gamma') \equiv Forget(s\cdot\gamma[r\cdot\gamma]_p)$, $Forget(\Delta_0')$ is $Forget(\Delta)$, $(s\cdot\gamma)\sigma \succ_{red} (s'\cdot\gamma')\sigma$, or $s \doteq s'$ and $s|_q$ is a variable for some prefix of q of p, and for every equation e in Δ_0', $\{s\cdot\gamma, t\cdot\gamma\}\sigma \succ_{red}^{mul} e\sigma$ or both sides of e are marked on top.
Now we show that the conclusion D of the previous inference has an instance $D\theta$ fulfilling $C\theta \succ_j^c D\theta$. Observe that, if $(s\cdot\gamma)\sigma \succ_{red} (s'\cdot\gamma')\sigma$, then $(s\cdot\gamma)\sigma \succ_j (s'\cdot\gamma')\sigma$ and also $(s\cdot\gamma)\theta \succ_j (s'\cdot\gamma')\theta$, since $\succ_{red} \subseteq \succ_j$ and $\succ_j$ is stable under substitutions. And, if $s \doteq s'$ and $s|_q$ is a variable for some prefix of q of p, then q is a marked position of $s\cdot\gamma$ and, hence, $(s\cdot\gamma)\theta \succ_j (s'\cdot\gamma')\theta$ by case (iii), since $(l\cdot\gamma)\theta \to (r\cdot\gamma)\theta \in R_S$. W.r.t. the equations e in Δ_0', either $\{s\cdot\gamma, t\cdot\gamma\}\sigma \succ_{red}^{mul} e\sigma$, and hence $\{s\cdot\gamma, t\cdot\gamma\}\theta \succ_j^{mul} e\theta$, or both sides of e are marked on top. In this latter case, if either $s\cdot\gamma$ or $t\cdot\gamma$ are not marked on top, then we also have that $\{s\cdot\gamma, t\cdot\gamma\}\sigma \succ_{red}^{mul} e\sigma$. If both $s\cdot\gamma$ and $t\cdot\gamma$ are marked on top, then we have that $\{s\cdot\gamma, t\cdot\gamma\}\theta \succ_j^{mul} e\theta$ by case (iv), for the following reasons: $Forget(\Delta_0\theta)$ is irreducible by $Forget(R_S)$ by Lemma 7.6.13; and $Forget((s\cdot\gamma \simeq t\cdot\gamma)\theta) \succ^{mul} Forget(e\theta)$, since $C\theta \succ^c C'\theta$ and $Forget((s\cdot\gamma \simeq t\cdot\gamma)\theta)$ is strictly maximal with respect to $\succ^{mul}$ in $Forget(C\theta)$, which gives us $Forget((s\cdot\gamma \simeq t\cdot\gamma)\theta) \succeq^{mul} Forget((l\cdot\gamma \simeq r\cdot\gamma)\theta) \succ^{mul} Forget(e\theta)$. Finally, as in case 1a, we derive a contradiction from saturatedness of $\langle S, T\rangle$: We have that this inference is redundant in S. Therefore, since $C\theta \in gnd_T(S)$, we have that (I) $Forget(gnd_T(S)^{\prec_{red}^c C\theta}) \cup Forget(gnd_T(S)^{\prec_{red}^c D\theta}) \models Forget(D\theta)$ or (II) there is a clause $D'\theta$ in $gnd_T(S)$ of a clause D' in S such that $D'\theta$ is a valid conclusion for the same premises as $\pi\sigma$ and the same set of terms T. If we are in case (I), since $C\theta \succ_j^c D\theta$, $\succ_{red} \subseteq \succ_j$ and, by minimality of $C\theta$, $R^* \models Forget(gnd_T(S)^{\prec_j^c C\theta})$,

then we also have that $R^* \models Forget(gnd_T(S)^{\prec^c_{red} C\theta}) \cup Forget(gnd_T(S)^{\prec^c_{red} D\theta})$, and hence $R^* \models Forget(D\theta)$; but, on the other hand, since $R^* \models (l \cdot \gamma)\theta \simeq (r \cdot \gamma)\theta$ and, by Lemma 7.6.13, $R^* \not\models Forget(\Delta'_0\theta)$, from $R^* \not\models Forget(C\theta)$ we get that $R^* \not\models Forget(D\theta)$, which is a contradiction. If we are in case (II), then we can assume w.l.o.g. that $D' \equiv D$, since no particular assumption has been made on the marks of D, and hence $D'\theta$ is smaller than $C\theta$ w.r.t. $\succ^c_j$ as well, which implies that $R^* \models Forget(D'\theta)$, contradicting $R^* \not\models Forget(C\theta)$ as before.

B. $l' \cdot \gamma' \to r' \cdot \gamma' \in R_T$. In this case, by Lemma 7.6.8, $l' \cdot \gamma'$ is not marked on top, $r' \cdot \gamma'$ is marked on top, and $Forget(l' \cdot \gamma') \equiv Forget(r' \cdot \gamma') \equiv l$ for some term l in T. Since $l' \cdot \gamma'$ is not marked on top and $(s \cdot \gamma)\theta|_p \equiv l' \cdot \gamma'$, then p is a non-marked position of $(s \cdot \gamma)\theta$, and hence of $s \cdot \gamma$. Therefore there is an inference by *marking right*,

$$\frac{\to s \cdot \gamma \simeq t \cdot \gamma, \Delta}{(\to s \cdot \gamma[(s \cdot \gamma|_p)^x]_p \simeq t \cdot \gamma, \Delta)\sigma},$$

where $\sigma = mgu(Forget(s \cdot \gamma|_p), l)$, and whose conclusion D, under the substitution θ, coincides with $(\to s \cdot \gamma[(s \cdot \gamma|_p)^x]_p \simeq t \cdot \gamma, \Delta)\theta$. Moreover, since p is a non-marked position of $s \cdot \gamma$, we have that either $s \gtrdot s'$, or $s \doteq s'$ and $Ran(\gamma) \succ^{mul}_m Ran(\gamma')$, where $s' \cdot \gamma'$ denotes $s \cdot \gamma[(s \cdot \gamma|_p)^x]_p$. Therefore, $s \cdot \gamma \succ_j s \cdot \gamma[(s \cdot \gamma|_p)^x]_p$ either by case (i) or (ii), and hence $C\theta \succ^{mul}_j D\theta$. Then a contradiction is derived as in the previous cases.

(c) $(t \cdot \gamma)\theta$ or $\Delta\theta$ are reducible by $R^C_S \cup R^C_T$ at some non-topmost position. The proof is like that of case 1b.

(d) None of the previous cases applies, and Δ is of the form $u \cdot \gamma \simeq v \cdot \gamma, \Delta'$, where $Forget((s \cdot \gamma)\theta) \equiv Forget((u \cdot \gamma)\theta)$ and $R^* \models Forget((t \cdot \gamma \simeq v \cdot \gamma)\theta)$, that is, the last condition of Definition 7.6.7 fails. We consider the following two situations:

i. Either $Forget((t \cdot \gamma)\theta)$ or $Forget((v \cdot \gamma)\theta)$ is reducible by R at a non-topmost position. We assume that $Forget((t \cdot \gamma)\theta)$ is reducible (the proof for the other case is analogous). Then $Forget((t \cdot \gamma)\theta)|_p \equiv Forget(l \cdot \delta)$ for some rule $l \cdot \delta \to r \cdot \delta$ in R_S and some position $p \neq \lambda$. In fact, since $\rhd \subseteq \succ$, no rule generated by a greater clause can reduce $Forget((t \cdot \gamma)\theta)$ at a non-topmost position, and hence $l \cdot \delta \to r \cdot \delta \in R^C_S$. Now recall that $(t \cdot \gamma)\theta$ is irreducible by R^C_S at non-topmost positions, since none of the previous cases applies. Therefore, the only possibility is that there are some positions q and p' such that $q = p \cdot p'$ and $Forget((t \cdot \gamma)\theta)|_q \equiv Forget((l \cdot \delta)\theta)|_{p'}$, and either q is a marked position of $(t \cdot \gamma)\theta$ and p' is a non-marked position of $(l \cdot \delta)\theta$, or q is a non-marked position of $(t \cdot \gamma)\theta$ and p' is a marked position of $(l \cdot \delta)\theta$.

- If q is a marked position of $(t \cdot \gamma)\theta$ then, since $C\theta \in gnd_T(S)$ and, moreover, for every marked term of the form $(l' \cdot \delta')^x$ in occurring in S we have that $Forget(l' \cdot \delta')$ is an instance of a term in T, we must have that $Forget((t \cdot \gamma)\theta|_q)$ is an instance of a term t in T. But then also $Forget((l \cdot \delta)\theta|_{p'})$ is an instance of t and, therefore, by Lemma 7.6.10 we have that $(l \cdot \delta)\theta|_{p'}$ is marked on top, contradicting that p' is a non-marked position of $(l \cdot \delta)\theta$.
- If q is a non-marked position of $(t \cdot \gamma)\theta$ and p' is a marked position of $(l \cdot \delta)\theta$, then we have that $(l \cdot \delta)\theta|_{p'} \equiv t$ for some term t in T by Lemma 7.6.9 and, as in case 1b, either the lifting argument applies or there is an inference by *marking right*, and hence a contradiction is derived as well.

ii. $Forget((t\cdot\gamma)\theta)$ and $Forget((v\cdot\gamma)\theta)$ are irreducible by R at non-topmost positions. In this case we consider the following inference by *equality factoring*,

$$\frac{\rightarrow s\cdot\gamma\simeq t\cdot\gamma, u\cdot\gamma\simeq v\cdot\gamma,\Delta'}{(t'\cdot\gamma'\simeq v'\cdot\gamma'\rightarrow s\cdot\gamma\simeq t\cdot\gamma,\Delta')\sigma},$$

where $\sigma = mgu(s\cdot\gamma, u\cdot\gamma)$, $Forget(t'\cdot\gamma')\equiv Forget(t\cdot\gamma)$, $Forget(v'\cdot\gamma')\equiv Forget(v\cdot\gamma)$, and $(u\cdot\gamma)\sigma \succ_{red} (t'\cdot\gamma')\sigma$ or $t'\cdot\gamma'$ is marked on top, and $(u\cdot\gamma)\sigma \succ_{red} (v'\cdot\gamma')\sigma$ or $v'\cdot\gamma'$ is marked on top. We clearly have that the conclusion D of the previous inference coincides with $(t'\cdot\gamma'\simeq v'\cdot\gamma'\rightarrow s\cdot\gamma\simeq t\cdot\gamma,\Delta')\theta$ under the substitution θ.
Now we show that $(u\cdot\gamma)\theta \succ_j (t'\cdot\gamma')\theta$ (the proof for $(u\cdot\gamma)\sigma \succ_j (v'\cdot\gamma')\theta$ is analogous). If $(u\cdot\gamma)\sigma \succ_{red} (t'\cdot\gamma')\sigma$ then $(u\cdot\gamma)\theta \succ_j (t'\cdot\gamma')\theta$, since $\succ_{red}\subseteq\succ_j$ and $\succ_j$ is stable under substitutions. If $t'\cdot\gamma$ is marked on top and $u\cdot\gamma$ is not, then $u\cdot\gamma \succ_{red} t'\cdot\gamma'$ and we conclude as before. If both $u\cdot\gamma$ and $t'\cdot\gamma'$ are marked on top, then we have that $(u\cdot\gamma)\theta \succ_j (t'\cdot\gamma')\theta$ by case (iv), since $Forget((u\cdot\gamma)\theta) \succ Forget((t'\cdot\gamma')\theta)$, and $Forget((t'\cdot\gamma')\theta)$ is assumed to be irreducible by R at non-topmost positions.
Then, since $\{(u\cdot\gamma)\theta,(v\cdot\gamma)\theta\} \succ_j^{mul} \{(t'\cdot\gamma')\theta,(t'\cdot\gamma')\theta,(v'\cdot\gamma')\theta,(v'\cdot\gamma')\theta\}$, we have that $C\theta \succ_j^c D\theta$, and from this fact a contradiction can be derived as in previous cases.

2. If C is a positive clause $\rightarrow s\cdot\gamma\simeq t\cdot\gamma,\Delta$, where $Forget((s\cdot\gamma\simeq t\cdot\gamma)\theta)$ is maximal but not strictly maximal with respect to $\succ^{mul}$ in $Forget(C\theta)$, then condition 1c of Definition 7.6.7 fails. If $s \not\succ_r t$ and $t\cdot\gamma$ is not marked on top, the reasoning of case 1a applies. If $(s\cdot\gamma\simeq t\cdot\gamma)\theta$ is reducible by R_S at a non-topmost position, then the same reasoning as in case 1b applies. Otherwise, the reasoning of case 1d applies.

3. If C is clause $\Gamma,\underline{s\cdot\gamma\simeq t\cdot\gamma}\rightarrow\Delta$ where $Forget((s\cdot\gamma)\theta)\equiv Forget((t\cdot\gamma)\theta)$, then there is an inference by *equality resolution*,

$$\frac{\Gamma,\underline{s\cdot\gamma\simeq t\cdot\gamma}\rightarrow\Delta}{(\Gamma\rightarrow\Delta)\sigma},$$

where $\sigma = mgu(s\cdot\gamma, t\cdot\gamma)$. Then we clearly have that $C\theta \succ_j^c D\theta$, where D denotes the conclusion of the previous inference. This fact, together with the saturatedness assumption, leads to a contradiction as in the other cases.

4. If C is a clause $\Gamma,\underline{s\cdot\gamma\simeq t\cdot\gamma}\rightarrow\Delta$ where $Forget((s\cdot\gamma)\theta)\not\equiv Forget((t\cdot\gamma)\theta)$, then assume w.l.o.g. that $Forget((s\cdot\gamma)\theta)\succ Forget((t\cdot\gamma)\theta)$. Let s' denote $Forget((s\cdot\gamma)\theta)$ and t' denote $Forget((t\cdot\gamma)\theta)$. Then, since $R^*\not\models Forget(C\theta)$, we have that $R^*\models s'\simeq t'$ and, since R is convergent by Lemma 7.6.12, there must be a rewrite proof of $s'\simeq t'$ with R, that is, s' and t' must rewrite into the same normal form with R. This implies that either t' is reducible at a non-topmost position or else s' is reducible. (Note that it cannot be the case that the only possible reduction step on $s'\simeq t'$ is at the topmost position of t'. By such a step, a new term t'' is obtained with $t'\succ t''$ and t'' again irreducible at non-topmost positions by Lemma 7.6.11; since $s'\succ t'$, such a sequence of topmost steps on t' can never produce s'). We consider the case where s' is reducible. (The other one is analogous.) Then there is an instance of a clause $\rightarrow l\cdot\gamma\simeq r\cdot\gamma,\Delta_0$ in S that has generated a rule $(l\cdot\gamma)\theta\rightarrow(r\cdot\gamma)\theta$ in R_S, such that $Forget((s\cdot\gamma)\theta)|_p\equiv Forget((l\cdot\gamma)\theta)$ for some position p. We have the following possibilities:

   (a) $(s\cdot\gamma)\theta|_p\equiv(l\cdot\gamma)\theta$. Then either the lifting argument applies like in case 1(b)iA, or else there is an inference by *paramodulation left*, which leads to a contradiction analogously to case 1(b)iiA.

(b) $(s\cdot\gamma)\theta|_p \not\equiv (l\cdot\delta)\theta$. Then, analogously to case 2b of Theorem 7.4.3, either the lifting argument applies or else there is an inference by *marking left*, and a contradiction is derived respectively as in cases 1(b)iB and 1(b)iiB of this theorem.

$\square$

Theorem 7.6.18 *If $\langle S_0,T_0\rangle,\langle S_1,T_1\rangle,\ldots$ is a fair derivation with respect to I_2, then Forget(S_0) is unsatisfiable if, and only if, $\square \in S_j$ for some j. Furthermore, if $\langle S_0,T_0\rangle,\langle S_1,T_1\rangle,\ldots\langle S_n,T_n\rangle$ is fair, then S_n is saturated and Forget(S_n) is logically equivalent to Forget(S_0).*

Proof. By Lemma 7.6.6 we have that $\langle S_\infty,T_\infty\rangle$ is saturated with respect to I_2. Then, by Theorem 7.6.17, we have that $\square \in S_\infty$ if *Forget*(S_∞) is unsatisfiable. Since, by definition of derivation, *Forget*(S_0) is logically equivalent to all *Forget*(S_i) with $i \geq 0$ and, by Lemma 7.6.5, to *Forget*(S_∞) as well, *Forget*(S_0) is unsatisfiable if, and only if, $\square \in S_\infty$ and hence $\square \in S_j$ for some j. Finally if $\langle S_0,T_0\rangle,\langle S_1,T_1\rangle,\ldots\langle S_n,T_n\rangle$ is fair then $S_\infty = S_n$ and hence S_n is saturated. $\square$

7.7 Conclusions

In this chapter we have improved on the results of Chapter 3 on completeness of ordered paramodulation with non-monotonic orderings, by adapting those refutation complete inference systems so that they are compatible with powerful redundancy notions like demodulation.

We have proposed some inference systems (for equations, Horn clauses and general first order clauses) that work with a pair of orderings: a west ordering $\succ$, which is used in the inferences, and a reduction ordering $\succ_r$, included in $\succ$, which is used in the redundancy notions. Our method is based on adding some marks that block terms for redundancy (but not for inferences). In it, if there are many equations that cannot be handled by the reduction ordering, then many terms will become blocked and we will have almost no redundancy, but the same amount of inferences by paramodulation as in Chapter 3. On the other hand, if many equations can be handled by the reduction ordering, then few terms will be blocked and we will have as much redundancy as in the paramodulation calculi for total reduction orderings. Therefore now it becomes interesting to obtain experimental results to see how do these inference systems behave in practice.

Using these results in the same way as in Chapter 4, we have obtained a Knuth-Bendix completion procedure, compatible with simplification by rewriting, that finds a convergent TRS for a given set of equations E and a (possibly non-totalizable) reduction ordering $\succ_r$ whenever it exists.

As a final remark, we want to note that a way to obtain, in practice, the orderings $\succ$ and $\succ_r$, is to define the reduction ordering $\succ_r$ with the *monotonic semantic path orderings* [BFR00], since then it is directly included, by definition, in a *semantic path ordering* [KL80], which can be used as the west ordering $\succ$.

Chapter 8

Conclusions and future work

In this thesis, a detailed study of the ordered paramodulation inference rule has been made. To this end, two aspects have been considered:

- which properties of the ordering used to paramodulate with are essential for ensuring refutation completeness of an ordered paramodulation based inference system, and
- what kind of redundancy notions are then compatible with refutation completeness in the resulting calculus.

We have focused mainly on non-monotonic orderings. We have shown that ordered paramodulation remains complete when using (well-founded) non-monotonic orderings (cf. Chapter 3), even if they contradict the subterm property (cf. Chapter 5). As a side effect, we have obtained Knuth-Bendix completion procedures for non-monotonic orderings (cf. Chapters 4 and 7). We think that these are important improvements, as defining a *good* ordering is a difficult task when considering deduction in an equational theory. On the other hand, these results somehow show the power of ordered paramodulation, and leave as the last question whether even well-foundedness is necessary. However, we have seen that there is a price to be paid, concerning redundancy elimination, when using such *weak* orderings: in some cases compatibility with redundancy elimination is lost, and in other cases we have not been able to prove it. For these situations, we have developed a modular notion of redundancy to exploit redundancy elimination in a refutation complete way, regardless of the theorem prover being used (cf. Chapter6). Regarding concrete redundancy notions we have shown that, in the context of paramodulation with non-monotonic orderings, tautology deletion and subsumption are correct in general, whereas demodulation (i.e., simplification by rewriting) is only possible when applied w.r.t. a reduction ordering which is included in the ordering which is used to paramodulate with (cf. Chapter 7). Some positive and negative results are best described by the following (counter)examples:

3.6.1 on page 44 shows that inferences on non-maximal positive atoms are necessary for refutation completeness of paramodulation with non-monotonic orderings.

3.6.2 on page 45 shows that also inferences on small sides of positive equations are needed.

3.6.3 on page 45 shows that also inferences on small sides of negative equations are needed.

3.6.4 ([Lyn97]) on page 45 shows that for arbitrary selection strategies, these techniques are incompatible with tautology deletion.

5.1.1 on page 51 shows that, sometimes, a goal-oriented (ordered) paramodulation proof can only be obtained if the ordering contradicts the subterm property.

7.4.8 on page 125 shows that if ordered paramodulation is applied w.r.t. a non-monotonic west ordering $\succ$ including a reduction ordering $\succ_r$, then demodulation w.r.t. $\succ_r$ and redundancy with smaller equations w.r.t. the subterm relation can cause incompleteness.

Although counterexamples showing the limitations of some extensions of our results have been given, there are many directions of research which are far from being closed. We emphasize the following two directions:

- One is about obtaining experimental results. Clearly, although we have obtained several completeness results, this is far to say that our ideas are applicable in practice. In fact, it is mainly after the results of Chapter 7, which show that demodulation is possible to some degree, that it becomes interesting to study how an inference system applying paramodulation w.r.t. a non-monotonic ordering will behave in practice.

- Another direction of future work is about studying completeness of ordered paramodulation when using weaker orderings (e.g., non-well-founded), or with a different combination of properties (e.g., monotonic but non-well-founded). One motivation for this kind of orderings is the fact that, in practice, if using a user-defined relation as the ordering, well-foundedness can be difficult or even impossible to be proved, whereas monotonicity is much easier to be guaranteed. We also need to investigate about refining redundancy notions in this context of weak orderings, to make the calculus useful in practice.

Here we list more particular open questions concerning the results we have presented. Deeper comments about most of this questions can be found in their respective chapters.

8.1 Open problems from Chapter 3

1. It becomes interesting to explore new completeness results for deduction modulo equational theories E for which monotonic, E-compatible total orderings do not exist, but that can be handled with the techniques described in Chapter 3.

2. There are results on redundancy, like the ones of Chapter 7, which directly apply to the setting we have presented in Chapter 3. On the other hand, we already know that, for the case of Horn clauses, our results are compatible with arbitrary selection strategies [BG01b]. However, when using arbitrary selection strategies, completeness can be lost unless some kind of tautologies are kept (see Example 3.6.4 of Chapter 3). Therefore, in order to improve efficiency, it should be more carefully studied to what amount our results are compatible with arbitrary selection strategies, and how can these strategies be combined with the results on redundancy.

8.2 Open problems from Chapter 4

Regarding Knuth-Bendix completion, after the results of Chapter 4 the following interesting questions remain unanswered:

1. Does Theorem 4.3.3 still hold when the strict ordered paramodulation rule of Definition 3.3.6 (where inferences at topmost positions of small sides w.r.t. $\succ$ are not needed) is used?

2. Would it hold when inferences with (or on) small sides w.r.t. $\succ_r$ are not computed at all? Or, in other words, is *superposition* w.r.t. arbitrary reduction orderings complete?

8.3 Open problems from Chapter 5

Some unanswered questions and interesting problems arising from Chapter 5 are:

1. From our result, it turns out that deduction modulo equational theories E is now always possible when E-compatibility of the ordering is needed, since, regardless of the theory E, there is always a well-founded E-compatible ordering. Therefore, new completeness results for deduction modulo built-in equational theories E should be investigated.

2. It becomes interesting to study to what amount the relaxation of the ordering properties, like requiring just well-foundedness and allowing subterm contradiction, can help in obtaining goal-oriented proofs.

3. We have presented a calculus in which, in some very special cases, inferences on variables are allowed. Are these inferences really necessary for refutation completeness?

4. Is subsumption fully compatible with the presented calculus for Horn clauses? To what amount demodulation and other more refined redundancy elimination methods can be applied?

5. Is the presented calculus compatible with equality and ordering constraint inheritance, and with the so-called *basic* strategies?

6. Is also paramodulation with well-founded orderings complete for the case of full first-order clauses? And, if this is the case, which redundancy notions are applicable in that setting?

7. Is even well-foundedness necessary for completeness of ordered paramodulation?

8.4 Open problems from Chapter 6

After Chapter 6, a main question remains open:

1. There is no evidence that our method would make theorem provers more efficient. A first task here could be, e.g., to do some modular experiments with the Saturate prover [NN93, GNN99], in combination with arbitrary provers, especially the ones that are known to be not compatible with the standard redundancy notions.

8.5 Open problems from Chapter 7

The following questions and problems arise from the results of Chapter 7:

1. In Chapter 5, we improved on the result of Chapter 3 by showing that well-foundedness of the term ordering suffices for completeness of (positive) ordered paramodulation for Horn clauses with equality. Do the redundancy notions we have presented in Chapter 7 preserve completeness in that case?

2. Although for the case of Horn clauses it has been shown that completeness can be lost if demodulation is fully applied, i.e., if markings are not introduced at all, no example justifying the need for the additional markings introduced in the case of general first order clauses has been given. So it becomes interesting to investigate finer approaches, as marking of terms will not allow much demodulation.

3. It has to be explored to what extent our results are applicable in practice. The first task here is to develop an experimental implementation of our inference systems. It should be carefully studied how the orderings which fit in our requirements behave in practice, like it happens with SPO and, in particular, it could be interesting to study if the use of non-monotonic (user supplied) orderings can sometimes improve efficiency.

Bibliography

[Bac91] Leo Bachmair. *Canonical equational proofs*. Birkhäuser, Boston, Mass., 1991.

[Bar99] R. Barták. Constraint programming: In pursuit of the holy grail. In *Proceedings of Week of Doctoral Students (WDS99), Part IV*, pages 555–564, Prague, June 1999. MatFyzPress.

[BD89] Leo Bachmair and Nachum Dershowitz. Completion for rewriting modulo a congruence. *Theoretical Computer Science*, 2 and 3(67):173–201, 1989.

[BD94] Leo Bachmair and Nachum Dershowitz. Equational inference, canonical proofs, and proof orderings. *J. of the Association for Computing Machinery*, 41(2):236–276, February 1994.

[BDH86] Leo Bachmair, Nachum Dershowitz, and Jieh Hsiang. Orderings for equational proofs. In *First IEEE Symposium on Logic in Computer Science (LICS)*, pages 346–357, Cambridge, Massachusetts, USA, June 16–18, 1986. IEEE Computer Society Press.

[BDP89] Leo Bachmair, Nachum Dershowitz, and David Plaisted. Completion Whitout Failure. In Hassan Aït-Kaci and Maurice Nivat, editors, *Resolution of Equations in Algebraic Structures*, volume 2: Rewriting Techniques, chapter 1, pages 1–30. Academic Press, New York, 1989.

[BFR00] C. Borralleras, M. Ferreira, and A. Rubio. Complete monotonic semantic path orderings. In David McAllester, editor, *Proceedings of the 17th International Conference on Automated Deduction (CADE-17)*, volume 1831 of *LNAI*, pages 346–364, Pittsburgh, USA, June 2000. Springer-Verlag.

[BG90] Leo Bachmair and Harald Ganzinger. On restrictions of ordered paramodulation with simplification. In Mark E. Stickel, editor, *10th International Conference on Automated Deduction (CADE)*, LNAI 449, pages 427–441, Kaiserslautern, FRG, July 24–27, 1990. Springer-Verlag.

[BG94a] Leo Bachmair and Harald Ganzinger. Associative-commutative superposition. In N. Dershowitz, editor, *Proc. 5th International Workshop on Conditional Term Rewriting Systems*, LNCS 968, pages 155–167, Jerusalem, July 11–15, 1994. Springer-Verlag.

[BG94b] Leo Bachmair and Harald Ganzinger. Rewrite-based equational theorem proving with selection and simplification. *Journal of Logic and Computation*, 4(3):217–247, 1994.

[BG98a] L. Bachmair and H. Ganzinger. Equational reasoning in saturation-based theorem proving. In W. Bibel and P.H. Schmitt, editors, *Automated Deduction — A Basis for Applications*, volume I, chapter 11, pages 353–397. Kluwer, 1998.

[BG98b] Leo Bachmair and Harald Ganzinger. Strict basic superposition. In Claude Kirchner and Hélène Kirchner, editors, *Proceedings of the 15th International Conference on Automated Deduction (CADE-98)*, volume 1421 of *LNAI*, pages 160–174, Berlin, July 5–10 1998. Springer.

[BG01a] Leo Bachmair and Harald Ganzinger. Resolution theorem proving. In J.A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume 1, chapter 2, pages 19–99. Elsevier Science Publishers and MIT Press, 2001.

[BG01b] Miquel Bofill and Guillem Godoy. On the completeness of arbitrary selection strategies for paramodulation. In *Automata, Languages and Programming, 28th Int. Colloquium (ICALP)*, volume 2076 of *Lecture Notes in Computer Science*, pages 951–962, Crete, Greece, July 2001. Springer-Verlag.

[BGLS92] Leo Bachmair, Harald Ganzinger, Christopher Lynch, and Wayne Snyder. Basic paramodulation and superposition. In Deepak Kapur, editor, *11th International Conference on Automated Deduction (CADE)*, LNAI 607, pages 462–476, Saratoga Springs, New York, USA, June 15–18, 1992. Springer-Verlag.

[BGLS95] L. Bachmair, H. Ganzinger, Chr. Lynch, and W. Snyder. Basic paramodulation. *Information and Computation*, 121(2):172–192, 1995.

[BGNR99] Miquel Bofill, Guillem Godoy, Robert Nieuwenhuis, and Albert Rubio. Paramodulation with non-monotonic orderings. In *14th IEEE Symposium on Logic in Computer Science (LICS)*, pages 225–233, Trento, Italy, July 2–5, 1999.

[BGNR00] Miquel Bofill, Guillem Godoy, Robert Nieuwenhuis, and Albert Rubio. Modular redundancy for theorem proving. In H. Kirchner and C. Ringeissen, editors, *Proceedings of the 3rd International Workshop on Frontiers of Combining Systems – FroCoS'2000*, volume 1794 of *Lecture Notes in Artificial Intelligence*, pages 186–199, Nancy, France, March 22–24 2000. Springer-Verlag.

[BGNR03] Miquel Bofill, Guillem Godoy, Robert Nieuwenhuis, and Albert Rubio. Paramodulation and Knuth-Bendix Completion with Nontotal and Nonmonotonic Orderings. *Journal of Automated Reasoning*, 30(1):99–120, 2003.

[BGW93] Leo Bachmair, Harald Ganzinger, and Uwe Waldmann. Superposition with simplification as a decision procedure for the monadic class with equality. In *3rd Kurt Gödel Colloquium: Computational Logic and Proof Theory*, LNCS 713, pages 83–96. SpringerVerlag, 1993.

[BR02] Miquel Bofill and Albert Rubio. Well-foundedness is sufficient for completeness of ordered paramodulation. In Andrei Voronkov, editor, *Automated Deduction – CADE-18*, volume 2392 of *Lecture Notes in Artificial Intelligence*, pages 456–470, Copenhagen, Denmark, July 27–30 2002. Springer-Verlag.

[BR04] Miquel Bofill and Albert Rubio. Redundancy notions for paramodulation with non-monotonic orderings. In *Proceedings of the 2nd International Joint Conference on Automated Reasoning (IJCAR-2004)*, Cork, Ireland, July 4–8 2004.

[Bra75] D. Brand. Proving theorems with the modification method. *SIAM Journal on Computing*, 4(4):412–430, December 1975.

[BS01] Franz Baader and Wayne Snyder. Unification theory. In J.A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume 1, chapter 8, pages 445–533. Elsevier Science Publishers and MIT Press, 2001.

[Che86] Philippe Le Chenadec. *Canonical Forms in Finitely Presented Algebras*. Research Notes in Theoretical Computer Science. Pitman, London; Wiley, New York, Toronto, 1986.

[CN00] H. Comon and R. Nieuwenhuis. Induction = I-axiomatization + first-order consistency. *Information and Computation*, 2000. To appear.

[CNNR98] Hubert Comon, Paliath Narendran, Robert Nieuwenhuis, and Michael Rusinowitch. Decision problems in ordered rewriting. In *13th IEEE Symposium on Logic in Computer Science (LICS)*, pages 410–422, Indianapolis, USA, June 27–30, 1998.

[Com01] Hubert Comon. Inductionless induction. In J.A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*. Elsevier Science Publishers and MIT Press, 2001.

[Dar68] J. Darlington. Automatic theorem proving with equality substitutions and mathematical induction. *Machine Intelligence*, 3:113–127, 1968.

[Deg79] Anatoli Degtyarev. The monotonic paramodulation strategy. In *5th All-Union Conference on Mathematical Logic*, Novosibirsk, 1979. (In Russian).

[Der82] Nachum Dershowitz. Orderings for term-rewriting systems. *Theoretical Computer Science*, 17(3):279–301, 1982.

[Der87] Nachum Dershowitz. Termination of rewriting. *Journal of Symbolic Computation*, 3:69–116, 1987.

[Der91] Nachum Dershowitz. Canonical sets of Horn clauses. In J. Leach Albert, B. Monien, and M. Rodríguez Artalejo, editors, *Proceedings of the Eighteenth International Colloquium on Automata, Languages and Programming (ICALP)*, LNCS 510, pages 267–278, Madrid, Spain, July 1991. Springer-Verlag.

[Dev90] Hervé Devie. Linear completion. In Stéphane Kaplan and Mitsuhiro Okada, editors, *Conditional and Typed Rewriting Systems, 2nd International Workshop*, LNCS 516, pages 233–245, Montreal, Canada, June 11–14, 1990. Springer-Verlag.

[Dev92] Hervé Devie. PhD thesis, Université de Paris-Sud, Orsay, France, 1992.

[DJ90] Nachum Dershowitz and Jean-Pierre Jouannaud. Rewrite systems. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B: Formal Models and Semantics, chapter 6, pages 244–320. Elsevier Science Publishers B.V., Amsterdam, New York, Oxford, Tokyo, 1990.

[DM79] Nachum Dershowitz and Zohar Manna. Proving termination with multiset orderings. *Comm. of ACM*, 22(8), 1979.

[DMT88] Nachum Dershowitz, Leo Marcus, and Andrzej Tarlecki. Existence, uniqueness, and construction of rewrite systems. *SIAM Journal on Computing*, 17(4):629–639, 1988.

[Dom92] Eric Domenjoud. A technical note on AC-unification. the number of minimal unifiers of the equation $\alpha x_1 + \ldots + \alpha x_p = \beta y_1 + \ldots + \beta y_q$. *Journal of Automated Reasoning*, 8(1):39–44, 1992.

[DP60] Martin Davis and Hilary Putnam. A computing procedure for quantification theory. *Journal of the ACM*, 7:201–215, 1960.

[DP01] Nachum Dershowitz and David Plaisted. Rewriting. In J.A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume 1, chapter 9, pages 535–610. Elsevier Science Publishers and MIT Press, 2001.

[DV86] Anatoli Degtyarev and Andrei Voronkov. Equality methods in machine theorem proving. *Cybernetics*, 22(3):298–307, 1986.

[DV01] Anatoli Degtyarev and Andrei Voronkov. Equality-reasoning in sequent-based calculi. In J.A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume 1, chapter 10, pages 609–704. Elsevier Science Publishers and MIT Press, 2001.

[Gal85] H. Gallaire. Logic programming: Further developments. In *Proceedings of the International Symposium on Logic Programming*, pages 88–99. IEEE Computer Society, Technical Committee on Computer Languages, The Computer Society Press, July 1985.

[Gan91] Harald Ganzinger. A completion procedure for conditional equations. *Journal of Symbolic Computation*, 11:51–81, 1991.

[GN87] Michael R. Genesereth and Nils J. Nilsson. *Logical Foundations of Artificial Intelligence*. Morgan Kaufmann, Los Altos, 1987.

[GN00] Guillem Godoy and Robert Nieuwenhuis. Paramodulation with built-in abelian groups. In *15th IEEE Symposium on Logic in Computer Science (LICS)*, pages 413–424, Santa Barbara, USA, June 26–29, 2000. IEEE Computer Society Press.

[GN01] Guillem Godoy and Robert Nieuwenhuis. On ordering constraints for deduction with built-in abelian semigroups, monoids and groups. In *Proceedings of the 16th Annual IEEE Symposium on Logic in Computer Science (LICS-01)*, pages 38–50, Los Alamitos, CA, June 16–19 2001. IEEE Computer Society.

[GN03] Guillem Godoy and Robert Nieuwenhuis. Paramodulation with completely built-in abelian groups. *Journal of Symbolic Computation*, 2003. To appear.

[GNN99] Harald Ganzinger, Robert Nieuwenhuis, and Pilar Nivela. The Saturate System, 1999. Available at: `http://www.mpi-sb.mpg.de/SATURATE/Saturate.html`.

[GW96] Harald Ganzinger and Uwe Waldmann. Theorem proving in cancellative abelian monoids. In M. A. McRobbie and J. K. Slaney, editors, *Thirteenth International Conference on Automated Deduction (CADE)*, volume 1104 of *LNAI*, pages 388–402, Berlin, July30 August–3 1996. Springer.

[HBVL97] Thomas Hillenbrand, Arnim Buch, Roland Vogt, and Bernd Löchner. WALDMEISTER — High-performance equational deduction. *Journal of Automated Reasoning*, 18(2):265–270, April 1997.

[Hil03] Th. Hillenbrand. CITIUS ALTIUS FORTIUS: lessons learned from the theorem prover WALDMEISTER (invited paper). In I. Dahn and L. Vigneron, editors, *Proceedings of the 4th International Workshop on First-Order Theorem Proving*, volume 86.1 of *Electronic Notes in Theoretical Computer Science*. Elsevier Science, 2003. Available at: `http://agent.informatik.uni-kl.de/waldmeister`.

[HR86] J. Hsiang and M. Rusinowitch. A new method for establishing refutational completeness in theorem proving. In Jörg H. Siekmann, editor, *Proceedings of the 8th International Conference on Automated Deduction*, volume 230 of *LNCS*, pages 141–152, Oxford, UK, July 1986. Springer.

[HR87] J. Hsiang and M. Rusinowitch. On word problems in equational theories. In T. Ottmann, editor, *Proc. 14th Int. Colloquium Automata, Languages and Programming*, LNCS 267, pages 54–71, Berlin, Germany, 1987. Springer-Verlag.

[HR91] J. Hsiang and M Rusinowitch. Proving refutational completeness of theorem proving strategies: the transfinite semantic tree method. *Journal of the ACM*, 38(3):559–587, jul 1991.

[HS88] M. Höhfeld and G. Smolka. Definite relations over constraint languages. Research Report 53, IBM Deutschland, WT LILOG, October 1988. To appear in Journal of Logic Programming.

[Hue72] G. Huet. *Constrained Resolution: A complete method for higher-order logic*. PhD thesis, Case Western Reserve University, 1972.

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

[JK86] Jean-Pierre Jouannaud and Hélène Kirchner. Completion of a set of rules modulo a set of equations. *SIAM Journal of Computing*, 15:1155–1194, 1986.

[JL87] Joxan Jaffar and Jean-Louis Lassez. Constraint logic programming. In *Fourteenth Annual ACM Symposium on Principles of Programming Languages*, pages 111–119, Munich, West Germany, January 21–23, 1987.

[JM94] Joxan Jaffar and Michael J. Maher. Constraint logic programming: A survey. *Journal of Logic Programming*, 19/20:503–581, 1994.

[Jou83] Jean-Pierre Jouannaud. Confluent and coherent equational term rewriting systems: Applications to proofs in abstract data types. In *Proc. 8th Colloquium on Trees in Algebra and Programming*, LNCS 59, pages 269–283. Springer-Verlag, 1983.

[JW86] Jean-Pierre Jouannaud and Bernard Waldmann. Reductive conditional term rewriting systems. In *Proc. Third IFIP Working Conference on Formal Description of Programming Concepts*, Ebberup, Denmark, 1986.

[Kan63] S. Kanger. A simplified proof method for elementary logic. In P. Braffort and D. Hirschberg, editors, *Computer Programming and Formal Systems*, pages 87–94. North-Holland, 1963. Reprinted as [Kan83].

[Kan83] S. Kanger. A simplified proof method for elementary logic. In J. Siekmann and G. Wrightson, editors, *Automation of Reasoning. Classical Papers on Computational Logic*, volume 1, pages 364–371. Springer-Verlag, 1983. Originally published as [Kan63].

[Kap84] Stephane Kaplan. Conditional rewrite rules. *Theoretical Computer Science*, 33:175–193, 1984.

[KB70] D.E. Knuth and P.B. Bendix. Simple word problems in universal algebras. In *J. Leech, ed., Computational Problems in Abstract Algebra*, pages 263–297. Pergamon Press, Oxford, 1970.

[KKR90] Claude Kirchner, Hélène Kirchner, and Michaël Rusinowitch. Deduction with symbolic constraints. *Revue Française d'Intelligence Artificielle*, 4(3):9–52, 1990.

[KL80] S. Kamin and J.-J. Levy. Two generalizations of the recursive path ordering. Unpublished note, Dept. of Computer Science, Univ. of Illinois, Urbana, IL, 1980.

[KN92] Deepak Kapur and Paliath Narendran. Double exponential complexity of computing complete sets of AC-unifiers. In *Seventh Annual IEEE Symposium on Logic in Computer Science*, pages 11–21, Santa Cruz, California, USA, June 22–25, 1992. IEEE Computer Society Press.

[KR91] Emmanuel Kounalis and Michael Rusinowitch. On word problems in Horn theories. *Journal of Symbolic Computation*, 11(1–2):113–128, 1991.

[Kru60] Joseph B. Kruskal. Well-quasi-ordering, the Tree Theorem, and Vazsonyi's conjecture. *Transactions of the American Mathematical Society*, 95:210–225, May 1960.

[Lan75] D. S. Lankford. Canonical inference. Technical Report ATP-32, Dept. of Mathematics and Computer Science, Univ. of Texas, Austin, TX, 1975.

[LB77] D. S. Lankford and A. M. Ballantyne. Decision procedures for simple equational theories with commutative-associative axioms: Complete sets of commutative-associative reductions. Technical Report Memo ATP-39, Dept. of Mathematics and Computer Science, Univ. of Texas, Austin, TX, August 1977.

[Lov78] Donald W. Loveland. *Automated Theorem Proving: a Logical Basis*. North-Holland, Amsterdam, 1 edition, 1978.

[LS93] Chris Lynch and Wayne Snyder. Redundancy criteria for constrained completion. In C. Kirchner, editor, *5th International Conference on Rewriting Techniques and Applications (RTA)*, LNCS 690, pages 2–16, Montreal, Canada, June 16–18, 1993. Springer-Verlag.

[Lus92] Ewing L. Lusk. Controlling redundancy in large search spaces: Argonne-style theorem proving through the years. In A. Voronkov, editor, *Proceedings of the International*

Conference on Logic Programming and Automated Reasoning (LPAR'92), volume 624 of *LNAI*, pages 96–106, St. Petersburg, Russia, July 1992. Springer Verlag.

[Lyn97] C. Lynch. Oriented equational logic programming is complete. *Journal of Symbolic Computation*, 23(1):23–46, January 1997.

[Mah92] Michael J. Maher. A CLP view of logic programming. In Hélène Kirchner and Giorgio Levi, editors, *Algebraic and Logic Programming, Third International Conference,*, volume 632 of *lncs*, pages 364–383, Volterra, Italy, 2–4 September 1992. Springer.

[Mar91] Claude Marché. On ground AC-completion. In Ronald V. Book, editor, *4th Int. Conf. Rewriting Techniques and Applications (RTA)*, LNCS 488, pages 411–422, Como, Italy, April 10–12, 1991. Springer-Verlag.

[Mar94] Claude Marche. Normalised Rewriting and Normalised Completion. In *Ninth Annual IEEE Symposium on Logic in Computer Science*, Paris, France, July 1994. IEEE Computer Society Press, Los Alamitos, CA, USA.

[Mar96] Claude Marché. Normalized rewriting: An alternative to rewriting Modulo a set of equations. *Journal of Symbolic Computation*, 21(3):253–288, March 1996.

[McC94] William McCune. Otter 3.0 Reference Manual and Guide. Technical Report ANL-94/6, Argonne National Laboratory, 1994. Software and documentation available at `http://www-unix.mcs.anl.gov/AR/otter`.

[McC97a] W. McCune. 33 basic test problems: A practical evaluation of some paramodulation strategies. In Robert Veroff, editor, *Automated Reasoning and its Applications: Essays in Honor of Larry Wos*, pages 71–114. MIT Press, 1997.

[McC97b] W. McCune. Well behaved search and the Robbins problem. In H. Comon, editor, *8th International Conference on Rewriting Techniques and Applications (RTA)*, LNCS 1232, pages 1–7, Sitges, Spain, 1997. Springer-Verlag.

[McC97c] William McCune. Solution of the Robbins problem. *Journal of Automated Reasoning*, 19(3):263–276, December 1997.

[MIL+97] Max Moser, Ortrun Ibens, Reinhold Letz, Joachim Steinbach, Christoph Goller, Johann Schumann, and Klaus Mayr. SETHEO and E-SETHEO—the CADE-13 systems. *Journal of Automated Reasoning*, 18(2):237–246, April 1997. Additional information available at: `http://www4.informatik.tu-muenchen.de/~schulz/WORK/e-setheo.html`.

[MN90] Ursula Martin and Tobias Nipkow. Ordered rewriting and confluence. In Mark E. Stickel, editor, *10th International Conference on Automated Deduction (CADE)*, LNAI 449, pages 366–380, Kaiserslautern, FRG, July 24–27, 1990. Springer-Verlag.

[NN91] Robert Nieuwenhuis and Pilar Nivela. Efficient deduction in equality horn logic by horn-completion. *Information Processing Letters*, 39(1):1–6, July 1991.

[NN93] Pilar Nivela and Robert Nieuwenhuis. Practical results on the saturation of full first-order clauses: Experiments with the saturate system. (system description). In C. Kirchner, editor, *5th International Conference on Rewriting Techniques and Applications (RTA)*, LNCS 690, pages 436–440, Montreal, Canada, June 16–18, 1993. Springer-Verlag.

[NO90] Robert Nieuwenhuis and Fernando Orejas. Clausal rewriting. In Stéphane Kaplan and Mitsuhiro Okada, editors, *Conditional and Typed Rewriting Systems, 2nd International Workshop*, LNCS 516, pages 246–258, Montreal, Canada, June 11–14, 1990. Springer-Verlag.

[NR91] Paliath Narendran and Michael Rusinowitch. Any ground associative commutative theory has a finite canonical system. In *4th Int. Conf. Rewriting Techniques and Applications (RTA)*, LNCS 488, pages 423–434, Como, Italy, April 10–12, 1991. Springer-Verlag.

[NR92a] Robert Nieuwenhuis and Albert Rubio. Basic superposition is complete. In B. Krieg-Brückner, editor, *European Symposium on Programming*, LNCS 582, pages 371–390, Rennes, France, February 26–28, 1992. Springer-Verlag.

[NR92b] Robert Nieuwenhuis and Albert Rubio. Theorem proving with ordering constrained clauses. In Deepak Kapur, editor, *11th International Conference on Automated Deduction (CADE)*, LNAI 607, pages 477–491, Saratoga Springs, New York, 1992.

[NR93] Paliath Narendran and Michael Rusinowitch. The Unifiability Problem in Ground AC Theories. In *Eighth Annual IEEE Symposium on Logic in Computer Science*, pages 364–370, Montreal, canada, June 19–23, 1993. IEEE Computer Society Press.

[NR94] Robert Nieuwenhuis and Albert Rubio. AC-Superposition with constraints: No AC-unifiers needed. In Allan Bundy, editor, *12th International Conference on Automated Deduction (CADE)*, LNAI 814, pages 545–559, Nancy, France, June 1994. Springer-Verlag.

[NR95] Robert Nieuwenhuis and Albert Rubio. Theorem Proving with Ordering and Equality Constrained Clauses. *Journal of Symbolic Computation*, 19(4):321–351, April 1995.

[NR97] Robert Nieuwenhuis and Albert Rubio. Paramodulation with Built-in AC-Theories and Symbolic Constraints. *Journal of Symbolic Computation*, 23(1):1–21, May 1997.

[NR01] Robert Nieuwenhuis and Albert Rubio. Paramodulation-based theorem proving. In J.A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume 1, chapter 7, pages 372–444. Elsevier Science Publishers and MIT Press, 2001.

[Pau92] E. Paul. A general refutational completeness result for an inference procedure based on associative-commutative unification. *Journal of Symbolic Computation*, 14(6):577–618, 1992.

[Pet83] Gerald E. Peterson. A technique for establishing completeness results in theorem proving with equality. *SIAM J. on Computing*, 12(1):82–100, February 1983.

[Pet90] Gerald E. Peterson. Complete sets of reductions with constraints. In Mark E. Stickel, editor, *10th International Conference on Automated Deduction (CADE)*, LNAI 449, pages 381–395, Kaiserslautern, FRG, July 24–27, 1990. Springer-Verlag.

[Pla93] David A. Plaisted. Equational reasoning and term rewriting systems. In Dov Gabbay, Christopher Hogger, and J. A. Robinson, editors, *The Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 1: Deductive Methodologies*, pages 274–367. Oxford University Press, Oxford, 1993.

[Plo72] Gordon Plotkin. Building in equational theories. *Machine Intelligence*, 7:73–90, 1972.

[PP91] J. Pais and G. Peterson. Using Forcing to Prove Completeness of Resolution and Paramodulation. *Journal of Symbolic Computation*, 11(1):3–19, 1991.

[PS81] G.E. Peterson and M.E. Stickel. Complete sets of reductions for some equational theories. *Journal Assoc. Comput. Mach.*, 28(2):233–264, 1981.

[RN95] Albert Rubio and Robert Nieuwenhuis. A total AC-compatible ordering based on RPO. *Theoretical Computer Science*, 142(2):209–227, May 15, 1995.

[Rob65] J.A. Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12(1):23–41, January 1965.

[Rob68] J. A. Robinson. The generalized resolution principle. In *Machine Intelligence*, volume 3, pages 77–93. Oliver and Boyd, Edinburgh, 1968. Reprinted in [SW83].

[Ros99] Francesca Rossi. Constraint (Logic) Programming: A Survey on Research and Applications. In *New Trends in Constraints: Joint ERCIM/Compulog Net Workshop, Paphos, Cyprus, October 1999. Selected Papers*, volume 1865 of *LNAI*, pages 40–74. Springer, 1999.

[RTA01] The RTA list of open problems, 2001.
Maintained at `http://www.lri.fr/~rtaloop/` (by Ralf Treinen).

[Rus91] Michael Rusinowitch. Theorem-proving with resolution and superposition. *Journal of Symbolic Computation*, 11(1–2):21–49, January–February 1991.

[RV95] Michael Rusinowitch and Laurent Vigneron. Automated deduction with associative commutative operators. *J. of Applicable Algebra in Engineering, Communication and Computation*, 6(1):23–56, 1995.

[RV02] Alexandre Riazanov and Andrei Voronkov. The design and implementation of Vampire. *AI Communications*, 15(2–3):91–110, 2002. Software and documentation available at: `http://www.cs.man.ac.uk/~riazanoa/Vampire`.

[RW69] G.A. Robinson and L.T. Wos. Paramodulation and theorem-proving in first order theories with equality. *Machine Intelligence*, 4:135–150, 1969.

[Sla74] James R. Slagle. Automated theorem-proving for theories with simplifiers, commutativity, and associativity. *Journal of the ACM*, 21(4):622–642, October 1974.

[SS98] Geoff Sutcliffe and Christian B. Suttner. The CADE-14 ATP system competition. Technical Report JCU-CS-98/01, Department of Computer Science, James Cook University, 3 March 1998.

[Stu98] Jürgen Stuber. Superposition theorem proving for abelian groups represented as integer modules. *Theoretical Computer Science*, 208(1–2):149–177, 28 November 1998.

[SW83] Jörg Siekmann and Graham Wrightson, editors. *Automation of Reasoning: Classical Papers in Computational Logic 1967–1970*, volume 2. Springer-Verlag, 1983.

[Toy87] Yoshihito Toyama. Counterexamples to termination for the direct sum of term rewriting systems. *Information Processing Letters*, 25(3):141–143, May 29, 1987.

[Vig94] Laurent Vigneron. Associative Commutative Deduction with constraints. In Allan Bundy, editor, *12th International Conference on Automated Deduction (CADE)*, LNAI 814, pages 530–544, Nancy, France, June 1994. Springer-Verlag.

[Wal98] Uwe Waldmann. Superposition for divisible torsion-free abelian monoids. In *Proceedings of the Fifteenth International Conference on Automated Deduction (CADE-98)*, volume 1421 of *LNAI*, pages 144–159, Berlin, 1998. Springer.

[Wal99] Uwe Waldmann. Cancellative superposition decides the theory of divisible torsion-free abelian groups. In *Logic Programming and Automated Reasoning, Int. Conf.*, LNAI 1705, pages 131–147, Tiblisi, Georgia, sept 1999. Springer-Verlag.

[Wan60] Hao Wang. Toward mechanical mathematics. *IBM Journal of Research and Development*, 4:2–22, 1960.

[WBH+02] Christoph Weidenbach, Uwe Brahm, Thomas Hillenbrand, Enno Keen, Christian Theobalt, and Dalibor Topic. SPASS version 2.0. In Andrei Voronkov, editor, *Automated Deduction – CADE-18*, volume 2392 of *Lecture Notes in Computer Science*, pages 275–279. Springer-Verlag, July 27-30 2002. Software and documentation available at: `http://spass.mpi-sb.mpg.de`.

[Wec91] W. Wechler. *Universal Algebra for Computer Scientists*, volume 25 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, Berlin, 1991.

[Wer92] Ulrich Wertz. First-order theorem proving modulo equations. Technical Report MPI-I-92-216, Max-Planck-Institut für Informatik, Saarbrücken, April 1992.

[Wos88] Larry Wos. *Automated Reasoning: 33 Basic Research Problems*. Prentice-Hall, Englewood Cliffs, 1988.

[Wos96] Larry Wos. *The Automation of Reasoning: An Experimenter's Notebook with OTTER Tutorial*. Academic Press, 1996.

[WRCS67] Lawrence Wos, George Robinson, Daniel F. Carson, and Leon Shalla. The concept of demodulation in theorem proving. *Journal of the ACM*, 14(4):698–709, October 1967.

[Zha88] H Zhang. *Reduction, Superposition, and Induction: Automated Reasoning in an Equational Logic*. PhD thesis, Renselaer Polytechnic Institute, 1988.

[Zha93] Hantao Zhang. A case study of completion modulo distributivity and Abelian groups. In Claude Kirchner, editor, *Proceedings of the 5th International Conference on Rewriting Techniques and Applications (RTA-93)*, volume 690 of *LNCS*, pages 32–46. Springer-Verlag, June 16–18 1993.

[ZK88] Hantao Zhang and Deepak Kapur. First-order theorem proving using conditional equations. In E. Lusk and R. Overbeek, editors, *Proceedings of the 9th International Conference on Automated Deduction (Argonne, Illinois)*, volume 310 of *LNCS*, pages 1–20, Berlin, May 1988. Springer-Verlag.

Abbreviations

a.k.a.	also known as
cf.	*confer* compare
e.g.	*exempli gratia* for example, for instance
i.e.	*id est* which is to say, in other words
iff	if and only if
resp.	respectively
s.t.	such that
w.l.o.g.	without loss of generality
w.r.t.	with respect to

Index