Inductive Proofs by Resolution and Paramodulation

Peter Padawitz
Fakultät für Mathematik und Informatik
Universität Passau
Postfach 2540
D-8390 Passau

Constructor-based sets of Horn clauses constitute a class of formulas for presenting verification problems occurring in data type specification as well as functional and logic programming. Inductive proofs of such clause sets can be carried out in a strict top-down manner by inductive expansion: the set is transformed via (linear) resolution and paramodulation into a case distinction, which covers all ground substitutions. Being a backward method, inductive expansion reduces the search space of corresponding forward proofs. The method does not put confluence or termination restrictions on the theorems to be proved such as procedures based on inductive completion do. Moreover, inductive expansion does not prescribe a strategy for controlling search trees so that the user may select "promising" paths according to specific applications.

1. Introduction

The mathematical models used in data type specification and program verification are *term-generated.* Each carrier element of the model is obtained by evaluating a *ground,* i.e. variable-free, functional expression. Hence a valid statement takes the form of an *inductive theorem,* which means that all ground instances are derivable. The proof is carried out by induction on the structure of ground terms (cf. [Bur69]) or, more generally, by induction with respect to a *Noetherian* relation on ground terms.

As one knows from inductive proofs in general, it might be difficult, not only to find a suitable Noetherian relation, but also to state an appropriate induction hypothesis, which often turns out to be a *generalization* of the theorem to be proved. While classical theorem proving provides explicit (more or less heuristic) induction rules to solve these problems, *inductive completion* (or *inductionless induction*) tries to get rid of induction steps by switching to *consistency* (or *conservative extension*) proofs (cf. [HH82], [JK86], [KM87], [Pad88a]).

Inductive completion puts strong restrictions not only on the underlying specification, but also on the theorems to be proved. Its requirement that axioms *and* theorems induce a *Church-Rosser* set of rewrite rules entails a number of syntactical restrictions, which might not be welcome, although some of these restrictions can be lowered if one uses weaker Church-Rosser criteria (cf. [HR87], [Pad88a]). In this paper, we describe an alternative method for proving inductive theorems based on traditional approaches like [BM79], [Hut86] and [GG88].

We start out from (Horn) clauses, written as $p \Leftarrow \gamma$, where p is an atom(ic formula) and γ is a finite set of atoms, called a *goal,* which consists of the premises under which p is required to hold. The existence of premises compels us to choose between two definitions of an inductive theorem:

Let AX be a set of axioms and $\vdash$ be a complete inference relation for valid clauses (cf. Section 2). For each clause $p \Leftarrow \gamma$ and each ground substitution f, let $p[f] \Leftarrow \gamma[f]$ denote the *instance of* $p \Leftarrow \gamma$ *by f,* i.e. the clause constructed from $p \Leftarrow \gamma$ by instantiating all variables according to f. By the first definition, $p \Leftarrow \gamma$ is an inductive theorem if for all ground substitutions f,

$$AX \cup \gamma[f] \vdash p[f]. \tag{1}$$

Alternatively, one may define: $p \Leftarrow \gamma$ is an inductive theorem if for all ground substitutions f,

$$AX \vdash \gamma[f] \text{ implies } AX \vdash p[f]. \tag{2}$$

(1) is equivalent to the validity of $p \Leftarrow \gamma$ in *all* term-generated models of AX (cf. [Pad88a], Cor. 4.3.3), while (2) characterizes the validity of $p \Leftarrow \gamma$ in the subclass of all *initial* models. (2) is weaker than (1): By (1), we may use $\gamma[f]$ in a proof of $p[f]$. By (2), we may also use formulas which occur in *every* derivation of $\gamma[f]$. When analyzing data type specifications one observes that crucial consequences of their axioms are valid in the sense of (2), but not in the sense of (1) (cf., e.g., [Pad88b] or [Pad88a], Ex. 4.3.4). In data base applications, the essence of (2) is known as the *closed world assumption* (cf. [Rei78]) that certain implications $p \Leftarrow q$ are in fact

equivalences. If, for deriving an instance of p *it is necessary* to derive the corresponding instance of q, then $q \Leftarrow p$ holds true as well, but, in general, only in the sense of (2). It is often the case that, for proving an implication inductively, one needs the inverse of an axiom as a lemma (cf. Ex. 4.8).

We present the proof method of *inductive expansion* in three steps. First, the (meta-)implication involved in (2) is eliminated. Therefore, a set *IN* of variables, called *input variables*, is separated from all other variables, which are called *output variables*. *Input terms* contain only input variables, *output terms* contain only output variables. Some (weak) conditions are put on the theorems to be proved such that (2) becomes a consequence of the following non-implicational property: For all ground substitutions f,

$$AX \vdash (\{p\} \cup \gamma)[f|IN+g] \quad \text{for } \textit{some } g. \tag{3}$$

(Here f+g stands for the parallel composition of f and g, which maps the domains of f and g to the images of f and g, respectively.) Moreover, we distinguish clause *sets* M such that M consists of inductive theorems if and only if for all ground substitutions f,

$$AX \vdash (\{p\} \cup \gamma)[f|IN+g] \quad \text{for } \textit{some } p \Leftarrow \gamma \in M \text{ and some } g. \tag{4}$$

This characterization is valid if M is *constructor-based*, i.e.,

- for all $p \Leftarrow \gamma \in M$, γ is a set of equations with input terms on the left-hand side and *constructors* on the right-hand side,
- the set of premises over all clauses of M constitutes a *complete* and *minimal* case distinction,

where constructors are output terms such that

- each two constructor instances c[f] and d[g] are *decomposable*, i.e. c[f] and d[g] are equivalent (w.r.t. the underlying axioms) only if c = d and f and g are equivalent.

In a second step, we aim at reducing the infinite number of *forward* proofs involved in (4) to a finite number of *backward* proofs (here called *expansions*) of the form

$$\begin{aligned} &\langle \{p_1\} \cup \gamma_1, id \rangle \vdash_{EX} \langle \delta_1, g_1 \rangle, \\ &\qquad \ldots \\ &\langle \{p_n\} \cup \gamma_n, id \rangle \vdash_{EX} \langle \delta_n, g_n \rangle \end{aligned} \tag{5}$$

such that $\{p_1 \Leftarrow \gamma_1, \ldots, p_n \Leftarrow \gamma_n\}$ covers the set of theorems to be proved and each ground substitution f is *subsumed* by some $\langle \delta_i, g_i \rangle$, i.e. $AX \vdash \delta_i[h]$ and $g_i[h] = f$ for some h. *id* denotes the identity substitution and $\vdash_{EX}$ stands for the inference relation generated by (linear) *resolution* [Rob65] and *paramodulation* [RW69].

The actual power of the approach is accomplished in a third step when *inductive* resolution and paramodulation rules are added to $\vdash_{EX}$. Applications of these rules simulate induction steps by resolving or paramodulating upon induction hypotheses. Furthermore, an inductive rule produces an atom of the form $fz >> z'$ where f is the substitution obtained so far, z is the sequence of input variables, z' is a copy of z and >> is a Noetherian relation, which justifies the induction step. Regarding $fz >> z'$ as a subgoal amounts to proving M *and* the soundness of the induction step simultaneously. This is what we call *inductive expansion*: resolving and paramodulating upon axioms, lemmas *and* induction hypotheses. The main result of the paper (Theorem 4.7) characterizes constructor-based inductive theorems as being provable by inductive expansion.

Section 2 presents basic notions concerning the syntax and semantics of Horn clause specifications with equality. Section 3 gives a precise definition of constructor sets and constructor-based clause sets along with their characterization as inductive theorems (Theorem 3.4). Section 4 starts from resolution and paramodulation as the basis of *backward* proofs and leads to the main result, given by Theorem 4.7 (see above).

At certain points, an inductive expansion relies on complete case distinctions (called *case matrices*). *The* requirement for completeness will be reduced to the question whether certain terms are *base-representable*. Theorem 5.3 tells us how this property can be proved by inductive expansion as well.

2. Preliminaries

Given a set S, an *S-sorted set* A is a family of sets, i.e. $A = (A_s \mid s \in S)$. For all $w = (s_1,...,s_n) \in S^*$, A_w denotes the cartesian product $A_{s_1} \times ... \times A_{s_n}$. A *signature* SIG = (S,OP,PR) consists of a set S of *sorts* and two S^+-sorted sets OP and PR the elements of which are called *function* (or *operation*) *symbols* and *predicate symbols*, respectively. S-sorted function symbols are called *constants*.

We assume that for all $s \in S$, PR_{ss} implicitly contains a predicate symbol $\equiv_s$, called the *equality predicate* for s. We also fix an S-sorted set X of *variables* such that for all $s \in S$, X_s is countably infinite.

Example 2.1 The signature of our running example throughout the paper provides constructor functions for Boolean values, natural numbers, sequences and bags (multisets) together with operations that will be axiomatized later.

SORT

sorts	bool, nat, seq, bag	
	symbol	*type*
opns	true	bool
	false	bool
	0	nat
	_+1	nat ⟶ nat
	ε	seq
	_&_	nat,seq ⟶ seq
	∅	bag
	add(_,_)	nat,bag⟶ bag
	le(_,_)	nat,nat ⟶ bool
	seqToBag(_)	seq ⟶ bag
	insert(_,_)	nat,seq ⟶ seq
	sort(_)	seq ⟶ seq
preds	_ ≤ _	nat,nat
	_ > _	nat,nat
	sorted(_)	seq
	_ >> _	seq,seq

We hope that the notation is self-explanatory. ∎

Given a signature SIG = (S,OP,PR), a *SIG-structure* A consists of an S-sorted set, also denoted by A, a function $F^A : A_w \longrightarrow A_s$ for each function symbol $F \in OP_{ws}$, $w \in S^*$, $s \in S$, and a relation $P^A \subseteq A_w$ for each predicate symbol $P \in PR_w$, $w \in S^+$. *T(SIG)* denotes the S^+-sorted set of terms (and term tuples) over SIG.

Given a term t, *root(t)*, *var(t)* and *single(t)* denote the leftmost symbol of t, the set of all variables of t, and the set of variables that occur exactly once in t, respectively. t is *ground* if var(t) is empty. *GT(SIG)* denotes the set of ground terms over SIG. We assume that SIG is *inhabited*, i.e. for each sort s there is a ground term t of sort s.

When speaking about terms in general, we use the prefix notation: *F* is placed in front of its argument list *t* to give the term *Ft*. In examples, however, the layout of terms is adapted to the underlying signature where infix, postfix or mixfix notations may occur as well.

Let A and B be S-sorted sets. An *S-sorted function* $f : A \longrightarrow B$ is a family of functions, i.e. $f = (f_s : A_s \longrightarrow B_s \mid s \in S)$. The set of S-sorted functions from A to B is denoted by B^A. The functions of $T(SIG)^X$ are called *substitutions*. Given a substitution f, *dom(f)*, the *domain of f*, is the set of all $x \in X$ such that $fx \neq x$. If dom(f) is empty, f is called the *identity substitution* and is denoted by *id*. If dom(f) is finite, say $dom(f) = \{x_1,...,x_n\}$, and if $fx_1 = t_1,..., fx_1 = t_n$, we also write $(t_1/x_1,...,t_n/x_n)$ instead of f. Given $V \subseteq X$, *f|V*, the restriction of f to V, is defined by (f|V)(x) = fx for all $x \in V$ and by (f|V)(x) = x for all $x \in X\text{-}V$. f is *ground* if the range of f consists of ground terms.

The *instance* of a term *t by f*, denoted by *t[f]*, is the term obtained from t by replacing all variables of t by their values under f. Conversely, one says that t *subsumes* t[f] or that t is a *prefix* of t[f]. f *unifies* t and t' if t[f] = t'[f]. The *sequential composition* of two substitutions f and g, denoted by f[g], is defined by (f[g])(x) = (fx)[g] for all $x \in X$. Accordingly, f[g] is an *instance* of f, and f *subsumes* f[g]. The *parallel composition* of f and g, denoted by f+g, is defined only if f and g have distinct domains. Then (f+g)(x) = fx if $x \in$ dom(f), and (f+g)(x) = gx otherwise.

Given $w \in S^+$, $P \in PR_w$ and $u \in T(SIG)_w$, the expression Pu is called an *atom*. If P is an equality predicate and thus w = (s,s) for some $s \in S$ and u = (t,t') for some $t,t' \in T(SIG)_w$, then Pu is called an *equation*, written as t≡t'. The notions *var*, *instance* and *unifier* extend from terms to atoms as if predicate symbols were function symbols.

Finite sets of atoms are called *goals*. A *clause* $p \Leftarrow \gamma$ consists of an atom p, the *conclusion* of $p \Leftarrow \gamma$, and a goal $\gamma = \{p_1,...,p_n\}$, the *premise* of $p \Leftarrow \gamma$. If p is an equation, then $p \Leftarrow \gamma$ is a *conditional equation*. If γ is empty, then $p \Leftarrow \gamma$ is *unconditional* and we identify $p \Leftarrow \gamma$ with the atom p. Note that unconditional clauses and goals are the same.

A *specification* is a pair (SIG,AX), consisting of a signature SIG and a set AX of clauses, comprising the axioms of the specification.

Example 2.1 (continued) The axioms of SORT, specifying *sort* as "insertion sort", are given by:

```
vars    x,y : nat;  s : seq;  b : bag
axms    seqToBag(ε) ≡ ∅                                          (BA1)
        seqToBag(x&s) ≡ add(x,seqToBag(s))                       (BA2)
        add(x,add(y,b)) ≡ add(y,add(x,b))                        (BA3)
        sort(ε) ≡ ε                                              (IS1)
        sort(x&s) ≡ insert(x,sort(s))                            (IS2)
        insert(x,ε) ≡ x&ε                                        (IN1)
        insert(x,y&s) ≡ x&y&s  ⇐  x ≤ y                          (IN2)
        insert(x,y&s) ≡ y&insert(x,s)  ⇐  x > y                  (IN3)
        le(0,x) ≡ true
        le(x+1,0) ≡ false
        le(x+1,y+1) ≡ le(x,y)
        x ≤ y  ⇐  le(x,y) ≡ true                                 (LE1)
        x > y  ⇐  le(x,y) ≡ false                                (LE2)
        sorted(ε)                                                (SO1)
        sorted(x&ε)                                              (SO2)
        sorted(x&y&s)  ⇐  x ≤ y, sorted(y&s)                     (SO3)
        x&s >> s                                                 (GR)  ∎
```

A clause $Pu \Leftarrow P_1u_1,...,P_nu_n$ is *valid* in a SIG-structure A if for all $b \in A^X$, $(\forall\ 1 \le i \le n : b^*u_i \in P_i^A)$ implies $b^*u \in P^A$, where b^* is the unique (SIG-) homomorphic extension of b to T(SIG). Given a clause set AX, A is a *SIG-model of AX* if each $p \Leftarrow \gamma \in$ AX is valid in A and if for all $s \in S$, $\equiv_s^A$ is the identity on A_s.

Let us fix a specification (SIG,AX). The *cut calculus with equality* consists of the congruence axioms for all equality symbols (w.r.t. SIG) and two inference rules:

(SUB)	For all substitutions f, $p \Leftarrow \gamma \ \vdash\ p[f] \Leftarrow \gamma[f]$.
(CUT)	$(p \Leftarrow \gamma \cup \{q\},\ q \Leftarrow \delta) \ \vdash\ p \Leftarrow \gamma \cup \delta$.

$\vdash_C$ denotes the corresponding inference relation. The class of all SIG-models of AX satifies a clause $p \Leftarrow \gamma$ if and only if p can be derived from AX∪γ via the cut calculus with equality such that the variables of γ need not be instantiated (cf. [Pad88a], Cor. 4.2.4).

Two terms t and t' are called *AX-equivalent* if $AX \vdash_C t \equiv t'$. Two substitutions f and g are AX-equivalent if for all $x \in X$, fx and gx are AX-equivalent.

Definition A clause $p \Leftarrow \gamma$ is called an *inductive AX-theorem* if for all ground substitutions f,

$$AX \vdash_C \gamma[f] \text{ implies } AX \vdash_C p[f].$$

The set of inductive AX-theorems is denoted by *ITh(AX)*. A set M of clauses is an inductive AX-theorem if all clauses of M are inductive AX-theorems. ∎

The model-theoretic counterpart of inductive theorems are *initial structures*:

A SIG-structure A is *initial w.r.t. AX* if A satisfies AX and each model B of AX admits a unique (SIG-) homomorphism from A to B. *Ini(AX)* denotes the (isomorphism) class of initial structures w.r.t. AX.

Theorem 2.2 (cf. [Pad88a], Thm. 4.4.3) *$p \Leftarrow \gamma \in$ ITh(AX) iff Ini(AX) satisfies $p \Leftarrow \gamma$.* ∎

Corollary 2.3 *ITh(ITh(AX)) = ITh(AX).* ∎

3. Constructor-based Clause Sets

General Assumption (part 1) Let *IN* be a fixed finite set of variables, called *input variables*. The elements of the complement *OUT* = X-IN are called *output variables*. *Input terms* are terms containing only input variables, *output terms* are terms containing only output variables. ∎

Definition A set T of output terms is *ground complete for a term t* if for all ground substitutions f there is $t' \in T$ such that t[f] is AX-equivalent to some instance of t'. T is *ground complete* if T is ground complete for all ground terms. T is a *set of constructors* if for all $c,d \in T$ and ground substitutions f,g such that c[f] and d[g] are AX-equivalent, c equals d and f|var(c) is AX-equivalent to g|var(c). ∎

In many applications, the constructor property can be checked easily by referring to a given initial structure A w.r.t. AX: Ground terms are AX-equivalent iff they denote the same element of A in terms of which the property is obvious. A "syntactical" constructor criterion is given in Section 6.

In order to define constructor-based clause sets we need a schema for presenting case distinctions.

Definition A finite set CM of finite sets of equations is a *(constructor-based) case matrix with input $IN_0 \subseteq$ X* if either $CM = \{\emptyset\}$ or $CM = \{\{t \equiv c_i\} \cup \gamma \mid 1 \le i \le n, \gamma \in CM_i\}$ for a term t, a ground complete set $\{c_1,...,c_n\}$ of output terms (constructors) for t, and (constructor-based) case matrices $CM_1,...,CM_n$ with input $IN_0 \cup var(c_1)$, ..., $IN_0 \cup var(c_n)$, respectively, such that $var(t) \subseteq IN_0$ and for all $1 \le i \le n$, $var(c_i) \cap IN_0 = \emptyset$. ∎

For instance, a constructor-based case matrix using SORT (cf. Ex. 2.1) is given by

$$\{\{s \equiv \varepsilon\},$$
$$\{s \equiv x\&\varepsilon\},$$
$$\{s \equiv x\&y\&s'\}, \{le(x,y) \equiv true\},$$
$$\{s \equiv x\&y\&s'\}, \{le(x,y) \equiv false\}\}$$

The case matrix condition is purely syntactic except for the ground completeness of $\{c_1,...,c_n\}$. How to prove this property is the topic of Section 5.

A case matrix covers the set of ground substitutions:

Proposition 3.1 *Let CM be a case matrix with input IN_0. Then for all ground substitutions f there are $\gamma \in CM$ and a substitution g such that $AX \vdash_C \gamma[f|IN_0+g]$.* ∎

Sometimes the case matrix condition is too restrictive (cf. the first covering derived in Ex. 4.8). In fact, it suffices to get a case matrix as an instantiation of a set of sets of equations:

Definition A set EM of sets of equations is *extendable to a case matrix* if there are output variables $x_1,...,x_n$ and output terms $c_1,...,c_n$ such that $CM = \{\gamma[c_1/x_1,...,c_n/x_n] \mid \gamma \in EM, 1 \le i \le n\}$ is a case matrix. ∎

Prop. 3.1 immediately implies

Proposition 3.2 *Let EM be extendable to a case matrix with input $IN_0 = IN$ (see above). Then for all ground substitutions f there are $\gamma \in CM$ and a substitution g such that $AX \vdash_C \gamma[f|IN+g]$.* ∎

For *un*conditional theorems, (2) and (3) (cf. Sect. 1) coincide. (Take IN = X.) If premises are involved, the equivalence of (2) and (3) is guaranteed only for constructor-based clause sets.

Definition A set M of clauses is *constructor-based* if there is a constructor-based case matrix CM with input IN such that

(a) $\gamma \in CM$ iff there is p with $p \Leftarrow \gamma \in M$,
(b) for all $p \Leftarrow \gamma \in M$, $var(p) \subseteq IN \cup var(\gamma)$,
(c) for all $p \Leftarrow \gamma, q \Leftarrow \delta \in M$ with $p \Leftarrow \gamma \neq q \Leftarrow \delta$, γ is not a subset of δ. ∎

Conditions (a)-(c) are purely syntactic. (c) forbids different clauses with subsuming premises. The "if" part of (a) can always be ensured by adding to M a clause $x \equiv x \Leftarrow \gamma$ for each "missing case" $\gamma \in CM$, provided that CM exists, i.e., predicates are available for specifying a complete case distinction. If the set of premises does not cover all ground substitutions, one may decompose it into a case matrix and a common condition ϑ and treat ϑ by *premise elimination* (cf. Sect. 4).

Constructor-based clause sets turned out to comprise a language used very frequently for writing functional and logic programs. From sorting algorithms via tree and graph manipulating functions up to interpreters, this language is powerful enough for bringing them into a concise *and* executable form (cf. [Pad87,88b,c,e]). The translation of a suitable sublanguage into PASCAL is described in [GHM88].

The following lemma is crucial for characterizing constructor-based inductive theorems (cf. Thm. 3.4). It says that each ground substitution satisfies at most one clause of a constructor-based clause set.

Lemma 3.3 *Let M be a constructor-based clause set, $p \Leftarrow \gamma, q \Leftarrow \delta \in M$ and f,g be ground substitutions such that $AX \vdash_C ((p) \cup \gamma)[f|IN+g] \cup \delta[f]$. Then $\gamma = \delta$ and $AX \vdash_C q[f]$.* ∎

Theorem 3.4 *A constructor-based clause set M is an inductive AX-theorem iff for all ground substitutions f there are $p \Leftarrow \gamma \in M$ and a substitution g such that $AX \vdash_C ((p) \cup \gamma)[f|IN+g]$.* ∎

Theorem 3.4 provides the basis for inductive proofs of constructor-based clause sets. In the next section, we turn from forward proofs using the cut calculus to backward proofs based on resolution and paramodulation.

4. Inductive Expansion

Derivations via the cut calculus proceed bottom-up from axioms to the theorems to be proved. In contrast, *resolution* and *paramodulation* work top-down from a goal by applying axioms backwards until the empty goal is achieved, indicating that the initial goal is *solvable*. A solution is built up stepwisely in the course of the proof. We call such a derivation an *expansion* in order to stress the "procedural interpretation" of Horn clauses underlying this kind of proof.

For guaranteeing the completeness of paramodulation it is well-known that in some (rare) cases *functionally-reflexive* axioms of the form $Fx \equiv Fx$ must be applied. In [Pad88a], Chapter 5, we have shown that these additional axioms need only occur as superterms of instances of other axioms. Hence, instead of adding all functionally-reflexive axioms to AX, we replace AX by the set of *prefixed axioms of AX*.

Definition *Pre(AX)*, the set of *prefixed axioms of AX*, is the smallest set of clauses, which contains all conditional equations of AX and satisfies the following closure property:

- If $u \equiv u' \Leftarrow \vartheta \in Pre(AX)$ and t is a term of the form $F(x_1,...,x_n)$ such that $sort(x_i) = sort(u)$ for some $1 \leq i \leq n$, then $t[u/x_i] \equiv t[u'/x_i] \Leftarrow \vartheta \in Pre(AX)$. ∎

Definition The *expansion calculus* consists of three rules (given below) for transforming pairs consisting of a goal and a substitution. We assume that the variables of a goal subjected to a derivation step belong to a set *GV* of variables, which do not occur in axioms. If the step brings axiom variables into the goal, they must be renamed as variables of GV before the derivation continues.

Resolution Rule Let γ be a goal, p be an atom, $q \Leftarrow \vartheta \in AX$, f be a substitution and g be a unifier of p and q. Then $\langle \gamma \cup (p), f \rangle \vdash \langle (\gamma \cup \vartheta)[g], f[g]|GV \rangle$.

Paramodulation Rule Let δ be a goal, $x \in \text{single}(\delta)$, t be a term, $u \equiv u' \Leftarrow \vartheta$ (or $u' \equiv u \Leftarrow \vartheta$) $\in \text{Pre}(AX)$, f be a substitution and g be a unifier of t and u. Then

$$\langle \delta[t/x], f \rangle \vdash \langle (\delta[u'/x] \cup \vartheta)[g], f[g] | GV \rangle.$$

Unification Rule Let γ be a goal, f be a substitution and g be a unifier of terms t and t'. Then

$$\langle \gamma \cup \{t \equiv t'\}, f \rangle \vdash \langle \gamma[g], f[g] \rangle.$$

An *expansion* is a sequence $\langle \gamma_1, f_1 \rangle, \ldots, \langle \gamma_n, f_n \rangle$ of goal-substitution pairs such that for all $1 \le i < n$, $\langle \gamma_{i+1}, f_{i+1} \rangle$ is obtained from $\langle \gamma_i, f_i \rangle$ by applying a rule of the expansion calculus.

$\vdash_{EX}$ denotes the corresponding inference relation.∎

Theorem 4.1 ([Pad88a], Thm. 5.3.5) *Let γ be a goal and f be a substitution such that $var(\gamma) \cup dom(f) \subseteq GV$. Then $AX \vdash_C \gamma[f]$ if and only if $\langle \gamma, id \rangle \vdash_{EX} \langle \emptyset, f \rangle$.* ∎

$\vdash_{EX}$ uses only (prefixed) *axioms* to resolve or paramodulate upon. Cor. 2.3 allows us to apply *lemmas* as well, i.e., Pre(AX) can be extended to the set ITh(AX) of all inductive AX-theorems. For ground terms, Thm. 4.1 remains valid:

Corollary 4.2 *Let γ be a goal and f be a ground substitution such that $var(\gamma) \cup dom(f) \subseteq GV$. Then $AX \vdash_C \gamma[f]$ iff $\langle \gamma, id \rangle \vdash_{EX} \langle \emptyset, f \rangle$.* ∎

Suppose we have a set of expansions

$$\langle \gamma, id \rangle \vdash_{EX} \langle \emptyset, g_1 \rangle,$$
$$\langle \gamma, id \rangle \vdash_{EX} \langle \emptyset, g_2 \rangle,$$
$$\ldots$$

such that each ground substitution f is subsumed by some g_i. Then, by Cor. 4.2, γ is an inductive theorem. Instead of expanding $\langle \gamma, id \rangle$ into the empty goal one may stop in a situation like

$$\langle \gamma, id \rangle \vdash_{EX} \langle \delta_1, g_1 \rangle,$$
$$\langle \gamma, id \rangle \vdash_{EX} \langle \delta_2, g_2 \rangle,$$
$$\ldots$$

where $\langle \delta_1, g_1 \rangle, \langle \delta_2, g_2 \rangle, \ldots$ represents a *ground complete* case distinction.

Definition A set GS of goal-substitution pairs is *ground complete* if for all ground substitutions f there are $\langle \delta, g \rangle \in GS$ and a substitution h such that $AX \cup EAX \vdash \delta[h]$ and $g[h] | IN$ is AX-equivalent to $f | IN$. ∎

The combination of Cor. 4.2 with the characterization of constructor-based clause sets (Thm. 3.4) leads to

Corollary 4.3 *Let M be a constructor-based clause set such that for all $p \Leftarrow \gamma \in M$ there is a ground substitution f with $AX \vdash_C \gamma[f]$. M is an inductive AX-theorem iff there is a countable set of expansions*

$$\langle \{p_1\} \cup \gamma_1, id \rangle \vdash_{EX} \langle \delta_1, g_1 \rangle,$$
$$\langle \{p_2\} \cup \gamma_2, id \rangle \vdash_{EX} \langle \delta_2, g_2 \rangle,$$
$$\ldots$$

such that $M = \{p_1 \Leftarrow \gamma_1, p_2 \Leftarrow \gamma_2, \ldots\}$ and $\{\langle \delta_1, g_1 \rangle, \langle \delta_2, g_2 \rangle, \ldots\}$ is ground complete. ∎

For checking the ground completeness of $\{\langle \delta_1, g_1 \rangle, \langle \delta_2, g_2 \rangle, \ldots\}$ one may, again, refer to case matrices:

Proposition 4.4 A finite set $\{\langle \delta_1, g_1 \rangle, \ldots, \langle \delta_k, g_k \rangle\}$ of goal-substitution pairs is ground complete if the set

$$\delta_1 \cup \{x \equiv g_1 x \mid x \in IN\}$$
$$\ldots$$
$$\delta_k \cup \{x \equiv g_k x \mid x \in IN\}$$

is extendable to a case matrix. ∎

With Prop. 4.4, the ground completeness of a set of goal-substitution pairs is reduced to the ground completeness of term sets (cf. Sect. 5).

So far, the proof procedure involved in Cor. 4.3 does not employ induction steps. Consequently, infinitely many expansions will often be needed in order to obtain a ground complete set of goal-substitution pairs. As in corresponding forward proofs, only the explicit use of induction hypotheses may reduce the search space to a finite proof tree. But how do induction hypotheses enter the expansion calculus?

In principle, the idea is as classical as the step from bottom-up derivations to top-down expansions. We find it, for instance, in Manna and Waldinger's deductive tableaus used for program synthesis (cf. [MW80], [MW87]), especially in the "formation of recursive calls". It amounts to including Noetherian relations into the specification, which allow us to distinguish certain instances of a clause as induction hypotheses.

Definition A binary relation R on a set A is *Noetherian* or *well-founded* if there are no infinite sequences $a_1,a_2,a_3,...$ of elements of A such that for all $i\geq 1$, $\langle a_i,a_{i+1}\rangle \in R$. ∎

Here we are interested in relations on GT(SIG) which arise from a binary predicate >>, being part of the specification (SIG,AX).

Definition Let $s \in S$ and $>> \in PR_{s,s}$ (cf. Sect. 2). Then

$$R(>>) = \{(t,t') \in GT(SIG)^2 \mid AX \vdash_C t >> t'\}.$$ ∎

The Noetherian property of R(>>) can be reduced to *one* of its interpretations:

Proposition 4.5 *R(>>) is Noetherian iff there is a SIG-model A of AX such that* $>>^A$, *the interpretation of >> on A, is Noetherian.* ∎

General Assumption (part 2; cf. Sect. 2) We order a subset of IN, say $\{z_1,...,z_n\}$, into a sequence, say $z = (z_1,...,z_n)$, and assume a predicate symbol $>> \in PR_{s,s}$ such that R(>>) is Noetherian.

For avoiding name clashes we also use a primed copy of $\{z_1,...,z_n\}$. So let $z' = (z_1',...,z_n')$, and for all clause sets M, let M' be M with all variables replaced by their primed counterparts. ∎

R(>>) is compatible with AX-equivalence: If $AX \vdash_C \{t>>t', t\equiv u, t'\equiv u'\}$, then by congruence axioms for ≡, $AX \vdash_C u>>u'$. In particular, $AX \vdash_C \{t>>t', t\equiv t'\}$ implies $AX \vdash_C t'>>t'$, which means that R(>>) can only be well-founded if it is *disjoint* from AX-equivalence. Therefore, R(>>) cannot agree with a reduction ordering needed for inductive completion (cf. Sects. 1 and 7): a reduction ordering *contains* an "oriented" version of AX-equivalence. This does not contradict the fact that the definition of a reduction ordering may use (parts of) the "semantic" relation R(>>) (cf. the semantic path orderings in [Der87a]).

Semantic relations, which are compatible with AX-equivalence, on the one hand and reduction orderings on the other hand are employed for different purposes. The former are a means for ensuring that inductive proofs of semantic properties are sound. The latter guarantee a purely syntactic condition: the well-foundedness of rewrite sequences.

Now think of a *forward* proof of $q \Leftarrow \mathfrak{d}$ by using induction hypotheses. Usually, one reduces the set of all ground substitutions to a finite *covering*, say $\{f_1,...,f_n\}$, presupposes the validity of all premise instances $\mathfrak{d}[f_i]$ and infers the corresponding conclusion instances $q[f_i]$. In the course of deriving $q[f_i]$ from $\mathfrak{d}[f_i]$, an induction step replaces a ground instance of $\mathfrak{d}$, say $\mathfrak{d}[g]$, by $q[g]$, provided that gz is "less than" f_iz (see the General Assumption). In other words, the clause

$$q \Leftarrow \mathfrak{d} \cup \{f_iz>>z\} \qquad (*)$$

is regarded as an additional axiom: $q[g]$ is the result of cutting (*) with $\mathfrak{d}[g]$ and $f_iz>>gz$. Indeed, (*) represents an induction hypothesis.

The forward proof will succeed only if a suitable covering $\{f_1,...,f_n\}$ has been guessed and if no *generalization* is needed, i.e., if $q \Leftarrow \mathfrak{d}$ is strong enough for generating induction hypotheseses. The backward proof, on the other hand, which proceeds by resolution and paramodulation on axioms, lemmas and induction hypotheses leads more or less automatically both to a covering and to necessary generalizations.

Definition Let M be a clause set. The *inductive expansion calculus (for M)* consists of the expansion calculus and two additional rules:

Inductive Resolution Rule Let γ be a goal, p be an atom, $q \Leftarrow \vartheta \in M'$, f be a substitution and g be a unifier of p and q. Then

$$\langle \gamma \cup \{p\}, f\rangle \vdash \langle (\gamma \cup \vartheta \cup \{fz >> z'\})[g], f[g]\rangle.$$

Inductive Paramodulation Rule Let δ be a goal, x $\in$ single(δ), t be a term, $u \equiv u' \Leftarrow \vartheta$ (or $u' \equiv u \Leftarrow \vartheta$) $\in M'$, f be a substitution and g be a unifier of t and u. Then

$$\langle \delta[t/x], f\rangle \vdash \langle (\delta[u'/x] \cup \vartheta \cup \{fz >> z'\})[g], f[g]\rangle.$$

An application of the Inductive Resolution or Paramodulation Rule is called an *M-induction step.* An M-induction step is *closed* if the output variables of the hypothesis resolved or paramodulated upon are regarded as constants (and thus prevented from subsequent instantiations).

An *inductive M-expansion* is a sequence $\langle \gamma_1, f_1\rangle, \ldots, \langle \gamma_n, f_n\rangle$ of goal-substitution pairs such that for all $1 \le i < n$, $\langle \gamma_{i+1}, f_{i+1}\rangle$ is obtained from $\langle \gamma_i, f_i\rangle$ by applying a rule of the inductive expansion calculus. If all M-induction steps in the sequence are closed, the expansion is called a *closed inductive M-expansion.*

$\vdash_{EX(M)}$ denotes the corresponding inference relation. ∎

Sometimes several clauses can only be proved by simultaneous induction. Therefore let us generalize clauses to formulas $\psi \Leftarrow \gamma$ where ψ *and* γ are goals. $\psi \Leftarrow \gamma$ stands for the union of all $p \Leftarrow \gamma$ over all $p \in \psi$. As before, we use Greek letters for goals and small Latin letters for atoms.

The question remains whether *inductive* M-expansions are sound. As the reader might expect, this can be proved by Noetherian induction with respect to R(>>).

Lemma 4.6 *Let M be a constructor-based clause set. If for all ground substitutions f there are $\psi \Leftarrow \gamma \in M$, a substitution g and an inductive expansion $\langle \psi \cup \gamma, id\rangle \vdash_{EX(M)} \langle \mathcal{B}, g\rangle$ such that g/IN and f/IN are AX-equivalent, then M is an inductive AX-theorem.* ∎

Of course, Lemma 4.6 does not *characterize* the set of those constructor-based clause sets which are inductive theorems. The inductive rules involved in $\vdash_{EX(M)}$ depend on the predicate >>. Instead, the important fact we conclude from Lemma 4.6 is the possibility of carrying out induction steps in backward proofs as well as in forward proofs, with the aim of achieving a *finite* proof. Moreover, backward induction improves over forward induction because it leads to linear proofs without any second-order arguments.

Yet we must cope with the restriction to constructor-based clause sets M, in particular with the requirement that the set of premises of M be a case matrix. In turn, this implies that the predicates used in the case matrix must be specified completely, the positive as well as the negative cases.

In fact, the restriction can be weakened. We can also handle conditional clause sets of the form $M \Leftarrow \vartheta$ where M is a constructor-based clause set, ϑ is an input goal, i.e., var(ϑ) $\subseteq$ IN, and $M \Leftarrow \vartheta$ stands for the set of all clauses $\psi \Leftarrow \gamma \cup \vartheta$ with $\psi \Leftarrow \gamma \in M$. The proof of $M \Leftarrow \vartheta$ proceeds as an inductive M-expansion, with possible applications of the following inference rule:

Premise Elimination Rule Let γ be a goal and f be a substitution. Then for all $\delta \subseteq \vartheta$,

$$\langle \gamma \cup \delta[f], f\rangle \vdash \langle \gamma, f\rangle.$$

As an immediate consequence of Lemma 4.6, Cor. 4.3 holds true for $\vdash_{EX(M)}$ as well as for $\vdash_{EX}$. Moreover, Prop. 4.4 provides a criterion for checking the ground completeness of the final set $\{\langle \delta_1, g_1\rangle, \langle \delta_2, g_2\rangle, \ldots\}$ of goal-substitution pairs. In summary, this yields

Theorem 4.7 *Let M be a constructor-based clause set. M (or $M \Leftarrow \vartheta$; see above) is an inductive AX-theorem if there is a finite set of expansions*

$$\langle \psi_1 \cup \gamma_1, id\rangle \vdash_{EX(M)} \langle \delta_1, g_1\rangle,$$

$$\ldots$$

$$\langle \psi_n \cup \gamma_n, id\rangle \vdash_{EX(M)} \langle \delta_n, g_n\rangle$$

such that $M = \{\psi_1 \Leftarrow \gamma_1, \ldots, \psi_n \Leftarrow \gamma_n\}$ and

$$\delta_1 \cup \{x \equiv g_1 x \mid x \in IN\},$$
$$\ldots$$
$$\delta_n \cup \{x \equiv g_n x \mid x \in IN\}$$

is extendable to a case matrix, called the derived covering. ∎

Example 4.8 Two equations capture the correctness of insertion sort as specified in Section 2: T1 says that *sort* returns a sorted sequence. T2 ensures that the sorted sequence is a permutation of the original one.

```
sorted(sort(s))                                  (T1)
seqToBag(sort(s)) ≡ seqToBag(s)                  (T2)
```

With IN = {s}, {{T1,T2}} is a constructor-based clause set. One obtains three inductive {{T1,T2}}-expansions using two lemmata, namely:

```
sorted(insert(x,s)) ⇐ sorted(s)                  (L1)
seqToBag(insert(x,s)) ≡ seqToBag(x&s)            (L2)
```

	goal	*substitution*	*axioms and lemmas applied*
1	sorted(sort(s)) (T1) seqToBag(sort(s)) ≡ seqToBag(s) (T2)		
1.1	sorted(sort(ε)) seqToBag(sort(ε)) ≡ seqToBag(ε)	ε/s	
	sorted(ε) seqToBag(ε) ≡ seqToBag(ε)		IS1
	∎		SO1, unification
1.2	sorted(sort(x&s')) seqToBag(sort(x&s')) ≡ seqToBag(x&s')	x&s'/s	
	sorted(insert(x,sort(s'))) seqToBag(insert(x,sort(s'))) ≡ seqToBag(x&s')		IS2 IS2
	sorted(sort(s')) seqToBag(x&sort(s')) ≡ seqToBag(x&s')		L1 L2
	x&s' >> s' seqToBag(x&sort(s')) ≡ seqToBag(x&s')		T1 as induction hypothesis
	add(x,seqToBag(sort(s'))) ≡ add(x,seqToBag(s'))		GR, BA2
	x&s' >> s' add(x,seqToBag(s')) ≡ add(x,seqToBag(s'))		T2 as induction hypothesis
	∎		GR, unification

The first induction step in expansion 1.2 applies the Inductive *Resolution* Rule, while the second one is an application of the Inductive *Paramodulation* Rule: T2 is applied from left to right to the subterm seqToBag(sort(s')). The covering derived by expansions 1.1 and 1.2 is the set {{s≡ε}, {s≡x&s'}}, which is a case matrix because {ε,x&s'} is ground complete for s (cf. Ex. 5.4).

With IN = {x,s}, the following inductive {{sorted(insert(x,s))}}-expansions yield a proof of L1.

2	sorted(insert(x,s))		
2.1	sorted(insert(x,ε))	ε/s	
	sorted(x&ε)		IN1
	↯		SO2
2.2	sorted(insert(x,y&s'))	y&s'/s	
2.2.1	sorted(x&y&s') x ≤ y		IN2
	sorted(y&s') x ≤ y		SO3
	x ≤ y		premise elimination
	le(x,y) ≡ true		LE1
2.2.2	sorted(y&insert(x,s')) x > y		IN3
2.2.2.1	sorted(y&insert(x,ε)) x > y	ε/s'	
	sorted(y&x&ε) x > y		IN1
	sorted(x&ε) y ≤ x, x > y		SO2
	y ≤ x, x > y		SO1
	x > y		y ≤ x ⇐ x > y
	le(x,y) ≡ false		LE2
2.2.2.2	sorted(y&insert(x,z&s")) x > y	z&s"/s'	
2.2.2.2.1	sorted(y&x&z&s") x ≤ z, x > y		IN2
	sorted(x&z&s") y ≤ x, x ≤ z, x > y		SO3

	sorted(x&z&s") x ≤ z, x > y	y ≤ x ⇐ x > y
	sorted(z&s") x ≤ z, x > y	SO3
	sorted(y&z&s") x ≤ z, x > y	sorted(z&s) ⇐ sorted(y&z&s) (L3)
	x ≤ z, x > y	premise elimination
	le(x,z) ≡ true le(x,y) ≡ false	LE1 LE2
2.2.2.2.2	sorted(y&z&insert(x,s")) x > z, x > y	IN3
	sorted(z&insert(x,s")) y ≤ z, x > z, x > y	SO3
	sorted(insert(x,z&s")) y ≤ z, x > z, x > y	IN3 (from right to left)
	sorted(z&s") z&s" >> s" y ≤ z, x > z, x > y	L1 as induction hypothesis
	sorted(z&s") y ≤ z, x > z, x > y	GR
	sorted(y&z&s") y ≤ z, x > z, x > y	sorted(z&s) ⇐ sorted(y&z&s) (L3)
	sorted(y&z&s") x > z, x > y	y ≤ z ⇐ sorted(y&z&s) (L4)
	x > z, x > y	premise elimination
	le(x,z) ≡ false le(x,y) ≡ false	LE2 LE2

The covering derived by these expansions is

s ≡ ε			(2.1)
s ≡ y&s'	le(x,y) ≡ true		(2.2.1)
s ≡ y&ε	le(x,y) ≡ false		(2.2.2.1)
s ≡ y&z&s"	le(x,y) ≡ false	le(x,z) ≡ true	(2.2.2.2.1)
s ≡ y&z&s"	le(x,y) ≡ false	le(x,z) ≡ false	(2.2.2.2.2)

It is extendable to a case matrix by replacing s' with ε and z&s", respectively. Note that lemmas L3 and L4 constitute the inverse of SO3. They are inductive theorems, but do not hold in all term-generated models of SORT. As inverses of an axiom, L3 and L4 are consequences of the *closed world assumption* (cf. Sect. 1).

Finally, L2 is proved by inductive {{L2}}-expansions:

3	seqToBag(insert(x,s)) ≡ seqToBag(x&s) (L2)		
3.1	seqToBag(insert(x,ε)) ≡ seqToBag(x&ε)	ε/s	
	seqToBag(x&ε) ≡ seqToBag(x&ε)		IN1
	∅		unification
3.2	seqToBag(insert(x,y&s')) ≡ seqToBag(x&y&s')	y&s'/s	
3.2.1	seqToBag(x&y&s') ≡ seqToBag(x&y&s') x ≤ y		IN2
	x ≤ y		unification
	le(x,y) ≡ true		LE1
3.2.2	seqToBag(y&insert(x,s')) ≡ seqToBag(x&y&s') x > y		IN3
	add(y,seqToBag(insert(x,s'))) ≡ add(x,seqToBag(y&s')) x > y		BA2
	y&s' >> s' add(y,seqToBag(x&s')) ≡ add(x,seqToBag(y&s')) x > y		L2 as induction hypothesis
	add(y,add(x,seqToBag(s'))) ≡ add(x,add(y,seqToBag(s'))) x > y		GR, BA2
	x > y		BA3
	le(x,y) ≡ false		LE2

The covering derived by these expansions is a case matrix, namely:

s ≡ ε		(3.1)
s ≡ y&s'	le(x,y) ≡ true	(3.2.1)
s ≡ y&s'	le(x,y) ≡ false	(3.2.2) ∎

5. How to Prove the Ground Completeness of Term Sets

Theorem 4.7 provides a proof method for inductive theorems where case matrices are presupposed both at the beginning and at the end of the proof; at the beginning because we have to start out from a constructor-based clause set the important property of which is that its premises constitute a (constructor-based) case matrix; at the end because the final goal-substitution pairs must correspond to a (not necessarily constructor-based) case matrix.

Apart from syntactic conditions, a set of goals is a case matrix if it is built up from ground complete sets of output terms. As we mentioned in Section 3, the constructor condition can be derived immediately from the Church-Rosser property of AX. Ground completeness, however, is a condition that needs its own proof methods.

Theorem provers devote a considerable amount of work to checking that the functions used have been defined completely (cf. [BM79], [Hut86]). At first sight, this does not seem to be necessary for proving theorems. But most proofs are carried out by case reasoning and thus the question arises whether a case distinction is complete. When it is presented as a case matrix CM, the question just amounts to whether the right-hand sides of CM-equations "cover" the left-hand sides. This leads to a new verification problem where induction is needed again. However, one may run into a cycle when *this* proof is also based on a case distinction. The problem can be overcome by expressing *these* cases on a "lower level", in terms of a particular set of *base* terms. In consequence, ground terms must be *base-representable,* which is indeed a sort of functional completeness. (For dealing with partial functions, non-base-representable terms are admitted, too. However, for simplifying the presentation, we do not consider such cases here.)

Moreover, ground completeness is an *existential* statement.

Both deviations from the kind of theorems considered in previous sections call for a particular method for proving ground completeness.

Definition Let *BOP* ⊆ OP be a set of *base operations. GBT* denotes the set of ground *base* terms, i.e. ground terms over BOP. We assume that for all $s \in S$, GBT_S is nonempty. A ground term Ft is *innermost* if $F \notin BOP$ and t is a base term (tuple).

A term t is *base-representable* if for all $f \in GBT^X$ there is a base term that is AX-equivalent to t[f]. *BR* denotes the set of base-representable terms. Subsets of BR are called base-representable sets. A substitution is base-representable if f(X) is base-representable. ∎

Base-representability can be expressed in terms of a base existential theorem, i.e. a goal with existentially quantified variables:

Definition A goal ψ is a *base existential theorem* if for all $f \in GBT^X$ there is $g \in GBT^X$ such that $AX \vdash_C \psi[f|IN+g]$. ∎

Proposition 5.1 *Let GEN be a set of input terms such that each innermost term is subsumed by some t ∈ GEN. GBT is ground complete (cf. Sect. 3) or, equivalently, GEN is base-representable, if for some output variables* $x_1,...,x_n$, $\psi(GEN) = (t_i \equiv x_i \mid 1 \le i \le n)$ *is a base existential theorem.* ∎

The analogue of Lemma 4.6 for proving base existential theorems reads as follows.

Lemma 5.2 *Let* ψ *be a goal. If for all* $f \in GBT^X$ *there are* $g \in BR^X$ *and a closed inductive M-expansion* $\langle \psi, id \rangle \vdash_{EX(\psi)} \langle \emptyset, g \rangle$ *such that f|IN and g|IN are AX-equivalent, then* ψ *is a base existential theorem.* ∎

While Lemma 4.6 establishes the correctness of inductive expansion w.r.t. *universally* quantified clauses, the previous lemma deals with existential theorems and thus requires *closed* expansions where the existentially quantified output variables of induction hypotheses are not instantiated. This is necessary because an induction hypothesis assures a validating instantiation, but in general not the one *non*-closed expansions would generate.

A second deviation of Lemma 5.2 from Lemma 4.6 concerns the range of substitutions. Since 5.2 deals with *base* theorems and with expansions into a *base* case matrix (see below), the given substitution f is a base substitution and the derived substitution g must be base-representable.

Finite coverings of GBT^X should be given as base case matrices:

Definition A set C of terms is *ground base complete* if each ground base term is subsumed by some $c \in C$. A finite set CM of finite sets of equations is a *base case matrix with input* $IN_0 \subseteq X$ if either $CM = \{\emptyset\}$ or $CM = \{\{t \equiv c_i\} \cup \gamma \mid 1 \le i \le n, \gamma \in CM_i\}$ for a base-representable term t, a ground base complete set $\{c_1,...,c_n\}$ of output base terms, and base case matrices $CM_1,...,CM_n$ with input $IN_0 \cup var(c_1)$, ..., $IN_0 \cup var(c_n)$, respectively, such that $var(t) \subseteq IN_0$ and for all $1 \le i \le n$, $var(c_i) \cap IN_0 = \emptyset$. ∎

The following result is concluded from Lemma 5.2 just as Theorem 4.7 is derived from Lemma 4.6.

Theorem 5.3 *Let GEN be a set of input terms and* $\psi = \psi(GEN)$ *(cf. Prop. 5.1) such that each innermost term is subsumed by some t ∈ GEN. GBT is ground complete or, equivalently, GEN is base-representable, if there is a finite set of closed inductive* ψ*-expansions*

$$\langle \psi, id \rangle \vdash_{EX(\psi)} \langle \delta_1, g_1 \rangle,$$
$$\ldots$$
$$\langle \psi, id \rangle \vdash_{EI(\psi)} \langle \delta_n, g_n \rangle$$

such that $g_1,\ldots,g_n \in BR^X$ *and*

$$\delta_1 \cup \{x \equiv g_1 x \mid x \in IN\}$$
$$\ldots$$
$$\delta_n \cup \{x \equiv g_n x \mid x \in IN\}$$

is a base case matrix, called the derived covering. ∎

Example 5.4 When claiming that the coverings derived in Example 4.8 are case matrices we have assumed that the sets C1 = {ε,y&s'), C2 = {ε,x&ε,x&y&s'} and C3 = {true,false} are ground complete for the terms s and le(x,y), respectively. For justifying this statement with the help of Theorem 5.3 we choose {true, false, 0, _+1, ε, _&_, ∅, add(_,_)} as the set BOP of base operations. Innermost terms are, for instance, given by sorted(0&ε) and insert(0+1,ε).

Of course, if GBT is ground complete, then, in particular, C1, C2 and C3 are ground complete for s and le(x,y), respectively, as required. Suppose that the base-representability of GEN_0 = {le(x,y), seqToBag(s)} has already been shown. Let GEN_1 = {insert(x,s), sort(s)}. Since each innermost term is subsumed by some t ∈ $GEN_0 \cup GEN_1$, GBT is ground complete if and only if GEN_1 is base-representable. Hence by Thm. 5.3, it is sufficient to construct closed inductive expansions of T1 = {insert(x,s)≡s_o} and T2 = {sort(s)≡s_o}.

	goal	*substitution*	*axioms and lemmas applied*
1	insert(x,s)≡s_o (T1)		
1.1	∅	ε/s, x&ε/s_o	IN1
1.2	x ≤ y	y&s'/s, x&y&s'/s_o	IN2
	le(x,y) ≡ true		LE1
1.3	y&insert(x,s') ≡ s_o x > y	y&s'/s	IN3
	y&s_1 ≡ s_o y&s' >> s' le(x,y) ≡ false		T1 as induction hypothesis LE2
	le(x,y) ≡ false	y&s_1/s_o	unification, GR

The derived covering is the base case matrix {(s≡ε), (s≡y&s', le(x,y)≡true), (s≡y&s', le(x,y)≡false)}. (By assumption, le(x,y) ∈ GEN_0 is base-representable.) By Thm. 5.3, expansions 1.1-1.3 imply that insert(x,s) is base-representable. Note that the induction step in expansion 1.3 is closed because the (output) variable s_1 is not replaced later on.

2	sort(s)≡s_o (T2)		
2.1	∅	ε/s, ε/s_o	IS1
2.2	insert(x,sort(s')) ≡ s_o	x&s'/s	IS3

$\mathtt{insert}(x,s_1) \equiv s_0$		T2 as induction hypothesis
$\emptyset$	$\mathtt{insert}(x,s_1)/s_0$	unification, GR

The derived covering is the base case matrix is $\{(s \equiv \varepsilon), (s \equiv x \& s')\}$. Since $\mathtt{insert}(x,s_1)$ is base-representable, we conclude from Thm. 5.3 that $\mathtt{sort}(s)$ is base-representable, too. ∎

6. Conclusion

We have presented a calculus for proving inductive theorems by resolving and paramodulating upon axioms, lemmas and induction hypotheses. The theorems must be given as constructor-based sets of Horn clauses. The derivations end up with a ground complete set of goal-substitution pairs. As a criterion for ground completeness, we introduced the notion of a case matrix, which reduces the completeness requirement from goal-substitution pairs to term sets. Constructors and ground complete term sets are the only non-syntactical notions associated with inductive expansion. As to ground completeness, we have shown in Section 5 how this property can be proved with the help of closed inductive expansions. As to constructors, one may refer to *goal reduction*, which extends term rewriting to a rule for transforming goals:

Reduction Rule Let δ be a goal, $x \in \mathrm{single}(\delta)$, $u \equiv u' \Leftarrow \vartheta \in AX$ and f be a substitution. Then
$\delta[u[f]/x] \vdash \delta[u'[f]/x] \cup \vartheta[f]$.

A goal reduction stops successfully if a goal consisting of reflexive equations has been obtained:

Success Rule Let γ be a goal consisting of equations of the form $t \equiv t$ such that the Reduction Rule is not applicable to γ. Then $\gamma \vdash \emptyset$.

$\vdash_R$ denotes the corresponding inference relation. AX is called *Church-Rosser* if all proofs using the cut calculus have a "reduction counterpart", i.e., $AX \vdash_C \gamma$ implies $\gamma \vdash_R \emptyset$. The literature is full of criteria for the Church-Rosser property (cf., e.g., [Pad88a]). It yields the following constructor criterion: If AX is Church-Rosser on ground goals and no left-hand side of a conditional equation of AX "overlaps" a term of a term set T, then T is a set of constructors.

Another consequence of the Church-Rosser property is the possibility of restricting paramodulation to the more effective rule of *narrowing*, invented by [Lan75]. Indeed, the crucial Lemma 4.6 could also be based upon rules different from resolution and paramodulation, provided that they are complete in the sense of Thm. 4.1 (perhaps only for a particular class of specifications, like Church-Rosser ones.) More detailed suggestions concerning this line of developing inductive proof methods are given in [Pad88e]. It must be noted, however, that every restriction of the inference rules might prevent induction hypotheses from being generated. For instance, narrowing does not admit applying an equation from right to left as we did in expansion 2.2.2.2.2 of Example 4.8. But this application was necessary for proceeding with an L1-induction step.

Inductive completion, the current alternative for proving inductive theorems (cf. Sect. 1), is also based upon the Church-Rosser property. In spite of the resemblance between narrowing steps and the basic steps of inductive completion, i.e., the construction of *critical pairs* (pointed out in [Der87b], Sect. 4.2), there is an important difference between "inductive narrowing" and inductive completion. In the latter case, the Church-Rosser property is extended from the axioms to the conjecture that is to be proved. In fact, sophisticated Church-Rosser criteria take into account the special role of the conjecture (cf., e.g., [Fri86], [Küc87], [HK88], [Pad88d]). Nevertheless, many examples have shown that the remaining conditions are more difficult to establish than the constructor-based clause set requirement.

References

[BM79] R.S. Boyer, J.S. Moore, A Computational Logic, Academic Press (1979)

[Bur69] R.M. Burstall, Proving Properties of Programs by Structural Induction, Comp. J. 12 (1969) 41-48

[Der87a] N. Dershowitz, Termination of Rewriting, J. Symbolic Comp. 3 (1987) 69-115

[Der87b] N. Dershowitz, Completion and its Applications, Report (1987)

[Fri 86] L. Fribourg, A Strong Restriction of the Inductive Completion Procedure, Proc. ICALP '86, Springer LNCS 226 (1986) 105-115

[GG88] S.J. Garland, J.V. Guttag, Inductive Methods for Reasoning about Abstract Data Types, Proc. POPL '88 (1988) 219-228

[GHM88] A. Geser, H. Hußmann, A. Mück, A Compiler for a Class of Conditional Term Rewriting Systems, Proc. Conditional Term Rewriting Systems '87, Springer LNCS 308 (1988) 84-90

[HH82] G. Huet, J.M. Hullot, Proofs by Induction in Equational Theories with Constructors, J. Comp. and Syst. Sci. 25 (1982) 239-266

[HK88] D. Hofbauer, R. Kutsche, Proving Inductive Theorems Based on Term Rewriting Systems, Proc. Algebraic and Logic Programming, Math. Research 49, Akademie-Verlag Berlin (1988) 180-190

[HR87] J. Hsiang, M. Rusinowitch, On Word Problems in Equational Theories, Proc. ICALP '87, Springer LNCS 267 (1987) 54-71

[Hut86] D. Hutter, Using Resolution and Paramodulation for Induction Proofs, Proc. 10th GWAI, Springer Informatik-Fachberichte 124 (1986) 265-276

[JK86] J.-P. Jouannaud, E. Kounalis, Automatic Proofs by Induction in Equational Theories without Constructors, IEEE Symp. Logic in Comp. Sci. (1986) 358-366

[KM87] D. Kapur, D.R. Musser, Proof by Consistency, Artificial Intelligence 31 (1987) 125-157

[Küc87] W. Küchlin, Inductive Completion by Ground Proof Transformation, Proc. Resolution of Equations in Algebraic Structures, Austin (1987)

[Lan75] D.S. Lankford, Canonical Inference, Report ATP-32, Univ. of Texas at Austin (1975)

[MW80] Z. Manna, R. Waldinger, A Deductive Approach to Program Synthesis, ACM TOPLAS 2 (1980) 90-121

[MW87] Z. Manna, R. Waldinger, How to Clear a Block: A Theory of Plans, J. Automated Reasoning 3 (1987) 343-377

[Pad87] P. Padawitz, ECDS - A Rewrite Rule Based Interpreter for a Programming Language with Abstraction and Communication, Report MIP-8703, Univ. Passau (1987)

[Pad88a] P. Padawitz, Computing in Horn Clause Theories, EATCS Monographs on Theor. Comp. Sci. 16, Springer (1988)

[Pad88b] P. Padawitz, Can Inductive Proofs be Automated? EATCS Bulletin 35 (1988) 163-170

[Pad88c] P. Padawitz, Program Verification Revisited (1988), submitted

[Pad88d] P. Padawitz, Proof by Consistency of Conditional Equations (1988), submitted

[Pad88e] P. Padawitz, Reduction and Narrowing for Horn Clause Theories (1988), submitted

[Rei78] R. Reiter, On Closed World Data Bases, in: H. Gallaire, J. Minker, eds., Logic and Data Bases, Plenum Press, New York (1978) 55-76

[Rob65] J.A. Robinson, A Machine-Oriented Logic Based on the Resolution Principle, J. ACM 12 (1965) 23-41

[RW69] G. Robinson, L. Wos, Paramodulation and Theorem-Proving on First-Order Theories with Equality, in: Machine Intelligence 4, Edinburgh Univ. Press (1969) 135-150