On generic representation of implicit induction procedures

D. Naidich

Computer Science/Department of Software Technology

CS-R9620 1996

Report CS-R9620
ISSN 0169-118X

CWI
P.O. Box 94079
1090 GB Amsterdam
The Netherlands

CWI is the National Research Institute for Mathematics and Computer Science. CWI is part of the Stichting Mathematisch Centrum (SMC), the Dutch foundation for promotion of mathematics and computer science and their applications.
SMC is sponsored by the Netherlands Organization for Scientific Research (NWO). CWI is a member of ERCIM, the European Research Consortium for Informatics and Mathematics.

Copyright © Stichting Mathematisch Centrum
P.O. Box 94079, 1090 GB Amsterdam (NL)
Kruislaan 413, 1098 SJ Amsterdam (NL)
Telephone +31 20 592 9333
Telefax +31 20 592 4199

On Generic Representation of Implicit Induction Procedures

Dimitri Naidich
CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands
e-mail: `dimitri@cwi.nl`

Abstract

We develop a generic representation of implicit induction proof procedures within the cover set induction framework. Our work further develops the approach of cover set induction on propositional orderings. We show that in order to represent a substantially wide range of implicit induction procedures it is necessary to generalize the induction on formulas (propositions) to the induction on formula instances. We present a generic induction procedure which captures virtually all the existing implicit induction procedures developed from the inductive completion framework. We establish the formal relationship between the generic induction procedure and the cover set induction scheme. We also demonstrate that the developed generic framework allows for easy generalization/modification of the existing procedures.

CR Subject Classification (1991): I.2.3, I.1.3.
Keywords & Phrases: automated induction, algebraic specifications, term rewriting.
Note: The author was supported by the NWO visiting research fellowship.

1 Introduction

1.1 Motivation

Reasoning by induction is important for verifying properties of the standard initial models of algebraic specifications. For this reason, such properties are often called *inductive*. Conventional methods for proving inductive properties employ some induction schemes like structural induction [Aub79, BM79, ZKK88, Wal94] or noetherian induction [Pad92, Fra93]. Given a conjecture to prove, a proper instantiation of such a scheme specifies the induction cases and the instances of the induction hypothesis which can be used in the proofs of these cases. Due to the explicit distinction of induction cases and induction hypotheses, the conventional methods are often called *explicit*.

In the seminal paper [Mus80], Musser proposed another approach to proving inductive conjectures. It was based on the fact that the (model-theoretical) consistency of a conjecture with a specification is sufficient for its validity in the initial model. To show the consistency, Knuth-Bendix completion was originally used. Since no induction cases and induction hypotheses were explicitly distinguished in the completion proofs, the method was initially called *inductionless induction*. However, yet Musser [Mus80] indicated the existence of some relation between his method and conventional induction: " ... the use of induction becomes 'meta', and proofs of inductive data type properties often can be carried out without explicitly invoking an induction principle."

Various optimizations, modifications and generalizations of the original inductive completion procedure have been developed since then, e.g. [HH80, Der82, JK86, KNZ86, Fri86, BK89].

Informal analogies between these procedures and conventional induction were emphasized elsewhere. However, the formal relationship is desirable for the following reasons. *Firstly,* due to the inherent incompleteness of inductive proof methods, the ability of user control is very important for inductive proofs, in general. The original inductive completion procedures were completely automated, hard to understand and did not facilitate user interaction [GG88]. In the case of proof failure there was nothing the user could do. Conventional interpretation of the inductive completion would allow to employ proof heuristics to improve proof flexibility. *Secondly,* many implicit induction procedures developed since then do not fit the completion and/or proof-by-consistency frameworks in the way they handle non-orientable conjectures, e.g. [Bac88, Gra89, BL90], and/or because they do not require the ground confluence of axioms, e.g. [HK88, KR90, BR95, BKR95, Bou95]. Still the similarity of many implicit induction procedures suggests the existence of a unifying framework. Such a framework would facilitate modification and generalization of the existing procedures.

In this paper we further develop the representation of implicit induction procedures within the framework of cover set induction on propositional orderings [Red90, BRH94]. In [Red90], Reddy presented the explicit *term rewriting induction* scheme (*TRI*) as the justification basis for the inductive completion procedures for orientable equations. TRI was based on the following two concepts. *Firstly,* ground equations were considered as the induction domain. *Secondly,* the concept of *cover set* was used to obtain a finite representation of that induction domain. Reddy further presented the *inductive completion procedure* (*ICP*) based on the notion of cover set. This procedure covers virtually all the existing inductive completion procedures for orientable equations such as [HH80, Der82, JK86, KNZ86, Fri86, BK89] and does not require the ground confluence of axioms. This approach was further developed in [BRH94]. There, TRI was generalized to the *noetherian induction with cover sets* (*NICS*) for the case of general formulas; the generic *implicit induction procedure* (*IIP*) generalized ICP to capture, apart from the inductive completion procedures for orientable equations, the implicit induction procedures [KR90, BR95, Bou95] that permit conditional rewrite systems as axioms and equational clauses as conjectures. Also, the explicit induction technique of *forward inferences* was uniformly merged with the implicit induction techniques.

1.2 Problems

The framework [Red90, BRH94] forms the basis for representing implicit induction procedures developed from the inductive completion. However, it still leaves room for improvement. We point out two major problems. *Firstly,* some advanced implicit induction procedures such as [Bac88, BL90, BKR95] are not captured by IIP and NICS. *Secondly,* there is some discrepancy between the proof procedures ICP, IIP and the induction schemes TRI, NICS. We address these issues in more detail below.

1.2.1 Representing Concrete Proof Procedures

The most distinctive aspect of the implicit induction procedures [Bac88, BL90, BKR95] which is not captured by IIP and NICS is the combination of the following features:

1. rewriting a conjecture by an inductive hypothesis at the top position, and
2. handling non-orientable conjectures.

We illustrate the first aspect by an example of compilation of arithmetic expressions. Consider the expressions $Expr$ of sums of natural numbers

$$Nat \subset Expr, \quad sum : Nat, Nat \rightarrow Expr,$$

and their interpretation $int : Expr \rightarrow Nat$, so that

$$\begin{aligned} int(n) &= n, \\ int(sum(exp_1, exp_2)) &= int(exp_1) + int(exp_2). \end{aligned}$$

Consider further the commands Com:

$$Nat \subset Com, \quad add :\rightarrow Com,$$

and their interpretation in the stack machine

$$run : List(Code), List(Nat) \rightarrow Nat, \text{ so that}$$

$$\begin{aligned} run(nil,\ n; nil) &= n, \\ run(n; P,\ S) &= run(P,\ n; S), \\ run(add; P,\ n_2; n_1; S) &= run(P,\ (n_1 + n_2); S), \end{aligned}$$

where nil, and ";" are the generic list constructors. Finally, consider the compilation function

$$comp : Expr \rightarrow List(Code), \text{ so that}$$

$$\begin{aligned} comp(n) &= n; nil, \\ comp(sum(exp_1, exp_2)) &= comp(exp_2); ; (comp(exp_1); ; (add; nil)), \end{aligned}$$

where ";;" denotes the lists concatenation defined as usual. The compilation correctness is formalized as

$$run(comp(exp), nil) = int(exp).$$

We consider its generalization

$$run(comp(exp); ; P,\ S) = run(P,\ int(exp); S). \tag{1}$$

The proof of (1) requires the associativity of code lists

$$(L_1; ; L_2); ; L3 = L_1; ; (L2; ; L3)$$

as a lemma. The critical case in proving (1) is when $exp = sum(exp_1, exp_2)$. By the axioms and the associativity lemma,

$$run(comp(sum(exp_1, exp_2)); ; P,\ S)$$

rewrites to

$$run(comp(exp_1); ; (comp(exp_2); ; (add; P)),\ S).$$

Rewriting this term at the top occurrence by (1) twice is essential for accomplishing the proof of this case.

The proof described above can be performed by ICP. However, this procedure is not suitable for proving non-orientable conjectures. The proof of (1) can be performed by, e.g. [BKR95], in the more general context permitting non-orientable conjectures.

However, we believe that the proof procedures like [BKR95] cannot be captured by the techniques [Red90, BRH94], because the induction domain of ground formulas is not quite adequate to the induction used in the justification of such implicit induction procedures. The analysis of the related justification techniques shows that the induction on the *ground instances of formulas* is more adequate there. In this context, a ground instance of a formula ϕ is a pair (ϕ, γ), where γ is a ground substitution. A ground instance (ϕ, γ) corresponds to ground formula $\phi\gamma$. The more elaborated structure of the ground instances allows for more flexible induction orderings.

1.2.2 Implicit vs. Explicit Induction: Formal Aspects

The informal relation between implicit and explicit induction has been discussed elsewhere. The formal relation, however, is straightforward just for the non-"hierarchical", as it is called in [Red90], implicit induction procedures such as [HK88, Gra89, Red90]. Such a procedure takes equational conjectures, generates their cover instances and rewrites them by the axioms - rewrite rules and the induction hypotheses, if the latter are orientable; it may also permit some restricted usage of the non-orientable hypotheses. The proof succeeds if all the cover instances are reduced to tautologies (or logical consequences of the axioms, in a more general setting). The relation of such procedures to TRI (more precisely, its extension for the case of multiple conjectures) is quite straightforward.

However, the practical proof procedures such as ICP are *hierarchical*, as far as they permit the nested induction for the cover instances that cannot be reduced to tautologies. The formal relation between ICP and TRI was not shown in [Red90]. ICP is a particular case of IIP, and TRI is a particular case of NICS. So there would be no problem if IIP could be justified by NICS. However, IIP *cannot* be justified by NICS as it was claimed in [BRH94]. The reason is that IIP employs the notion of *conditioned cover set*, while NICS is based on the notion of cover set compatible with that of [Red90]. We address this problem in more detail below. We superpose the definitions of the compared notions in order to make the comparison easier.

Definition 1.1 (Cover sets and conditioned cover sets [BRH94]) Let Ax be axioms, and $\succcurlyeq$ a quasi-order on formulas. A set of formulas $\{\psi_i\}_i$ is a *cover set* (*conditioned cover set*) for a formula ϕ if, for every ground substitution γ, there exist a formula ψ_i and ground substitution γ' such that

1. $\psi_i\gamma' \preccurlyeq \phi\gamma$ ($\psi_i\gamma' \prec \phi\gamma$), and
2. $Ax \models \psi_i\gamma' \Rightarrow \phi\gamma$

The concept of cover set [Red90, BRH94] is close to that of test set used in the inductive completion [KNZ86] and implicit induction [KR90], and to the concept of cover set used in the explicit induction scheme [ZKK88]. The concept of conditioned cover set is analogous to that of cover set used in the proofs by consistency [Bac88, Gra89, BL90]. The difference between the cover sets and conditioned cover sets causes the difference between the corresponding induction schemes. We reproduce the definition of NICS and make explicit the definition of the *induction with conditioned cover sets* (*ICCS*) as it is implicitly used in [BRH94].[1]

Proposition 1.2 (NICS and ICCS [BRH94]) *Let Ax be axioms, and $\succcurlyeq$ a well-founded quasi-order on formulas such that $\succ$[2] ($\succcurlyeq$) is stable. Given a proposition ϕ, $Ax \models_{ind} \phi$ if there exists a cover set (conditioned cover set) of formulas $\{\psi_i\}_i$ w.r.t. Ax, $\succcurlyeq$ such that, for each ψ_i, there exists a set of substitutions $\{\theta_j\}_j$ such that*

1. *for each j, $\phi\theta_j \prec \psi_i$ ($\phi\theta_j \preccurlyeq \psi_i$), and*
2. *$Ax \models \wedge_j \phi\theta_j \Rightarrow \psi_i$*

The notion of conditioned cover set is stronger than that of cover set. On the other hand, the ordering condition in the proposition above is weaker for the conditioned cover sets. Therefore, the two modification of cover set induction are generally incompatible.

[1] The reader is referred to Section 2.1 for the definitions of stability and $\models_{ind}$.

[2] In [BRH94], the stability of $\succ$ is incorrectly replaced with the stability of $\succcurlyeq$.

We believe that ICP still can be justified by NICS. We do not consider this issue in the paper, though, because the existing generalizations of ICP like [BKR95] cannot be covered by NICS (as well as ICCS) anyway. We rather consider ICP in the generalized framework of induction on formula instances.

The induction with cover sets is closer to the conventional induction than the induction with conditioned cover sets. E.g., the structural induction scheme [Aub79] is an instance of NICS, while it is not an instance of ICCS. Therefore, we adopt the original approach [Red90] rather than that of induction with conditioned cover sets [BRH94].

1.3 Results

In this paper we generalize NICS for the domain of formula instances. This allows us to capture, to the best of our knowledge, practically all the existing implicit induction procedures developed from the inductive completion framework, including the advanced procedures [Bac88, BL90, BKR95]. Akin [BRH94], we represent implicit induction procedures by means of a generic inference system, so that the concrete proof procedures correspond to proof strategies within the inference system. We establish the formal relation between the generic inference system and the cover set induction scheme.

We also show that our generic framework immediately allows for easy generalization/modification of the existing induction procedures. This development was stimulated by the fact that the induction procedure for conditional equations [BR95] does not fully generalize the the induction procedure for pure equations [BKR95]. Our approach immediately allows for the proper modification of [BR95] which generalizes [BKR95]. Further generalizations are possible on the ground that we ignore the refutational aspects of implicit induction procedures [Gra90, Bev91]. Along with incorporating the conventional conjecture generalization, it also allows us to eliminate some technical restrictions on the goal simplification like the specialization ordering [Bac88].

The paper is organized as follows. In Section 2 we introduce the *generalized cover set induction* (*GCSI*), and the *generic inference system* $\mathcal{I}$ to represent implicit induction procedures. We establish the formal relation between $\mathcal{I}$ and GCSI. The inference system $\mathcal{I}$ is essentially "descriptive". It specifies the requirements on the cover set generation and simplification techniques that are not computable, in general.

In the rest of the paper we consider "computable" (i.e. posing computable requirements) instantiations of $\mathcal{I}$ for the case of axioms-clauses. In Section 3 we consider the methods of computing cover sets. The presented instantiations of $\mathcal{I}$ correspond to the two instances of the orderings on clause witnesses defined in Section 4. In Section 5 we consider the simplification techniques corresponding to these orderings, and show that these techniques satisfy the requirements imposed by $\mathcal{I}$. Finally, in Section 6, we combine the cover set and simplification components presented in the previous sections to get the "computable" inference systems $\mathcal{J}$ and $\mathcal{K}$. The proofs of the presented propositions are put in Appendix A.

The relations between the induction procedures and induction schemes discussed in this paper are summarized in Figure 1. There, the induction schemes and "descriptive" inference systems are enclosed in ovals, and the "computable" inference systems and concrete proof procedures are enclosed in boxes. We group together concrete induction procedures covered by the same generic procedures. The arrows denote the instantiation relations between the procedures. The dotted arrow denotes the "almost"-instantiation relation (cf. Section 6).

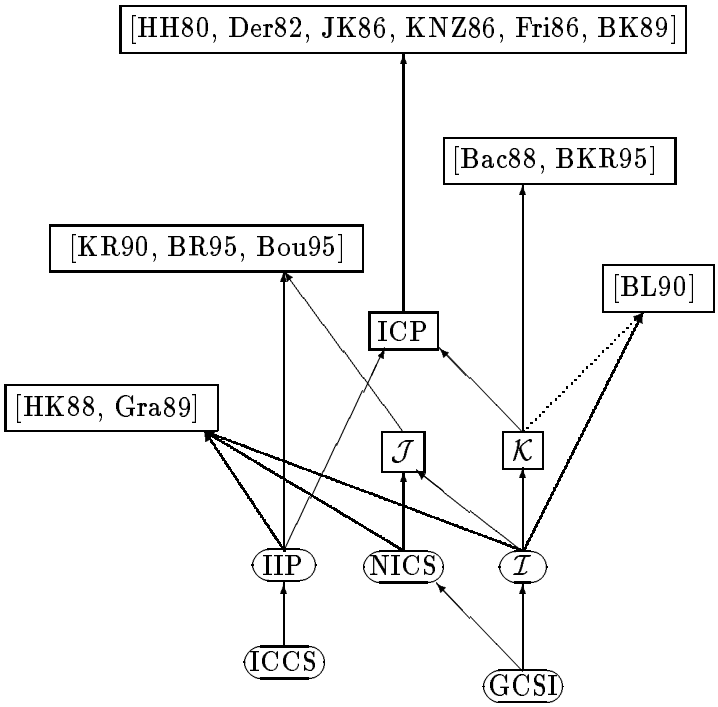

Figure 1: Implicit Induction Procedures and Cover Set Induction Schemes

2 Implicit Induction: Abstract Framework

2.1 Basic Notions

A formula ϕ is a *semantic consequence* of axioms Ax, denoted $Ax \models \phi$, if ϕ is valid in every model of Ax. A formula ϕ is an *inductive consequence* of axioms Ax, denoted $Ax \models_{ind} \phi$, if $Ax \models \phi\gamma$ for any ground substitution γ. $\models_{ind}$ is the conventional notion of inductive consequence corresponding to the validity of a conjecture in all the term generated models of the axioms.

A *quasi-order* is a reflexive transitive relation. A *(partial) order* is an irreflexive and transitive relation. An *equivalence* is a reflexive, transitive, and symmetric relation. Given a quasi-order $\succcurlyeq$, the *strict part* of $\succcurlyeq$ is the order $\succ$ defined as follows: $a \succ b$ iff $a \succcurlyeq b$ and $b \not\succcurlyeq a$. The *equivalence part* of $\succcurlyeq$ is the equivalence $\sim$ defined as follows: $a \sim b$ iff $a \succcurlyeq b$ and $b \succcurlyeq a$. On the other hand, any partial order $\succ$ and equivalence $\sim$ such that $\succ \cap \sim = \emptyset$ define the quasi-order $\succcurlyeq \equiv \succ \cup \sim$. Given a quasi-order $\succcurlyeq$, we write $a \eqcirc b$ if neither $a \succcurlyeq b$, nor $b \succcurlyeq a$. A quasi-order $\succcurlyeq$ is *well-founded* if there is no infinite strictly descending sequence $a_1 \succ a_2 \succ \ldots$ of elements. A relation $\gg$ is *stable* if $a \gg b$ implies $a\sigma \gg b\sigma$ for any substitution σ. A quasi-order $\succcurlyeq$ is *strongly stable* if $\succ$ and $\sim$ are stable.

2.2 Cover Set Induction

The induction domain of the pairs "formula-substitution" is the core of our abstract representation of implicit induction procedures. We call such pairs formula witnesses because the term "formula instance" is usually interpreted as a formula.

Definition 2.1 (Formula witnesses) A *formula witness* is a pair $\langle \phi, \sigma \rangle$, where ϕ is a formula, and σ is a substitution. A formula ϕ is considered as the formula witness $\langle \phi, id \rangle$. The *application of a substitution* θ to a formula witness $\langle \phi, \sigma \rangle$, denoted $\langle \phi, \sigma \rangle \theta$, yields the formula witness $\langle \phi, \sigma\theta \rangle$.

The concept of cover set is used to get a finite representation of the induction domain. The definition of cover set below directly generalizes definition 1.1 for the case of formula witnesses.

Definition 2.2 (Cover set of formula witnesses) Let Ax be axioms, and $\succcurlyeq$ a quasi-order on formula witnesses. A set of formula witnesses $\{\langle \psi_i, \sigma_i \rangle\}_i$ is a *cover set* for a formula ϕ if, for every ground substitution γ, there exist a formula witness $\langle \psi_i, \sigma_i \rangle$ and ground substitution γ' such that

1. $\langle \psi_i, \sigma_i \rangle \gamma' \preccurlyeq \langle \phi, \gamma \rangle$, and
2. $Ax \models \psi_i \sigma_i \gamma' \Rightarrow \phi\gamma$

We describe methods of generating cover sets in Section 3.

Next, we properly modify NICS, definition 1.2.

Proposition 2.3 (Cover set induction on formula witnesses) *Let Ax be axioms, and $\succcurlyeq$ a well-founded quasi-order on formula witnesses such that $\succ$ is stable.*

Given a proposition ϕ, $Ax \models_{ind} \phi$ if there exists a cover set of formula witnesses $\{\langle \psi_i, \sigma_i \rangle\}_i$ w.r.t. Ax, $\succcurlyeq$ such that, for each $\langle \psi_i, \sigma_i \rangle$, there exists a set of substitutions $\{\theta_j\}_j$ such that

1. *for each j, $\langle \phi, \theta_j \rangle \prec \langle \psi_i, \sigma_i \rangle$, and*
2. *$Ax \models \wedge_j \phi\theta_j \Rightarrow \psi_i\sigma_i$*

As it was emphasized in [BRH94], the induction on formulas facilitates mutual induction, i.e. the mutual usage of the induction hypotheses in the proofs of multiple conjectures. The induction on multiple conjectures is highly relevant to the representation of implicit induction procedures. So we generalize the proposition above directly for the case of multiple formulas. We introduce a concise notation akin [BRH94] before. We consider possibly infinite conjunctions as implication premises. Given a possibly infinite set of formulas Φ, we write $\wedge\Phi$ for the conjunction of the elements of Φ.[3] We write $Ax \models \bigwedge \Phi \Rightarrow \phi$ if there exists a finite subset Φ' of Φ such that $Ax \models \bigwedge \Phi' \Rightarrow \phi$. Given a binary relation on formula witnesses $\ll$, a formula witness $\langle \phi, \sigma \rangle$, and a set of formula witnesses $\langle \Psi \rangle$, we write $\langle \Psi \rangle_{\ll \langle \phi, \sigma \rangle}$ to denote the possibly infinite conjunction $\wedge\{\psi\theta\sigma' | \langle \psi, \theta \rangle \in \langle \Psi \rangle, \langle \psi, \theta \rangle \sigma' \ll \langle \phi, \sigma \rangle\}$.

Proposition 2.4 (Generalized cover set induction (GCSI)) *Let Ax be axioms, and $\succcurlyeq$ a well-founded quasi-order on formula witnesses such that $\succ$ is stable. Given a set of propositions Φ, $Ax \models_{ind} \Phi$ if, for each $\phi \in \Phi$, there exists a cover set of formula witnesses $\langle \Psi \rangle_\phi$ w.r.t. Ax, $\succcurlyeq$ such that, for each $\langle \psi, \sigma \rangle \in \langle \Psi \rangle_\phi$,*

$$Ax \models \Phi_{\prec \langle \psi, \sigma \rangle} \Rightarrow \psi\sigma \tag{2}$$

2.3 A Generic Induction Procedure

Following the paradigm "Algorithms = Logic + Control" [Kow79], the implicit induction procedures have been formalized by inference systems, e.g. [Bac88, Red90], specifying possible proof state transformations. The presented generic inference system $\mathcal{I}$, Figure 2, is a modification of the *mutual induction procedure* [BRH94] for the case of formula witnesses. Axioms Ax and a

[3] We define $\wedge\emptyset$ as the propositional truth.

Generate

$$\frac{(P \cup \{\phi\}, H)}{(P \cup P', H \cup \{\phi\})}$$

if $\langle \Psi \rangle$ is a cover set of formula witnesses for ϕ w.r.t. Ax, $\succcurlyeq$ such that, for any $\langle \psi, \sigma \rangle \in \langle \Psi \rangle$, $Ax \models (P \cup P' \cup H \cup \{\phi\})_{\prec \langle \psi, \sigma \rangle} \Rightarrow \psi\sigma$

Simplify

$$\frac{(P \cup \{\phi\}, H)}{(P \cup P', H)}$$

if $Ax \models (P \cup P' \cup H)_{\preccurlyeq \phi} \Rightarrow \phi$

Figure 2: Inference system $\mathcal{I}(Ax, \succcurlyeq)$

quasi-order on formula witnesses $\succcurlyeq$ are the parameters of $\mathcal{I}$. A proof state of $\mathcal{I}(Ax, \succcurlyeq)$ is a pair of sets of formulas (P, H), where P represents proof goals, and H induction hypotheses.

The inference rules of $\mathcal{I}(Ax, \succcurlyeq)$ are essentially descriptive. They specify the requirements on the proof state transformations to be satisfied by concrete proof procedures. Rule *Generate* specifies the generation of the subgoals, the induction cases of a proof goal, and their initial simplification. Rule *Simplify* specifies "regular" simplifications of proof goals which also may result in multiple subgoals, as well as in the goal elimination. The descriptive nature of $\mathcal{I}(Ax, \succcurlyeq)$ facilitates the modular development of diverse simplification and cover set generation techniques, while providing us with a uniform framework for their justification. We address the cover generation techniques in Section 3. We generalize the existing simplification techniques in Section 5.

A typical inference strategy of $\mathcal{I}(Ax, \succcurlyeq)$ amounts to, *first*, the simplification of current goals by *Simplify* and, *second*, the generation of induction cases for the simplified goals by *Generate*. During the simplification, the goals either are eliminated, or form lemmas subjected to further induction cycles. The instances of the conjectures used in the simplification are determined "on the fly". The soundness of their usage follows from the reduction of the goal complexity during the simplification. The proven conjectures are inductive theorems if all the goals are eventually eliminated.

The basic notion relating $\mathcal{I}(Ax, \succcurlyeq)$ to the inductive proofs is that of successful derivation. We write $(P, H) \vdash (P', H')$ to denote that a proof state (P', H') is obtained from a proof state (P, H) by an application of an inference rule. Given a set of formulas P, a *successful derivation* is an inference sequence $(P, \emptyset) \vdash \ldots \vdash (\emptyset, H)$ for some H.

Proposition 2.5 (Soundness of $\mathcal{I}(Ax, \succcurlyeq)$) *Let Ax be axioms, and $\succcurlyeq$ be a strongly stable well-founded quasi-order on formula witnesses. Given a set of propositions P, $Ax \models_{ind} P$ if there exists a successful derivation for P by $\mathcal{I}(Ax, \succcurlyeq)$.*

2.4 On Concrete Proof States

We will address the relation of the generic procedure to concrete implicit induction procedures in more detail in the course of the paper. Here we describe the general correspondence between the generic proof state and the proof states of the concrete procedures. While the generic proof state is essentially the same as the proof states of the inductive completion procedures and the test set induction procedures, it looks different from the proof states of the proof-by-consistency procedures [Bac88, Gra89, BL90]. However, the correspondence is easy to establish taking into account the marking mechanisms of those proof procedures.

The proof-by-consistency procedures [Bac88, Gra89, BL90] are primarily aimed at the refutation of the proof goals. The "positive proofs" are obtained by making sure that no refutation is possible. The notion of *fair derivation* is the basic means to detect the impossibility of refutation. Some marking is employed to determine the fair derivations. A fair derivation terminates when all the proof goals are marked. Therefore, although the proof state there is just a set of formulas, one can distinguish marked and unmarked proof goals. The unmarked goals correspond to the first component of the generic proof state, and the marked goals to the second.

2.5 Implicit vs. Explicit Induction

We prove the soundness of the generic inference system, Proposition 2.5, by establishing the formal relation between the successful derivations and the proofs by GCSI, Proposition 2.4. This facilitates understanding of implicit induction proofs within the conventional induction framework.

Proposition 2.6 ($\mathcal{I}$ vs. GCSI) *Let $\succcurlyeq$ be a stable quasi-order on formula witnesses. If $(P^\dagger, \emptyset) \vdash \ldots \vdash (\emptyset, H^\dagger)$ by $\mathcal{I}(Ax, \succcurlyeq)$ then, for any $\phi \in H^\dagger$, there exists a cover set of formula witnesses $\langle\Psi\rangle_\phi$ w.r.t. Ax, $\succcurlyeq$ such that, for each $\langle\psi, \sigma\rangle \in \langle\Psi\rangle_\phi$, condition (2) of Proposition 2.4 holds. Also, $Ax \cup H^\dagger \models P^\dagger \setminus H^\dagger$.*

The validity of condition (2) of Proposition 2.4 implies the existence of a finite set of formula witnesses $\{\langle\phi_i, \theta_i\rangle\}_i$ such that $\phi_i \in \Phi$ and $\langle\phi_i, \theta_i\rangle \prec \langle\psi, \sigma\rangle$ for any i, and $Ax \models \wedge_i \phi_i\theta_i \Rightarrow \psi\sigma$. Therefore, $\mathcal{J}$ specifies strategies of finding the instances of induction hypotheses for the meta-level cover set induction proofs. The merit of Proposition 2.6 is that the correspondence between implicit and explicit induction is uniformly determined for all the implicit induction procedures representable by $\mathcal{I}$.

The inference system $\mathcal{I}$ is actually equivalent to GCSI w.r.t. the set of provable conjectures, because any proof by GCSI corresponds to a successful derivation by *Generate*, where $P' = \emptyset$. The distinction of the intermediate goals P' in *Generate* and *Simplify* reflects the step-wise simplification of the proof goals as the main component of the implicit induction proof search strategies.

2.6 Instantiations of the Abstract Framework

An instantiation of the generic induction procedure $\mathcal{I}(Ax, \succcurlyeq)$ amounts to defining the following components:

- classes of axioms Ax and goals P,
- an induction ordering $\succcurlyeq$ on formula witnesses,
- a method of generating the cover sets w.r.t. $\succcurlyeq$, and
- simplification techniques justifiable by $\succcurlyeq$.

In this paper we present the instantiations of $\mathcal{I}(Ax, \succcurlyeq)$ for the case of equational clauses. The proof procedures $\mathcal{J}(Ax, \succcurlyeq_t)$ and $\mathcal{K}(Ax, \succcurlyeq_t)$, Section 6, cover and generalize, as far as we know, virtually all the existing hierarchical implicit induction procedures developed from the inductive completion framework. $\succcurlyeq_t$ is a term ordering used to generate the two induction orderings on clause witnesses for $\mathcal{J}$ and $\mathcal{K}$, Section 4. The methods of generating cover sets described in Section 3 exploit the common properties of the two orderings; so the cover set components of $\mathcal{J}$ and $\mathcal{K}$ are the same. The difference between the induction orderings results in the difference between the simplification techniques of $\mathcal{J}$ and $\mathcal{K}$ presented in Section 5.

3 Cover Sets

In this section we consider some practical refinements of the cover set notion. The notion itself is very abstract to permit practically useless cover sets. E.g., $\{\phi\}$ is always a cover set for ϕ. The reason for considering such an abstract notion is that it is sufficient for justifying the soundness of implicit induction procedures.

3.1 Cover Substitutions

A useful cover set of a goal should necessarily provide for the possibility of simplification of the cover instances when the goal itself cannot be simplified; no successful derivation can be generated, otherwise. In practice, the simplification of a goal amounts to the simplification of the terms occurring in it. That is why in [Red90], the notion of cover set was formulated on the level of terms and substitutions. We reproduce this notion below.

Given axioms Ax and substitutions σ, σ', we write $Ax \models \sigma = \sigma'$ if $Ax \models x\sigma = x\sigma'$ for any variable x. Given a quasi-order on terms $\succcurlyeq_t$ and substitutions σ, σ', we write $\sigma \succcurlyeq_t \sigma'$ if $x\sigma \succcurlyeq_t x\sigma'$ for any variable x.

Definition 3.1 (Cover substitutions) Let Ax be axioms, and $\succcurlyeq_t$ a quasi-order on terms. A *cover set of substitutions* $CS(Ax, \succcurlyeq_t)$ is a set of substitutions $\{\sigma_i\}_i$ such that, for any ground substitution γ, there exists a substitution σ_i and a ground substitution γ' such that $\gamma \succcurlyeq_t \sigma_i\gamma'$ and $Ax \models \gamma = \sigma_i\gamma'$.

The superposition substitutions of inductive completion and proof-by-consistency procedures such as [Fri86, Bac88, BK89, BL90] and the test substitutions of test set induction procedures [KR90, BKR95, BR95, Bou95] are the two representative instances of cover substitutions w.r.t. rewrite relations defined by axioms.

The relation between cover substitutions and cover witnesses can be formulated in terms of the ordering compatibility.

Definition 3.2 (Ordering compatibility) A quasi-order $\succcurlyeq$ on formula witnesses is *compatible* with a quasi-order on terms $\succcurlyeq_t$ if $\sigma \succcurlyeq_t \sigma'$ implies $\langle \phi, \sigma \rangle \succcurlyeq \langle \phi, \sigma' \rangle$ for any formula ϕ.

Proposition 3.3 (Cover substitutions and cover witnesses) *Let a quasi-order on formula witnesses $\succcurlyeq$ be compatible with a quasi-order on terms $\succcurlyeq_t$. Then, for any formula ϕ, $\{\langle \phi, \sigma \rangle\}_{\sigma \in CS(Ax, \succcurlyeq_t)}$ is a cover set of formula witnesses w.r.t. Ax, $\succcurlyeq$.*

3.2 Other Cover Sets

Other cover sets correspond to refinements of the induction ordering on formula witnesses. Consider, for example, the notion of max-extension ordering for clauses [BRH94].

Definition 3.4 (Max-extension [BRH94]) Let $\succcurlyeq_a$ be a quasi-order on atoms. The *max-extension* of $\succcurlyeq_a$ is the quasi-order on clauses $\succcurlyeq_a^{max}$ defined as follows. Let $max(C)$ denote the multiset of the atoms C maximal w.r.t. $\succcurlyeq_a$. Then $C \succcurlyeq_a^{max} C'$ if $max(C) \succcurlyeq_a^{mul} max(C')$, where $\succcurlyeq_a^{mul}$ is the multiset extension[4] of $\succcurlyeq_a$.

For a max-extension ordering, the following cover set [BRH94] is possible:

Proposition 3.5 (Max-cover set [BRH94]) *Let $\succcurlyeq_a$ be a stable quasi-order on atoms, and Ax be axioms. Let C be a clause, and A an atom such that $A \notin max(C \vee A)$ and $\neg A \notin max(C \vee \neg A)$. Then $\{C \vee A,\ C \vee \neg A\}$ is a cover set for C w.r.t. Ax, $\succcurlyeq_a^{max}$.*

[4] Cf. Definition 4.1.

However, in order to be useful for the cover set induction (cf. Proposition 2.4) the induction ordering $\succcurlyeq_a^{max}$ should be strongly stable.

Proposition 3.6 *$\succcurlyeq_a^{max}$ is strongly stable if $\succcurlyeq_a$ is strongly stable and $\doteqdot_a$ is stable.*

The stability of $\doteqdot_a$ in the above proposition is a quite restrictive condition. It holds, e.g., for any ordering of atoms based on the sort information of its terms, but such an ordering is quite useless by itself.

The effect of max-extension is to make some parts of formulas irrelevant to the formula comparison. This could be done explicitly by selecting the comparable parts of formulas, but such a selection would complicate the structure of proof goals. Although such a development, argued in [WB94], is reasonable in general, it is out of the scope of this paper.

It is still possible to modify the notion of max-extension to make it useful within the scope of induction on formulas.

Definition 3.7 (Combined extension) Let $\succcurlyeq_a$ be a quasi-order on atoms, and $\succcurlyeq_{ms}$ a quasi-order on multisets of atoms. The *combined extension* of $\succcurlyeq_a$, $\succcurlyeq_{ms}$ is the quasi-order on clauses $\succcurlyeq_{com}$ defined as follows. Let $max(C)$ denote the multiset of atoms in C maximal w.r.t. $\succcurlyeq_a$. Then $C \succcurlyeq_{com} C'$ if $max(C) \succcurlyeq_{ms} max(C')$.

The notion of combined extension is more general than that of max-extension, as it employs an arbitrary ordering $\succcurlyeq_{ms}$ instead of $\succcurlyeq_a^{mul}$. This makes the combined extension orderings more flexible.

Proposition 3.8 *$\succcurlyeq_{com}$ is strongly stable if $\succcurlyeq_a$, $\succcurlyeq_{ms}$ are strongly stable and $\doteqdot_a$ is stable.*

Proposition 3.5 obviously holds for the combined extensions as well, which results in another technique of generating cover sets. This technique can be lifted directly for the case of formula witnesses.

The two methods of generating cover sets described above can be combined together. However, we do not develop this line further. As for the existing implicit induction procedures, the notion of cover substitutions is sufficient for their justification. We proceed with this development in the following sections.

4 Induction Orderings

The ordering on formula witnesses $\succcurlyeq$ is a fundamental parameter of $\mathcal{I}(Ax, \succcurlyeq)$. Its refinements determine the requirements on the cover set generation and simplification techniques of concrete proof procedures. For the majority of the existing implicit induction procedures, such a refinement $\succcurlyeq_w$ is a function of a term ordering $\succcurlyeq_t$ compatible with $\succcurlyeq_t$, and the corresponding cover formula witnesses are defined in terms of cover substitutions (cf. Proposition 3.3). Therefore, different methods of the cover set generation used in the concrete induction procedures are usually quite interchangeable. For instance, the superpositional substitutions [Fri86, Bac88, BK89, BL90] and the test set substitutions [KR90, BR95, BKR95, Bou95] are interchangeable in the corresponding methods. What really makes difference in these procedures is their simplification techniques, and this difference is caused by the different induction orderings used for their justification.

In this section we present the two induction orderings for equational clauses, $\succcurlyeq_c$ in Section 4.2, and $\succcurlyeq_{cw}$ in section 4.3. These orderings form the basises of the two proof procedures presented in Section 6. $\succcurlyeq_c$ and $\succcurlyeq_{cw}$ are generated by orderings on terms. $\succcurlyeq_c$ is equivalent to an ordering on clauses, and, therefore, it fits the framework of the induction on propositional orderings [Red90, BRH94]; $\succcurlyeq_{cw}$ is essentially an ordering on clause witnesses.

4.1 Basic Notions

In the rest of the paper we refer to equational clauses as clauses. As usual, we consider equations as multisets of terms, and clauses as multisets of atoms, i.e. we abstract from the order of terms in equations, and atoms in clauses. The expression $(t = s)^{\varepsilon}$ denotes the atom $t = s$ if $\varepsilon = +$ and the atom $\neg(t = s)$ if $\varepsilon = -$.

The notion of multiset extension is extensively used in the definitions of the induction orderings presented in this section.

Definition 4.1 (Multiset extension [DM79]) Given a quasi-order $\succcurlyeq$, the multiset extension of $\succcurlyeq$, denoted $\succcurlyeq^{mul}$, is defined by:

$$\begin{aligned} X \sim^{mul} Y \quad &\text{if} \quad X \setminus_{\sim} Y = X \setminus_{\sim} Y = \emptyset, \\ X \succ^{mul} Y \quad &\text{if} \quad X \setminus_{\sim} Y \neq \emptyset \wedge \forall a \in (Y \setminus_{\sim} X) \exists b \in (X \setminus_{\sim} Y) b \succ a, \end{aligned}$$

where

$$\begin{aligned} X \cup \{\{a\}\} \setminus_{\sim} Y \cup \{\{b\}\} &= X \setminus_{\sim} Y \quad \text{if } a \sim b, \text{ and} \\ X \setminus_{\sim} Y &= X, \qquad \text{otherwise.} \end{aligned}$$

Proposition 4.2 ([DM79]) *For any quasi-order $\succcurlyeq$, $\succcurlyeq^{mul}$ is a quasi-order. If $\succcurlyeq$ is stable (strongly stable), so is $\succcurlyeq^{mul}$.*

A relation on terms $\gg$ is *monotonic* if $s \gg t$ implies $f(\ldots s \ldots) \gg f(\ldots t \ldots)$. A quasi-order on terms $\succcurlyeq$ is *strongly monotonic* if $\sim$ and $\succ$ are monotonic. A *reduction quasi-order* is a strongly stable and strongly monotonic well-founded quasi-order.

4.2 An Induction Ordering

The ordering on clause witnesses $\succcurlyeq_c$ defined below is essentially the ordering on equations [Red90] lifted to the level of clause witnesses.

Definition 4.3 (Induction ordering $\succcurlyeq_c$) Given a quasi-order on terms $\succcurlyeq_t$, the complexity measure μ on atoms is defined by

$$\mu((s = t)^{\varepsilon}) = \{\{s, t\}\}$$

The relation on atoms $\succcurlyeq_a$ is defined by

$$AT \succcurlyeq_a AT' \text{ iff } \mu(AT) \succcurlyeq_t^{mul} \mu(AT').$$

The relation on clause witnesses $\succcurlyeq_c$ is defined as follows. For any clauses $C \equiv \vee_i AT_i$, $C \equiv \vee_j AT'_j$ and substitutions σ, σ',

$$\langle C, \sigma \rangle \succcurlyeq_c \langle C', \sigma' \rangle$$

if $\{\{AT_i\sigma\}\}_i \succcurlyeq_a^{mul} \{\{AT'_i\sigma'\}\}_j$.

The properties of $\succcurlyeq_c$ essential for implicit induction are formulated in the following proposition.

Proposition 4.4 (Properties of $\succcurlyeq_c$) *If $\succcurlyeq_t$ is a reduction quasi-order then $\succcurlyeq_c$ is a strongly stable well-founded quasi-order compatible with $\succcurlyeq_t$.*

4.3 Another Induction Ordering

The quasi-order $\succcurlyeq_{cw}$ on clause witnesses defined below is a generalization of the proof orderings [Bac88, BKR95] which are essentially orderings on equation witnesses. It is also a simplifying generalization of the ordering on inconsistency witnesses [BL90] (cf. Section 5.3) which is a partial order on the witnesses of conditional equations.

Definition 4.5 (Induction ordering $\succcurlyeq_{cw}$) Given a quasi-order on terms $\succcurlyeq_t$, the complexity measure ν on atom witnesses is defined by

$$\begin{aligned} \nu(\langle s = t, \sigma\rangle) &= \begin{cases} \{\{s\sigma\}\} & \text{if } s \succ_t t, \\ \{\{t\sigma\}\} & \text{if } t \succ_t s, \\ \{\{s\sigma, t\sigma\}\}, & \text{otherwise} \end{cases} \\ \nu(\langle \neg s = t, \sigma\rangle) &= \{\{s\sigma, t\sigma\}\} \end{aligned}$$

The relation on atom witnesses $\succcurlyeq_{aw}$ is defined by

$$\langle AT, \sigma\rangle \succcurlyeq_{aw} \langle AT', \sigma'\rangle \text{ iff } \nu(\langle AT, \sigma\rangle) \succcurlyeq_t^{mul} \nu(\langle AT', \sigma'\rangle).$$

The relation on clause witnesses $\succcurlyeq_{cw}$ is defined as follows. For any clauses $C \equiv \vee_i AT_i$, $C \equiv \vee_j AT'_j$ and substitutions σ, σ',

$$\langle C, \sigma\rangle \succcurlyeq_{cw} \langle C', \sigma'\rangle$$

if $\{\{\langle AT_i, \sigma\rangle\}\}_i \succcurlyeq_{aw}^{mul} \{\{\langle AT'_i, \sigma'\rangle\}\}_j$.

Proposition 4.6 (Properties of $\succcurlyeq_{cw}$) *If $\succcurlyeq_t$ is a reduction quasi-order then $\succcurlyeq_{cw}$ is a strongly stable well-founded quasi-order compatible with $\succcurlyeq_t$.*

4.4 Other Induction Orderings

The induction orderings presented above allow us to represent a wide range of implicit induction procedures as instances of the generic induction procedure $\mathcal{I}$. However, these orderings certainly do not exhaust all the possibilities. E.g., the underlying induction ordering of the non-hierarchical induction procedures [HK88, Gra89] is based on comparing the left-hand sides of conjectures-equations w.r.t. an ordering on terms. Other orderings are possible, in general, as well. The induction orderings determine the corresponding simplification techniques (cf. Section 5). A new ordering may be necessary for justifying new simplification techniques that are adequate to a given proof problem.

5 Simplification Techniques

In this section we present the simplification techniques corresponding to the induction orderings $\succcurlyeq_c$ and $\succcurlyeq_{cw}$, Section 4. These simplification techniques are based on term rewriting, cases analysis and clause subsumption akin [BL90, BR95, Bou95]. The simplification techniques corresponding to $\succcurlyeq_c$ are presented in Sections 5.2, 5.4, 5.6; the simplification techniques corresponding to $\succcurlyeq_{cw}$ are presented in Sections 5.3, 5.5, 5.6.

5.1 Basic Notions

A *conditional equation* is a clause with a single positive atom. An expression $\wedge\{e_i\}_i \Rightarrow e$ denotes the clause $\vee_i \neg e_i \vee e$. Given a reduction quasi-order on terms $\succcurlyeq_t$, a *rewrite rule* $\wedge\{a_i = b_i\}_i \Rightarrow t \rightarrow s$ is the conditional equation $\wedge\{a_i = b_i\}_i \Rightarrow t = s$ such that $t \succ_t s$, and $t \succ_t a_i$, $t \succ_t b_i$ for

any i. We also say that $\wedge\{a_i = b_i\}_i \Rightarrow t \rightarrow s$ is oriented w.r.t. $\succ_t$. A *rewrite system* is a set of rewrite rules. Given a clause C, $prem(C)$ denotes the set of the equations in the negative atoms of C. Given a set of equations E, $\leftrightarrow_E$ stands for the least symmetric and monotonic relation including E. We write $t[u]_\omega \leftrightarrow^\omega_E t[v]_\omega$ if $u \leftrightarrow_E v$. Given a binary relation $\gg$, $\gg^*$ denotes its reflexive and transitive closure.

The contextual term rewriting relation [BL90] enhanced w.r.t. rewriting by non-orientable conditional equations akin [BKR95] forms the basis of the simplification techniques presented in this paper.

Definition 5.1 (Contextual rewriting) Let $\succcurlyeq_t$ be a reduction quasi-order, W a set of clauses. Given terms t, t' and a clause C, we write

$$t \rightarrow_{W,C} t'$$

if $t \succ_t t'$ and

either $t \leftrightarrow^*_{prem(C)} t'$

or there exist a conditional equation $\wedge\{a_i = b_i\}_i \Rightarrow a = b$ in W and a substitution θ and such that $t \leftrightarrow_{\{a\theta=b\theta\}} t'$ and, for any i,

1. $t \succ_t a_i\theta$ and $t \succ_t b_i\theta$,
2. $a_i\theta \rightarrow^*_{W,C} c_i \leftrightarrow^*_{prem(C)} d_i \leftarrow^*_{W,C} b_i\theta$ for some c_i, d_i.

Since the rewrite relation above is meant to form the simplification techniques for clausal proof procedures, it is convenient to consider W as clauses, although only conditional equations of W take part in the rewriting.

Akin [BR95], we make the straightforward generalization of $\rightarrow$ to permit the term replacements not compatible with the underlying term ordering.

Definition 5.2 (Relaxed contextual rewriting) Let $\succcurlyeq_t$ be a reduction quasi-order, W a set of clauses. Given terms t, t', a clause C, we write

$$t \looparrowright_{W,C} t'$$

if

either $t \leftrightarrow^*_{prem(C)} t'$

or there exist a conditional equation $CE \equiv \wedge\{a_i = b_i\}_i \Rightarrow a = b$ in W and substitution θ such that $t \leftrightarrow^\omega_{\{a\theta=b\theta\}} t'$ and, for any i,

1. $t \succ_t a_i\theta$ and $t \succ_t b_i\theta$,
2. $a_i\theta \rightarrow^*_{W,C} c_i \leftrightarrow^*_{prem(C)} d_i \leftarrow^*_{W,C} b_i\theta$ for some c_i, d_i.

Note that the definition of $\looparrowright$ is not recursive, it rather directly reduces to the definition of $\rightarrow$. We write $t \looparrowright^{CE,\omega,\theta}_{W,C} t'$ to distinguish the matching equation CE, substitution θ and position ω in the definition of $\looparrowright$.

Some of the presented simplification techniques are based on clause subsumption. A clause C is *subsumed* by a clause C' if there exists a substitution σ such that, for any atom $a = b$ in C', there exists a term t such that $t[a\sigma] = t[b\sigma] \in C$ and, for any $\neg a = b \in C'$, $\neg a\sigma = b\sigma \in C$.

We write $s \triangleright t$ if t is a strict subterm of s.

The notion of *decreasing* quasi-order below is important for justifying the presented simplification techniques. It is a straightforward generalization of the notion of *decreasing (partial) order* [Red90].

Definition 5.3 (Decreasing quasi-order) A *decreasing* quasi-order $\succcurlyeq_t$ is a reduction quasi-order such that $s \triangleright t$ implies $s \succ_t t$.

5.2 A Simplification by Rewriting

A simplification by rewriting amounts to rewriting terms in proof goals so that the complexity of a proof goal w.r.t. a given induction ordering decreases. In this section we define such rewriting techniques corresponding to $\succcurlyeq_c$.

Definition 5.4 (Rewrite relations $\rightsquigarrow$, $\mapsto$) Let $\succcurlyeq_t$ be a reduction quasi-order, W and V sets of clauses. Given clauses C and C_1, let

1. $C \equiv (t = s)^{\epsilon} \vee C'$, $C_1 \equiv (t' = s)^{\epsilon} \vee C'$,
2. $t \overset{CE,\omega,\theta}{\underset{W \cup V, C'}{\hookrightarrow}} t'$ w.r.t. $\succcurlyeq_t$, $CE \equiv E \Rightarrow e \in W \cup V$.

We write $C \rightsquigarrow_{W[V]} C_1$ if $t \succcurlyeq_t t'$ and

either $CE \in W$ (a)
or $CE \in V$ and
either $\omega \neq \epsilon$ (b)
or $\omega = \epsilon$ and
either $s \succ_t t'$ (c)
or $s \sim_t t'$ and $\neg E\theta \preccurlyeq_c C'$ (d)

Let $\rightsquigarrow_{\rightarrow W[V]}$ denote the relation defined exactly as $\rightsquigarrow_{W[V]}$ except that, in condition (d), $\neg E\theta \preccurlyeq_c C'$ is strengthen to $\neg E\theta \prec_c C'$. (d')
We write $C \mapsto_{W[V]} C_1$ if $C \rightsquigarrow_{\rightarrow W[V]} C_1$ and $t \succ_t t'$.

Note that the $\succcurlyeq_c$ part of condition (d) is obviously satisfied when $E = \emptyset$ for any $\succcurlyeq_c$; the $\succ_c$ part of condition (d') is satisfied when $E = \emptyset$ and $C' \neq \emptyset$. It is easy to see that $\mapsto \subset \rightsquigarrow$. The only difference between the parameters W and V of $\rightsquigarrow_{W[V]}$ and $\mapsto_{W[V]}$ lies in the restrictions on using the conditional equations in V for the rewriting at top positions.

The usage of $\rightsquigarrow$ and $\mapsto$ as simplification relations in the scope of the generic induction procedure is justified by the following proposition:

Proposition 5.5 (Properties of $\rightsquigarrow$, $\mapsto$) *Let $\succcurlyeq_t$ be a decreasing quasi-order. Then*

1. $C \rightsquigarrow_{W[V]} C_1$ *implies* $W \models (V \cup \{C_1\})_{\preccurlyeq_c C} \Rightarrow C$,
2. $C \mapsto_{W[V]} C_1$ *implies* $W \models (V \cup \{C_1\})_{\prec_c C} \Rightarrow C$.

The strict subterm property of $\succcurlyeq_t$ in the proposition above is essential for the justification of the presented simplification techniques. One may note that the existing implicit induction techniques require just reduction quasi-orders rather than decreasing quasi-orders, and, therefore, the additional restriction reduces the generality of the presented techniques. However, the closer analysis of the existing simplification techniques reveals that the subterm property always appears as an implicit requirement. The point is that the justification of the existing techniques depends just on the strict part of the underlying reduction quasi-orders. For any reduction quasi-order $\succcurlyeq_t$, there exists a decreasing quasi-order $\succcurlyeq_{t\triangleright}$ such that $\succ_{t\triangleright}$ is an extension of $\succ_{t\triangleright}$, i.e. $\succ_t \subseteq \succ_{t\triangleright}$. It is $\succ_{t\triangleright}$ that is usually implicitly used for the justification of the simplification techniques parameterized by a reduction quasi-order $\succcurlyeq_t$. However, not any reduction quasi-order $\succcurlyeq_t$ can be extended to a decreasing quasi-order $\succcurlyeq_{t\triangleright}$ so that $\succ_t \subseteq \succ_{t\triangleright}$ *and* $\sim_t \subseteq \sim_{t\triangleright}$, because

$\triangleright \cap \sim_t$ may be non-empty. Since the presented simplification techniques essentially depend on the equivalence parts of the underlying quasi-orders as well, we have to make the strict subterm requirement explicit.

According to Proposition 5.5, an instantiation $\mathcal{I}(Ax, \succcurlyeq_c)$ can contain the following inference rules:

Generate

$$\frac{(P \cup \{C\}, H)}{(P \cup \{C^\sigma \mid \sigma \in CS(Ax, \succcurlyeq_t)\}, H \cup \{C\})}$$

if for every $\sigma \in CS(Ax, \succcurlyeq_t)$, $C\sigma \mapsto_{Ax[P \cup H \cup \{C\}]} C^\sigma$

Simplify

$$\frac{(P \cup \{C\}, H)}{(P \cup \{C_1\}, H)}$$

if $C \rightsquigarrow_{Ax[P \cup H]} C_1$

The most close analogue of the simplification relations above is the simplification by *inductive rewriting* ([BR95], section 6.1). It corresponds to the cases (a) and (b) of the definition of $\mapsto$. I.e., the *Generate* and *Simplify* rules of [BR95] correspond to the *Generate* and *Simplify* rules above with the properly restricted simplification relation.[5] The rewriting at top positions by the components of proof states is not permitted in [BR95] due to the choice of the underlying induction ordering. The simplification corresponding to $\rightsquigarrow \setminus \mapsto$ is not used in [BR95] because it would corrupt the refutational properties of [BR95].

5.3 Another Simplification by Rewriting

In this section we define a simplification by rewriting compatible with $\succcurlyeq_{cw}$.

Definition 5.6 (Rewrite relations $\rightsquigarrow$, $\twoheadrightarrow$) Let $\succcurlyeq_t$ be a reduction quasi-order, W and V sets of clauses. Given a clause witness $\langle C, \sigma \rangle$ and a clause C_1, let

1. $C \equiv (t = s)^\varepsilon \vee C'$, $C_1 \equiv (t' = s\sigma)^\varepsilon \vee C'\sigma$,
2. $t\sigma \twoheadrightarrow^{CE,\omega,\theta}_{W \cup V, C'} t'$ w.r.t. $\succcurlyeq_t$, $CE \equiv E \Rightarrow e \in W \cup V$.

We write $\langle C, \sigma \rangle \rightsquigarrow_{W[V]} C_1$ if

either $t\sigma \succcurlyeq_t t'$ and
  either $CE \in W$ (a)
  or $CE \in V$ and
    either $\omega \neq \epsilon$ (b)
    or $\omega = \epsilon$, CE is oriented w.r.t. $\succ_t$ and
      either $\varepsilon = -$ (c)
      or $\varepsilon = +$ and
        either $t \not\succ_t s$ (d)
        or $t \succ_t s$ and $\langle \neg E, \theta \rangle \preccurlyeq_{cw} \langle C', \sigma \rangle$ (e)
or $t\sigma \not\succcurlyeq_t t'$, $\varepsilon = +$ and $s\sigma \succ_t t'$ (f)

[5] We find the optimized simplification by relaxed contextual rewriting ([BR95], Section 8) incorrect. We give a counterexample in Appendix B.

Let $\rightsquigarrow_{\rightarrow W[V]}$ denote the relation defined exactly as $\rightsquigarrow_{W[V]}$ except that, in conditions (a)–(e), $t\sigma \succcurlyeq_t t'$ is strengthen to $t\sigma \succ_t t'$, and, in condition (e), $\langle \neg E, \theta\rangle \preccurlyeq_{cw} \langle C', \sigma\rangle$ is strengthen to $\langle \neg E, \theta\rangle \prec_{cw} \langle C', \sigma\rangle$. (e')
We write $\langle C, \sigma\rangle \twoheadrightarrow_{W[V]} C_1$ if $\langle C, \sigma\rangle \rightsquigarrow_{\rightarrow W[V]} C_1$ and $s \not\succcurlyeq_t t$.

The remarks about $\rightsquigarrow_{W[V]}$ and $\mapsto_{W[V]}$ made after definition 5.4 are relevant for the relations $\rightsquigarrow_{W[V]}$ and $\twoheadrightarrow_{W[V]}$ as well.

Now we compare the relations $\rightsquigarrow_{W[V]}$, $\rightsquigarrow_{W[V]}$ and $\mapsto_{W[V]}$, $\twoheadrightarrow_{W[V]}$ pairwise. The relations $\rightsquigarrow$ and $\rightsquigarrow$ differ, *firstly*, w.r.t. the rewriting by V at top positions. As one of the conditions (d), (e) in the definition of $\rightsquigarrow$ is often satisfied, while the conditions (c) and (d) in the definition of $\rightsquigarrow$ are not, $\rightsquigarrow$ is more general than $\rightsquigarrow$. *Secondly*, $\rightsquigarrow$ permits replacing t with t' so that $t \not\succcurlyeq_t t'$, while $\rightsquigarrow$ do not. Thus, $\rightsquigarrow$ permits using non-orientable conditional equations for the simplification, and $\rightsquigarrow$ is more general than $\rightsquigarrow$ in this respect, too.

The remarks about $\rightsquigarrow$ and $\rightsquigarrow$ above are relevant to the comparison of $\twoheadrightarrow$ and $\mapsto$ as well. However, $\twoheadrightarrow$ is more restrictive than $\mapsto$ as far as it does not permit simplifying t when $s \succ_t t$.

The usage of $\rightsquigarrow$ and $\twoheadrightarrow$ as simplification relations in scope of the generic induction procedure is justified by the following proposition:

Proposition 5.7 (Properties of $\rightsquigarrow$, $\twoheadrightarrow$) *Let $\succcurlyeq_t$ be a decreasing quasi-order. Then*

1. $\langle C, \sigma\rangle \rightsquigarrow_{W[V]} C_1$ *implies* $W \models (V \cup \{C_1\})_{\preccurlyeq_{cw}\langle C,\sigma\rangle} \Rightarrow C\sigma$,
2. $\langle C, \sigma\rangle \twoheadrightarrow_{W[V]} C_1$ *implies* $W \models (V \cup \{C_1\})_{\prec_{cw}\langle C,\sigma\rangle} \Rightarrow C\sigma$.

According to the proposition above, an instantiation $\mathcal{I}(Ax, \succcurlyeq_{cw})$ can contain the following inference rules:

Generate

$$\frac{(P \cup \{C\}, H)}{(P \cup \{C^\sigma \mid \sigma \in CS(Ax, \succcurlyeq_t)\}, H \cup \{C\})}$$

if for every $\sigma \in CS(Ax, \succcurlyeq_t)$, $\langle C, \sigma\rangle \twoheadrightarrow_{Ax[P\cup H\cup\{C\}]} C^\sigma$

Simplify

$$\frac{(P \cup \{C\}, H)}{(P \cup \{C_1\}, H)}$$

if $C \rightsquigarrow_{Ax[P\cup H]} C_1$

The most close subset of the simplification relation $\rightsquigarrow$ for the case of unconditional equations is the simplification relation for unconditional equations defined by the *Simplify* rules [BKR95]. It is equivalent to $\twoheadrightarrow$ enforced with the simplification by subsumption (cf. Section 5.6). The ordering on the witnesses of equations used in [BKR95] is just $\succ_{aw}$ on positive atoms. The simplification corresponding to $\rightsquigarrow \setminus \twoheadrightarrow$ is not considered in [BKR95] for the refutational considerations.

The simplification by rewriting in [BL90], rule *Simplification*, is a close subset of the restriction of $\rightsquigarrow$ for the case of conditional equations, where $t\sigma \succ_t t'$. The simplification by orientable instances of non-orientable conditional equations as well as by non-orientable conditional equations itself was not considered there.[6]

The ordering on the witnesses of conditional equations underlying the simplification relation in [BL90] is determined by the following complexity:

$$c(\langle E \Rightarrow s = t, \sigma\rangle) = \begin{cases} (\{\{s\sigma\}\}, \{\{E\sigma\}\}, \{\{s\}\}, t\sigma) & \text{if } E \Rightarrow s \rightarrow t \text{ w.r.t. } \succ_t, \\ (\{\{t\sigma\}\}, \{\{E\sigma\}\}, \{\{t\}\}, s\sigma) & \text{if } E \Rightarrow s \rightarrow t \text{ w.r.t. } \succ_t, \\ (\{\{s\sigma, t\sigma\} \cup \{\{E\sigma\}\}, \{\{E\sigma\}\}, \{\{s, t\}\}, -), & \text{otherwise} \end{cases}$$

[6] We consider the subsumption by non-orientable conditional equations [BL90] in Section 5.7.

The partial order $>_{\mathrm{GP}}$ [BL90] corresponds to the comparison of the complexities above by the lexicographic combination of twice $\succcurlyeq_t^{mul}$, the multiset ordering induced by the *specialization ordering* [Bac88], and $\succcurlyeq_t$.

Orderings $>_{\mathrm{GP}}$ and $\succcurlyeq_{cw}$ are incomparable, in general. The simplification by rewriting in [BL90] can be generalized to become incomparable with $\rightsquigarrow$. Actually, the subsumption by conditional equation in [BL90] cannot be fully justified by $\succcurlyeq_{cw}$ already (cf. Section 5.7).

5.4 A Simplification by Cases

A simplification by cases amounts to some case analysis of a goal followed by simplification of the cases by rewriting. The simplification by cases defined below corresponds to the simplification relation $\mapsto$.

Definition 5.8 (Simplification by cases) Let $\succcurlyeq_t$ be a reduction quasi-order, W and V sets of clauses. Let $\overset{d}{\mapsto}$ stand for the relation defined exactly as $\mapsto$ except for omitting the case (d'). Given a clause C and a set of clauses P, we write

$$C \mapsto\!\circ_{W[V]} P$$

if

1. $CE_i \equiv E_i \Rightarrow e_i \in W \cup V$, $E_i \neq \emptyset$,
2. $\neg E_i\theta_i \vee C \overset{d}{\mapsto}_{W[V]}^{CE_i,\theta_i} \neg E_i\theta_i \vee C_1^i$,
3. $P \equiv P_1 \cup P_2$, where $P_1 \equiv \{E_i\theta_i \Rightarrow C_1^i\}_i$ and $P_2 \equiv \mathit{CNF}(\vee_i E_i\theta_i)$,

where $\mathit{CNF}(\phi)$ denotes the set of clauses corresponding to a conjunctive normal form of ϕ.

The usage of $\mapsto\!\circ$ as a simplification relation in scope of the generic induction procedure is justified by the following proposition:

Proposition 5.9 (Properties of $\mapsto\!\circ$) *Let $\succcurlyeq_t$ be a decreasing quasi-order. Then $C \mapsto\!\circ_{W[V]} P$ implies $W \models (V \cup P)_{\prec_c C} \Rightarrow C$.*

Note that it is not possible to define a case analysis relation $\rightsquigarrow\!\circ$ in the same way as $\mapsto\!\circ$, so that $C \rightsquigarrow\!\circ_{W[V]} P$ would imply $W \models (V \cup P)_{\preccurlyeq_c C} \Rightarrow C$, but not $W \models (V \cup P)_{\prec_c C} \Rightarrow C$. The reason is that, for any $C' \in P_1$ in such a definition, $C \setminus C'$ should be a singleton, while $C' \setminus C$ should not, and, therefore, $C \succcurlyeq_c C'$ should imply $C \succ_c C'$. Also, as $\neg E_i\theta_i \not\subset C$, $C \succcurlyeq_c CE_i\theta_i$ should imply $C \succ_c CE_i\theta_i$ for any $CE_i \in V$.

According to the proposition above, an instantiation $\mathcal{I}(Ax, \succcurlyeq_c)$ can contain the following inference rules:

Generate

$$\frac{(P \cup \{C\}, H)}{(P \cup \cup_\sigma P^\sigma, H \cup \{C\})}$$

if, for every $\sigma \in CS(Ax, \succcurlyeq_t)$, $C\sigma \mapsto\!\circ_{Ax[P \cup H \cup \{C\}]} P^\sigma$

Simplify

$$\frac{(P \cup \{C\}, H)}{(P \cup P', H)}$$

if $C \mapsto\!\circ_{Ax[P \cup H]} P'$

The *case rewriting* [BR95] correspond to the case (a) of the definition of $\mapsto$, as it occurs in $\mapsto\!\circ$. The case rewriting by the components of proof states are not considered in [BR95], although it is justifiable by the underlying induction ordering. We have an example [ND96], where the usage of an induction hypothesis in the case analysis is essential.

5.5 Another Simplification by Cases

The simplification by cases defined here corresponds to the simplification relation $\twoheadrightarrow$.

Definition 5.10 (Another simplification by cases) Let $\succcurlyeq_t$ be a reduction quasi-order, W and V sets of clauses. Let $\overset{e}{\twoheadrightarrow}$ stand for the relation defined exactly as $\twoheadrightarrow$ except for omitting the case (e'). Given a clause witness $\langle C, \sigma \rangle$ and a set of clauses P, we write

$$\langle C, \sigma \rangle \twoheadrightarrow\!\circ_{W[V]} P$$

if

1. $CE_i \equiv E_i \Rightarrow e_i \in W \cup V$, $E_i \neq \emptyset$,
2. $\langle E_i\theta_i \Rightarrow C, \sigma \rangle \overset{e}{\twoheadrightarrow}_{W[V]}^{CE_i,\theta_i} E_i\theta_i \Rightarrow C_1^i$,
3. $P \equiv P_1 \cup P_2$, where $P_1 \equiv \{E_i\theta_i \Rightarrow C_1^i\}_i$ and $P_2 \equiv CNF(\vee_i E_i\theta_i)$.

The usage of $\twoheadrightarrow\!\circ$ as a simplification relation in scope of the generic induction procedure is justified by the following proposition:

Proposition 5.11 (Properties of $\twoheadrightarrow\!\circ$) *Let $\succcurlyeq_t$ be a decreasing quasi-order. Then $\langle C, \sigma \rangle \twoheadrightarrow\!\circ_{W[V]} P$ implies $W \models (V \cup P)_{\prec_{cw}\langle C,\sigma\rangle} \Rightarrow C\sigma$.*

The arguments about a possible generalization of $\twoheadrightarrow\!\circ$ to $\rightsquigarrow\!\circ$ are the same as for $\mapsto\!\circ$ in Section 5.4.

According to the proposition above, an instantiation $\mathcal{I}(Ax, \succcurlyeq_{cw})$ can contain the following inference rules:

Generate

$$\frac{(P \cup \{C\}, H)}{(P \cup \cup_\sigma P^\sigma, H \cup \{C\})}$$

if, for every $\sigma \in CS(Ax, \succcurlyeq_t)$, $\langle C, \sigma \rangle \twoheadrightarrow\!\circ_{Ax[P \cup H \cup \{C\}]} P^\sigma$

Simplify

$$\frac{(P \cup \{C\}, H)}{(P \cup P', H)}$$

if $C \twoheadrightarrow\!\circ_{Ax[P \cup H]} P'$

The generation of *contextual critical pairs* (CCP) and *superpositional instances* (SI) [BL90] by conditional theories is an instance of the rule *Generate* above. I.e., *firstly*, given a goal C, the substitutions $\{\sigma_i\}_i$ corresponding to CCP and SI on an *inductively reducible set of positions* in C is a cover set of substitutions for C. *Secondly,* $CCP \cup SI$ corresponds to $\cup_i P_{1i}$ when $\langle C, \sigma_i \rangle \twoheadrightarrow\!\circ_{Ax[\emptyset]} P_{1i} \cup P_{2i}$ (cf. Definition 5.10). The P_{2i} components are implicit in [BL90], because they are inductive consequences of Ax by the properties of the inductively reducible sets of positions.

5.6 Simplification by Subsumption

The simplification by subsumption is based on the subsumption of a clause by another clause.

Definition 5.12 (Inductive subsumption) Let W and V be sets of clauses. Given a clause C, let C' be a clause in $W \cup V$ such that C is subsumed by C' with a matching substitution θ. Then we write $C \sqsupseteq_{W \cup V}$. We write $C \sqsupset_{W[V]}$ if either $C' \in W$ or $C' \in V$ and $C'\theta \neq C$.

Proposition 5.13 (Properties of $\supseteq$, $\supset$) *Let $\succcurlyeq_t$ be a decreasing quasi-order. Then*

1. *$C \supseteq_{W \cup V}$ implies $W \models V_{\preccurlyeq_c C} \Rightarrow C$ and $W \models V_{\preccurlyeq_{cw} C} \Rightarrow C$.*
2. *$C \supset_{W[V]}$ implies $W \models V_{\prec_c C} \Rightarrow C$ and $W \models V_{\prec_{cw} C} \Rightarrow C$.*

According to the proposition above, instantiations of $\mathcal{I}(Ax, \succcurlyeq_c)$ and $\mathcal{I}(Ax, \succcurlyeq_{cw})$ can contain the following inference rules:

Generate

$$\frac{(P \cup \{C\}, H)}{(P, H \cup \{C\})}$$

if, for every $\sigma \in CS(Ax, \succcurlyeq_t)$, $C\sigma \supset_{Ax[P \cup H \cup \{C\}]}$

Simplify

$$\frac{(P \cup \{C\}, H)}{(P, H)}$$

if $C \supseteq_{Ax \cup P \cup H}$

The simplification by subsumption is employed in many implicit induction procedures like [Gra89, BL90, BR95]. The following generalization rule is a useful combination of the simplification by subsumption with lemma generation:

Generalize by subsumption

$$\frac{(P \cup \{C\}, H)}{(P \cup \{C_1\}, H)}$$

if $C \supseteq_{\{C_1\}}$

The generalization by subsumption is a particular kind of lemma generation, where the form of the lemma is directly prompted by the form of the generalized conjecture. Other kinds of generalization in the implicit induction framework, e.g. [Gra89], can be covered by combining *Lemma* inference rule (cf. Appendix A, proof of Proposition 2.6) with the simplification by subsumption.

5.7 Other Simplifications

The simplification techniques presented so far form the basic simplification techniques of the existing implicit induction procedures. Other simplification techniques can be covered within our generic framework as well. To justify this claim w.r.t. the discussed concrete induction procedures, we still need to consider the *simplification of constructors* and *complimenting simplification* techniques employed in [BR95] and the *subsumption by non-orientable conditional equations* [BL90].

Definition 5.14

(Right simplify of constructors) Let $C \equiv f(\bar{t}) = f(\bar{s}) \vee C'$ and $\bar{t} = (t_i)_i$, $\bar{s} = (s_i)_i$.
Then we write $C \circlearrowleft \{s_i = t_i \vee C'\}_i$.

(Left simplify of constructors) Let $C \equiv \neg(f(\bar{t}) = f(\bar{s})) \vee C'$ and $\bar{t} = (t_i)_i$, $\bar{s} = (s_i)_i$. Let also $f(\ldots x_i \ldots) = f(\ldots y_i \ldots) \Rightarrow x_i = y_i \in W$ for any i, where x_i, y_i are variables.
Then we write $C \curvearrowright_W \vee_i \neg(s_i = t_i) \vee C'$.

(Complement) Let $C \equiv \neg(l\sigma = s) \vee C'$, $C_1 \equiv r\sigma = s \vee C'$, $\neg(l = r) \in W$, and $l\sigma \succcurlyeq_t r\sigma$. Then we write $C \not\curvearrowright_W C_1$.

The simplification relations defined above are the generalizations of the corresponding relations in [BR95] resulted from ignoring their refutational properties. The *right* and *left* simplifications are particular forms of generalization. The complementing simplification [BR95] was introduced mainly for the refutational considerations. We reproduce it here just to complete the consideration of [BR95].

Proposition 5.15 (Properties of $\circlearrowleft$, $\curvearrowright$, $\not\curvearrowright$) *Let $\succcurlyeq_t$ be a decreasing quasi-order. Then*

1. *$C \circlearrowleft P$ implies $\models P_{\prec_c C} \Rightarrow C$ and $\models P_{\prec_{cw} C} \Rightarrow C$.*
2. *$C \curvearrowright_W C_1$ implies $W \models \{C_1\}_{\prec_c C} \Rightarrow C$ and $\models \{C_1\}_{\prec_{cw} C} \Rightarrow C$*
3. *$C \not\curvearrowright_W C_1$ implies $W \models \{C_1\}_{\preccurlyeq_c C} \Rightarrow C$ and $W \models \{C_1\}_{\preccurlyeq_{cw} C} \Rightarrow C$.*

Now we consider the subsumption by non-orientable conditional equations [BL90]. The proof state in [BL90] is, essentially, a set of conditional equations (cf. section 2.4); in our setting, the *Subsumption* inference rule [BL90] is as follows. Let $\overrightarrow{W}$ stand for the set of the conditional rewrite rules in a set of clauses W.

Subsumption by an equation

$$\frac{(P \cup \{E \Rightarrow t = s\}, H)}{(P, H)}$$

if there exists a non-orientable conditional equation $CE_1 \equiv \wedge\{a_i = b_i\}_i \Rightarrow a = b \in (Ax \cup P \cup H)$ such that

1. $t \leftrightarrow_{\{a\theta = b\theta\}} s$,
2. for every i, $a_i\theta \rightarrow^*_{\overrightarrow{W \cup V}, \neg E} c_i \leftrightarrow^*_E d_i \leftarrow^*_{\overrightarrow{W \cup V}, \neg E} b_i\theta$ for some c_i, d_i, and
3. $CE_1\theta <_{\mathrm{GP}} E \Rightarrow t = s$ if $CE_1 \in (P \cup H)$,

As $>_{\mathrm{GP}}$ and $\succcurlyeq_{cw}$ are incomparable, the subsumption above cannot be justified by $\succcurlyeq_{cw}$. However, it can be justified in the scope of the presented framework directly by $>_{\mathrm{GP}}$.

We could also introduce an analogy of the above relation for $\succcurlyeq_{cw}$ which would not be justifiable by $>_{\mathrm{GP}}$. Below, we define such a very simple analogy without mentioning $\succcurlyeq_{cw}$ explicitly.

Definition 5.16 (Subsumption by conditional equation) Let $\succcurlyeq_t$ be a reduction quasi-order, W and V sets of clauses. Given a clause $C \equiv t = s \vee C'$ and conditional equation $CE \equiv E \Rightarrow e \in W \cup V$, let $t \twoheadrightarrow^{CE,\omega,\theta}_{W \cup V, C'} s$ w.r.t. $\succcurlyeq_t$. We write $C \sqsupseteq_{W[V]}$ if

either $CE \in W$
or $CE \in V$ and
either $\omega \neq \epsilon$
or $\omega = \epsilon$ and $\langle \neg E, \theta \rangle \preccurlyeq_{cw} C'$

Proposition 5.17 (Properties of $\sqsupseteq$) *Let $\succcurlyeq_t$ be a decreasing quasi-order. Then $C \sqsupseteq_{W[V]}$ implies $W \models V_{\preccurlyeq_{cw} C} \Rightarrow C$.*

Generate

$$\frac{(P \cup \{C\}, H)}{(P \cup (\cup_\sigma P^\sigma, H \cup \{C\})}$$

if, for every $\sigma \in CS(Ax, \succcurlyeq_t)$,

either	$C\sigma$ is a tautology	and $P^\sigma = \emptyset$,
or	$C\sigma \supset_{Ax[P \cup H \cup \{C\}]}$	and $P^\sigma = \emptyset$,
or	$C\sigma \mapsto_{Ax[P \cup H \cup \{C\}]} C_1$	and $P^\sigma = \{C_1\}$,
or	$C\sigma \mapsto^{\circ}_{Ax[P \cup H \cup \{C\}]} P^\sigma$	

Simplify

$$\frac{(P \cup \{C\}, H)}{(P \cup P', H)}$$

if either	C is a tautology	and $P' = \emptyset$,
or	$C \supseteq_{Ax \cup P \cup H}$	and $P' = \emptyset$,
or	$C \rightsquigarrow_{Ax[P \cup H]} C_1$	and $P' = \{C_1\}$,
or	$C \mapsto^{\circ}_{Ax[P \cup H]} P'$	

Generalize

$$\frac{(P \cup \{C\}, H)}{(P \cup \{C_1\}, H)} \quad \text{if } C \supseteq_{\{C_1\}}$$

Lemma

$$\frac{(P, H)}{(P \cup \{C\}, H)}$$

Figure 3: Inference system $\mathcal{J}(Ax, \succcurlyeq_t)$

6 Two Induction Procedures for Equational Clauses

In this section we combine the simplification techniques corresponding to the orderings $\succcurlyeq_c$ and $\succcurlyeq_{cw}$ to get two induction procedures as direct instantiations of the generic procedure $\mathcal{I}$. The inference systems $\mathcal{J}(Ax, \succcurlyeq_t)$ and $\mathcal{K}(Ax, \succcurlyeq_t)$ transforming pairs of sets of clauses are presented at Figures 3 and 4 resp.

Proposition 6.1 (Correctness of $\mathcal{J}(Ax, \succcurlyeq_t)$ and $\mathcal{K}(Ax, \succcurlyeq_t)$) *Let Ax be a set of axioms-clauses and $\succcurlyeq_t$ a decreasing quasi-order. Given a set of clauses P, $Ax \models_{ind} P$ if there exists a successful derivation for P by either $\mathcal{J}(Ax, \succcurlyeq_t)$ or $\mathcal{K}(Ax, \succcurlyeq_t)$.*

The relations of $\mathcal{J}$ and $\mathcal{K}$[7] to the existing proof procedures, established in the course of the direct comparison, are summarized in Figure 1. As we can see, the procedures $\mathcal{J}$ and $\mathcal{K}$ cover and generalize a representative range of the existing implicit induction procedures. The procedure [BL90], i.e. its subsumption technique, cannot be completely covered by $\mathcal{K}$ due to the difference of the underlying induction orderings. However, it can be considered and generalized as a direct instantiation of $\mathcal{I}$ corresponding to $>_{\mathrm{GP}}$.

[7] When completed with the auxiliary simplification techniques, Section 5.7.

Generate

$$\frac{(P \cup \{C\}, H)}{(P \cup (\cup_\sigma P^\sigma, H \cup \{C\})}$$

if, for every $\sigma \in CS(Ax, \succcurlyeq_t)$,

either	$C\sigma$ is a tautology	and $P^\sigma = \emptyset$,
or	$C\sigma \supset_{Ax[P \cup H \cup \{C\}]}$	and $P^\sigma = \emptyset$,
or	$\langle C, \sigma \rangle \twoheadrightarrow_{Ax[P \cup H \cup \{C\}]} C_1$	and $P^\sigma = \{C_1\}$,
or	$\langle C, \sigma \rangle \twoheadrightarrow^{\circ}_{Ax[P \cup H \cup \{C\}]} P^\sigma$	

Simplify

$$\frac{(P \cup \{C\}, H)}{(P \cup P', H)}$$

either	C is a tautology	and $P' = \emptyset$,
or	$C \supseteq_{Ax \cup P \cup H}$	and $P' = \emptyset$,
or	$C \rightsquigarrow_{Ax[P \cup H]} C_1$	and $P' = \{C_1\}$,
or	$C \twoheadrightarrow^{\circ}_{Ax[P \cup H]} P'$	

Generalize

$$\frac{(P \cup \{C\}, H)}{(P \cup \{C_1\}, H)} \quad \text{if } C \supseteq_{\{C_1\}}$$

Lemma

$$\frac{(P, H)}{(P \cup \{C\}, H)}$$

Figure 4: Inference system $\mathcal{K}(Ax, \succcurlyeq_t)$

7 Conclusion

In this paper we further developed the approach of [Red90, BRH94] towards the cover set induction representation of implicit induction procedures. We presented a generic implicit induction procedure which covers a representative class of the existing implicit induction procedures. This generic representation allows for easy modification and generalization of the existing procedures, and greatly simplifies their justification. It is also directly interpreted in the explicit cover set induction framework.

The discussed induction procedures have many common features, and the differences between them are often rather technical. To demonstrate their specific features, examples are certainly desirable. This is left for a next version. The emphasis in the current version is to demonstrate the power of the presented generic framework that allows for

- easy justification of many implicit induction techniques, and
- their direct interpretation in the explicit induction framework.

The phenomenon of implicit induction as induction on the proof meta-level cannot be defined formally. Therefore, strictly speaking, no one can claim that a certain formal framework covers all possible implicit induction procedures. Currently we are aware of the two implicit induction procedures [Pad92, Pro94] based on the induction schemes different from the cover set induction. Still the cover set induction captures the major part of the existing implicit induction procedures.

References

[Aub79] Raymond Aubin. Mechanizing structural induction: Parts (i) and (ii). *Theoretical Computer Science*, 9:329–362, 1979.

[Bac88] Leo Bachmair. Proof by consistency in equational theories. In *3rd IEEE symposium on Logic in Computer science*, pages 228–233, 1988.

[Bev91] Eddy Bevers. An abstract framework for proof by consistency. Unpublished manuscript, 1991.

[BK89] Reinhard Bündgen and Wolfgang Küchlin. Computing ground reducibility and inductively complete positions. In *3rd International Conference on Rewriting Techniques and Applications*, volume 355 of *LNCS*, pages 59–75, 1989.

[BKR95] Adel Bouhoula, Emmanuel Kounalis, and Michaël Rusinowitch. Automated mathematical induction. *Journal of Logic and Computation*, 5(5):631–668, 1995. Also Technical Report #1663, INRIA/Lorraine, 1992.

[BL90] Eddy Bevers and Johan Lewi. Proof by consistency in conditional equational theories. In *Conditional and Typed Rewriting Systems, 2nd International Workshop*, volume 516 of *LNCS*, pages 194–205, 1990.

[BM79] Robert S. Boyer and J Strother Moore. *A Computational Logic.* ACM Monograph Series. Academic Press, 1979.

[Bou95] Adel Bouhoula. Fundamental results on automated theorem proving by test set induction. Technical Report 2478, INRIA/Lorraine, 1995.

[BR95] Adel Bouhoula and Michaël Rusinowitch. Implicit induction in conditional theories. *Journal of Automated Reasoning*, 14(2):189–235, 1995.

[BRH94] Francois Bronsard, Uday S. Reddy, and Robert W. Hasker. Induction using term orderings. In *12th International Conference on Automated Deduction*, volume 814 of *LNCS*, pages 102–117, 1994.

[Der82] Nachum Dershowitz. Applications of the Knuth-Bendix completion procedure. In *Seminaire d'Informatique Theorique*, pages 95–111, Paris, France, 1982.

[DM79] Nachum Dershowitz and Zohar Manna. Proving termination with multiset orderings. *Communications of the ACM*, 22(8):465–476, 1979.

[Fra93] Ulrich Fraus. A calculus for conditional inductive theorem proving. In *Conditional and Typed Rewriting System, 3th International Workshop*, volume 656 of *LNCS*, pages 357–362, 1993.

[Fri86] Laurent Fribourg. A strong restriction of the inductive completion procedure. In *13th EATCS International Conference on Automata, Languages and Programming*, volume 226 of *LNCS*, pages 105–115, 1986.

[GG88] Stephen J. Garland and John V. Guttag. Inductive methods for reasoning about abstract data types. In *ACM Symposium on Principles of Programming Languages*, pages 219–228, 1988.

[Gra89] Bernhard Gramlich. Induction theorem proving using refined unfailing completion techniques. Technical Report SR89-14, Universität Kaiserslautern, 1989.

[Gra90] Bernhard Gramlich. Completion based inductive theorem proving: An abstract framework and its applications. In *9th European Conference on Artificial Intelligence*, pages 314–319, 1990.

[HH80] Gérard Huet and Jean-Marie Hullot. Proofs by induction in equational theories with constructors. In *21th Symposium on Foundations of Computer Science*, 1980. Also J. Computer and System Sciences 25:2, 239–266, 1982.

[HK88] Dieter Hofbauer and Ralf-Detlef Kutsche. Proving inductive theorems based on term rewriting. In *1st International Workshop On Algebraic And Logic Programming*, pages 180–190. Academie Verlag, 1988.

[JK86] Jean-Pierre Jouannaud and Emmanuel Kounalis. Automatic proofs by induction in equational theories without constructors. In *2nd IEEE Symposium on Logic in Computer Science*, pages 358–366, 1986.

[KNZ86] Deepak Kapur, P. Narendran, and Hantao Zhang. Proof by induction using test sets. In *8th International Conference on Automated Deduction*, volume 230 of *LNCS*, 1986.

[Kow79] Robert A. Kowalski. Algorithm = logic + control. *Communications of the ACM*, 22(7):424–436, 1979.

[KR90] Emmanuel Kounalis and Michaël Rusinowitch. Mechanizing inductive reasoning. In *Conf. of the American Association for Artificial Intelligence*, pages 240–245, 1990.

[Mus80] David R. Musser. On proving inductive properties of abstract data types. In *7th ACM Symp. on Principles of Programming Languages*, pages 154–162, 1980.

[ND96] Dimitri Naidich and T. B. Dinesh. Implicit induction techniques for the analysis of PIM – a transformational toolkit for compilers. In preparation, 1996.

[Pad92] Peter Padawitz. *Deduction and Declarative Programming*. Cambridge University Press, 1992.

[Pro94] Martin Protzen. Lazy generation of induction hypotheses. In *12th International Conference on Automated Deduction*, volume 814 of *LNCS*, pages 42–56, 1994.

[Red90] Uday S. Reddy. Term rewriting induction. In *10th International Conference on Automated Deduction*, volume 449 of *LNCS*, pages 162–177, 1990.

[Wal94] Christoph Walther. Mathematical induction. In *Handbook on Logic in Artificial Intelligence and Logic Programming*, volume 2, pages 127–228. Clarendon Press, Oxford, 1994.

[WB94] Claus-Peter Wirth and Klaus Becker. Abstract notions and inference systems for proofs by mathematical induction. In *Conditional and Typed Rewriting System, 4th International Workshop*, volume 968 of *LNCS*, pages 353–373, 1994.

[ZKK88] Hantao Zhang, Deepak Kapur, and M. S. Krishnamoorthy. A mechanizable induction principle for equational specifications. In *9th International Conference on Automated Deduction*, volume 310 of *LNCS*, pages 162–181, 1988.

A Proofs

Proof of proposition 2.3
Proposition 2.3 an instance of Proposition 2.4.

Proof of proposition 2.4
By contradiction, let $\langle \phi, \gamma \rangle$ be a $\succcurlyeq$-minimal ground formula witness among those of the formulas in Φ such that $Ax \not\models \phi\gamma$. Then, by definition 2.2, there exist a formula witness $\langle \psi, \sigma \rangle \in \langle \Psi \rangle_\phi$ and a ground substitution γ' such that $\langle \psi, \sigma \rangle \gamma' \preccurlyeq \langle \phi, \gamma \rangle$, and $Ax \not\models \psi\sigma\gamma'$. Then, by (2), there exist a formula $\phi' \in \Phi$ and a substitution θ such that $\langle \phi', \theta \rangle \prec \langle \psi, \sigma \rangle$ and $Ax \not\models \phi'\theta\gamma'$. By the stability of $\succ$, $\langle \phi', \theta\gamma' \rangle = \langle \phi, \theta \rangle \gamma' \prec \langle \psi, \sigma \rangle \gamma'$, which contradicts to the minimum property of $\langle \phi, \gamma \rangle$.

Proof of proposition 2.5
Follows directly from Proposition 2.4, 2.6.

Proof of proposition 2.6
We prove this proposition by transforming the derivations by the implicit induction procedure.

I. First, we introduce the auxiliary inference rules *Delete* and *Lemma*:

Delete

$$\frac{(P \cup \{\phi\}, H)}{(P, H)}$$

if $Ax \models (P \cup H)_{\preccurlyeq \phi} \Rightarrow \phi$

Lemma

$$\frac{(P, H)}{(P \cup \{C\}, H)}$$

Any application of *Simplify* is equivalent to a consecutive application of *Lemma* and *Delete*. So we consider rules *Lemma* and *Delete* instead of *Simplify*.[8]

II. Any consecutive application of *Delete* and another rule ρ (either *Generate* or *Lemma*) in a successful derivation can be replaced, to give another successful derivation, with

either an application of rule ρ when ρ re-introduces the conjecture deleted on the previous step,

or a consecutive application of rules ρ, *Delete*, otherwise.

III. Any consecutive application of a rule ρ and *Lemma* in a successful derivation can be replaced with a consecutive application of rules *Lemma*, ρ to give another successful derivation.

By applying the transformations above to the derivation

$$(P^\dagger, \emptyset) \vdash \ldots \vdash (\emptyset, H^\dagger)$$

we obtain a derivation $\mathcal{D}$

$$\begin{array}{l}(P^\dagger, \emptyset) \vdash_{Lemma} \ldots \vdash_{Lemma} \\ (P^\ddagger, \emptyset) \vdash_{Generate} \ldots \vdash_{Generate} \\ (P^\diamond, H^\dagger) \vdash_{Delete} \ldots \vdash_{Delete} \\ (\emptyset, H^\dagger)\end{array}$$

[8] *Delete* and *Lemma*, when $P \neq \emptyset$, are actually instances of *Simplify*, but it is not important for the current proof.

Consider the last application of *Delete* in $\mathcal{D}$ to a formula ϕ. Obviously, $Ax \models H^{\dagger}_{\preccurlyeq\phi} \Rightarrow \phi$. Hence, by the stability of $\preccurlyeq$, any condition of the form $Ax \models (\Psi \cup \{\phi\})_{\preccurlyeq\psi} \Rightarrow \psi$ in any previous application of *Delete* in $\mathcal{D}$ can be replaced with $Ax \models (\Psi \cup H^{\dagger})_{\preccurlyeq\psi} \Rightarrow \psi$. Therefore, by induction,

$$Ax \models H^{\dagger}_{\preccurlyeq\phi} \Rightarrow \phi \text{ for any } \phi \in P^{\diamond} \tag{3}$$

Next, consider an application of *Generate* to a formula $\phi \in H^{\dagger}$:

$$\frac{(P \cup \{\phi\}, H)}{(P \cup P', H \cup \{\phi\})},$$

where $\langle\Psi\rangle_{\phi}$ is a cover set of formula witnesses for ϕ w.r.t. Ax, $\succcurlyeq$, and $Ax \models (P \cup P' \cup H \cup \{\phi\})_{\prec\langle\psi,\sigma\rangle} \Rightarrow \psi\sigma$ for any $\langle\psi,\sigma\rangle \in \langle\Psi\rangle_{\phi}$. Since $(P \cup P' \cup H \cup \{\phi\}) \subset (P^{\diamond} \cup H^{\dagger})$ and $\preccurlyeq$ is stable, (3) implies $Ax \models H^{\dagger}_{\prec\langle\psi,\sigma\rangle} \Rightarrow \psi\sigma$ for any $\langle\psi,\sigma\rangle \in \langle\Psi\rangle_{\phi}$.

If $\phi \in P^{\dagger} \setminus H^{\dagger}$ then $\phi \in P^{\diamond}$. Therefore, $Ax \cup H^{\dagger} \models P^{\dagger} \setminus H^{\dagger}$ by (3).

Proof of proposition 3.3
Consider a ground formula witness $\langle\phi,\gamma\rangle$. There exist $\sigma \in CS(Ax, \succcurlyeq_t)$ and a ground substitution γ' such that $\gamma \succcurlyeq_t \sigma\gamma'$ and $Ax \models \gamma = \sigma\gamma'$. Then $Ax \models \phi\sigma\gamma' \Rightarrow \phi\gamma$ and $\langle\phi,\sigma\rangle\gamma' \preccurlyeq \langle\phi,\gamma\rangle$ by the compatibility of $\succcurlyeq$.

Proof of proposition 3.6
If $\succcurlyeq_a$ is strongly stable then $max(C\sigma) \subseteq max(C)\sigma$. The stability of $\eqslantgtr_a$ is essential to insure that $max(C\sigma) = max(C)\sigma$.

Proof of proposition 3.8
The proof is analogous to that of Proposition 3.6.

Proof of proposition 4.4
The proof is analogous to that of Proposition 4.6.

Proof of proposition 4.6
This proposition follows from Propositions A.1, A.2.

Proposition A.1 *$\succcurlyeq_{cw}$ is a strongly stable well-founded quasi-order compatible with $\succcurlyeq_t$ if so is $\succcurlyeq_{aw}$.*

Proof
$\succcurlyeq_{cw}$ is a strongly stable well-founded quasi-order because so is $\succcurlyeq^{mul}_{aw}$.
$\succcurlyeq_{cw}$ is compatible with $\succcurlyeq_t$ because $\{\{a_i\}\}_i \succcurlyeq^{mul}_{aw} \{\{a'_i\}\}_i$ if $a_i \succcurlyeq_{aw} a'_i$ for any i.

Proposition A.2 *If $\succcurlyeq_t$ is a reduction quasi-order then $\succcurlyeq_{aw}$ is a strongly stable well-founded quasi-order compatible with $\succcurlyeq_t$.*

Proof
$\succcurlyeq_{aw}$ is a well-founded quasi-order because so is $\succcurlyeq^{mul}_{t}$.
For any atom witness $\langle AT\rangle$, $\nu(\langle AT\rangle\theta) = \nu(\langle AT\rangle)\theta$. Therefore, $\succcurlyeq_{aw}$ is strongly stable because $\succcurlyeq^{mul}_{t}$ is.

$\succcurlyeq_{aw}$ is compatible with $\succcurlyeq_t$ because $\succcurlyeq_t$ is monotonic, and $\{\{t_i\}\}_i \succcurlyeq_t^{mul} \{\{t'_i\}\}_i$ if $t_i \succcurlyeq_t t'_i$ for any i.

Proof of proposition 5.5.*1* (5.5.*2*)

Let V' be the set of instances of conditional equations $CE'\theta'$ such that $CE' \in V$ and CE' is used in contextual rewriting of the terms in $E\theta$ with matching substitution θ'. We show that

1. $W \models \wedge V' \wedge CE\theta \wedge C_1 \Rightarrow C$,
2. $CE'\theta' \prec_c C$ for any $CE'\theta' \in V'$,
3. $C_1 \preccurlyeq_c C$ $(C_1 \prec_c C)$.
4. If $CE \in V$ then $CE\theta \preccurlyeq_c C$ $(CE\theta \prec_c C)$.

1: This property follows directly from the definition of relaxed contextual rewriting.

2: By the definition of relaxed contextual rewriting for a decreasing quasi-order $\succcurlyeq_t$, for any $CE'\theta' \in V'$ and any term u occurring in $CE'\theta'$, $u \prec_t t$. Hence, $CE'\theta' \prec_c (t = s)^\varepsilon$. Also, we have $(t = s)^\varepsilon \preccurlyeq_c C$. Therefore, $CE'\theta' \prec_c C$.

3: In all the cases of definition 5.4 $(t = s)^\varepsilon \succcurlyeq_a (t' = s)^\varepsilon$ $((t = s)^\varepsilon \succ_a (t' = s)^\varepsilon)$. Therefore, $C \succcurlyeq_c C_1$ $(C \succ_c C_1)$.

4: Because $\neg E\theta \prec_c (t = s)^\varepsilon$, it is sufficient to show that either $e\theta \prec_a (t = s)^\varepsilon$, or $e\theta \sim_a (t = s)^\varepsilon$ and $\neg E\theta \preccurlyeq_c C'$ $(\neg E\theta \prec_c C')$.

(a) Trivial.
(b)–(c),(e) $e\theta \prec_a (t = s)^\varepsilon$
(d),((d')) $e\theta \sim_a (t = s)^\varepsilon$ and $\neg E\theta \preccurlyeq_c C'$ $(\neg E\theta \prec_c C')$.

Proof of proposition 5.7.*1* (5.7.*2*)

Let $\langle V' \rangle$ be the set of witnesses of conditional equations $\langle CE', \theta' \rangle$ such that $CE' \in V$ and CE' is used in contextual rewriting of terms in $E\theta$ with matching substitution θ'. We show that

1. $W \models \wedge \{CE'\theta' | \langle CE', \theta' \rangle \in \langle V' \rangle\} \wedge CE\theta \wedge C_1 \Rightarrow C\sigma$,
2. $\langle CE', \theta' \rangle \prec_{cw} \langle C, \sigma \rangle$ for any $\langle CE', \theta' \rangle \in \langle V' \rangle$,
3. $C_1 \preccurlyeq_{cw} \langle C, \sigma \rangle$ $(C_1 \prec_{cw} \langle C, \sigma \rangle)$.
4. If $CE \in V$ then $\langle CE, \theta \rangle \preccurlyeq_{cw} \langle C, \sigma \rangle$ $(\langle CE, \theta \rangle \prec_{cw} \langle C, \sigma \rangle)$.

1: This property follows directly from the definition of relaxed contextual replacement.

2: By the definition of contextual rewriting for a decreasing quasi-order $\succcurlyeq_t$, for any $\langle CE', \theta' \rangle \in \langle V' \rangle$, and any term u occurring in $CE'\theta'$, $u \prec_t t\sigma$. Hence, $\langle CE', \theta' \rangle \prec_{cw} \langle (t = s)^\varepsilon, \sigma \rangle$. Also, we have $\langle (t = s)^\varepsilon, \sigma \rangle \preccurlyeq_{cw} \langle C, \sigma \rangle$. Therefore, $\langle CE', \theta' \rangle \prec_{cw} \langle C, \sigma \rangle$.

3: In all the cases of definition 5.6 $\langle (t = s)^\varepsilon, \sigma \rangle \succcurlyeq_{aw} (t' = s\sigma)^\varepsilon$ $(\langle (t = s)^\varepsilon, \sigma \rangle \succ_{aw} (t' = s\sigma)^\varepsilon)$. Therefore, $\langle C, \sigma \rangle \succcurlyeq_{cw} C_1$ $(\langle C, \sigma \rangle \succ_{cw} C_1)$.

4: Because $\langle \neg E, \theta \rangle \prec_{cw} \langle (t = s)^{\varepsilon}, \sigma \rangle$, it is sufficient to show that either $\langle e, \theta \rangle \prec_{aw} \langle (t = s)^{\varepsilon}, \sigma \rangle$, or $\langle e, \theta \rangle \sim_{aw} \langle (t = s)^{\varepsilon}, \sigma \rangle$ and $\langle \neg E, \theta \rangle \preccurlyeq_{cw} \langle C', \sigma \rangle$ ($\langle \neg E, \theta \rangle \prec_{cw} \langle C', \sigma \rangle$).

(a) Trivial.

(b)–(d),(f) $\langle e, \theta \rangle \prec_{aw} \langle (t = s)^{\varepsilon}, \sigma \rangle$

(e),((e')) $\langle e, \theta \rangle \sim_{aw} \langle (t = s)^{\varepsilon}, \sigma \rangle$ and $\langle \neg E, \theta \rangle \preccurlyeq_{cw} \langle C', \sigma \rangle$ ($\langle \neg E, \theta \rangle \prec_{cw} \langle C', \sigma \rangle$).

Proof of proposition 5.9
We show that

1. $W \models \wedge(P \cup \{ CE_i\theta_i | CE_i \in V \}) \Rightarrow C$,
2. $C' \prec_c C$ for any $C' \in P$,
3. If $CE_i \in V$ then $CE_i\theta_i \prec_c C$.

1: This property follows directly from the definition.

2,3: For any i, $C \equiv AT^i \vee C'_i$, $C_1^i \equiv AT_1^i \vee C'_i$, and, by the definition of $\mapsto\circ$ (cf. proof of Proposition 5.5.2),

(a) $AT^i \succ_a AT_1^i$,
(b) $AT^i \succ_a e_i\theta_i$ if $CE_i \in V$ and
(c) $AT^i \succ_a AT'\theta_i$ for any $AT' \in \neg E_i$.

Therefore, $CE_i\theta_i \prec_c C$ if $CE_i \in V$ and $C' \prec_c C$ for any $C' \in P_1$.

Moreover, $AT_i \equiv (t_i = s_i)^{\varepsilon_i}$ and $t_i \succ_t t'$ for any term in $E_i\theta_i$. Therefore, as all the terms in $CNF(\vee_i E_i\theta_i)$ occur in $E_i\theta_i$ for some i, by the definition of $\succcurlyeq_c$, $C' \prec_c C$ for any $C' \in P_2$ as well.

Proof of proposition 5.11
We show that

1. $W \models \wedge(P \cup \{ CE_i\theta_i | CE_i \in V \}) \Rightarrow C\sigma$,
2. $C' \prec_{cw} \langle C, \sigma \rangle$ for any $C' \in P$,
3. If $CE_i \in V$ then $\langle CE_i, \theta_i \rangle \prec_{cw} \langle C, \sigma \rangle$.

1: This property follows directly from the definition.

2,3: For any i, $C \equiv AT^i \vee C'_i$, $C_1^i \equiv AT_1^i \vee C'_i\sigma$, and, by the definition of $\twoheadrightarrow\circ$ (cf. proof of Proposition 5.7.2),

(a) $\langle AT^i, \sigma \rangle \succ_{aw} AT_1^i$,
(b) $\langle AT^i, \sigma \rangle \succ_{aw} \langle e_i, \theta_i \rangle$ if $CE_i \in V$ and
(c) $\langle AT^i, \sigma \rangle \succ_{aw} \langle AT', \theta_i \rangle$ for any $AT' \in \neg E_i$.

Therefore, $\langle CE_i, \theta_i \rangle \prec_{cw} \langle C, \sigma \rangle$ if $CE_i \in V$ and $C' \prec_{cw} \langle C, \sigma \rangle$ for any $C' \in P_1$.

Moreover, $AT_i \equiv (t_i = s_i)^{\varepsilon_i}$ and $t_i\sigma \succ_t t'$ for any term in $E_i\theta_i$. Therefore, as all the terms in $CNF(\vee_i E_i\theta_i)$ occur in $E_i\theta_i$ for some i, by the definition of $\succcurlyeq_{cw}$, $C' \prec_{cw} \langle C, \sigma \rangle$ for any $C' \in P_2$ as well.

Proof of proposition 5.13.*1*(5.13.*2*)
We show that

1. $\models C'\theta \Rightarrow C$,
2. $\langle C', \theta\rangle \preccurlyeq_{cw} C$ and $C'\theta \preccurlyeq_c C$ ($\langle C', \theta\rangle \prec_{cw} C$ and $C'\theta \prec_c C$ If $C' \in V$).

1: This property follows directly from the definition of subsumption.

2: This property follows directly from the strict subterm property of $\succcurlyeq_t$.

Proof of proposition 5.15
1: $\models P \Rightarrow C$. Also, by the proper subterm property of $\succcurlyeq_t$, $C' \prec_c C$ and $C' \prec_{cw} C$ for any $C' \in P$.
2: $W \models C_1 \Rightarrow C$. Also, $C_1 \prec_c C$ and $C_1 \prec_{cw} C$.
3: $W \models C_1 \Rightarrow C$. Also, $C_1 \preccurlyeq_c C$ and $C_1 \preccurlyeq_{cw} C$.

Proof of proposition 5.17
The proof is analogous to that of Proposition 5.7.
Proof of proposition 6.1
This proposition is a direct consequence of Proposition 2.5, Propositions 4.4, 5.5, 5.9, 5.13 for the case of $\mathcal{J}$, and Propositions 4.6, 5.7, 5.11 for the case of $\mathcal{K}$.

B Errata

In [BR95], Section 8, the simplification by relaxed contextual rewriting is captured by the inference rule formulated, in our notation, as follows:

Simplify'

$$\frac{(P \cup \{(t = s)^e \vee C'\}, H)}{(P \cup \{(t' = s)^e \vee C'\}, H)}$$

if $t \hookrightarrow^{\omega}_{H \cup P \cup R; C'} t'$, $\omega \neq \epsilon$, and $t' = s \prec_e t = s$, where the partial order on equations $\prec_e$ is defined below.

First, the complexity ν of an equation $t = s$ is defined by

$$\nu(t = s) = \begin{cases} (\{\{t\}\}, \{\{s\}\}) & \text{if } t \succ_t s \\ (\{\{s\}\}, \{\{t\}\}) & \text{if } s \succ_t t \\ (\{\{t, s\}\}, \emptyset) & \text{otherwise} \end{cases}$$

Then, $t = s \succ_e u = v$ iff $\nu(t = s)$ is greater than $\nu(u = v)$ by the lexicographic comparison of their components by $\succ_t^m$.

Thus, the optimized proof procedure [BR95] permits the derivations specified by the properly restricted rules *Generate* and *Simplify*, Section 5.2, rule *Subsume*, section 5.6, and rule *Simplify'*. It also trivially permits deleting tautologies. We denote the inference system consisting of these inference rules by $\mathcal{J}'(Ax, \succcurlyeq_t)$. We show that $\mathcal{J}'(Ax, \succcurlyeq_t)$ allows for a successful derivation of inconsistent conjectures. This derivation is necessarily technical to satisfy the conditions of $\mathcal{J}'(Ax, \succcurlyeq_t)$, but, basically, it exploits the non-stability of $\succ_e$.

Proposition B.1 (Counterexample) *Consider the rewrite system $\mathcal{R}$ defining functions f and g*

$$\begin{array}{ll} f(0) \to 0 & g(0) \to s(0) \\ f(s(x)) \to s(f(x)) & g(s(x)) \to s(g(x)) \end{array}$$

and the auxiliary functions f' and i

$$\begin{array}{l} f'(0) \to 0 \\ f'(s(x)) \to s(f'(x)) \\ f'(0) \to f(0) \\ i(x) \to x \end{array}$$

Consider the set of conjectures $P \equiv \{g(x) = i(f'(x)),\ g(y) = i(f(y)),\ f'(z) = f(z)\}$. Then

1. *$\mathcal{R} \not\models_{ind} P$, and*
2. *there exist a decreasing quasi-order $\succcurlyeq_{pol}$ and a successful derivation for P by $\mathcal{J}'(\mathcal{R}, \succcurlyeq_{pol})$.*

Proof of proposition B.1
1: E.g., $\mathcal{R} \not\models g(0) = i(f(0))$.
2: First, we define the induction ordering. Let $\succcurlyeq_{pol}$ be the reflexive closure of the polynomial ordering defined by the following weights $\lceil\ \rceil$:

$$\lceil 0 \rceil = 0,\ \lceil s \rceil(n) = n+1,\ \lceil f \rceil(n) = 3n^2+1,\ \lceil g \rceil(n) = 2n^2+4,\ \lceil f' \rceil(n) = 2n^2+2,\ \lceil i \rceil(n) = n+1.$$

$\succcurlyeq_{pol}$ is a decreasing quasi-order, R is oriented w.r.t. $\succ_{pol}$, and

$$g(y) \eqslantgtr_{pol} i(f(y)),\ g(y) \succ_{pol} i(f'(y)).$$

The latter implies that

$$g(y) = i(f(y))\ (\succ_{pol})_e\ g(y) = i(f'(y)).$$

Note that $g(0) = i(f(0))\ (\succ_{pol})_e\ g(0) = i(f'(0))$ does not hold.

Also, $\{\{x \to 0\}, \{x \to s(x')\}\}$ is a cover set of substitutions for $\mathcal{R}$.

The critical derivation by $\mathcal{J}'(\mathcal{R}, \succcurlyeq_{pol})$ is as follows:

$(\{g(x) = i(f'(x)),\ g(y) = i(f(y)),\ \underline{f'(z) = f(z)}\}, \emptyset)$
$\vdash \ldots \vdash$
$(\{\underline{g(x) = i(f'(x))},\ g(y) = i(f(y))\}, \{f'(z) = f(z)\})$
$\vdash_{Generate}$
$(\{\underline{g(0) = i(f(0))},\ g(s(x')) = i(s(f'(x'))),\ \underline{g(y) = i(f(y))}\}, \{f'(z) = f(z),\ g(x) = i(f'(x))\})$
$\vdash_{Subsume}$
$(\{g(s(x')) = i(s(f'(x'))),\ \underline{g(y) = i(f(y))}\}, \{\underline{f'(z) = f(z)},\ g(x) = i(f'(x))\})$
$\vdash_{Simplify'}$
$(\{g(s(x')) = i(s(f'(x'))),\ \underline{g(y) = i(f'(y))}\}, \{f'(z) = f(z),\ \underline{g(x) = i(f'(x))}\})$
$\vdash_{Subsume}$
$(\{\underline{g(s(x')) = i(s(f'(x')))}\}, \{f'(z) = f(z),\ g(x) = i(f'(x))\})$
$\vdash_{Simplify}$
$(\{\underline{s(g(x')) = i(s(f'(x')))}\}, \{f'(z) = f(z),\ \underline{g(x) = i(f'(x))}\})$
$\vdash_{Simplify}$
$(\{\underline{s(i(f'(x'))) = i(s(f'(x')))}\}, \{f'(z) = f(z),\ g(x) = i(f'(x))\})$
$\vdash_{Simplify} \vdash_{Simplify}$
$(\{\underline{s(f'(x')) = s(f'(x'))}\}, \{f'(z) = f(z),\ g(x) = i(f'(x))\})$
$\vdash_{Delete}$
$(\emptyset, \{f'(z) = f(z),\ g(x) = i(f'(x))\})$