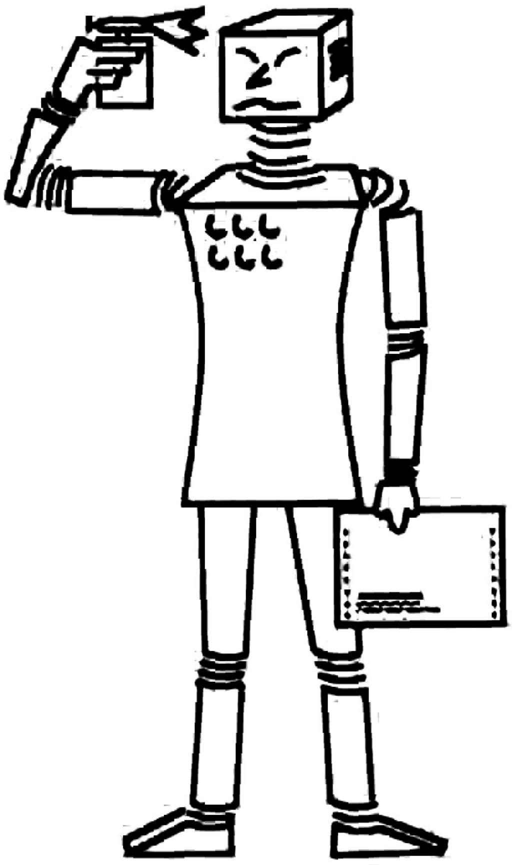
Fachbereich Informatik
Universität Kaiserslautern
D-67663 Kaiserslautern

SEKI - REPORT



How to Prove Ground Confluence

Klaus Becker

SEKI Report SR–96–02

How to Prove Ground Confluence

Klaus Becker

Universität Kaiserslautern,67663 Kaiserslautern, Germany
email: klbecker@informatik.uni-kl.de

Abstract

We show how to prove ground confluence of term rewrite relations that are induced by reductive systems of clausal rewrite rules. According to a well-known critical pair criterion it suffices for such systems to prove ground joinability of a suitable set of 'critical clauses'. We outline how the latter can be done in a systematic fashion, using mathematical induction as a key concept of reasoning.

1 Introduction and Motivation

The notion of (ground) confluence is of great importance in the field of term rewriting: If the rewrite relation in focus is (ground) confluent, then one knows that term rewriting — considered as a computation mechanism — provides unique computation results. Note that the supplement "ground" is added if term rewriting is considered on ground terms (i.e. terms without variables) only. This restricted notion of confluence suffices in many applications/situations. Generally, ground confluence is easier to achieve than full confluence: There are rewrite systems inducing a rewrite relation which is ground confluent, but not confluent. However, ground confluence is much harder to prove than confluence: Even in the case of finite Noetherian unconditional rewrite systems, ground confluence is an undecidable notion (see [KNO90]), whereas confluence can easily be tested by a critical pair criterion.

This paper is about how to prove ground confluence. We assume that we are given a system $\mathcal{R}$ of positive/negative conditional rewrite rules (we call them clausal rules), inducing a terminating rewrite relation on the ground terms of the specification language. Due to a well-known critical pair criterion it then suffices to prove ground joinability of the set of "critical" clauses induced by $\mathcal{R}$. Our approach is based on this criterion, providing techniques in order to verify ground joinability of a set of clauses. These techniques are mainly based on the following reasoning concepts: (1) case splitting, (2) induction based reasoning, (3) contextual simplification and elimination. We briefly illustrate the relevance of these reasoning techniques for the problem of discourse by some simple examples.

The first example is to demonstrate case splitting and inductive reasoning.

Example 1 *Let $\mathcal{R}$ consist of the following rewrite rules:*

$$\begin{array}{llcll}
(R1) & & \Rightarrow & \mathit{even}(0) & \rightarrow t \\
(R2) & \mathit{even}(x) \neq t & \Rightarrow & \mathit{even}(s(x)) & \rightarrow t \\
(R3) & & \Rightarrow & \mathit{even}(s(s(x))) & \rightarrow \mathit{even}(x)
\end{array}$$

This rewrite system induces the critical clause $C : \Rightarrow \mathit{even}(x) = t, \mathit{even}(s(x)) = t$. The comma is to be understood as a logical '$\vee$'. Proving ground joinability of this clause C cannot be done in a uniform 'schematized' fashion, using the same reduction steps for all ground instances of C. E.g., concerning the instantiation $x \mapsto 0$, the first equation is joinable whereas the second equation is not. Concerning the instantiation $x \mapsto s(0)$, the second equation is joinable whereas the first equation is not. One can directly see that different ground instances of C require different joinability proofs. Our proof method consists in generating a finite number of cases such that (i) there is a schematized joinability proof for every case and (ii) every (relevant) ground instance is covered by one of the cases. In the present example it is convenient to consider the case splitting $\{x \mapsto 0, x \mapsto s(x')\}$. Below we briefly comment on the completeness of such case splittings. The first case has already been discussed above. So we only have to consider the second one. Instantiation of C with $x \mapsto s(x')$ leads to the clause $C' : \Rightarrow \mathit{even}(s(x')) = t, \mathit{even}(s(s(x'))) = t$. In order to simplify C' we perform a schematized rewrite step using $R3$. One obtains the clause $C'' : \Rightarrow \mathit{even}(s(x')) = t, \mathit{even}(x') = t$. This clause C'' has the same structure as the original clause C. Thus, we have traced back every instance of C wrt. an instantiation $x \mapsto s(x')$ to an instance of C wrt. an instantiation $x \mapsto x'$. Using an inductive argument — in the paper we are going to develop the related theory — we can complete the proof.

In order to show the relevance of contextual simplification techniques, we briefly outline another example.

Example 2 *Let $\mathcal{R}$ consist of the following rules:*

$$\begin{array}{llcll}
(R1) & & \Rightarrow & \mathit{even}(0) & \rightarrow t \\
(R2) & \mathit{even}(x) \neq f & \Rightarrow & \mathit{even}(s(x)) & \rightarrow f \\
(R3) & \mathit{even}(x) \neq t & \Rightarrow & \mathit{even}(s(x)) & \rightarrow t
\end{array}$$

The critical clause induced by $\mathcal{R}$ is $C : \Rightarrow f = t, \mathit{even}(x) = f, \mathit{even}(x) = t$. First note that the equation $f = t$ can be eliminated because no rewrite step is possible. We consider the cases $x \mapsto 0$ and $x \mapsto s(x')$. The first case is simple. The second one leads to the clause $C' : \Rightarrow \mathit{even}(s(x')) = f, \mathit{even}(s(x')) = t$. We want to use the rewrite rule $R2$ in order to simplify the first conclusion equation and thus to finish the proof. To do so we have to verify that the (suitably instantiated) condition of $R2$ is satisfied within the present proof context. In the paper we show that it suffices to verify ground joinability of the "condition" clause $C'' : \mathit{even}(x') = f \Rightarrow \mathit{even}(s(x')) = t$. This can be understood by writing C'' in the (logically equivalent) form $\mathit{even}(s(x')) \neq t \Rightarrow \mathit{even}(x') \neq f$: One has to show that within the context $\mathit{even}(s(x')) \neq t$, the condition $\mathit{even}(x') \neq t$ of $R2$ is satisfied. Actually, this can be done successfully by the means described above (see also the appendix for a complete discussion).

To summarize, our — even very simple — examples show that one needs strong reasoning concepts in order to infer ground confluence of the rewrite relation in focus. Next we briefly comment on the problems which arise with these reasoning concepts and on our solutions.

Firstly, the generation of case splitting is a non-trivial problem. The main difficulty is the verification of its completeness (i.e.: all relevent instantiations have to be covered). In this paper we describe the case splittings of interest on a rather abstract level such that the techniques developed in the literature (like narrowing with $\mathcal{R}$ into a condition equation) are covered. In our examples (see e.g. above) we make use of the fact that our approach is hierarchical, using "constructor-based" instantiations. But, our approach is not limited to such restricted instantiations, it is also applicable if one is interested in arbitrary instantiations. One possibility is to show additionally that the non-constructor operations are totally defined wrt. the constructor domains. This can be done with the same reasoning techniques we employ in order to show ground joinability of clauses (see e.g. [Be93]). Another possibility is to declare all operations to be constructors.

Secondly, equational simplification steps may cause problems when verifying joinability: To see this note that correctness proofs for the transformation steps usually proceed indirectly by showing that one does not lose all counterexamples (for the respective notion). Now let $C : s = t \Rightarrow \Box$ be a given ground clause. We assume that C is not joinable wrt. the given rewrite relation $\longrightarrow$. Thus $s = t$ is joinable wrt. $\longrightarrow$. If we perform a rewrite step $s \longrightarrow s'$ in order to simplify C, then $s' = t$ may not be joinable wrt. $\longrightarrow$ because $\longrightarrow$ need not be ground confluent. As a consequence, the resulting clause $C' : s' = t \Rightarrow \Box$ may become joinable wrt. $\longrightarrow$. Thus, by choosing the wrong rewrite path one may lose a counterexample for joinability . In order to solve this problem we are going to introduce the concept of "confluence below a given bound": If $\longrightarrow$ is known to be confluent below a bound b and if C is bounded by b in the sense that every term occuring in C is smaller than b, then C' is a counterexample for joinability too. In the paper we develop a theory on how to handle such confluence bounds. Note that we have to handle them carefully as we assume confluence (up to such bounds) in order to show confluence.

Thirdly, the design of correct simplification techniques causes problems (additionally to those discussed above) because one has to combine equational reasoning, induction-based reasoning and contextual reasoning. The combination of these techniques is known from inductive theorem proving. But, due to the necessity of handling confluence bounds, one is faced with more complex argumentations. In order to facilitate them and to make things more transparent, we develop an abstract framework for inductive reasoning which enables one as well to model contextual reasoning on a conceptual level. This framework allows different "concrete" instantiations, of which one is well suited for our purposes. More precisely, the framework developed in the paper abstractly models a method which allows one to prove a property P (wrt. certain syntactic constructs) by mathematical induction wrt. a given well-founded ordering $\succ_i$. Once the framework is developed, we only have to instantiate it in the right way in order to obtain a method for proving ground joinability of a set of clauses. E.g., the concretization of the predicate P captures the fact that a ground clause C is joinable wrt. the rewrite relation $\longrightarrow_{(\Sigma_0,\mathcal{R})}$ provided this rewrite relation $\longrightarrow_{(\Sigma_0,\mathcal{R})}$ is confluent below the bound b related to C. The performance of contextual reasoning is also solved on the abstract level. We design a calculus which uses conditional derivations as a tool for modeling strong contextual reasoning. We follow here the idea (see the example above) that contextual simplification techniques are transparently designed by handling conditions — which are to be verified contextually — as additional "verification goals". The latter goals then can be treated in just the same way as the original "verification goals".

To summarize, our approach handles the complexity of the matter by designing a proof method in a strict conceptual and top-down fashion. For that purpose we develop new conceptual tools like 'abstract inductive theorem proving' and 'contextual simplification by conditional derivations'. These tools allow one to design a method for verifying ground confluence of clausal (positive/negative conditional) rewriting which is, on the one hand, transparent and, on the other hand, very powerful.

We briefly comment on related work. Note that we do not consider syntactic confluence criteria here (see [Wi95] for a recent overview). There is only little special work on proving ground confluence in the literature. In [Pl85] one can find a semantic method which is in principle very powerful, which however indicates no way to automatize the ground confluence test. [Fr86, Ga87] use Knuth-Bendix-like completion. This method has the disadvantage of producing often too many new consequences which are not needed for a successful proof. The work in [Go87, KoRu91, Be93] is based on mathematical induction. This method seems to be most adequate for the problem of discourse. In the present paper we follow the latter direction. We improve the techniques developed in [Go87, KoRu91, Be93] such that proofs can be performed which are not possible in [Go87, KoRu91, Be93]. On the one hand, we extend the range of the proof method to the very general class of clausal rewrite systems. On the other hand, we refine the proof techniques presented in [Go87, KoRu91, Be93].

The paper is organized as follows. Section 2 introduces the basic notions and results. In section 3 we develop an abstract framework for proofs by mathematical induction. This abstract framework is suitably instantiated in section 4 for the purpose of proving ground joinability of clauses. We assume that the reader is familiar with the basic notions of term rewriting (see e.g. [DeJo90, Av95]).

2 Basic notions and results

2.1 Syntactic notions

A *signature* is a triple $\Sigma = (S, F, \alpha)$ consisting of a set S of sort symbols, a set F of function symbols and a mapping α that assigns to every function symbol $f \in F$ an arity declaration $f : s_1, \ldots, s_n \to s$ (with $s_i, s \in S$). Whenever Σ is a given signature, we assume that we are given a system $(V_s)_{s \in S}$ of disjoint sets of variables for the sorts of Σ. Let V be the union of all these sets of variables. Furthermore, we assume that we are given a sub-signature $\Sigma_0 = (S, F_0, \alpha|_{F_0})$ of Σ. The latter is to provide the 'constructors'. Let $T(\Sigma, V)$ (resp. $T(\Sigma_0, V)$) denote the set of terms induced by Σ (resp. Σ_0) and V and let $T(\Sigma)$ (resp. $T(\Sigma_0)$) denote the set of ground terms induced by Σ (resp. Σ_0). In addition to the terms induced by Σ and V we assume that there are two pseudo-ground terms $\top$ and $\bot$. The latter are introduced for technical reasons.

Equational formulae over a given signature Σ are defined in terms of multisets. An *equation* E (*over* Σ) is a multiset $\{s, t\}$, usually written $E : s = t$, consisting of two terms of the same sort. A *clause* C (*over* Σ) is a pair (Φ, Ψ) of multisets of equations, written $C : \Phi \Rightarrow \Psi$. We call Φ the *antecedent* and Ψ the *succedent* of the clause $\Phi \Rightarrow \Psi$. In the sequel we write Φ_1, Φ_2 and Φ, L instead of $\Phi_1 \cup \Phi_2$ and $\Phi \cup \{L\}$. A clause $A_1, \ldots, A_m \Rightarrow B_1, \ldots, B_n$ represents an implication $A_1 \wedge \ldots \wedge A_m \Rightarrow B_1 \vee \ldots \vee B_n$. We denote the empty multiset by $\Box$. So $\Box \Rightarrow \Psi$ denotes a clause with an empty antecedent. We also write such a clause in the form $\Rightarrow \Psi$ or even simply Ψ. Finally, if $C : \Phi \Rightarrow \Psi$ is a given clause, then *Mult*(C) denotes the multiset of all terms s with $s \in A \in \Phi, \Psi$.

A *directed equation* (*over* Σ) is a pair (u, v) of terms of the same sort, written $u \rightarrow v$. A *directed clause* (*over* Σ) is obtained from a clause by directing exactly one succedent equation. We write $\Gamma \Rightarrow u \rightarrow v, \Delta$ to indicate such a directed clause. Here $u \rightarrow v$ is the particular directed succedent equation. The term $u = lhs(R)$ is called the *left-hand side* of $R : \Gamma \Rightarrow u \rightarrow v, \Delta$. A *clausal rule* (*over* Σ) is a directed clause $R : \Gamma \Rightarrow u \rightarrow v, \Delta$. Note that clausal rules are often presented in a positive/negative conditional fashion (turning succedent equations into antecedent disequations). A *term rewriting system* (or briefly TRS) is a pair $(\Sigma, \mathcal{R})$, consisting of a signature Σ and a system $\mathcal{R}$ of clausal rules over Σ.

Term rewriting is captured abstractly by a binary relation on $T(\Sigma)$. The definition of the concrete rewrite relation will be given in section 2.2, here we introduce some frequently used notions and notations. Let $\longrightarrow$ denote a binary relation on $T(\Sigma)$. Then $\longleftrightarrow$ denotes the reflexive closure of $\longrightarrow$ and $\longrightarrow^*$ denotes the transitive reflexive closure of $\longrightarrow$. We write $s \downarrow t$ for $s, t \in T(\Sigma)$ iff there is some $w \in T(\Sigma)$ such that $s \longrightarrow^* w$ and $t \longrightarrow^* w$. The relation $\longrightarrow$ is said to be *terminating* iff there is no infinite reduction sequence $s_0 \longrightarrow s_1 \longrightarrow \cdots$. We say that $\longrightarrow$ is (*locally*) *ground confluent* iff for all $s, s_1, s_2 \in T(\Sigma)$, whenever $s \longrightarrow^* s_i$ $(i = 1, 2)$ (resp. $s \longrightarrow s_i$ $(i = 1, 2)$), then $s_1 \downarrow s_2$. A ground equation $s = t$ is *joinable* (*wrt.* $\longrightarrow$) iff $s \downarrow t$. A ground clause $\Phi \Rightarrow \Psi$ is *joinable* (*wrt.* $\longrightarrow$) iff the following implication holds: Whenever A is joinable for all $A \in \Phi$, then there exists $B \in \Psi$ such that B is joinable. As a consequence, $\Phi \Rightarrow \Psi$ is not joinable iff every $A \in \Phi$ and no $B \in \Psi$ is joinable.

A Σ-*substitution* is a substitution with codomain $T(\Sigma, V)$. If C is some construct (like a clause or a rule), then a Σ-substitution for C is a Σ-substitution the domain of which contains all variables occuring in C. Let *GInst*(Σ, C) denote the set of all ground instances $\tau(C)$ where τ is a Σ-ground substitution for C. The latter means that $\tau(x)$ is a ground term for all variables x occuring in C. An arbitrary clause C is Σ-*ground joinable* iff every ground instance from *GInst*(Σ, C) is joinable. Finally, a set $\mathcal{C}$ of clauses is Σ-ground joinable iff every element of $\mathcal{C}$ is Σ-ground joinable. Just analogously one defines notions which refer to Σ_0-substitutions.

Finally, if $\succ$ is a partial ordering on $T(\Sigma)$, then $\succ_m = \succ\succ$ extends $\succ$ to multisets. Let $\succsim_m = \succ_m \cup \sim$, where $\sim$ denotes the equality of multisets.

2.2 Term rewriting

Let $(\Sigma, \mathcal{R})$ be a given TRS, let Σ_0 be an arbitrary sub-signature of Σ. Note that the case $\Sigma_0 = \Sigma$ is always covered in the sequel. We always assume below that $(\Sigma_0, \mathcal{R})$ is reductive in the following sense:

Definition 2.1 $(\Sigma_0, \mathcal{R})$ *is said to be* reductive wrt. $\succ$ *iff the following conditions are satisfied for all* $R : \Gamma \Rightarrow u \to v, \Delta$ *with* $R \in GInst(\Sigma_0, \mathcal{R})$*: (i)* $s[u] \succ s[v]$ *for all* $s[u] \in T(\Sigma)$*, (ii)* $s[u] \succeq u$ *for all* $s[u] \in T(\Sigma)$ *and (iii)* $u \succ w$ *for all* $w \in Mult(\Gamma \Rightarrow \Delta)$*.*

Next we define of positive/negative conditional rewriting (see also [Ka88]).

Definition 2.2 *Let* $(\Sigma_0, \mathcal{R})$ *be reductive wrt.* $\succ$*. Let* $\longrightarrow_{(\Sigma_0,\mathcal{R})}$ *be the (uniquely existing) binary relation on* $T(\Sigma)$*, satisfying for all* $s, t \in T(\Sigma)$*:* $s \longrightarrow_{(\Sigma_0,\mathcal{R})} t$ *iff there exists* $R : \Gamma \Rightarrow u \to v, \Delta \in GInst(\Sigma_0, \mathcal{R})$ *such that (i)* $s \equiv s[u]$*,* $t \equiv s[v]$ *and such that (ii)* $\Gamma \Rightarrow \Delta$ *is not joinable wrt.* $\longrightarrow_{(\Sigma_0,\mathcal{R})}$*.*

Note that antecedent equations are treated positively and succedent equations negatively wrt. the notion of joinability. Thus, succedent equations are treated as disequations, using negation as failure as an evaluation strategy.

2.3 A refined ground confluence criterion

Next we develop a critical pair criterion for ground confluence which refines the standard criterion in [Ka88]. We follow the idea that ground confluence can be shown by Noetherian induction as follows: Let $Q(t)$ be true iff $\longrightarrow_{(\Sigma_0,\mathcal{R})}$ is ground confluent "below t" (the exact definition will follow). Now $\longrightarrow_{(\Sigma_0,\mathcal{R})}$ is ground confluent iff $Q(t)$ is true for all ground terms t. In order to model a proof method based on this idea it is convenient to take into consideration both, the critical clause as well as the term giving rise to the critical clause (see also [Go87] where this idea is used in the unconditional rewrite case). This term will be used as a "confluence bound". Next we formalize these ideas by introducing suitable notions.

Let $\succ$ be a given well-founded partial ordering on $T(\Sigma)$. Let $\top \succ t \succ \bot$ for every $t \in T(\Sigma)$. Let $\succ_m = \succ\succ'$ and let $\succsim_m = \succ_m \cup \sim$, where $\sim$ denotes the equality of multisets. In the sequel we write singleton multisets $\{s\}$ often in the form s.

Definition 2.3 *A* bound *is a multiset of (pseudo-)terms. A* bounded clause (over Σ) *is a pair* $(C|b)$ *consisting of a clause* C *and a bound* b*.*

Now we introduce the notion of a bounded critical clause.

Definition 2.4 *Let $R : \Gamma \Rightarrow u \to v, \Delta$ and $R' : \Gamma' \Rightarrow u' \to v', \Delta'$ be two rules from $\mathcal{R}$ that share no variables. Let p be a non-variable position of u such that $u|_p$ and u' are unifiable with most general Σ_0-unifier μ. Then the bounded clause*

$$(\mu(\Gamma), \mu(\Gamma') \Rightarrow \mu(u)[\mu(v')]_p = \mu(v), \mu(\Delta), \mu(\Delta') \mid \{\mu(u)\})$$

is called a bounded critical clause between R and R'. *Let $BCrit(\Sigma_0, \mathcal{R})$ denote the set of all bounded critical clauses between the rules from $\mathcal{R}$.*

Note that bounds resulting from critical overlaps consist of one single term only. There are cases too where we want to admit non-singleton multisets as bounds (see section 4.2 below).

If $(C|b)$ is a bounded clause, then the bound b is to majorize the terms from $Mult(C)$ in the following fashion.

Definition 2.5 *Let $(C|b)$ be ground. We say that C is* bounded *by b iff $b \succ_m w$ for all $w \in Mult(C)$. Now let $(C|b)$ be an arbitrary non-ground bounded clause. C is said to be* bounded *by b iff $\tau(C)$ is bounded by $\tau(b)$ for all Σ_0-ground substitutions τ for $(C|b)$.*

Next we introduce a confluence notion which refers to a given bound.

Definition 2.6 *Let b be a ground bound. We say that $\longrightarrow_{(\Sigma_0,\mathcal{R})}$ is* confluent below b *iff for all $s, s_1, s_2 \in T(\Sigma)$, whenever $b \succ_m s$ and $s \longrightarrow^*_{(\Sigma_0,\mathcal{R})} s_i$ (i=1,2), then there exists $s_0 \in T(\Sigma)$ such that $s_i \longrightarrow^*_{(\Sigma_0,\mathcal{R})} s_0$ (i=1,2).*

Finally, we slightly weaken the notion of joinability.

Definition 2.7 *Let $(C|b)$ be a bounded clause that is ground. $(C|b)$ is said to be* weakly joinable (wrt. $\longrightarrow_{(\Sigma_0,\mathcal{R})}$) *iff the following condition holds: If $\longrightarrow_{(\Sigma_0,\mathcal{R})}$ is confluent below b, then C is joinable wrt. $\longrightarrow_{(\Sigma_0,\mathcal{R})}$. An arbitrary bounded clause C is* weakly Σ_0-ground joinable *iff every ground instantiation from $GInst(\Sigma_0, C)$ is weakly joinable. Finally, a set $\mathcal{C}$ of bounded clauses is weakly Σ_0-ground joinable iff every element of $\mathcal{C}$ is weakly Σ_0-ground joinable.*

The following theorem now refines the standard critical pair criterion.

Theorem 2.1 *Let $(\Sigma_0, \mathcal{R})$ be reductive wrt. a well-founded ordering $\succ$. Then $\longrightarrow_{(\Sigma_0,\mathcal{R})}$ is ground confluent iff $BCrit(\Sigma_0, \mathcal{R})$ is weakly Σ_0-ground joinable wrt. $\longrightarrow_{(\Sigma_0,\mathcal{R})}$.*

It follows that it suffices to design a prover for weak ground joinability of a set of bounded clauses in order to obtain a prover for the notion of ground confluence. The description of this prover will be given in the sections to follow. For reasons of clarity we separate conceptual from technical issues: We first design an abstract framework for induction-based reasoning that captures the main conceptual decisions. Thereafter we present a (more technical) instantiation of this framework which constitutes a prover for the property of discourse.

3 A framework for proofs by induction

We next describe a framework that allows one to model inductive reasoning on an abstract level. This framework captures and extends ideas that have been developed in the field of implicit inductive theorem proving [Ba88, KoRu90, Re90]. It can be used for different purposes like proving inductive validity (see [Wi96]) or — as we do here — proving ground joinability. The reader is referred to [WiBe94] for additional details and discussions.

Generally, an (inductive) theorem prover is given by an inference system that operates on syntactic constructs such that certain (inductively defined) properties remain invariant. We capture the latter using an abstract terminology.

3.1 Derivations

We assume that we are given a set of *syntactic constructs.* These syntactic constructs form the basis of the prover states the prover operates on. For the purpose of inductive theorem proving it is convenient to model the *prover states* as pairs $(\mathcal{H};\mathcal{G})$ of multisets of syntactic constructs. The set $\mathcal{G}$ contains the so-called actual *goals.* These are the syntactic constructs which still have to be treated. The set $\mathcal{H}$ contains the constructs that are additionally available for inductive reasoning. Usually, these are syntactic constructs which have already been treated successfully. We call them *hypotheses.* A *derivation relation* is a binary relation $\vdash$ on prover states. If $(\mathcal{H};\mathcal{G}) \vdash (\mathcal{H}';\mathcal{G}')$, then we say that $(\mathcal{H};\mathcal{G})$ is transformed by $\vdash$ into $(\mathcal{H}';\mathcal{G}')$. A sequence of transformations $(\mathcal{H};\mathcal{G}) \vdash^* (\mathcal{H}';\mathcal{G}')$ is called a *derivation.*

We are mainly interested in derivations $(\emptyset;\mathcal{G}) \vdash_{\mathcal{I}} \cdots \vdash_{\mathcal{I}} (\mathcal{H}';\emptyset)$ which start with an empty set of hypotheses and end with an empty set of goals. The latter means that all goals have been treated successfully.

Next we introduce semantic notions that are used to state the invariants of the prover. Every syntactic construct G gives rise to a set of so-called *semantic units.* The latter are pairs (G,τ) where τ equips G with some extra information. If $G \in \mathcal{G}$, then a semantic unit (G,τ) is also called a *$\mathcal{G}$-instance.* To have a concretization in mind one may understand by G a clause and by τ a ground substitution for G.

The *property* to be proved is captured abstractly by a predicate P which is defined on the set of all semantic units. Let $P(G)$ be true iff $P(G,\tau)$ is true for all G-instances (G,τ). Let $P(\mathcal{G})$ be true iff $P(G)$ is true for all $G \in \mathcal{G}$. We say that $\mathcal{G}$ is *P-valid* iff $P(\mathcal{G})$ is true. If $P(G,\tau)$ is false, then (G,τ) is said to be a *counterexample (for P).* If $G \in \mathcal{G}$, then we also speak of a $\mathcal{G}$-counterexample.

Now we turn to the problem of proving P-validity. In order to enable inductive reasoning we assume that we are given a well-founded quasi-ordering $\succsim_i$ on the semantic units. The strict part $\succ_i$ of $\succsim_i$ is called *induction ordering.* A careful analysis of inductive argumentation reveals that the following notion is convenient for proving P-validity by induction wrt. $\succ_i$.

Definition 3.1 *A prover state* $(\mathcal{H};\mathcal{G})$ *is called* inductive (wrt. P and $\succ_i$) *iff the following condition is satisfied: For every* $\mathcal{G}$*-counterexamples* (G,τ) *there exists a* $\mathcal{H}$*-counterexample* (H,π) *with* $(G,\tau) \succ_i (H,\pi)$.

To associate an intuition to the notion of inductiveness it is convenient to consider the inverse of this notion. If $(\mathcal{H};\mathcal{G})$ is not inductive, then $(\mathcal{H};\mathcal{G})$ gives rise to a so-called *inductive counterexample* (G,τ) *for* $(\mathcal{H};\mathcal{G})$. The latter is a $\mathcal{G}$-counterexample (G,τ) such that $P(H,\pi)$ is true for all $\mathcal{H}$-instances (H,π) with $(G,\tau) \succ_i (H,\pi)$.

The notion of inductiveness has the following interesting properties which enable one to show Theorem 3.1 below:

Lemma 3.1
(a) Every prover state $(\mathcal{H};\emptyset)$ *is inductive.*
(b) A prover state $(\emptyset;\mathcal{G})$ *is inductive iff* $P(\mathcal{G})$ *is true.*
(c) If $(\mathcal{H};\mathcal{G})$ *is inductive and if* $P(\mathcal{H})$ *is true, then* $P(\mathcal{G})$ *is true too.*

Definition 3.2 *A derivation relation* $\vdash$ *is called* inductively sound (wrt. P and $\succ_i$) *iff for all derivation steps* $(\mathcal{H};\mathcal{G}) \vdash (\mathcal{H}';\mathcal{G}')$*, whenever* $(\mathcal{H}';\mathcal{G}')$ *is inductive, then* $(\mathcal{H};\mathcal{G})$ *is inductive.*

Turned the other way around, inductive soundness means that one does not lose all inductive counterexamples by a derivation step. Now we directly obtain by Lemma 3.1 the following result, indicating how to prove $P(\mathcal{G})$.

Theorem 3.1 *Let* $\vdash$ *be inductively sound. Let* $(\mathcal{H};\mathcal{G}) \vdash \cdots \vdash (\mathcal{H}';\emptyset)$*. Then* $(\mathcal{H};\mathcal{G})$ *is inductive. If* $\mathcal{H}$ *is empty, then* $P(\mathcal{G})$ *is true.*

3.2 Inference rules

Derivation relations usually are induced by inference rules. An *inference rule* relates goals to two multisets of goals. We write

$$I:\ G \rightsquigarrow (\mathcal{G}_c, \mathcal{G}_r)$$

if the inference rule I relates the goal G to the two multisets of goals $\mathcal{G}_c$ and $\mathcal{G}_r$ resp. if $(G,(\mathcal{G}_c,\mathcal{G}_r)) \in I$. Furthermore we call $G \rightsquigarrow (\mathcal{G}_c,\mathcal{G}_r)$ an *inference by* I. The intuition to be associated with such a rule is as follows: The goal G is transformed by I into the (possibly empty) set $\mathcal{G}_r$ of *result goals.* The (possibly empty) set $\mathcal{G}_c$ describes so-called *condition goals* which have to be considered additionally. Note that if $\mathcal{G}_r$ is empty, then we have a deletion rule. If we model case splitting, then $\mathcal{G}_r$ usually consists of several goals. If we model rewrite-based simplification rules, then $\mathcal{G}_r$ consists of one singe result goal G' which results from G by replacing some subterm. The additional condition goals represent in that case the conditions which have to be verified in order to ensure a correct rewrite step. Next we associate derivation relations to a given set $\mathcal{I}$ of inference rules.

Definition 3.3 *Let $(\mathcal{H};\mathcal{G}) \vdash_{con} (\mathcal{H}';\mathcal{G}')$ iff there exists an inference $I : G \leadsto (\mathcal{G}_c, \mathcal{G}_r)$ with $I \in \mathcal{I}$ such that (i) $\mathcal{G} = G, \mathcal{G}_t$, (ii) $\mathcal{H}' = \mathcal{H}, G$, (iii) $\mathcal{G}' = \mathcal{G}_t, \mathcal{G}_r$ and such that (iv) $(\mathcal{H}';\mathcal{G}_c)$ is inductive.*

Note that the comma represents the union of multisets. The derivation relation $\vdash_{con}$ replaces the goal G by the set $\mathcal{G}_r$ provided the condition goals $\mathcal{G}_c$ are satisfied in the sense that the prover state $(\mathcal{H}';\mathcal{G}_c)$ is inductive. The goal G is stored as a hypothesis in order to be available for further (inductive) reasoning.

Next we develop conditions which allow one to infer inductive soundness of this derivation relation.

Definition 3.4 *Let $(\mathcal{H};\mathcal{G})$ be a given prover state. Let $\mathcal{G} = \mathcal{G}', G$. We call an inference rule I* inductively sound wrt. *$(\mathcal{H};\mathcal{G})$ iff the following condition holds for all inferences $I: G \leadsto (\mathcal{G}_c, \mathcal{G}_r)$: For all G-instances (G,τ), if (G,τ) is a counterexample, then there exists a $\mathcal{H}$-counterexample (H,π) with $(G,\tau) \succ_i (H,\pi)$, or there exists a $\mathcal{G}' \cup \mathcal{G}_c \cup \mathcal{G}_r$-counterexample (G',τ') with $(G,\tau) \succsim_i (G',\tau')$. A system $\mathcal{I}$ of inference rules is called inductively sound iff every rule from $\mathcal{I}$ is inductively sound.*

Note that our concept of performing inductive reasoning differs from that in [Ba88, KoRu90, Re90]. The main point is that we do not require that goals are replaced by strictly smaller ones (note that $(G,\tau) \succsim_i (G',\tau')$ is required only). This enables one e.g. to simplify the design of case splitting rules (see [WiBe94] for a discussion of the advantages of such an approach).

Lemma 3.2 *If $\mathcal{I}$ consist is inductively sound, then $\vdash_{con}$ is inductively sound.*

The problem with the derivation relation $\vdash_{con}$ is that the verification of the condition refers to the invariance notion "inductive soundness". For practical applications one needs a derivation relation where the 'condition check' is operationalized. The following derivation relation can be considered as such an operationalized version of $\vdash_{con}$. We first define auxiliary derivation relations $\vdash_i$ for all natural numbers i.

Definition 3.5 *Let $\vdash_0$ be the empty relation $\emptyset$. Let $\vdash_i$ be defined already. Now let $(\mathcal{H};\mathcal{G}) \vdash_{i+1} (\mathcal{H}';\mathcal{G}')$ iff $(\mathcal{H};\mathcal{G}) \vdash_i (\mathcal{H}';\mathcal{G}')$ or if there exists an inference $I : G \leadsto (\mathcal{G}_c, \mathcal{G}_r)$ with $I \in \mathcal{I}$ such that (i) $\mathcal{G} = G, \mathcal{G}_r$, (ii) $\mathcal{H}' = \mathcal{H}, G$, (iii) $\mathcal{G}' = \mathcal{G}_r, \mathcal{G}_r$ and such that (iv) $(\mathcal{H}';\mathcal{G}_c) \vdash_i^* (\mathcal{H}'';\emptyset)$. Finally let $\vdash_{rec}$ be the union of all these relations $\vdash_i$.*

Hence, $\vdash_{rec}$ replaces the goal G by the set $\mathcal{G}_r$ provided the condition goals $\mathcal{G}_c$ are satisfied in the sense that the prover state $(\mathcal{H}';\mathcal{G}_c)$ can be transformed by the given inference system into a final prover state with an empty set of goals. Note that this derivation relation is defined in a recursive way which is similar to conditional rewriting. Indeed, we use this kind of conditional derivation below in order to simulate contextual conditional inductive rewriting. We finally show that inductive soundness carries over from $\vdash_{con}$ to $\vdash_{rec}$.

Lemma 3.3 *If $\mathcal{I}$ is inductively sound, then* $\vdash_{rec} \subseteq \vdash_{con}$.

Corollary 3.1 *If $\mathcal{I}$ is inductively sound, then $\vdash_{rec}$ is inductively sound.*

4 A framework for proving ground joinability

Now we instantiate the notions of section 3 in order to obtain a framework for proving weak ground joinability of a set of bounded clauses. We assume in the sequel that $(\Sigma, \mathcal{R})$ is a TRS and that Σ_0 is a sub-signature of Σ.

Let $\succ$ be a well-founded partial ordering on $T(\Sigma)$ such that $(\Sigma_0, \mathcal{R})$ is reductive wrt. $\succ$. Let $\succ_m = \succ\succ$ and let $\succsim_m = \succ_m \cup \sim$, where $\sim$ denotes the equality of multisets. We write $s \succ t$ for two arbitrary terms $s, t \in T(\Sigma, V)$ iff $\tau(s) \succ \tau(t)$ for all Σ_0-ground substitutions τ for s, t. We use an analogous notation when comparing multisets of arbitrary terms. Finally, we abbreviate $\longrightarrow_{(\Sigma_0, \mathcal{R})}$ by $\longrightarrow$.

4.1 The basic setting

First we specify the syntactic constructs in the instantiated setting. Syntactic constructs mainly consist of clauses which are enriched by some bound and, additionally, by a so-called 'weight'. The weights are constructs (in our case we use multisets of terms) that enable one to perform strong inductive reasoning. We refer to [WiBe94] for a detailed discussion of weights and some illustrating examples.

Definition 4.1 *A* syntactic construct *is a triple $(C|b|w)$ consisting of a clause C and two multisets b and w of terms from $T(\Sigma, V)$ such that C is bounded by b. We call b the (confluence) bound and w the (induction) weight of the syntactic construct.*

The main part of a syntactic construct $(C|b|w)$ is the clausal formula C. The bound and the weight are to enrich the formula with additional information. Informally, b is used to measure up to which bound the relation $\longrightarrow$ is known to be ground confluent and w is used to measure up to which weight hypotheses are known to be true. Usually, such information is extracted from the formula C. However, separating it from C as indicated above enables one to design more powerful inference rules. The main point is that formulae generally become 'smaller' when applying simplification techniques. This leads to a loss of ordering information if the latter is extracted from the formula itself. Our definition of syntactic constructs allows one to keep bounds and weights (mainly) unchanged and thus to preserve the full information given by the initial formula.

In order to make the setting outlined above applicable we have to clarify how bounded clauses are turned into syntactic constructs. If $(C|b)$ is a given bounded clause, then there are different ways to choose w (see [WiBe94] for a general discussion). One simple possibility is to let w be identical to b. Another possibility is to choose a variable occuring in $(C|b)$. The latter then amounts in selecting an induction variable.

For the considerations to come, the exact knowledge of w is irrelevant. The reader who wants to have a concretization in mind may identify b and w. We only assume that there is a procedure which adds weights to bounded clauses.

In the sequel we are going to identify bounded clauses and the syntactic constructs which are related to them. Thus, $BCrit(\Sigma_0, \mathcal{R})$ e.g. is to be considered as a set of bounded clauses and as a set of syntactic constructs.

Next we take semantic issues into consideration. A *semantic unit* is a pair (G, τ) consisting of a syntactic construct G and a ground substitution $\tau \in GInst(\Sigma_0, G)$. The following definition is crucial for the prover as it fixes the property P to be proved.

Definition 4.2 *Let $((C|b|w), \tau)$ be a semantic unit. We define $P((C|b|w), \tau)$ to be true iff $\tau(C|b)$ is weakly joinable wrt. $\longrightarrow$.*

Note that this definition does not refer to the weight w. We thus have a complete separation between semantic and inductive issues. It follows directly that a set $\mathcal{C}$ of bounded clauses is weakly Σ_0-ground joinable iff $P(\mathcal{C})$ is true. The instantiation of the abstract framework given above thus is well suited for our purposes: According to Theorem 2.1 we have to show weak Σ_0-ground joinability of $BCrit(\Sigma_0, \mathcal{R})$. The latter is done if we have shown that $BCrit(\Sigma_0, \mathcal{R})$ is P-valid.

To finish the instantiation of the abstract framework we have to define the quasi-ordering $\succsim_i$ which is used to control inductive reasoning. In our approach here we relate $\succsim_i$ to the given ordering $\succ$ (for a discussion of other ways see [WiBe94]): Let $((C|b|w), \tau) \succsim_i ((C'|b'|w'), \tau')$ iff $\tau(w) \succsim_m \tau'(w')$. Thus, semantic units will be compared by their weights when performing inductive reasoning.

Once we have instantiations for P and $\succsim_i$, we directly obtain the invariant of our 'concrete' prover framework. Recall that this invariant — inductive soundness — is defined in terms of P and $\succsim_i$. What still has to be done is the design of inference rules which are inductively sound wrt. the instantiated notions.

4.2 The inference system

In order to keep the presentation of the inference system clear, we continue in a top-down fashion by describing only a few general rules. We briefly outline (without describing all details) that these general rules subsume most of the technical reasoning strategies used in the literature (in order to treat related problems).

There are at least three different reasoning concepts leading to a powerful prover: (a) case splitting, (b) (inductive) contextual conditional simplification and (c) literal elimination. We formalize them next. An illustration by some examples will follow.

4.2.1 Case splitting

As motivated in the introduction, case splitting is essential for obtaining finite schematized joinability proofs. We first provide a rather general case splitting rule. Thereafter we show that the familiar case splitting techniques are covered by this rule. Note that our approach allows one to split cases without performing a succeeding reduction step.

A *case splitting for a clause* C is a finite (possibly empty) set S of pairs $(\mu, \Gamma \Rightarrow \Delta)$, consisting of a Σ_0-substitution μ and a clause $\Gamma \Rightarrow \Delta$. A case splitting S for C is said to be *complete* (*wrt.* $(\Sigma_0, \mathcal{R})$) iff for all $\longrightarrow$-irreducible $\tau \in \mathit{GInst}(\Sigma_0, C)$, if $\tau(C)$ is not joinable wrt. $\longrightarrow$, then there exists $(\mu, \Gamma \Rightarrow \Delta) \in S$ and a Σ_0-ground substitution τ' such that (i) $\tau = \tau'\mu$ and such that (ii) $\tau'(\Gamma \Rightarrow \Delta)$ is not joinable wrt. $\longrightarrow$.

Rule 1 **[Case splitting]**
Let $(\mathcal{H}; \mathcal{G}, G)$ be a prover state. Let $G : (C|b|w)$ with $C : \Phi \Rightarrow \Psi$. Let S be a complete case splitting for C such that $\Gamma \Rightarrow \Delta$ is bounded by $\mu(b)$ for all $(\mu, \Gamma \Rightarrow \Delta) \in S$. Let $\mathcal{G}'$ consist of all syntactic constructs $(\Gamma, \mu(\Phi) \Rightarrow \mu(\Psi), \Delta|\mu(b)|\mu(w))$ with $(\mu, \Gamma \Rightarrow \Delta) \in S$. Then $I_1 : G \rightsquigarrow (\emptyset, \mathcal{G}')$.

Note that the elements of $\mathcal{G}'$ are well-formed syntactic constructs satisfying the boundedness condition.

We briefly discuss three special sub-rules of this general rule. The first is obtained by considering complete case splittings of the type $S = \{(\mu_i, \Box \Rightarrow \Box) \mid i = 1, \ldots, n\}$. This models case splitting by mere instantiation. Note that the generation of complete case splittings of this type is usually much simplified if one introduces a sub-signature Σ_0 with a few constructors only. However, the verification of completeness of Σ_0 wrt. Σ then may become more difficult.

The second sub-rule models case splitting by addition of context equations. The easiest way to do it is to let $S = \{(id, A \Rightarrow \Box), (id, \Box \Rightarrow A)\}$, where id is the identity and A is an equation. Obviously S is a complete case splitting for every clause C.

The third sub-rule allows one to produce case splittings that enable successive schematized rewriting steps. The techniques involving narrowing are based on this sub-rule. Let $C : \Phi, s = t \Rightarrow \Psi$ (the case $C : \Phi, s! \Rightarrow \Psi$ is treated analogously). Now let $(\mu, \mu(\Gamma) \Rightarrow \mu(\Delta)) \in S$ if there is a non-variable position p in s (or analogously in t) and a rule $\Gamma \Rightarrow u \rightarrow v, \Delta \in \mathcal{R}$ such that μ is the most general Σ_0-unifier of $s|_p$ and u. Let further $(\mu, \Box \Rightarrow \Box) \in S$ if μ is the most general Σ_0-unifier of s and t. One easily verifies that S is a complete case splitting for C. Note that it is essential here that $s = t$ is an antecedent equation. The situation is different if $s = t$ is a succedent equation. In that case, the narrowing-based method produces a complete case splitting provided $s = t$ is ground reducible (see [Ba88]). The notion of ground reducibility however is undecidable in the conditional rewrite case.

4.2.2 Inductive contextual simplification

Our simplification concept allows one to use actual goals, actual hypotheses and so-called lemmata for rewriting and subsuming a given syntactic construct. A *lemma* is a syntactic construct $(C|b|\Box)$ such that $(C|b)$ is (known to be) weakly Σ_0-ground joinable wrt. $\longrightarrow$. Let $\mathcal{L}$ denote a set of lemmata in the sequel. Below we briefly discuss ways to generate lemmata.

Rule 2 [Contextual rewriting]
Let $(\mathcal{H}; \mathcal{G}, G)$ be the actual prover state. Let $K : (\Gamma \Rightarrow u = v, \Delta | b' | w')$ be a syntactic construct from $\mathcal{H} \cup \mathcal{G} \cup \mathcal{L}$. Let σ be an arbitrary Σ-substitution such that $\sigma(K)$ is a syntactic construct (satisfying the boundedness condition) too. Let $G : (C|b|w)$ with $C : \Phi \Rightarrow E, \Psi$ resp. $C : \Phi, E \Rightarrow \Psi$, where the (pseudo-)equation $E : s[\sigma(u)] = t$ contains an instance of u. Let $\mathcal{G}_c$ contain all the constructs $G_c^A : (\Phi \Rightarrow \sigma(A), \Psi | b | w)$ with $A \in \Gamma$, $G_c^B : (\Phi, \sigma(B) \Rightarrow \Psi | b | w)$ with $B \in \Delta$ and $G_c^x : (\Phi \Rightarrow \sigma(x)!, \Psi | b | w)$ such that $x \in \mathit{var}(K)$ and $\sigma(x) \notin T(\Sigma_0, V)$. Finally let G' result from G by replacing the subterm $\sigma(u)$ in $s[\sigma(u)]$ by $\sigma(v)$. Thus, $G' : (C'|b|w)$ with $C' : \Phi \Rightarrow s[\sigma(v)] = t, \Psi$ resp. $C' : \Phi, s[\sigma(v)] = t \Rightarrow \Psi$. Let the following conditions be satisfied:
(a) $w \succ_m \sigma(w')$ if $K \in \mathcal{H}$ and $w \succsim_m \sigma(w')$ if $K \in \mathcal{G}$.
(b) $b \succsim_m \sigma(b')$.
(c) $b \succ_m \{s[\sigma(v)]\}$ and $b \succ_m \{\sigma(x)\}$ for all $x \in \mathit{var}(K)$ and $\sigma(x) \notin T(\Sigma_0, V)$.
Then $I_2 : G \rightsquigarrow (\mathcal{G}_c, \{G'\})$.

First note that all newly generated constructs are indeed syntactic constructs in the sense of Definition 4.1. On the one hand we use condition (c) to infer that G' satisfies the boundedness condition. On the other hand we use condition (b) together with the fact that $\sigma(K)$ is a syntactic construct to ensure that all constructs G_c^A and G_c^B satisfy the boundedness condition. The second part of condition (c) finally ensures that G_c^x satisfies the boundedness condition.

This rule allows one to use $\sigma(K)$ for schematized rewriting, replacing the subterm $\sigma(u)$ of G by $\sigma(v)$. In order to be applicable, the conditions $\sigma(\Gamma)$ and $\sigma(\Delta)$ must be guaranteed to be 'true' (wrt. the notion of ground joinablability) within the present situation. Note that the present situation allows us to use Φ and Ψ as additional context. Note that the substitution σ need not be a Σ_0-substitution. However, $\sigma(x)$ must be guaranteed to be 'defined' for all relevant variables x in order to be able to pull a Σ-ground substitution $\tau\sigma$ down to a Σ_0-ground substitution τ'.

The next rule models contextual subsumption. We omit a detailed discussion as it proceeds just as in the rewrite case.

Rule 3 [Contextual subsumption]
Let $(\mathcal{H}; \mathcal{G}, G)$ be the actual prover state. Let $K : (\Gamma_0, \Gamma_1 \Rightarrow \Delta_0, \Delta_1 | b' | w')$ be a syntactic construct from $\mathcal{H} \cup \mathcal{G} \cup \mathcal{L}$. Let σ be an arbitrary Σ-substitution such that $\sigma(K)$ is a syntactic construct too. Let $G : (C|b|w)$ with $C : \sigma(\Gamma_0), \Phi_1 \Rightarrow \sigma(\Delta_0), \Psi_1$. Let $\mathcal{G}_c$ contain all the syntactic constructs $G_c^A : (\Phi_1 \Rightarrow \sigma(A), \Psi_1 | b | w)$ with $A \in \Gamma_1$, $G_c^B : (\Phi_1, \sigma(B) \Rightarrow \Psi_1 | b | w)$ with $B \in \Delta_1$ and $G_c^x : (\Phi_1 \Rightarrow \sigma(x)!, \Psi_1 | b | w)$ such that $x \in \mathit{var}(K)$ and such that $\sigma(x) \notin T(\Sigma_0, V)$. Let the following conditions be satisfied:
(a) $w \succ_m \sigma(w')$ if $K \in \mathcal{H}$ and $w \succsim_m \sigma(w')$ if $K \in \mathcal{G}$.
(b) $b \succsim_m \sigma(b')$.
(c) $b \succ_m \{\sigma(x)\}$ for all $x \in \mathit{var}(K)$ and $\sigma(x) \notin T(\Sigma_0, V)$.
Then $I_3 : G \rightsquigarrow (\mathcal{G}_c, \emptyset)$.

First note that all newly generated constructs are indeed syntactic constructs in the sense of Definition 4.1. First, we use condition (b) together with the fact that $\sigma(K)$ is a syntactic construct to ensure that all constructs G_c^A and G_c^B satisfy the boundedness condition. Condition (c) ensures that G_c^x satisfies the boundedness condition as well.

Next we briefly outline how to obtain lemmata. Lemmata can be generated according to different strategies:

First note that if a clause C (without a bound) is known to be Σ_0-ground joinable wrt. $\longrightarrow$, then $(C|\mathit{Mult}(C)|\Box)$ is a lemma.

Every rule $R : \Gamma \Rightarrow u \to v, \Delta \in \mathcal{R}$ can be turned into a lemma $(R|\{u, \bot\}|\Box)$ (note that $(R|\{u, \bot\}|\Box)$ indeed satisfies the boundedness condition). As a consequence, rewriting with $\mathcal{R}$ turns out to be a special case of rewriting with lemmata.

Lemmata may result from former runs of our prover.

Finally, there are general heuristics that indicate how to obtain lemmata. For instance, every equation $A : s = s$ (resp. $A : s!$ with $s \in T(\Sigma_0, V)$) induces a lemma $(\Box \Rightarrow A \mid \mathit{Mult}(A) \mid \Box)$. If $C : A \Rightarrow A$, then $(C|\mathit{Mult}(A)|\Box)$ is a lemma too.

Lemmata can be used to model a kind of contextual rewriting which uses (skolemized) antecedent equations for rewriting and subsuming an equation occuring in the remaining clause. We briefly explain how this can be done in our framework.

Let $G : (C|b|w)$ with $C : \Phi, u = v \Rightarrow s[u] = t$. Suppose that $b \succ_m \{s[v]\}$. Then we are allowed to replace u by v. This is because we can use the lemma $K : (u = v \Rightarrow u = v \mid \{u, v\} \mid \Box)$ (wrt. the substitution $\sigma = id$) for rewriting. Note that $(\mathcal{H}, G; \mathcal{G}_c)$ is trivially inductive (and thus satisfied in a conditional fashion) as the only element $(\Phi, u = v \Rightarrow u = v \mid b \mid w)$ of $\mathcal{G}_c$ is Σ_0-ground joinable.

Now let $G : (C|b|w)$ with $C : \Phi \Rightarrow u = v, \Psi$. If there is a derivation $u \equiv u_0 \leftrightarrow_\Phi u_1 \leftrightarrow_\Phi \cdots \leftrightarrow_\Phi u_n \equiv v$ using equations from Φ (with variables treated as constants) such that $b \succ_m \{u_i\}$ for $i = 1, \ldots, n-1$, then we are allowed to subsume G by the lemma $K : (\Phi \Rightarrow u = v|b|\Box)$.

4.2.3 Literal elimination

This concept of reasoning allows one to eliminate atoms from a clause that turn out to be redundant. As in [BaGa92] we first provide a very general rule.

Rule 4 [Literal elimination]
Let $(\mathcal{H}; \mathcal{G}, G)$ be the current prover state. Let $G : (C|b|w)$ with $C : \Phi, A \Rightarrow \Psi$ (resp. $C : \Phi \Rightarrow A, \Psi$). Let $G' : (C'|b|w)$ with $C' : \Phi \Rightarrow \Psi$. Let $G_c : (C_c|b|w)$ with $C_c : \Phi \Rightarrow A, \Psi$ (resp. $C_c : \Phi, A \Rightarrow \Psi$). Then $I_4 : G \rightsquigarrow (\{G_c\}, \{G'\})$.

First note that the newly generated constructs obviously satisfy the boundedness condition.

We briefly discuss three applications of this rule. Consider first the case where the clause $G : (\Phi \Rightarrow A, A, \Psi \mid b \mid w)$ contains two identical succedent atoms. In that case we have $G_c : (\Phi, A \Rightarrow A, \Psi \mid b \mid w)$. Note that$(\mathcal{H}, G; G_c)$ is trivially inductive as $P(G_c)$ is true. If $G : (\Phi, s = s \Rightarrow \Psi \mid b \mid w)$, then $G_c : (\Phi \Rightarrow s = s, \Psi \mid b \mid w)$. It follows that

$P(G_c)$ is true and that $(\mathcal{H}, G; G_c)$ is inductive. Hence $s = s$ can be eliminated in G. Finally, if $G : (\Phi \Rightarrow s = t, \Psi \mid b \mid w)$ is such that $S = \emptyset$ is a complete case splitting for $\Phi, s = t \Rightarrow \Psi$, then $s = t$ can be eliminated as well in G.

4.3 The main result

In this section we join together all the previously developed parts of our theory. Note that section 4.1 and 4.2 provide an instantiation of the abstract notions we have introduced in section 3 to state the abstract theory. In order to make applicable this abstract theory, one still has the verify that all the rules from section 4.2 satisfy the invariant property. For that purpose let $\mathcal{I}$ consist of the four rules I_1, I_2, I_3, I_4 discussed in section 4.2. One easily verifies the following correctness properties of $\mathcal{I}$.

Lemma 4.1 *All rules from $\mathcal{I}$ are inductively sound.*

Corollary 4.1 *The derivation relations $\vdash_{con}$ and $\vdash_{rec}$ induced by $\mathcal{I}$ are inductively sound.*

The following theorem provides the main result of this paper. It is obtained in a straightforward manner from previously stated results.

Theorem 4.1 *Let $(\Sigma_0, \mathcal{R})$ be reductive wrt. a well-founded partial ordering $\succ$. Let $\vdash_{\mathcal{I}} = \vdash_{rec}$ or $\vdash_{\mathcal{I}} = \vdash_{con}$. Let $BCrit(\Sigma_0, \mathcal{R}) \subseteq \mathcal{G}$. Let $(\emptyset; \mathcal{G}) \vdash_{\mathcal{I}} \cdots \vdash_{\mathcal{I}} (\mathcal{H}'; \emptyset)$. Then $\longrightarrow_{(\Sigma_0, \mathcal{R})}$ is ground confluent.*

Acknowledgements: I would like to thank J. Avenhaus and C.-P. Wirth for many valuable discussions and helpful suggestions.

References

[Av95] J. Avenhaus, *Reduktionssysteme,* (Springer, 1995).

[Ba88] L. Bachmair, Proof by Consistency in Equational Theories, in: *3rd LICS* (1988) pp. 228-233.

[BaGa92] L. Bachmair and H. Ganzinger, Rewrite-based theorem proving with selection and simplification, Technical Report MPI-I-91-208, Max-Planck-Institut für Informatik, Saarbrücken.

[Be93] K. Becker, Proving ground confluence and inductive validity in constructor based equational specifications, *TAPSOFT '93,* LNCS 668 (Berlin, 1993) pp. 46-60.

[DeJo90] N. Dershowitz and J. P. Jouannaud, Rewriting systems, in: J. van Leeuwen, ed., *Handbook of Theoretical Computer Science, Vol. B* (Elsevier, Amsterdam, 1990) pp. 241-320.

[Fr86] L. Fribourg, A strong restriction of the inductive completion procedure, *13th ICALP 86,* LNCS 266 (Springer, 1991) pp. 105-115.

[Ga87] H. Ganzinger, Ground term confluence in parametric conditional equational specifications, in: *STACS '87,* LNCS 247 (Springer, 1987) pp. 286-298.

[Go87] R. Göbel, Ground confluence, in: *2nd RTA 87*, LNCS 256 (Berlin, 1991) pp. 156-167.

[Ka88] S. Kaplan, Positive/Negative Conditional Rewriting, in: *Conditional Term Rewriting Systems*, LNCS 308 (Springer, 1987) pp. 129-143.

[KNO90] D. Kapur, P. Narendran and F. Otto, On ground-confluence of term rewriting systems, *Inform. and Comp.* 86 1990, pp. 14-31.

[KoRu90] E. Kounalis and M. Rusinowitch, Mechanizing Inductive Reasoning, in: *Proc. of 8th AAAI '90* (MIT Press, 1990) pp. 240-245.

[KoRu91] E. Kounalis and M. Rusinowitch, Studies on the ground convergence property of conditional theories, in: *Algebraic Methodology and Software Technology '91* (Springer, 1992) pp. 363-376.

[Pl85] D. Plaisted, Semantic confluence tests and completion methods, *Inform. and Control* 65(2/3) 1985, pp. 182-215.

[Re90] Term rewriting induction, in: *10th CADE '90*, LNAI 449 (Springer 1990) pp. 162-177.

[Wi95] C.-P. Wirth, Syntactic confluence criteria for positive/negative-conditional rewriting systems, SEKI Report SR-95-09 (SFB), Universität Kaiserslautern.

[Wi96] C.-P. Wirth, PhD-thesis, Universität Kaiserslautern, to appear 1996.

[WiBe94] C.-P. Wirth and K. Becker, Abstract notions and inference systems for proofs by mathematical induction, in: *4th CTRS '94*, to appear.

A Proofs

In the sequel we always abbreviate $\longrightarrow_{(\Sigma_0,\mathcal{R})}$ by $\longrightarrow$.

Theorem 2.1 Let $(\Sigma_0, \mathcal{R})$ be reductive wrt. a well-founded partial ordering $\succ$. Then $\longrightarrow_{(\Sigma_0,\mathcal{R})}$ is ground confluent iff $BCrit(\Sigma_0, \mathcal{R})$ is weakly Σ_0-ground joinable wrt. $\longrightarrow_{(\Sigma_0,\mathcal{R})}$.

Proof: We first define an auxiliary predicate Q on $T(\Sigma)$: For $t \in T(\Sigma)$ let $Q(t)$ be true iff the following implication holds for all $t_1, t_2 \in T(\Sigma)$: whenever $t \longrightarrow t_i$ (i=1,2), then there exists $t_0 \in T(\Sigma)$ such that $t_i \longrightarrow^* t_0$ (i=1,2). Thus, if $Q(t)$ is shown to be true for all $t \in T(\Sigma)$, then it follows that $\longrightarrow$ is locally ground confluent. As $\longrightarrow$ is terminating, $\longrightarrow$ is even ground confluent.

In the sequel we show that $Q(t)$ is true for all $t \in T(\Sigma)$, using Noetherian induction wrt. $\succ$. Let $Q(s)$ be true for all $s \in T(\Sigma)$ satisfying $t \succ s$. It follows directly that $\longrightarrow$ is locally confluent and thus confluent below $\{t\}$.

Now consider an overlap $t \longrightarrow t_i$ (i=1,2). Let $t \longrightarrow t_1$ with the rule $R : \Gamma \Rightarrow u \to v, \Delta$ and the Σ_0-ground substitution τ_1 at the position $p \in O(t)$. Further, let $t \longrightarrow t_2$ with the rule $R' : \Gamma' \Rightarrow u' \to v', \Delta'$ and the Σ_0-ground substitution τ_2 at the position $q \in O(t)$. We assume that R and R' share no variables. Thus, we can make one substitution τ from τ_1 and τ_2.

The case where p and q are disjoint positions is obvious. Thus, we only consider the case where (say) q is below p. Let $q = pr$ for a suitable position r. So we have $\tau(u)|_r \equiv \tau(u')$.

Let first $r \in O(u)$ and $u|_r \notin V$. Then $u|_r$ and u' are Σ_0-unifiable with a most general Σ_0-unifier μ. Let $\tau = \rho\mu$ for a suitable Σ_0-ground substitution ρ. Consider the bounded critical clause $(C|\{\mu(u)\})$ resulting from R and R'. First note that $\longrightarrow$ is confluent below $\{\rho\mu(u)\}$. This follows from the fact that $t \succ s \succeq \tau(u) \equiv \rho\mu(u)$ (note that $(\Sigma_0, \mathcal{R})$ is reductive wrt. $\succ$) and that $\longrightarrow$ is confluent below $\{t\}$ (according to the induction hypothesis). Furthermore note that $\rho\mu(\Gamma \Rightarrow \Delta)$ and $\rho\mu(\Gamma' \Rightarrow \Delta')$ are unjoinable because the two rules R and R' are applicable wrt. the Σ_0-ground substitution τ. As $BCrit(\mathcal{R})$ is weakly joinable wrt. $\longrightarrow$ it follows that $\rho(\mu(u)[\mu(v')]_p) = \rho\mu(v)$ is joinable wrt. $\longrightarrow$. Thus, $t_1 = t_2$ is joinable as well, which has to be shown.

Now let $r \notin O(u)$ or $r \in O(u)$ and $u|_r \in V$. Hence, $\tau(u')$ is a sub-term of $\tau(x)$ for some variable $x \in var(u)$. Let $\tau(x)|_{r'} \equiv \tau(u')$. We define a new Σ_0-ground substitution τ' as follows: Let $\tau'(x) \equiv \tau(x)[\tau(v')]_{r'}$ and let $\tau'(y) \equiv \tau(y)$ for all $y \not\equiv x$. Then $\tau(z) \longrightarrow^* \tau'(z)$ for all variables z. Next we show that $\tau'(u) \longrightarrow \tau'(v)$. Using this relation we easily obtain the desired result.

We know that $\tau(u) \longrightarrow \tau(v)$. Thus $\tau(\Gamma \Rightarrow \Delta)$ is unjoinable wrt. $\longrightarrow$. We show that $\tau'(\Gamma \Rightarrow \Delta)$ is unjoinable wrt. $\longrightarrow$ too. Let first $c = d \in \Gamma$. We have $\tau'(c) \longleftarrow^* \tau(c) \downarrow \tau(d) \longrightarrow^* \tau'(d)$. Now we use the fact that $t \succ s \succeq \tau(u) \succ \tau(c), \tau(d)$. As $\longrightarrow$ is confluent below $\{t\}$ according to the induction hypothesis, we can infer that $\tau'(c) \downarrow \tau'(d)$. Next let $c = d \in \Delta$. We have $\tau(c) \longrightarrow^* \tau'(c)$ and $\tau(d) \longrightarrow^* \tau'(d)$.

Furthermore, $\tau(c) = \tau(d)$ is not joinable wrt. $\longrightarrow$. Suppose, $\tau'(c) = \tau'(d)$ were joinable wrt. $\longrightarrow$. As $t \succ s \succeq \tau(u) \succeq \tau'(u) \succ \tau'(c), \tau'(d)$ and as $\longrightarrow$ is confluent below $\{t\}$ according to the induction hypothesis we could infer that $\tau(c) = \tau'(d)$ were joinable wrt. $\longrightarrow$ too. □

Lemma 3.2 Let $\mathcal{I}$ be inductively sound. Then $\vdash_{con}$ is inductively sound.

Proof: Let $(\mathcal{H}; \mathcal{G}, G) \vdash_{con} (\mathcal{H}, G; \mathcal{G}, \mathcal{G}_r)$. Thus, by the definition of $\vdash_{con}$, $(\mathcal{H}, G; \mathcal{G}_c)$ is inductive. Now let $(\mathcal{H}, G; \mathcal{G}, \mathcal{G}_r)$ be inductive. It directly follows that $(\mathcal{H}, G; \mathcal{G}, \mathcal{G}')$ is inductive, where $\mathcal{G}' = \mathcal{G}_c \cup \mathcal{G}_r$. We show that $(\mathcal{H}; \mathcal{G}, G)$ is inductive, using Noetherian induction wrt. $\succ_i$. For that purpose let Q be an auxiliary predicate which is defined on $\mathcal{G} \cup \{G\}$-instances as follows:

Let $Q(K, \rho)$ be true for a $\mathcal{G} \cup \{G\}$-instance (K, ρ) iff the following condition holds: If $P(K, \rho)$ is false, then there exists a $\mathcal{H}$-counterexample (H, π) such that $(K, \rho) \succ_i (H, \pi)$.

We show that $Q(K, \rho)$ is true for all $\mathcal{G} \cup \{G\}$-instances (K, ρ). $(\mathcal{H}; \mathcal{G}, G)$ is then inductive according to the definition.

Let (K, ρ) be a $\mathcal{G} \cup \{G\}$-instance. We assume that $Q(K', \rho')$ is true for all $\mathcal{G} \cup \{G\}$-instances (K', ρ') with $(K, \rho) \succ_i (K', \rho')$. Let $P(K, \rho)$ be false.
Case 1: $K = G$. Thus, $P(G, \rho)$ is false. As I is inductively sound, there are two cases to be considered:
Case 1.1: There exists a $\mathcal{H}$-counterexample (H, π) such that $(G, \rho) \succ_i (H, \pi)$. In that case we can directly infer that $Q(K, \rho)$ is true.
Case 1.2: There exists a $\mathcal{G} \cup \mathcal{G}'$-counterexample (G', τ') such that $(G, \rho) \succ_i (G', \tau')$. As $(\mathcal{H}, G; \mathcal{G}, \mathcal{G}')$ is inductive, there exists a $\mathcal{H} \cup \{G\}$-counterexample (K', ρ') such that $(G', \tau') \succ_i (K', \rho')$.
Case 1.2.1: $K' = G$. According to the induction hypothesis, $Q(K', \tau')$ is true. It follows that there exists a $\mathcal{H}$-counterexample (H, π) with $(K', \tau') \succ_i (H, \pi)$. As $(K, \rho) \succ_i (G', \tau') \succ_i (H, \pi)$, $Q(K, \rho)$ is true too.
Case 1.2.2: $K' \in \mathcal{H}$. In that case we can directly infer that $Q(K, \rho)$ is true.
Case 2: $K \in \mathcal{G}$. As $(\mathcal{H}, G; \mathcal{G}, \mathcal{G}')$ is inductive, there exists a $\mathcal{H} \cup \{G\}$-counterexample (K', ρ') such that $(G', \tau') \succ_i (K', \rho')$. Now we can proceed just as in case 1.2. □

Lemma 3.3 Let $\mathcal{I}$ be inductively sound. Then $\vdash_{rec} \subseteq \vdash_{con}$.

Proof: We show (by using induction on i) that the following two implications are true for all natural numbers i:
(a) If $(\mathcal{H}; \mathcal{G}) \vdash_i (\mathcal{H}'; \mathcal{G}')$, then $(\mathcal{H}; \mathcal{G}) \vdash_{con} (\mathcal{H}'; \mathcal{G}')$.
(b) If $(\mathcal{H}; \mathcal{G}) \vdash_i^* (\mathcal{H}'; \emptyset)$, then $(\mathcal{H}; \mathcal{G})$ is inductive.
The claim then follows directly.

Let first $i = 0$. The first statement (a) is trivially satisfied. For the second statement (b) note that $\vdash_0^0$ is the identity relation. But, if $(\mathcal{H}; \mathcal{G})$ is such that $\mathcal{G} = \emptyset$, then we know from Lemma 3.1 above that $(\mathcal{H}; \mathcal{G})$ is inductive.
Now we assume that the statements hold for a given i. Consider first the statement (a). Let $(\mathcal{H}; \mathcal{G}) \vdash_{i+1} (\mathcal{H}'; \mathcal{G}')$. Hence, we have $(\mathcal{H}'; \mathcal{G}_c) \vdash_i^* (\mathcal{H}''; \emptyset)$. Using the induction hypothesis we can infer that $(\mathcal{H}'; \mathcal{G}_c)$ is inductive. It then follows directly that

$(\mathcal{H};\mathcal{G}) \vdash_{con} (\mathcal{H}';\mathcal{G}')$. Now consider the statement (b). Let $(\mathcal{H};\mathcal{G}) \vdash^*_{i+1} (\mathcal{H}';\emptyset)$. Then, as we have just proved, $(\mathcal{H};\mathcal{G}) \vdash^*_{con} (\mathcal{H}';\emptyset)$. As $\mathcal{I}$ consists of sound inference rules, $\vdash_{con}$ is inductively sound. It follows that $(\mathcal{H};\mathcal{G})$ is inductive. □

Before we prove Lemma 4.1 we briefly consider an auxiliary lemma.

Lemma Let $G : (C|b|w)$ be a syntactic construct. Let σ be a Σ-substitution such that $\sigma(G)$ is a syntactic construct too. If $(\sigma(G), \tau)$ is a counterexample and if $\tau\sigma \longrightarrow^* \rho$ such that ρ is a Σ_0-ground substitution, then (G, ρ) is a counterexample with $(\sigma(G), \tau) \succsim_i (G, \rho)$.

Proof: Let $C : \Phi \Rightarrow \Psi$. Let $(\sigma(G), \tau)$ be a counterexample. Thus, $\longrightarrow$ is confluent below $\tau\sigma(b)$ and $\tau\sigma(C)$ is unjoinable wrt. $\longrightarrow$. Let ρ be a Σ_0-ground substitution such that $\tau\sigma(x) \longrightarrow^* \rho(x)$. We show that (G, ρ) is a counterexample too.

First we show that $\rho(C)$ is unjoinable wrt. $\longrightarrow$. Let $c = d \in \Phi$. As $\tau\sigma(c = d)$ is joinable we have $\rho(c) \longleftarrow^* \tau\sigma(c) \downarrow \tau\sigma(d) \longrightarrow^* \rho(d)$. As $\tau\sigma(b) \succ_m \tau\sigma(c), \tau\sigma(d)$ and as $\longrightarrow$ is confluent below $\tau\sigma(b)$, we obtain $\rho(c) \downarrow \rho(d)$. Now let $c = d \in \Delta$. If $\rho(c) \downarrow \rho(d)$, then we would directly obtain $\tau\sigma(c) \downarrow \tau\sigma(d)$. Hence, $\rho(C)$ is indeed unjoinable wrt. $\longrightarrow$. As $\tau\sigma(b) \succsim_m \rho(b)$, $\longrightarrow$ is confluent below $\rho(b)$. Thus, (G, ρ) is indeed a counterexample. As $\tau\sigma(w) \succsim_m \rho(w)$ it follows that $(\sigma(G), \tau) \succsim_i (G, \rho)$. □

Lemma 4.1 I_1 is inductively sound.

Proof: Let $(\mathcal{H};\mathcal{G}, G)$ be a prover state. Let $G : (C|b|w)$ with $C : \Phi \Rightarrow \Psi$. Let S be a complete case splitting for C such that $\Gamma \Rightarrow \Delta$ is bounded by $\mu(b)$ for all $(\mu, \Gamma \Rightarrow \Delta) \in S$. Let $\mathcal{G}'$ consist of all syntactic constructs $(\Gamma, \mu(\Phi) \Rightarrow \mu(\Psi), \Delta|\mu(b)|\mu(w))$ with $(\mu, \Gamma \Rightarrow \Delta) \in S$. Thus, let $I_1 : G \leadsto (\emptyset, \mathcal{G}')$.

Suppose that (G, τ) is a G-counterexample. Thus, $\longrightarrow$ is confluent below $\tau(b)$ and $\tau(C)$ is unjoinable wrt. $\longrightarrow$. According to the lemma above we can assume that τ is irreducible wrt. $\longrightarrow$. As S is a complete case splitting for C, there exists $(\mu, \Gamma \Rightarrow \Delta) \in S$ and a ground substitution τ' such that $\tau = \tau'\mu$ and such that $\tau'(\Gamma \Rightarrow \Delta)$ is unjoinable wrt. $\longrightarrow$. Now consider $G' : (C'|\mu(b)|\mu(w))$, where $C' : \Gamma, \mu(\Phi) \Rightarrow \mu(\Psi), \Delta$. First note that $\tau'(C')$ is unjoinable. As $\tau = \tau'\mu$, $\longrightarrow$ is confluent below $\tau'\mu(b)$. Thus, (G', τ') is a counterexample. Furthermore, we obviously have $(G, \tau) \succsim_i (G', \tau')$ as $\tau(w) = \tau'\mu(w)$.
□

Lemma 4.1 I_2 is inductively sound.

Proof: Let $(\mathcal{H};\mathcal{G}, G)$ be the actual prover state. Let $K : (\Gamma \Rightarrow u = v, \Delta|b'|w')$ be a syntactic construct from $\mathcal{H} \cup \mathcal{G} \cup \mathcal{L}$. Let σ be an arbitrary Σ-substitution such that $\sigma(K)$ is a syntactic construct too. Let $G : (C|b|w)$ with $C : \Phi, E \Rightarrow \Psi$, where the (pseudo-)equation $E : s[\sigma(u)] = t$ contains an instance of u. (The case $C : \Phi \Rightarrow E; \Psi$ is treated just analogously.) Let $\mathcal{G}_c$ contain all the constructs $G^A_c : (\Phi \Rightarrow \sigma(A), \Psi|b|w)$ with $A \in \Gamma$, $G^B_c : (\Phi, \sigma(B) \Rightarrow \Psi|b|w)$ with $B \in \Delta$ and $G^x_c : (\Phi \Rightarrow \sigma(x)!, \Psi|b|w)$ such that $x \in var(k)$ and $\sigma(x) \notin T(\Sigma_0, V)$. Let $G' : (C'|b|w)$ with $C' : \Phi \Rightarrow s[\sigma(v)] = t, \Psi$ resp. $C' : \Phi, s[\sigma(v)] = t \Rightarrow \Psi$. Finally, let $I_2 : G \leadsto (\mathcal{G}_c, \{G'\})$. Thus the following conditions are satisfied:

(a) $w \succ_m \sigma(w')$ if $K \in \mathcal{H}$ and $w \succsim_m \sigma(w')$ if $K \in \mathcal{G}$.
(b) $b \succsim_m \sigma(b')$.
(c) $b \succ_m \{s[\sigma(v)]\}$ and $b \succ_m \{\sigma(x)\}$ for all $x \in var(K)$ and $\sigma(x) \notin T(\Sigma_0, V)$.

Suppose that (G, τ) is a G-counterexample. Thus, $\longrightarrow$ is confluent below $\tau(b)$ and $\tau(C)$ is unjoinable wrt. $\longrightarrow$.
Case 1: $\tau(C_c)$ is unjoinable wrt. $\longrightarrow$ for some $G_c : (C_c \mid b \mid w)$ with $G_c \in \mathcal{G}_c$. As $\longrightarrow$ is confluent below the bound $\tau(b)$, (G_c, τ) is a counterexample. As G_c has the same weight as G, we have $(G, \tau) \succsim_i (G_c, \tau)$.
Case 2: $\tau(C_c)$ is joinable wrt. $\longrightarrow$ for all $G_c : (C_c \mid b \mid w)$ with $G_c \in \mathcal{G}_c$. In particular, $\tau(\Phi \Rightarrow \sigma(x)!, \Psi)$ is joinable wrt. $\longrightarrow$. As $\tau(\Phi \Rightarrow \Psi)$ is unjoinable wrt. $\longrightarrow$ by the assumption, it follows that there exists a Σ_0-ground substitution τ' such that $\tau\sigma(x) \longrightarrow^* \tau'(x)$ for all x.
Case 2.1: $\tau\sigma(\Gamma \Rightarrow u = v, \Delta)$ is unjoinable wrt. $\longrightarrow$. Note that $\longrightarrow$ is confluent below $\tau\sigma(b')$ as $\tau(b) \succsim \tau\sigma(b')$ and as $\longrightarrow$ is confluent below $\tau(b)$. Thus, $(\sigma(K), \tau)$ is a counterexample. It follows from the auxiliary lemma above that (K, τ') then is a counterexample too, satisfying $(\sigma(K), \tau) \succsim_i (K, \tau')$. If $K \in \mathcal{H}$, then $\tau(w) \succ_m \tau\sigma(w') \succsim_m \tau'(w')$. If $K \in \mathcal{G}$, then $\tau(w) \succsim_m \tau\sigma(w') \succsim_m \tau'(w')$. The case $K \in \mathcal{L}$ is impossible as lemmata cannot give rise to counterexamples. Thus, (K, τ') is a counterexample with $(G, \tau) \succ_i (K, \tau')$ resp. $(G, \tau) \succsim_i (K, \tau')$.
Case 2.2: $\tau\sigma(\Gamma \Rightarrow u = v, \Delta)$ is joinable wrt. $\longrightarrow$. Thus, as $\tau\sigma(\Gamma \Rightarrow \Delta)$ is unjoinable by the assumption, $\tau\sigma(u = v)$ is joinable wrt. $\longrightarrow$. Now consider the antecedent (pseudo-)equation $E' : s[\sigma(v)] = t$. We show that $\tau(E')$ is joinable wrt. $\longrightarrow$. We have $\tau(s)[\tau\sigma(v)] \downarrow \tau(s)[\tau\sigma(u)] \downarrow \tau(t)$. The latter relation holds as $\tau(C)$ is unjoinable. As $\tau(b) \succ \tau(s)$ and as $\longrightarrow$ is confluent below $\tau(b)$, we can infer that $\tau(s)[\tau\sigma(v)] \downarrow \tau(t)$. Thus, $\tau(C')$ is unjoinable. As $\longrightarrow$ is confluent below the bound $\tau(b)$, (G', τ) is a counterexample. As G and G' have the same weights, $(G, \tau) \succsim_i (G', \tau)$. □

Lemma 4.1 I_3 is inductively sound.

Proof: Let $(\mathcal{H}; \mathcal{G}, G)$ be the actual prover state. Let $K : (\Gamma_0, \Gamma_1 \Rightarrow \Delta_0, \Delta_1 | b' | w')$ be a syntactic construct from $\mathcal{H} \cup \mathcal{G} \cup \mathcal{L}$. Let σ be an arbitrary Σ-substitution such that $\sigma(K)$ is a syntactic construct too. Let $G : (C|b|w)$ with $C : \sigma(\Gamma_0), \Phi_1 \Rightarrow \sigma(\Delta_0), \Psi_1$. Let $\mathcal{G}_c$ contain all the syntactic constructs $G_c^A : (\Phi_1 \Rightarrow \sigma(A), \Psi_1 | b | w)$ with $A \in \Gamma_1$, $G_c^B : (\Phi_1, \sigma(B) \Rightarrow \Psi_1 | b | w)$ with $B \in \Delta_1$ and $G_c^x : (\Phi_1 \Rightarrow \sigma(x)!, \Psi_1 | b | w)$ such that $x \in var(K)$ and such that $\sigma(x) \notin T(\Sigma_0, V)$. Finally, let $I_3 : G \rightsquigarrow (\mathcal{G}_c, \emptyset)$. Thus, the following conditions are satisfied:
(a) $w \succ_m \sigma(w')$ if $K \in \mathcal{H}$ and $w \succsim_m \sigma(w')$ if $K \in \mathcal{G}$.
(b) $b \succsim_m \sigma(b')$.
(c) $b \succ_m \{\sigma(x)\}$ for all $x \in var(K)$ and $\sigma(x) \notin T(\Sigma_0, V)$.

Now assume that (G, τ) is a G-counterexample. Thus, $\longrightarrow$ is confluent below $\tau(b)$ and $\tau(C)$ is unjoinable wrt. $\longrightarrow$.
Case 1: $\tau(C_c)$ is unjoinable wrt. $\longrightarrow$ for some $G_c : (C_c \mid b \mid w)$ with $G_c \in \mathcal{G}_c$. As $\longrightarrow$ is confluent below the bound $\tau(b)$, (G_c, τ) is a counterexample. As G_c has the same weight as G, we have $(G, \tau) \succsim_i (G_c, \tau)$.
Case 2: $\tau(C_c)$ is joinable wrt. $\longrightarrow$ for all $G_c : (C_c \mid b \mid w)$ with $G_c \in \mathcal{G}_c$. In

particular, $\tau(\Phi_1 \Rightarrow \sigma(x)!, \Psi_1)$ is joinable wrt. $\longrightarrow$ for all $x \in var(K)$ such that $\sigma(x) \notin T(\Sigma_0, V)$. As $\tau(\Phi_1 \Rightarrow \Psi_1)$ is unjoinable wrt. $\longrightarrow$ due to the fact that $\tau(C)$ is unjoinable by assumption, it follows that there exists a Σ_0-ground substitution τ' such that $\tau\sigma(x) \longrightarrow^* \tau'(x)$ for all x.

We next show that $\tau\sigma(\Gamma_0, \Gamma_1 \Rightarrow \Delta_0, \Delta_1)$ is unjoinable wrt. $\longrightarrow$. Assume the contrary. Thus, $\tau\sigma(\Gamma_0, \Gamma_1 \Rightarrow \Delta_0, \Delta_1)$ is joinable wrt. $\longrightarrow$. As $\tau(C_c)$ is joinable wrt. $\longrightarrow$ and as $\tau(\Phi_1 \Rightarrow \Psi_1)$ is unjoinable wrt. $\longrightarrow$, it follows that $\tau\sigma(\Gamma_1 \Rightarrow \Delta_1)$ is unjoinable wrt. $\longrightarrow$. According to the assumption, $\tau\sigma(\Gamma_0 \Rightarrow \Delta_0)$ is joinable wrt. $\longrightarrow$. But then it follows that $\tau(C)$ is joinable wrt. $\longrightarrow$, which contradicts the fact that (G, τ) is a counterexample.

Thus, $\tau\sigma(\Gamma_0, \Gamma_1 \Rightarrow \Delta_0, \Delta_1)$ is indeed unjoinable wrt. $\longrightarrow$. Note that $\longrightarrow$ is confluent below $\tau\sigma(b')$ because $\tau(b) \succsim \tau\sigma(b')$. It follows from the auxiliary lemma above that (K, τ') is a counterexample too. If $K \in \mathcal{H}$, then $\tau(w) \succ_m \tau\sigma(w') \succsim_m \tau'(w')$. If $K \in \mathcal{G}$, then $\tau(w) \succsim_m \tau\sigma(w') \succsim_m \tau'(w')$. The case $K \in \mathcal{L}$ is impossible as lemmata cannot give rise to counterexamples. Thus, (K, τ') is a counterexample with $(G, \tau) \succ_i (K, \tau')$ resp. $(G, \tau) \succsim_i (K, \tau')$. □

Lemma 4.1 I_4 is inductively sound.

Proof: Let $(\mathcal{H}; \mathcal{G}, G)$ be the current prover state. Let $G : (C|b|w)$ with $C : \Phi, A \Rightarrow \Psi$. (The case $C : \Phi \Rightarrow A, \Psi$ is treated just analogously.) Let $G' : (C'|b|w)$ with $C' : \Phi \Rightarrow \Psi$. Let $G_c : (C_c|b|w)$ with $C_c : \Phi \Rightarrow A, \Psi$ (resp. $C_c : \Phi, A \Rightarrow \Psi$). Thus, $I_4 : G \rightsquigarrow (\{G_c\}, \{G'\})$.

Assume that (G, τ) is a G-counterexample. Thus, $\longrightarrow$ is confluent below $\tau(b)$ and $\tau(C)$ is unjoinable wrt. $\longrightarrow$. Then $\tau(C')$ is unjoinable wrt. $\longrightarrow$ too. As G and G' have the same weights, (G', τ) is a counterexample with $(G, \tau) \succsim_i (G', \tau)$. □

Theorem 4.1 Let $(\Sigma_0, \mathcal{R})$ be reductive wrt. a well-founded partial ordering $\succ$. Let $\vdash_{\mathcal{I}} = \vdash_{rec}$ or $\vdash_{\mathcal{I}} = \vdash_{con}$. Let $BCrit(\Sigma_0, \mathcal{R}) \subseteq \mathcal{G}$. Let $(\emptyset; \mathcal{G}) \vdash_{\mathcal{I}} \cdots \vdash_{\mathcal{I}} (\mathcal{H}'; \emptyset)$. Then $\longrightarrow_{(\Sigma_0, \mathcal{R})}$ is ground confluent.

Proof: Let $(\emptyset; \mathcal{G}) \vdash_{\mathcal{I}} \cdots \vdash_{\mathcal{I}} (\mathcal{H}'; \emptyset)$. As $\mathcal{I}$ is inductively sound it follows from Lemma 3.2 that $\vdash_{\mathcal{I}}$ is inductively sound too. Using Theorem 3.1 it follows that $P(\mathcal{G})$ is true. The latter means that $BCrit(\Sigma_0, \mathcal{R})$ is weakly ground joinable. Thus, $\longrightarrow_{(\Sigma_0, \mathcal{R})}$ is ground confluent according to Theorem 2.1. □

B Examples

We briefly discuss the introductory examples once again, using now our proving method in a formal manner. In order to simplify notations we write sx instead of $s(x)$.

Let $\mathcal{R}$ consist of the following rules — written now in a clausal fashion:

$$\begin{array}{llllll}
(R1) & \Rightarrow & even(0) & \rightarrow & t & \\
(R2) & \Rightarrow & even(sx) & \rightarrow & t & , \ even(x) = t \\
(R3) & \Rightarrow & even(ssx) & \rightarrow & even(x) &
\end{array}$$

Let Σ_0 be the sub-signature of Σ that consists of the constructors $0, s, t, f$. $(\Sigma_0, \mathcal{R})$ is obviously reductive wrt. a suitable recursive path ordering. We show that $\longrightarrow_{(\Sigma_0,\mathcal{R})}$ is ground confluent, using Theorem 4.1. Our goal is to show that the set $BCrit(\Sigma_0, \mathcal{R})$ — which contains the bounded clause $C : (\Rightarrow even(x) = t, even(sx) = t \mid \{even(ssx)\})$ only — is weakly Σ_0-ground joinable wrt. $\longrightarrow_{(\Sigma_0,\mathcal{R})}$. The initial weight chosen for this bounded clause is the bound $\{even(ssx)\}$. We use the derivation relation $\vdash_{rec}$ induced by $\mathcal{I}$. Let
$G_{0.0} : (\Rightarrow even(x) = t, even(sx) = t \mid \{even(ssx)\} \mid \{even(ssx)\})$
be the initial goal. We first use rule I_1 wrt. the complete case splitting $S = \{(x \mapsto 0, \Box \Rightarrow \Box), (x \mapsto sx, \Box \Rightarrow \Box)\}$. The first case is easy, so we do not consider it further. Let
$G_{0.1} : (\Rightarrow even(sx) = t, even(ssx) = t \mid \{even(sssx)\} \mid \{even(sssx)\})$
result from $G_{0.0}$ by applying the instantiation $x \mapsto sx$. Now we can use the inference rule I_2 in order to simplify $G_{0.1}$. More concretely, we use the rewrite rules $R3$ as a lemma in order to rewrite $even(ssx)$ into $even(x)$. The application of this lemma causes no difficulties because the rewrite rule is unconditional. Note further that the condition concerning the bounds is satisfied. Now one obtains the goal
$G_{0.2} : (\Rightarrow even(sx) = t, even(x) = t \mid \{even(sssx)\} \mid \{even(sssx)\})$
As outlined in the introduction we want to subsume $G_{0.2}$ by the initial goal $G_{0.0}$. The latter indeed can be done by using rule I_3. Note that the striking condition concerning the weights is satisfied: The goal $G_{0.0}$ is an element of $\mathcal{H}$. Thus, we have to check whether $\{even(sssx)\} \succ_m \{even(ssx)\}$. But the latter is obviously true. It thus follows that $BCrit(\Sigma_0, \mathcal{R})$ is indeed weakly Σ_0-ground joinable wrt. $\longrightarrow_{(\Sigma_0,\mathcal{R})}$ and that $\longrightarrow_{(\Sigma_0,\mathcal{R})}$ is thus ground confluent.

Let $\mathcal{R}$ consist now of the following rules (which are written in a clausal fashion):

$$\begin{array}{llllll}
(R1) & \Rightarrow & even(0) & \rightarrow & t & \\
(R2) & \Rightarrow & even(sx) & \rightarrow & f & , even(x) = f \\
(R3) & \Rightarrow & even(sx) & \rightarrow & t & , even(x) = t
\end{array}$$

Let Σ_0 be the sub-signature of Σ that consists of the constructors $0, s, t, f$. $(\Sigma_0, \mathcal{R})$ is obviously reductive. We show that $\longrightarrow_{(\Sigma_0,\mathcal{R})}$ is ground confluent, using Theorem 4.1, proceeding as above. Our goal is to show that the set $BCrit(\Sigma_0, \mathcal{R})$ — which contains

the bounded clause $C_0 : (\Rightarrow f = t, even(x) = f, even(x) = t \mid \{even(sx)\})$ only — is weakly Σ_0-ground joinable wrt. $\longrightarrow_{(\Sigma_0,\mathcal{R})}$. Thus, let
$G_{0.0} : (\Rightarrow f = t, even(x) = f, even(x) = t \mid \{even(sx)\} \mid \{even(sx)\})$
be the initial goal. First we want to delete the equation $f = t$ using rule I_4. For that purpose we have to verify that the goal
$G_{1.0} : (f = t \Rightarrow even(x) = f, even(x) = t \mid \{even(sx)\} \mid \{even(sx)\})$
can be deleted by our calculus. This is indeed possible by using the case splitting rule I_1, because the empty set $S = \emptyset$ is a complete case splitting for $f = t \Rightarrow even(x) = f, even(x) = t$. The latter follows from the general fact that narrowing with $\mathcal{R}$ into an antecedent equation provides a complete case splitting and from the fact that in the particular case above no narrowing step into the equation $f = t$ is possible. Thus, the literal elimination rule is applicable and it provides the new goal
$G_{0.1} : (\Rightarrow even(x) = f, even(x) = t \mid \{even(sx)\} \mid \{even(sx)\})$.
Now we apply rule I_1 wrt. the complete case splitting $S = \{(x \mapsto 0, \Box \Rightarrow \Box), (x \mapsto sx, \Box \Rightarrow \Box)\}$. The new goal obtained by the instantiation $x \mapsto 0$ is easy to handle. So we only treat the new goal
$G_{0.2} : (\Rightarrow even(sx) = f, even(sx) = t \mid \{even(ssx)\} \mid \{even(ssx)\})$.
Next we want to apply the simplification rule I_2 by using the rewrite rule $R2$ as a lemma (in order to simplify the first equation of $G_{0.2}$). If such an application is possibe, then the transformed goal contains the succedent equation $f = f$, indicating a successful joinability proof. Formally, one deletes the resulting transformed goal by using the rule I_3, because is can be subsumed by the lemma $f = f$. In order to enable the mentioned rewrite step by $R2$ we have to verify that the condition of $R2$ is satisfied wrt. the actual context (note that the condition concerning the bounds is satisfied). Concretely, we have to further treat the condition goal
$G_{1.1} : (even(x) = f \Rightarrow even(sx) = t \mid \{even(ssx)\} \mid \{even(ssx)\})$.
There are several possibilities to proceed. One could e.g. produce a case splitting by narrowing with $\mathcal{R}$ into the antecedent equation. We proceed by applying the rule I_2 wrt. the rewrite rule $R3$ (considered as a lemma) in order to produce the succedent equation $t = t$. This leads to a successful treatment of the goal $G_{1.1}$. Again, we have to treat the related condition goal which now has the form
$G_{2.0} : (even(x) = f, even(x) = t \Rightarrow \Box \mid \{even(ssx)\} \mid \{even(ssx)\})$.
Now it is convenient to produce a case splitting by narrowing with $\mathcal{R}$ into one of the antecedent equations, say $even(x) = f$. This case splitting has the form $S = \{(x \mapsto 0, \Box \Rightarrow \Box), (x \mapsto sx, \Box \Rightarrow even(x) = f), (x \mapsto sx, \Box \Rightarrow even(x) = f)\}$. We do not further treat the goal resulting from the mere instantiation $x \mapsto 0$. First we consider the goal
$G_{2.1} : (even(sx) = f, even(sx) = t \Rightarrow even(x) = f \mid \{even(sssx)\} \mid \{even(ssx)\})$.
This goal can be transformed by using I_2 wrt. the lemma $R2$. Concretely, we use $R2$ in order to produce the antecedent equation $f = t$. The newly generated goal is then deletable because the empty set provides a complete case splitting for this goal. The verification that $R2$ is indeed applicable is easy now: The related condition clause contains the equation $even(x) = f$ as an antecedent and succedent equation. One can use now a suitable lemma to subsume this condition clause. The second goal to be considered is

$G_{2.2}: (even(sx) = f, even(sx) = t \Rightarrow even(x) = t \mid \{even(sssx)\} \mid \{even(sssx)\})$.
Here we can proceed just analogously.

Now, all goals are treated successfully. It follows that $BCrit(\Sigma_0, \mathcal{R})$ is indeed weakly Σ_0-ground joinable wrt. $\longrightarrow_{(\Sigma_0,\mathcal{R})}$ and that $\longrightarrow_{(\Sigma_0,\mathcal{R})}$ is thus ground confluent.

Surely, these examples are rather simple. But there are much harder examples which can be treated successfully by our proving method.