Inductive Theorem Proving
for Design Specifications

January 2, 1997

Peter Padawitz
Fachbereich Informatik, Universität Dortmund, Germany
peter@ls5.informatik.uni-dortmund.de

Abstract

We present a number of new results on inductive theorem proving for design specifications based on Horn logic with equality. Induction is explicit here because induction orderings are supposed to be part of the specification. We show how the automatic support for program verification is enhanced if the specification satisfies a bunch of rewrite properties, summarized under the notion of canonicity. The enhancement is due to inference rules and corresponding strategies whose soundness is implied by the specification's canonicity. The second main result of the paper provides a method for proving canonicity by using the same rules, which are applied in proofs of conjectures about the specification and the functional-logic programs it contains.

Contents

1 Introduction

This paper presents a summary as well as extensions of main results of our monograph *Deduction and Declarative Programming* [33], which deals with a number of program and proof issues centering around the following paradigmatic equations of functional-logic programming:

$$\begin{gathered} \textit{programs} = \textit{axioms} = \textit{Horn clauses} \\ \textit{requirements} = \textit{theorems} = \textit{Gentzen clauses} \end{gathered}$$

Not only functional-logic programs, but also imperative ones are amenable to logic-oriented design methods, provided that the programming language has a kind of declarative semantics. Of course, Horn clauses $g \Leftarrow h$ are not sufficient as a formal setting for all requirements to a program. Hence we admit *Gentzen clauses,* which generalize Horn clauses in that a conclusion of a Gentzen clause has the form

$$\exists X_1 g_1 \vee \ldots \vee \exists X_n g_n$$

with $X_i, 1 \leq i \leq n$, being a set of variables occurring in the goal (= set of atoms) g_i[1]. The use of Horn clauses for programs and Gentzen clauses for requirements complies well with the *initial semantics* that implicitly underlies any design specification.

It is well-known that conjunctions of Horn clauses are the most general first-order formulas for which an initial model always exists. This is not just *a* model, it is *the* model, which the programmer, more or less consciously, has in mind when designing a data type. Since the initial model is a single model (up to isomorphism) and not a class of models, we need not restrict ourselves to Horn clauses when defining the theory of a design specification. On the contrary, many non-Horn Gentzen clauses are direct consequences of the *closed world assumption* of initial semantics (cf. [37]; [33], Section 1.5). These clauses often provide crucial lemmas used in program verification.

1.1 Expander

Some of the results presented in this paper yield the theoretical foundation of several inference rules implemented in Expander (cf. [34]), which is mainly a proof checker tailored to abstract data types and declarative programs. Using Expander, a proof of a Gentzen clause $gs \Leftarrow hs$[2] is carried out interactively. One starts with singleton lists $front = [gs]$ and $rear = [hs]$ and extends them stepwise into lists

$$front = [gs, gs_1, \ldots, gs_k] \quad \text{and} \quad rear = [hs, hs_1, \ldots, hs_n],$$

consisting of successively inferred goal sets. *front* is a backward proof of the clause $gs \Leftarrow gs_k$, while *rear* is a forward proof of $hs_n \Leftarrow hs$. The proof of $gs \Leftarrow hs$ is complete if there are $k, n \geq 1$ such that hs_n *subsumes* gs_k. This syntactical condition implies that $gs_k \Leftarrow hs_n$ is inductively valid (cf. [34], Section 3.5). From the validity of $gs \Leftarrow gs_k$, $gs_k \Leftarrow hs_n$ and $hs_n \Leftarrow hs$ one concludes that the original conjecture $gs \Leftarrow hs$ holds true. Some of the inference rules used for building up *front* and *rear* are sound only if the underlying specification is *canonical* (cf. 1.4).

1.2 Proof by term rewriting

If we forget the *non*-equality predicates that a Horn clause specification SP may include, SP can be regarded as a *conditional term rewriting system* (a CTRS for short; cf. e.g. [23], [43], [5]). In fact, fundamental notions

[1] If the existentially quantified variables of a Gentzen clause are known from the context, we sometimes denote $\exists X_1 g_1 \vee \ldots \vee X_n g_n$ by the *goal set* $gs = \{g_1, \ldots, g_n\}$ (cf. Section 2).

[2] $gs \Leftarrow hs$ is an abbreviation of the conjunction over all $gs \Leftarrow h$ with $h \in hs$.

and results from CTRS theory have been integrated into our approach to inductive theorem proving. But since our application area is not classical algebra, but the verification of declarative programs on constructor-based data types, we had to generalize the notions and results used in the CTRS community. These generalizations have been worked out in [33], Chapters 6 and 7. In particular, Sections 7.4 and 7.5 of [33] are devoted to a detailed comparison between *inductive completion* or *proof by consistency* (cf. [25]; [5], Section 8.5; [8], Section 7.3) and our method of inductive expansion. Given a CTRS R, inductive completion mixes the derivation of theorems from R with a proof that R is ground confluent. Inductive expansion separates both proof obligations from each other, which makes the proofs more transparent and allows us to drop certain serious restrictions of inductive completion. We discuss this matter in more detail at the end of Section 7.

For readers familiar with CTRS theory, we sketch the main generalizations introduced in [33], Sections 6.1 - 6.3. At first, a set of Horn clauses viewed as a CTRS may introduce *fresh* or *extra variables* (cf. [17], Section 2.4). Fresh variables are indispensable when term rewriting systems shall represent non-trivial declarative programs. At second, a set of Horn clauses may be strongly terminating in the sense of [33] even if it is neither reductive (cf. [22]) nor simplifying (cf. [24]) nor decreasing (cf. [7]) nor quasi-reductive (cf. [4]). For a Horn clause $p \Leftarrow g$ to be strongly terminating in the sense of [33] it is sufficient that only those ground instances $g\sigma$ of g are smaller than $p\sigma$, which are *convergent* or joinable (cf. [7]). At third, a CTRS may be ground confluent even if it generates critical pairs, which are neither feasible (cf. [24]) nor joinable. What we really may assume about each ground instance g of the premise of a critical pair is that g is not only convergent, but *strongly* convergent, i.e. that all reducts of g are convergent.

Examples taken from typical design specifications have motivated these generalizations. At the end of [33], Section 6.2, we claimed that only a strongly terminating CTRS can be proved ground confluent because only a reduction ordering allows us to induce from the premise to the conclusion of a conditional rule. In contrast to this conjecture, [6], Thm. 4, tells us that the well-foundedness of the rewrite relation is sufficient for confluence provided that all feasible critical pairs are *convergent overlays*. Fortunately, a feasible critical pair cp generated by a design specification is almost always an overlay. However, the proof of cp's convergence often depends on the assumption that the ground instances of cp's premise are strongly convergent (cf. [33], Example 6.11).

1.3 Proof by expansion

In [33], Sections 5.4 and 6.4, we have shown the soundness of two calculi, **inductive expansion** for proving inductive theorems and **subreductive expansion** for proving ground confluence. A second view on both calculi and their soundness proofs reveals their strong similarity. Each calculus is hierarchical in the sense that its rules apply Gentzen clauses, which are already known to be valid with respect to the same theory for which the calculus is sound. The main difference between inductive expansion and subreductive expansion is the set TH of ground goals where the respective validity notion is based upon. In the first case, TH is given by *all* valid ground goals. In the second case, TH consists of all strongly convergent ground goals (see above). But the set $Gen(TH, GS)$ of Gentzen clauses for which inductive (repectively subreductive) expansion is sound is defined the same in both cases: $gs \Leftarrow h \in Gen(TH, GS)$ iff for all ground substitutions σ, $h\sigma \in TH$ implies $g\tau \in TH$ for some $g \in gs$ and a ground substitution τ, which agrees with σ on all universally quantified variables of $gs \Leftarrow h$. Besides TH we must also parameterize the set of ground substitutions where σ is taken from. In the case of inductive expansion σ is arbitrary, but in the case of subreductive expansion σ is taken from a proper subset of the set of all ground substitutions.

To sum up, inductive and subreductive expansion are two instances of a **generic expansion calculus** we introduce and prove sound with respect to $Gen(TH, GS)$ in Section 3. Section 4 presents inductive expansion as an instance of the generic calculus. Section 6 does the same for subreductive expansion.

The generic calculus has a number of advantages. Firstly, one gains more insight into the exact applicability conditions of expansion rules. Secondly, we mentioned that Expander (cf. 1.1) realizes inductive expansion. If the derived subgoals are *bounded* (cf. Section 6), an inductive expansion is also a subreductive expansion. Hence Expander is also a tool for proving ground confluence. Thirdly, the subreductive instance of the generic calculus is even more powerful than the corresponding calculus defined in [33]. Here it includes inductive rules and thus we may prove ground confluence by induction. In CTRS terms, this means that we may use a critical pair as an induction hypothesis when proving that this critical pair is convergent. This is *not* inductive completion (cf. 1.2) where the subreductive expansion given by the process of joining a critical pair *cp* may not employ *cp* as an induction hypothesis. Theorem 6.6 justifies the method for proving ground confluence by building subreductive expansions of critical clauses.

1.4 Canonical specifications

What do we gain from a design specification SP with ground confluent axioms? Life becomes easier because of the following proof-theoretical implications of ground confluence.

1. **Base consistency.** If SP is built up in a hierarchical manner, we must ensure consistency with respect to the base specifications of SP, i.e. each ground goal over the base signature should follow from SP-axioms only if it already follows from base axioms. An almost syntactical consistency criterion is obtained if SP is ground confluent and strongly terminating (cf. [33], Corollary 6.19). Base consistency is crucial for proving program equivalence (cf. [33], Theorem 3.17) or the correctness of specification refinements (cf. [33], Section 7.6).

2. **Constructors.** Certain inference rules rely on the distinction of the set of operations of SP into a set of *constructors* and a set of *defined functions*. Terms built up of constructors are called *normal forms*. Ground normal forms yield the actual representations of data specified by SP. *SP has free constructors* if these representations are unique (cf. 5.2). The generalization of CTRS to include fresh variables (cf. 1.2) leads to reduction and narrowing calculi where fresh variables are only replaced by normal forms (cf. Section 5). If SP has free constructors, then the goals of a derivation can be simplified more efficiently, which results in shorter and more understandable proofs (cf. 1.5).

3. **Refutation.** Testing a declarative program means solving a goal g with respect to SP. Moreover, a negative answer to the question *Is g solvable?* is an affirmative answer to the question *Is $\neg g$ valid?* The *narrowing calculus* (cf. Section 8) mostly gives us an answer, provided that SP is ground confluent.

4. **Proof by narrowing.** For the topic of this paper the most important consequence of ground confluence is the soundness of *narrowing-strategy-controlled goal generation* as an inference rule used in proofs of Gentzen clauses (cf. Section 8). We mentioned that Expander proves a Gentzen clause $gs \Leftarrow hs$ by stepwise transforming gs and hs until hs subsumes gs (cf. 1.1). Narrowing steps are always sound in the backward transformation of gs, but SP must be ground confluent if they are used in the forward transformation of hs. Indeed, this part of the proof is often facilitated considerably by narrowing steps because they produce case analyses automatically.

Ground confluence is related to strong termination because the method of proving ground confluence by subreductive expansion (cf. 1.3) is correct only if SP is strongly terminating (cf. Section 6). Hence Section 7 recapitulates the *path calculus* for proving strong termination, which was introduced in [33], Section 6.2. This calculus generates a reduction ordering both from a syntactical signature ordering and from well-founded semantic relations of SP.

A specification SP is called *canonical* if it is ground confluent, strongly terminating and *normal form complete*, i.e. if all ground terms have normal forms. The precise definitions are given in Section 5.

1.5 Simplification

Sections 8 to 10 review and extend results of [30] and [32], which deal with narrowing, narrowing strategies and *simplification*. Simplification steps are automatically performed equivalence transformations, which make expansion proofs more transparent and more efficient. Moreover, they are combined with narrowing without loosing the completeness properties of narrowing provided that the underlying simplifier is *reductive*, i.e. monotonic with respect to the reduction ordering that makes the specification strongly terminating (cf. 1.4). Theorem 10.6 will generalizes all previous results on reduced or normalizing narrowing (cf. [12]; [38]; [30], Section 8.7; [18], Section 6.5.2; [14], Chapter 5; [17]) where simplification is confined to rewriting terms into reduced ones. For instance, the simplifier of Expander (cf. 1.1) realizes goal set normalization, the partial evaluation of standard functions and predicates, β-reduction of λ-expressions and continuation passing (cf. [34], Section 4). The simplifier may carry out various program transformations and thus control tests or proofs of a program in a similar way a compiler controls its execution.

Although Expander simplifies the reduct of each derivation step, we do not attach simplification to the inference rules themselves such as in, e.g., *rewriting modulo* a theory (cf. [20]) or an algebra (cf. [2]). A deduction step may properly weaken or strengthen a formula, while a simplifier always transforms formulas into a (inductively) equivalent ones. Hence, by simplifying the reduct of an inference step, the soundness of the applied inference rule is not affected. For the sake of flexible and transparent theorem proving we clearly separate deduction rules, which are fixed, but applied through user interaction, from the simplifier, which is applied automatically, but amenable to modifications due to particular base theories. This differs from *theory resolution* (cf. [41]) and *rewriting modulo* where theories or models are "built into" inference rules. The theory that is built into *our* rules is always the inductive theory, i.e. the theory of the initial model of the specification we actually deal with. This means that we may apply lemmas as in "natural proofs", but this has nothing to do with built-in normalization or simplification.

Simplifiers and the (weak) conditions they must satisfy in order to comply with inference rules, especially with strategy-controlled goal generation (cf. 1.4), are treated in Section 10.

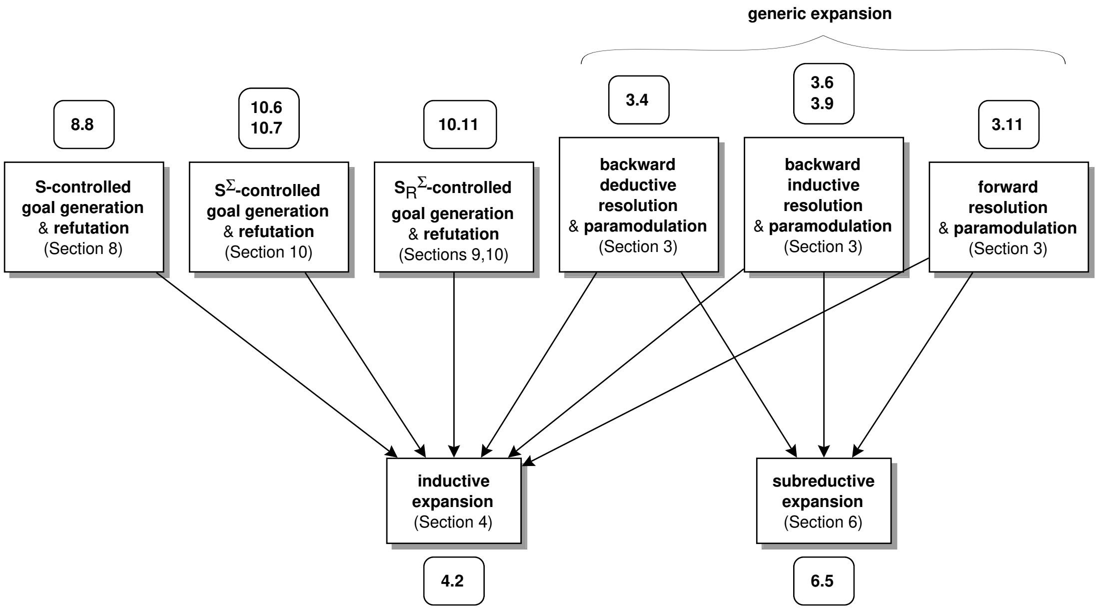

Figure 1. Generic, inductive and subreductive expansion

Fig. 1 summarizes the rules of inductive and subreductive expansion and the numbers of the results showing their soundness.

2 Preliminaries

We assume some familiarity with the basic notions of algebraic specification and Horn logic (cf., e.g., [15], [11], [30], [42] or [33]) and briefly recapitulate basic notations used in this paper.

We fix a set **S** of sorts, an infinite S-sorted set **X** of variables and an S-sorted signature **SIG = (S,OP,PR)** with a set OP of operation or function symbols and a set PR of predicate symbols. For each sort s, an **equality predicate**, denoted by $\equiv$, implicitly belongs to SIG. All other predicates of SIG are called **logical predicates**. $\mathbf{T_{SIG}(X)}$ $(\mathbf{T_{SIG}})$ denotes the S-sorted set of (ground) SIG-terms over X. SIG is assumed to be **inhabited**, i.e. for all sorts s there is a ground term of sort s. Atom(ic formula)s over SIG and X are defined as usual. Equations $t \equiv u$ between terms t and u are particular atoms. A binary relation R on $T_{SIG}(X)$ is SIG-**compatible** if for all $t = F(t_1, \ldots, t_n), u = F(u_1, \ldots, u_n) \in T$, $(t_1, u_1), \ldots, (t_n, u_n) \in R$ implies $(t, u) \in R$. A SIG-**congruence** is a SIG-compatible equivalence relation.

Sub (GSub) denotes the set of (ground) substitutions over SIG, i.e. S-sorted functions from X to $T_{SIG}(X)$ (T_{SIG}). As usually, we write $x\sigma$ for $\sigma(x)$. For all $\sigma \in Sub$, $\mathbf{dom}(\sigma) =_{def} \{x \in X \mid x\sigma \neq x\}$. $[t/x]$ denotes the substitution σ defined by $x\sigma = t$ and $dom(\sigma) = \{x\}$. Let $\sigma, \tau \in Sub$ such that for all $x \in dom(\sigma) \cap dom(\tau)$, $x\sigma = x\tau$. Then $x(\sigma + \tau) =_{def} x\sigma$ for all $x \in dom(\sigma)$ and $x(\sigma + \tau) =_{def} x\tau$ for all $x \in X \setminus dom(\sigma)$. **id** denotes the substitution σ with $dom(\sigma) = \emptyset$.

Let $V, Z \subseteq X$ and $\sigma, \tau \in Sub$. Then $V\sigma =_{def} \{x\sigma \mid x \in V\}$ and

$$\mathbf{EQ}(\sigma) \ =_{def} \ \{x \equiv x\sigma \mid x \in dom(\sigma)\}.$$

σ is a **renaming of** V **away from** Z if $V\sigma \subseteq X$, $|V\sigma| = |V|$ and $V\sigma \cap Z = \emptyset$. $\sigma =_{\mathbf{V}} \tau$ means $\sigma(x) = \tau(x)$ for all $x \in V$. The substitution $\sigma_{\mathbf{V}}$ is defined by $dom(\sigma_V) \subseteq V$ and $\sigma_V =_V \sigma$. We also write σ for the unique SIG-homomorphic extension σ^* of σ to T_{SIG}. Hence for all $x \in X$, $x\sigma\tau = \tau^*(x\sigma)$. A term t **matches** or **subsumes** a term u, written $t \leq u$, if $t\sigma = u$ for some $\sigma \in Sub$. Then u is the **instance of** t **by** σ. σ subsumes τ, written $\sigma \leq \tau$, if $\sigma\rho = \tau$ for some $\rho \in Sub$. σ **unifies** the terms t and u if $t\sigma = u\sigma$. σ is a **minimal** or **most general unifier** of t and u if σ unifies t and u and subsumes all unifiers of t and u.

A **goal** over SIG is a set of atoms over SIG, which stands for the conjunction of its elements, or the **contradictory goal** $FALSE$. A goal without variables is **ground**. Given $V \subseteq X$ and a goal g, the expression $\exists Vg$ is called an **existential goal**. The elements of V resp. $X \setminus V$ are called **existential** resp. **universal variables of** g. For a substitution σ, the goal $(\exists Vg)\sigma$ is defined as $\exists Vg\sigma_{X\setminus V}$.

Given $V \subseteq X$ and a goal g, the expression $\forall Vg$ is called a **universal goal**, which is just an abbreviation of the (infinite) goal $\bigcup\{g\sigma \mid \sigma$ is a ground substitution over SIG such that $dom(\sigma) = V\}$. Actually, the variables of V in $\forall Vg$ do not exist. Hence they are regarded neither as existential nor as universal variables of g.

A **goal set** is a non-empty set of existential goals, which stands for the disjunction of its elements. The goal set $gs \cup \{FALSE\}$ is identified with gs.

Given a goal set gs and a universal goal h, an expression $gs \Leftarrow h$ is called a **Gentzen clause** over SIG. If $h = \emptyset$, we write gs instead of $gs \Leftarrow h$. If gs consists of a single non-empty universal goal g, then $gs \Leftarrow h$ is a **Horn clause** and we write $g \Leftarrow h$ instead of $gs \Leftarrow h$. $g \Leftarrow h$ is a **conditional equation** if $g = \{t \equiv t'\}$ for some terms t, t'. Otherwise $g \Leftarrow h$ is called a **non-equational Horn clause**. Given a further goal set hs, the clause $gs \Leftarrow hs$ is an abbreviation of the set $\{gs \Leftarrow h \mid h \in hs\}$ of Gentzen clauses.

The set of all variables occurring in a Gentzen clause c is denoted by $\mathbf{var(c)}$. If c is a conditional equation, say $c = t \equiv u \Leftarrow h$, $x \in var(c)$ is called a **fresh** or **extra variable** of c if x occurs in u or h, but not in t. If c

is a non-equational Horn clause, say $c = g \Leftarrow h$, $x \in var(c)$ is a fresh variable of c if x occurs in h, but not in g. **fresh(c)** denotes the set of fresh variables of c.

A **specification** is a pair (SIG, AX) consisting of a signature SIG and a set AX of Horn clauses. A specification SP to be processed by Expander (cf. 1.1) may also include a `theorems` section containing Gentzen clauses representing lemmas or constraints used in proofs of those Gentzen clauses, which are listed in a `conjectures` section of SP (cf. Examples 4.2, 4.3 and 7.4).

3 Generic expansion

In the sequel, we separate a finite set $\mathbf{X_{in}}$ of **input variables** from a finite set $\mathbf{X_{out}}$ of **output variables**. Given a proof of a Gentzen clause c, input variables are the universally quantified variables of c, output variables are the existentially quantified variables of c and all variables, which stem from an axiom or lemma used in the proof of c. These imported variables are automatically existentially quantified. For all $\sigma \in Sub$, let $\sigma_{\mathbf{in}} = \sigma_{X_{in}}$ and $\sigma_{\mathbf{out}} = \sigma_{X_{out}}$. For a clause c, **in(c)** denotes the set of input variables of c.

The generic expansion calculus depends on two parameters: a set **TH** of ground atoms and a set **GS** of ground substitutions such that

- TH is symmetric, i.e. for all ground terms t and u, $t \equiv u \in TH$ implies $u \equiv t \in TH$,
- for all $\sigma \in GS$ and $\tau \in GSub$, $\sigma_{in} + \tau_{out} \in GS$.

Definition 3.1 A Gentzen clause $c = gs \Leftarrow h$ **follows from TH by induction on GS** if for all $\sigma \in GS$, $h\sigma \subseteq TH$ implies $g\tau \subseteq TH$ for some $g \in gs$ and $\tau \in GSub$ with $\tau =_{in(c)} \sigma$. **Gen(TH,GS)** denotes the set of all Gentzen clauses that follow from TH by induction on GS. ❑

Note that TH is always a subset of $Gen(TH, GS)$.

Definition 3.2 A goal $g[t/x]$ is **(TH,GS)-equivalence compatible** if for all terms u, the clause

$$g[u/x] \;\Leftarrow\; g[t/x] \cup \{t \equiv u\}$$

follows from TH by induction on GS. ❑

Proposition 3.3 Let $\sigma \in Sub$ and the goal $g[t/x]\sigma$ be (TH, GS)-equivalence compatible. Then

$$g[t/x]\sigma_{out} \;\Leftarrow\; g[u/x]\sigma \cup \{t\sigma \equiv u\sigma\} \cup EQ(\sigma_{in})$$

follows from TH by induction on GS. ❑

Given TH and GS as above, the two non-inductive rules of the generic calculus are defined as follows. Both rules transform goal sets and thus perform steps in the backward part of a proof of a Gentzen clause (cf. 1.1).

Backward deductive resolution Let $\sigma \in Sub$ and $\{\exists X_1 h_1\sigma, \ldots, \exists X_n h_n\sigma\} \Leftarrow h \in Gen(TH, GS)$ such that for all $1 \le i \le n$, $h_i\sigma$ is (TH, GS)-equivalence compatible and $X_i \cap var(g_i\sigma \cup X_{in}\sigma \cup X_{in}) = \emptyset$.

$$\frac{\{g_1 \cup h_1, \ldots, g_n \cup h_n\}}{\{g_1\sigma \cup \ldots \cup g_n\sigma \cup h \cup EQ(\sigma_{in})\}}$$

Backward deductive paramodulation Let $\sigma \in Sub$, $\{\exists X_1(t \equiv t_1)\sigma, \ldots, \exists X_n(t \equiv t_n)\sigma\} \Leftarrow h \in Gen(TH, GS)$ and $x \in var(g_1 \cap \ldots \cap g_n)$ such that for all $1 \le i \le n$, $X_i \cap var(g_i[t/x]\sigma \cup X_{in}\sigma \cup X_{in}) = \emptyset$.

$$\frac{\{g_1[t_1/x], \ldots, g_n[t_n/x]\}}{\{g_1[t/x]\sigma \cup \ldots \cup g_n[t/x]\sigma \cup h \cup EQ(\sigma_{in})\}}$$

Theorem 3.4 (backward deductive rules are sound) Let the goal set hs be obtained from the goal set $gs = \{g_1, \ldots, g_m\}$ by a single deductive resolution or paramodulation step such that gs and hs consist of (TH, GS)-equivalence compatible goals. Then the clause $gs \Leftarrow hs$ follows from TH by induction on GS.

Proof. W.l.o.g. suppose that hs is a singleton, say $hs = \{h\}$. Let $\tau \in GS$ such that $h\tau \subseteq TH$. We must infer

(1) $\qquad g_i(\tau_{in} + \xi_{out}) \subseteq TH$ for some $1 \leq i \leq m$ and $\xi \in GSub$.

We consider both rules separately.

Case 1. h is obtained from gs by a resolution step. Then

$$gs \;=\; \{(g_1' \cup h_1), \ldots, (g_n' \cup h_n)\}$$

and

$$h \;=\; g_1'\sigma \cup \ldots \cup g_n'\sigma \cup f \cup EQ(\sigma_{in})$$

such that $\sigma \in Sub$ and $\{\exists X_1 h_1\sigma, \ldots, \exists X_n h_n\sigma\} \Leftarrow f \in Gen(TH, GS)$ such that for all $1 \leq i \leq n$, $h_i\sigma$ is (TH, GS)-equivalence compatible and $X_i \cap var(g_i'\sigma \cup X_{in}\sigma \cup X_{in}) = \emptyset$. Since $f\tau \subseteq h\tau \subseteq TH$, there are $1 \leq i \leq k$ and $\xi \in GSub$ such that $h_i\sigma\xi \subseteq TH$ and $\xi =_{X \setminus X_i} \tau$.

$X_i \cap var(g_i'\sigma) = \emptyset$ implies $g_i'\sigma\xi = g_i'\sigma\tau \subseteq h\tau \subseteq TH$, while $X_i \cap var(X_{in}\sigma \cup X_{in}) = \emptyset$ implies $\xi_{in} = \tau_{in}$ and $EQ(\sigma_{in})\xi = EQ(\sigma_{in})\tau$. Hence $\xi \in GS$. Since $EQ(\sigma_{in})\tau \subseteq h\tau \subseteq TH$, we conclude

$$(g_i' \cup h_i)(\tau_{in} + (\sigma\xi)_{out}) = (g_i' \cup h_i)(\xi_{in} + (\sigma\xi)_{out}) = (g_i' \cup h_i)\sigma_{out}\xi \subseteq TH$$

from $(g_i' \cup h_i)\sigma\xi \subseteq TH$ and Prop. 3.3 because $g_i'\sigma$ and $h_i\sigma$ are (TH, GS)-equivalence compatible. This implies (1).

Case 2. h is obtained from gs by a paramodulation step. Then

$$gs = \{g_1'[t_1/x], \ldots, g_n'[t_n/x]\}$$

and

$$h = g_1'[t/x]\sigma \cup \ldots \cup g_n'[t/x]\sigma \cup f \cup EQ(\sigma_{in})$$

such that $\sigma \in Sub$, $\{\exists X_1(t \equiv t_1)\sigma, \ldots, \exists X_n(t \equiv t_n)\sigma\} \Leftarrow f \in Gen(TH, GS)$ and $x \in var(g_1' \cap \ldots \cap g_n')$ such that for all $1 \leq i \leq n$, $X_i \cap var(g_i'[t/x]\sigma \cup X_{in}\sigma \cup X_{in}) = \emptyset$. Since $f\tau \subseteq g\tau \subseteq TH$, there are $1 \leq i \leq k$ and $\xi \in GSub$ such that $(t_i \equiv t)\sigma\xi \subseteq TH$ and $\xi =_{X \setminus X_i} \tau$.

$X_i \cap var(g_i'[t/x]\sigma \cup X_{in}) = \emptyset$ implies $g_i'[t/x]\sigma\xi = g_i'[t/x]\sigma\tau \subseteq h\tau \subseteq TH$ and $\xi_{in} = \tau_{in}$. Hence $\xi \in GS$. $X_i \cap var(X_{in}\sigma \cup X_{in}) = \emptyset$ implies $EQ(\sigma_{in})\xi = EQ(\sigma_{in})\tau \subseteq h\tau \subseteq TH$, and we conclude

$$g_i'[t_i/x](\tau_{in} + (\sigma\xi)_{out}) = g_i'[t_i/x](\xi_{in} + (\sigma\xi)_{out}) = g_i'[t_i/x]\sigma_{out}\xi \subseteq TH$$

from $(g_i'[t/x] \cup \{t_i \equiv t\})\sigma\xi \subseteq TH$ and Prop. 3.3 because $g_i'[t/x]\sigma$ is (TH, GS)-equivalence compatible. Again, this implies (1). ❏

Induction hypotheses are also applied by resolution or paramodulation. In this case the resolvent or paramodulant is extended by a **descent condition** of the form $t >> t'$ (cf. [33], Section 5.2). Given $w \in S^+$ and a tuple $\mathbf{x_{in}} \in \mathbf{X_w}$ consisting of all actual input variables, SIG is supposed to include a **descent function** $>>$: $w \times w \to bool$, which induces the following **induction ordering** $>>_{\mathbf{TH}}$ on $T_{SIG,w}$:

$$t >>_{TH} t' \quad \Longleftrightarrow_{def} \quad t >> t' \equiv true \;\in\; TH.$$

$>>_{TH}$ is assumed to be well-founded. If TH is the ground theory of a set AX of Horn clauses (cf. Section 4), then $>>_{TH}$ is well-founded iff the interpretation of $>>$ in the initial AX-model has this property. This is equivalent to the condition that *some* model of AX interprets $>>$ as a well-founded relation. The well-foundedness of $>>$ is a semantical assumption needed for the soundness of inductive resolution and paramodulation w.r.t. the initial model. A proof of this condition is not part of the inductive proof. The latter, however, includes a proof of the descent condition generated by an inductive step. For this purpose, $>>$ must be defined by suitable axioms upon which the descent conditions can be resolved or paramodulated (cf. Exs. 4.3 to 4.5). $>>$ is a Boolean function and not a predicate because *forward* resolution and paramodulation will produce negative descent conditions (cf. 3.10).

As there are infinitely many induction orderings, the set of inductive proofs w.r.t. AX is not enumerable. Hence no restriction of inductive rules to specific induction orderings or schemas can be expected to capture the whole inductive theory of AX. The range of conjectures, which express the correctness of programs specified by AX and which are provable with the same induction ordering, is just too limited.

Definition 3.5 Let TH and GS be as above, CS be a set of Gentzen clauses and $\rho : X \to X$ be a renaming of X_{in} away from X_{in}. The **generic expansion calculus upon** (TH, GS) **with induction hypotheses from** CS consists of deductive resolution and paramodulation (see above) and the following two rules:

Backward inductive resolution Let $c = (\{\exists X_1 h_1, \ldots, \exists X_n h_n\} \Leftarrow h) \in CS$ and $\sigma \in Sub$ be a renaming of $Z = X_1 \cup \ldots \cup X_n$ away from $var(c)$ such that for all $1 \leq i \leq n$, $f_i\sigma = h_i\rho\sigma$ is (TH, GS)-equivalence compatible and $Z\sigma \cap var(g_i\sigma \cup X_{in}\sigma \cup X_{in}) = \emptyset$, and for all $\tau \in GS$, $\rho\sigma\tau \in GS$.

$$\frac{\{g_1 \cup f_1, \ldots, g_n \cup f_n\}}{\{g_1\sigma \cup \ldots \cup g_n\sigma \cup h\rho\sigma \cup \{x_{in} >> x_{in}\rho\sigma \equiv true\} \cup EQ(\sigma_{in})\}}$$

Backward inductive paramodulation Let $c = (\{\exists X_1(u \equiv t_1), \ldots, \exists X_n(u \equiv t_n)\} \Leftarrow h) \in CS$, $x \in var(g_1 \cap \ldots \cap g_n)$ and $\sigma \in Sub$ be a renaming of $Z = X_1 \cup \ldots \cup X_n$ away from $var(c)$ such that for all $1 \leq i \leq n$, $u_i\sigma = t_i\rho\sigma$ and $Z\sigma \cap var(g_i[u\rho/x]\sigma \cup X_{in}\sigma \cup X_{in}) = \emptyset$, and for all $\tau \in GS$, $\rho\sigma\tau \in GS$.

$$\frac{\{g_1[u_1/x], \ldots, g_n[u_n/x]\}}{\{g_1[u\rho/x]\sigma \cup \ldots \cup g_n[u\rho/x]\sigma \cup h\rho\sigma \cup \{x_{in} >> x_{in}\rho\sigma \equiv true\} \cup EQ(\sigma_{in})\}}$$

A sequence $gs_1, \ldots, gs_n$ of goal sets is called an **expansion of** gs_1 **into** gs_n **upon** (TH, GS) **with induction hypotheses from** CS if

- for all $1 \leq i < n$, gs_{i+1} is obtained from gs_i by a single deductive or inductive resolution or paramodulation step,
- for all $1 \leq i \leq n$, gs_i consists of (TH, GS)-equivalence compatible goals.

The corresponding inference relation is denoted by $\vdash_{\mathbf{CS}}$. ❑

Theorem 3.6 (generic expansions are sound) Let $c = \{g_1, \ldots, g_m\} \Leftarrow h$ be a Gentzen clause and

$$CS \;=\; \{gs_1 \Leftarrow h_1, \ldots, gs_n \Leftarrow h_n\}$$

be a set of Gentzen clauses such that for all $1 \leq i \leq m$ and $1 \leq j \leq n$, $h \Leftarrow h_j \in Gen(TH, GS)$ and

(1) $gs_j \Leftarrow g_i \in Gen(TH, GS)$ or

(2) for all $\tau \in GSub$, $gs_j \Leftarrow g_i\tau_{out} \cup h_j \in Gen(TH, GS)$.

Then $\{g_1, \ldots, g_m\} \vdash_{CS} \{h\}$ implies $\{c\} \cup CS \subseteq Gen(TH, GS)$.

For an inductive proof of c one may use *consequences* of c as induction hypotheses CS. (1) admits hypotheses of the form $\{g'_1,\dots,g'_m\} \Leftarrow h'$ with $g'_i \subseteq g_i$. (2) is a generalization of (1) that is applicable if for all $1 \le i \le m$, the "subcase" $gs' \Leftarrow g_i\tau_{out} \cup h'$ of $gs' \Leftarrow h'$ is already known to be valid.

Proof of Theorem 3.6. Let $\{g_1,\dots,g_m\} \vdash_{CS} \{h\}$. Since (2) follows from (1), it is sufficient to show $\{c\} \cup CS \subseteq Gen(TH, GS)$ under Assumption (2). There is an expansion $gs_1,\dots,gs_k$ such that $gs_1 = \{g_1,\dots,g_m\}$ and $gs_k = \{h\}$. $CS \subseteq Gen(TH, GS)$ holds true if $c \in Gen(TH, GS)$, and this agrees with the case $j = k$ of the following condition:

(3) $gs_1 \Leftarrow gs_j \in Gen(TH, GS)$.

By the definition of $Gen(TH, GS)$, (3) is equivalent to (4):

(4) For all $\tau \in GS$, $1 < j \le k$ and $g \in gs_j$, $g\tau \subseteq TH$ implies $g_i(\tau_{in} + \xi_{out}) \subseteq TH$ for some $1 \le i \le m$ and $\xi \in GSub$.

Hence it remains to show (4), which will be done by induction on (τ_{in}, j) along $(>>_{TH}, >)$.

Let $\tau \in GS$, $1 < j \le k$ and $g \in gs_j$ such that $g\tau \subseteq TH$. If the expansion $gs_1,\dots,gs_j$ does not include inductive steps, then, by Theorem 3.4, $g_i(\tau_{in} + \xi_{out}) \subseteq TH$ for some $1 \le i \le m$ and $\xi \in GSub$. Otherwise there are $1 < l \le j$ and a goal set gs such that $gs_{l-1} = \{g'_1,\dots,g'_n\} \cup gs$ and $gs_l = \{g'\} \cup gs$, $\{g'_1,\dots,g'_n\}$ is transformed into $\{g'\}$ by an inductive step and there is an inductionless expansion of gs_l into gs_j such that, by Theorem 3.4, $g'(\tau_{in} + \xi_{out}) \subseteq TH$ for some $\xi \in GSub$. Assume that

(5) $\quad g'_i(\tau_{in} + \delta_{out}) \subseteq TH$ holds true for some $1 \le i \le n$ and $\delta \in GSub$.

Since $gs_1,\dots,gs_{l-1}$ is a proper subexpansion of $gs_1,\dots,gs_j$, the induction hypothesis and $\tau_{in} + \delta_{out} \in GS$ imply $g_i(\tau_{in} + \eta_{out}) \in TH$ for some $1 \le i \le m$ and $\eta \in GSub$, and the proof is complete. Hence it remains to show (5).

Let $\tau' = \tau_{in} + \xi_{out}$. Then $\tau' \in GS$ and $g'\tau' \in TH$.

Case 1. $\{g'\}$ is obtained from $\{g'_1,\dots,g'_n\}$ by an inductive resolution step. Then for all $1 \le i \le n$, $g'_i = g''_i \cup f_i$ and

$$g' \;=\; g''_1\sigma \cup \ldots \cup g''_n\sigma \cup h'\rho\sigma \cup \{x_{in} >> x_{in}\rho\sigma \equiv true\} \cup EQ(\sigma_{in})$$

such that $c' = (\{\exists X_1 h_1,\dots,\exists X_n h_n\} \Leftarrow h') \in CS$, σ is a renaming of $Z = X_1 \cup \ldots \cup X_n = var(c') \cap X_{out}$ away from $var(c')$ such that for all $1 \le i \le n$, $f_i\sigma = h_i\rho\sigma$ is (TH, GS)-equivalence compatible and $Z\sigma \cap var(g''_i\sigma \cup X_{in}\sigma \cup X_{in}) = \emptyset$, and for all $\theta \in GS$, $\rho\sigma\theta \in GS$. From

$$(\{x_{in} >> x_{in}\rho\sigma \equiv true\} \cup EQ(\sigma_{in}))\tau' \subseteq g''\tau' \subseteq TH$$

we conclude

(6) $\quad x_{in}\tau' >> x_{in}\rho\sigma\tau' \equiv true \in TH.$

Since $\tau' \in GS$, $\rho\sigma\tau' \in GS$. Since $h'\rho\sigma\tau' \subseteq g'\tau' \subseteq TH$ and since by (2), $h \Leftarrow h' \in Gen(TH, GS)$, we have $h\rho\sigma\tau' \in TH$. Since $\{h\} = gs_k$, the induction hypothesis implies

$$g_i((\rho\sigma\tau')_{in} + \lambda_{out}) \subseteq TH$$

for some $1 \le i \le m$ and $\lambda \in GSub$. Since $h'\rho\sigma\tau' \subseteq TH$ and since by (2), the clause $\{\exists X_1 h_1,\dots,\exists X_n h_n\} \Leftarrow g_i\lambda_{out} \cup h'$ follows from TH by induction on GS, we obtain

$$(7) \qquad h_i((\rho\sigma\tau')_{in} + \delta_{out}) \subseteq TH$$

for some $1 \le i \le n$ and $\delta \in GSub$. Since σ is a renaming of $Z = var(c') \cap X_{out}$ away from $var(c')$, σ^{-1} is defined on $Z\sigma$. Hence $Z\sigma \cap var(X_{in}\sigma) = \emptyset = dom(\rho) \cap X_{out}$ and $h_i\rho\sigma = f_i\sigma$ imply

$$
\begin{aligned}
(8) \qquad h_i((\rho\sigma\tau')_{in} + \delta_{out}) &= h_i\rho((\sigma\tau')_{in} + \delta_{out}) \\
&= h_i\rho((\sigma\tau')_{in} + \delta_Z) \\
&= h_i\rho((\sigma\tau')_{in} + (\sigma\sigma^{-1}\delta)_Z) \\
&= h_i\rho\sigma(\tau'_{var(X_{in}\sigma)} + (\sigma^{-1}\delta)_{Z\sigma} \\
&= h_i\rho\sigma(\tau'_{var(X_{in}\sigma)} + (\sigma^{-1}\delta)_{Z\sigma} + \tau'_Q) \\
&= f_i\sigma(\tau'_{var(X_{in}\sigma)} + (\sigma^{-1}\delta)_{Z\sigma} + \tau'_Q) \\
&= f_i\sigma\eta
\end{aligned}
$$

where $Q = var(X_{out}\sigma) \setminus Z\sigma$ and $\eta = \tau'_{var(X_{in}\sigma)} + (\sigma^{-1}\delta)_{Z\sigma} + \tau'_Q$. By (7) and (8), $f_i\sigma\eta \subseteq TH$. $Z\sigma \cap var(g''_i\sigma) = \emptyset$ implies $g''_i\sigma\eta = g''_i\sigma\tau' \subseteq g'\tau' \subseteq TH$. Hence $g'_i\sigma\eta \subseteq TH$.

$Z\sigma \cap var(X_{in}\sigma \cup X_{in}) = \emptyset$ implies $\eta_{in} = \tau'_{in}$ and $EQ(\sigma_{in})\eta = EQ(\sigma_{in})\tau'$. Hence $\eta \in GS$. Since $EQ(\sigma_{in})\tau' \subseteq g'\tau' \subseteq TH$ and $g''_i\sigma$ and $f_i\sigma$ are (TH, GS)-equivalence compatible, Prop. 3.3 implies $g'_i\sigma_{out}\eta = (g''_i \cup f_i)\sigma_{out}\eta \subseteq TH$. Hence

$$g'_i(\tau_{in} + (\sigma\eta)_{out}) = g'_i(\tau'_{in} + (\sigma\eta)_{out}) = g'_i(\eta_{in} + (\sigma\eta)_{out}) = g'_i\sigma_{out}\eta \subseteq TH,$$

and the proof of (5) is complete.

Case 2. $\{g'\}$ is obtained from $\{g'_1, \ldots, g'_n\}$ by an inductive paramodulation step. Then for all $1 \le i \le n$, $g'_i = g''_i[u_i/x]$ and

$$g' \;=\; g''_1[u\rho/x]\sigma \cup \ldots \cup g''_n[u\rho/x]\sigma \cup h'\rho\sigma \cup \{x_{in} >> x_{in}\rho\sigma \equiv true\} \cup EQ(\sigma_{in})$$

such that $c' = (\{\exists X_1(u \equiv t_1), \ldots, \exists X_n(u \equiv t_n)\} \Leftarrow h') \in CS$, σ is a renaming of $Z = X_1 \cup \ldots \cup X_n = var(c') \cap X_{out}$ away from $var(c')$, for all $1 \le i \le n$, $u_i\sigma = t_i\rho\sigma$ and $Z\sigma \cap var(g''_i[u\rho/x]\sigma X_{in}\sigma X_{in}) = \emptyset$, and for all $\theta \in GS$, $\rho\sigma\theta \in GS$. As in Case 1 (cf. (6)) we conclude

$$(9) \qquad x_{in}\tau' >> x_{in}\rho\sigma\tau' \equiv true \in TH$$

and thus

$$g_i((\rho\sigma\tau')_{in} + \lambda_{out}) \subseteq TH$$

for some $1 \le i \le m$ and $\lambda \in GSub$ by induction hypothesis. Since $h'\rho\sigma\tau' \subseteq g'\tau' \subseteq TH$ and since by (2), the clause $\{\exists X_1(t_1 \equiv u), \ldots, \exists X_n(t_n \equiv u)\} \Leftarrow g_i\lambda_{out} \cup h'$ follows from TH by induction on GS, we obtain

$$(10) \qquad (t_i \equiv u)((\rho\sigma\tau')_{in} + \delta_{out}) \in TH$$

for some $1 \le i \le n$ and $\delta \in GSub$. Since σ is a renaming of $Z = var(c') \cap X_{out}$ away from $var(c')$, σ^{-1} is defined on $Z\sigma$. Hence, analogously to (8), $t_i\rho\sigma = u_i\sigma$ implies

$$(11) \qquad (t_i \equiv u)((\rho\sigma\tau')_{in} + \delta_{out}) \;=\; (u_i \equiv u\rho)\sigma\eta$$

where $Q = var(X_{out}\sigma) \setminus Z\sigma$ and $\eta = \tau'_{var(X_{in}\sigma)} + (\sigma^{-1}\delta)_{Z\sigma} + \tau'_Q$. By (10) and (11), $(u_i \equiv u\rho)\sigma\eta \in TH$. $Z\sigma \cap var(g''_i[u\rho/x]\sigma \cup X_{in}) = \emptyset$ implies

$$g''_i[u\rho/x]\sigma\eta = g''_i[u\rho/x]\sigma\tau' \subseteq g'\tau' \subseteq TH$$

and $\eta_{in} = \tau'_{in}$. Hence $\eta \in GS$. $Z\sigma \cap var(X_{in}\sigma \cup X_{in}) = \emptyset$ implies $EQ(\sigma_{in})\eta = EQ(\sigma_{in})\tau' \subseteq g'\tau' \subseteq TH$. We conclude

$$g'_i(\tau_{in}+(\sigma\eta)_{out}) = g'_i(\tau'_{in}+(\sigma\eta)_{out}) = g''_i[u_i/x](\tau'_{in}+(\sigma\eta)_{out}) = g''_i[u_i/x](\eta_{in}+(\sigma\eta)_{out}) = g''_i[u_i/x]\sigma_{out}\eta \subseteq TH$$

from $(g''_i[u\rho/x] \cup \{u_i \equiv u\rho\})\sigma\eta \subseteq TH$ and Prop. 3.3 because $g''_i[u\rho/x]\sigma$ is (TH, GS)-equivalence compatible. Hence the proof of (5) is complete. ❑

Theorem 3.6(2) includes the case of n Horn clauses $g_1 \Leftarrow h_1, \ldots, g_n \Leftarrow h_n$ with a common *guard* g and the rest of $h_1, \ldots, h_n$ being a *g-minimal* goal set (cf. [33], Chapter 2):

Definition 3.7 A Horn clause $c = g \Leftarrow g' \cup h$, written as $g \Leftarrow g' : h$, is a **guarded clause** if $fresh(c) \subseteq var(h)$. g' is the **guard** of c, h is the **body** of c. Given a goal g, a goal set gs is **g-minimal w.r.t.** (TH, GS) if for all $h, h' \in gs$ and $\sigma, \tau \in GS$ with $\sigma =_{var(g)} \tau$,

$$g\sigma \cup h\sigma \cup h'\tau \subseteq TH \quad \text{implies} \quad h = h' \quad \text{and} \quad \{x\sigma \equiv x\tau \mid x \in var(h)\} \subseteq TH. \text{ ❑}$$

A g-minimal goal set gs yields a minimal case analysis insofar as two ground instances of two different goals $h, h' \in gs$ by the same substitution σ do not exist if $g\sigma \subseteq TH$. For instance, with respect to a usual specification of integers, $\{\{f(x) \equiv 0\}, \{f(x) \equiv succ(y)\}\}$ is an $\{f(x) \geq 0\}$-minimal goal set.

The notation $g \Leftarrow g' : h$ stems from concurrent logic programming (cf. [Sha 89]) where the guard g' on the one hand and the body h on the other hand are evaluated in different ways. Semantically, $g \Leftarrow g' : h$ agrees with the Horn clause $g \Leftarrow g' \cup h$. A set HS of guarded clauses with the same guard g is equivalent to a single Gentzen clause, provided that the bodies of HS constitute a g-minimal goal set:

Lemma 3.8 (Guarded clauses and Gentzen clauses) Let $HS = \{g_1 \Leftarrow g : h_1, \ldots, g_n \Leftarrow g : h_n\}$ be a set of guarded clauses such that $\{h_1, \ldots, h_n\}$ is g-minimal w.r.t. (TH, GS) and for all $1 \leq i, j \leq n$, g_i is (TH, GS)-equivalence compatible, $X_i =_{def} fresh(g_i \Leftarrow g : h_i)$ and $h_i = h_j$ implies $g_i = g_j$. Let

$$c = \{\exists X_1(g_1 \cup h_1), \ldots, \exists X_n(g_n \cup h_n)\} \Leftarrow g \quad \text{and} \quad c' = \{\exists X_1 h_1, \ldots, \exists X_n h_n\} \Leftarrow g.$$

(1) For all $1 \leq i, j \leq n$ and $\tau \in GSub$, $g_j \Leftarrow (g_i \cup h_i)\tau_{X_i} \cup g \cup h_j$ follows from TH by induction on GS.

(2) $c \in Gen(TH, GS)$ implies $HS \subseteq Gen(TH, GS)$.

(3) $HS \cup \{c'\} \subseteq Gen(TH, GS)$ implies $c \in Gen(TH, GS)$.

Proof. (1) Let $1 \leq i, j \leq n$, $\tau \in GSub$ and $\sigma \in GS$ such that $(g_i \cup h_i)(\sigma_{X \setminus X_i} + \tau_{X_i}) \cup g\sigma \cup h_j\sigma \subseteq TH$. Since $\{h_1, \ldots, h_n\}$ is g-minimal w.r.t. (TH, GS), we conclude $h_i = h_j$ and $x\sigma \equiv x\tau \in TH$ for all $x \in var(h_i)$. Hence $g_i = g_j$, $X_i = X_j$ and thus $g_j(\sigma_{X \setminus X_i} + \tau_{X_i}) \subseteq TH$. This implies $g_j\sigma \subseteq TH$ because $X_i \subseteq var(h_i)$ and g_j is (TH, GS)-equivalence compatible.

(2) immediately follows from (1).

(3) Let $HS \cup \{c'\} \subseteq Gen(TH, GS)$ and $\sigma \in GS$ such that $g\sigma \subseteq TH$. Since $c' \in Gen(TH, GS)$, there are $1 \leq i \leq n$ and $\tau \in GSub$ such that $h_i(\sigma_{X \setminus X_i} + \tau_{X_i}) \subseteq TH$. $var(g) \cap X_i = \emptyset$ implies $g\sigma = g(\sigma_{X \setminus X_i} + \tau_{X_i})$. Hence $(g \cup h_i)(\sigma_{X \setminus X_i} + \tau_{X_i}) \subseteq TH$ and thus $g_i(\sigma_{X \setminus X_i} + \tau_{X_i}) \subseteq TH$ because $HS \subseteq Gen(TH, GS)$. ❑

Hence a proof of the above Gentzen clause c may use guarded clauses from HS as induction hypotheses:

Corollary 3.9 (generic expansions with guarded clause hypotheses are sound) Let $HS = \{g_1 \Leftarrow g : h_1, \ldots, g_n \Leftarrow g : h_n\}$ be a set of guarded clauses such that $\{h_1, \ldots, h_n\}$ is g-minimal w.r.t. (TH, GS) and for all $1 \leq i, j \leq n$, g_i is (TH, GS)-equivalence compatible, $X_i =_{def} fresh(g_i \Leftarrow g : h_i)$ and $h_i = h_j$ implies $g_i = g_j$.

Let c be as in 3.8. $\{\exists X_1(g_1 \cup h_1), \ldots, \exists X_n(g_n \cup h_n)\} \vdash_{HS} \{g\}$ implies $\{c\} \cup HS \subseteq Gen(TH, GS)$.

Proof. Lemma 3.8(1) and Theorem 3.6(2). ❑

For applications of this result, cf. [33], Chapt. 2, Sect. 5.5 and Chapt. 8. The original motivation for the step from CS to c is to simplify the premises of CS. The smaller the premise is, the greater is the chance that the clause can be proved backward by transforming the conclusion into the premise (cf. 1.1). The next section deals with the case where TH is given by all ground atoms that are valid w.r.t. a set of Horn clause axioms. Example 4.4 presents two proofs of a guarded Horn clause. One of them consists of backward as well as forward steps. The other one uses Cor. 3.9 for a completely backward expansion.

The above inference rules can only be used in a backward proof. The redex of a rule in the actual goal set is unified with the conclusion (or left-hand side) of a clause $c \in Gen(TH, GS)$, while the reduct obtained by applying the rule is an instance of the premise (or right-hand side) of the applied clause. With slight modifications, resolution and paramodulation can also be applied conversely and thus be used in a forward proof. Then no substitution σ is to be generated by the rule application and $EQ(\sigma_{in})$ is empty. Moreover, if an inductive rule is inverted, the descent condition $b \equiv true$ becomes part of the rule antecedent. Such a rule would be applicable only in a few cases. However, a simple propositional law allows us to move the descent condition from the antecedent to the succedent. Roughly said, each application of an inductive rule of 3.5 establishes a theorem of the form

$$\{g\} \cup gs \;\Leftarrow\; \{h \cup \{b \equiv true\}\} \cup gs,$$

while the corresponding application of a forward inductive rule establishes the equivalent theorem

$$\{h\} \cup gs \;\Rightarrow\; \{g\} \cup \{h \cup \{b \equiv false\}\} \cup gs.$$

The following inference rules are added to the generic expansion calculus (cf. 3.5):

Definition 3.10 (generic forward rules) Let CS be a set of Gentzen clauses and $\rho : X \to X$ be a renaming of X_{in} away from X_{in}.

Forward deductive resolution Let $\{\exists X_1 h_1, \ldots, \exists X_n h_n\} \Leftarrow h \in Gen(TH, GS)$ such that for all $1 \le i \le n$, $X_i \cap var(g \cup X_{in}) = \emptyset$.

$$\frac{\{g \cup h\}}{\{g \cup h_1, \ldots, g \cup h_n\}}$$

Forward deductive paramodulation Let $\{\exists X_1(t \equiv t_1), \ldots, \exists X_n(t \equiv t_n)\} \Leftarrow h \in Gen(TH, GS)$ and $x \in var(g)$ such that for all $1 \le i \le n$, $X_i \cap var(g[t/x] \cup X_{in}) = \emptyset$.

$$\frac{\{g[t/x] \cup h\}}{\{(g \cup \{t \equiv x\})[t_1/x], \ldots, (g \cup \{t \equiv x\})[t_n/x]\}}$$

Forward inductive resolution Let $c = (\{\exists X_1 h_1, \ldots, \exists X_n h_n\} \Leftarrow h) \in CS$ and $\sigma \in Sub$ be a renaming of $Z = X_1 \cup \ldots \cup X_n$ away from $var(c)$ such that $f = h\rho\sigma$, $Z\sigma \cap var(g \cup X_{in}) = \emptyset$ and for all $\tau \in GS$, $\rho\sigma\tau \in GS$.

$$\frac{\{g \cup f\}}{\{g \cup h_1\rho\sigma, \ldots, g \cup h_n\rho\sigma, g \cup f \cup \{x_{in} >> x_{in}\rho\sigma \equiv false\}\}}$$

Forward inductive paramodulation Let $c = (\{\exists X_1(u \equiv t_1), \ldots, \exists X_n(u \equiv t_n)\} \Leftarrow h) \in CS$, $x \in var(g) \setminus var(f)$ and $\sigma \in Sub$ be a renaming of $Z = X_1 \cup \ldots \cup X_n$ away from $var(c)$ such that $f = h\rho\sigma$, $t = u\rho\sigma$, $Z\sigma \cap var(g[t/x] \cup X_{in}) = \emptyset$ and for all $\tau \in GS$, $\rho\sigma\tau \in GS$.

$$\frac{\{g[t/x] \cup f\}}{\{(g \cup \{t \equiv x\})[t_1\rho\sigma/x], \ldots, (g \cup \{t \equiv x\})[t_n\rho\sigma/x], g[t/x] \cup f \cup \{x_{in} >> x_{in}\rho\sigma \equiv false\}\}} \quad ❑$$

The equation $\ldots \equiv false$ in the succedent of the above inductive rules is redundant for their soundness. However, the succedent is stronger with the equation $\ldots \equiv false$ than without it. Succedents of rules applied in a forward proof should be as strong as possible, otherwise the given conclusion might not be achieved.

Theorem 3.11 (forward rules are sound) Suppose that $x \equiv x \in Gen(TH, GS)$ and, for descent functions $>>$, $\{\{x >> y \equiv true\}, \{x >> y \equiv false\}\} \in Gen(TH, GS)$. Let CS be a set of Gentzen clauses, $\rho : X \to X$ be a renaming of X_{in} away from X_{in} and the goal set gs be obtained from the goal set hs by a single forward resolution or paramodulation step such that gs and hs consist of (TH, GS)-equivalence compatible goals. Then $hs \vdash_{CS} gs$ and thus, by Thm. 3.6(1), $hs \Leftarrow g \in Gen(TH, GS)$ for all $g \in gs$.

Proof. Deductive forward resolution is the inverse of deductive resolution with $\sigma = id$ and $g_1 = \ldots = g_n = g$. Deductive forward paramodulation is the inverse of deductive paramodulation with $\sigma = id$ and $g_1 = \ldots = g_n = g \cup \{t \equiv x\}$, followed by deductive resolution upon $x \equiv x$. Hence $hs \vdash_{CS} gs$ if the step from gs to hs is performed by deductive forward resolution or paramodulation.

Let $b = (x_{in} >> x_{in}\rho\sigma)$.

If the step from gs to hs is performed by inductive forward resolution, we have

$$hs \;=\; \{g \cup f_1), \ldots, g \cup f_n, g \cup f \cup \{b \equiv false\}\},$$

$gs = \{g \cup f\}$, $f = h\rho\sigma$ and $f_i = h_i\rho\sigma$ for all $1 \leq i \leq n$ and some $c = (\{\exists X_1 h_1, \ldots, \exists X_n h_n\} \Leftarrow h) \in CS$. Inductive resolution upon c with $\sigma =_{X \setminus X_{in}\rho} id$ and $g_1 = \ldots = g_n = g$ reads as follows:

$$\frac{\{g \cup f_1, \ldots, g \cup f_n\}}{\{g \cup f \cup \{b \equiv true\}\}}.$$

By applying this rule to hs we obtain

$$hs \;\vdash_{CS}\; gs' =_{def} \{g \cup f \cup \{b \equiv true\},\; g \cup f \cup \{b \equiv false\}\}.$$

By assumption, $\{\{b \equiv true\}, \{b \equiv false\}\} \in Gen(TH, GS)$ and thus $gs' \vdash_{CS} gs$ by deductive resolution. Hence $hs \vdash_{CS} gs$.

If the step from gs to hs is performed by inductive forward paramodulation, we have

$$hs \;=\; \{(g \cup \{t \equiv x\})[u_1/x], \ldots, (g \cup \{t \equiv x\})[u_n/x], g[t/x] \cup f \cup \{b \equiv false\}\},$$

$gs = \{g[t/x] \cup f\}$, $f = h\rho\sigma$, $t = u\rho\sigma$ and $u_i = t_i\rho\sigma$ for all $1 \leq i \leq n$ and some $c = (\{\exists X_1(u \equiv t_1), \ldots, \exists X_n(u \equiv t_n)\} \Leftarrow h) \in CS$. Inductive paramodulation with $\sigma =_{X \setminus X_{in}\rho} id$ and $g_1 = \ldots = g_n = g \cup f \cup \{t \equiv x\}$ reads as follows:

$$\frac{\{(g \cup f \cup \{t \equiv x\})[u_1/x], \ldots, (g \cup f \cup \{t \equiv x\})[u_n/x]\}}{\{g[t/x] \cup f \cup \{t \equiv t, b \equiv true\}\}}.$$

By applying this rule to hs we obtain

$$hs \;\vdash_{CS}\; gs' =_{def} \{g[t/x] \cup f \cup \{t \equiv t, b \equiv true\},\; g[t/x] \cup f \cup \{b \equiv false\}\}.$$

By deductive resolution upon $x \equiv x$,

$$gs' \;\vdash_{CS}\; gs'' =_{def} \{g[t/x] \cup f \cup \{b \equiv true\},\; g[t/x] \cup f \cup \{b \equiv false\}\}.$$

By assumption, $\{\{b \equiv true\}, \{b \equiv false\}\} \in Gen(TH, GS)$ and thus $gs'' \vdash_{CS} gs$ by deductive resolution. Hence $hs \vdash_{CS} gs$. ❑

Implementations of forward paramodulation should admit the simultaneous replacement of several occurrences of the same term because the sequential replacement performed by several applications of the rule might result in a weaker succedent, which cannot be transformed via further forward steps into the given conclusion.

4 Inductive expansion

The inductive expansion calculus defined in [31] and [33] is an instance of the generic expansion calculus. The set GS of ground substitutions is defined as $GSub$, the set of all ground substitutions, and, given a set AX of Horn clauses over SIG, the set TH of ground atoms is the **ground theory of AX, TH(AX)**, which consists of all ground atoms that are derivable by the **cut calculus** for the **specification SP = (SIG,AX)**. This calculus consists of the following two inference rules for deriving Horn clauses from AX and congruence axioms for equality predicates of SIG:

Cut

$$\frac{g \Leftarrow d \cup g',\ g' \Leftarrow d'}{g \Leftarrow d \cup d'}$$

Substitution Let $\sigma \in$ Sub.

$$\frac{g \Leftarrow d}{g\sigma \Leftarrow d\sigma}$$

We write $\mathbf{SP} \vdash_{\mathbf{cut}} \mathbf{g} \Leftarrow \mathbf{h}$ if $g \Leftarrow h$ is derivable from AX and congruence axioms by applying the above rules.

Definition 4.1 (inductive theory) Given a specification $SP = (SIG, AX)$, the set of derivable equations induces a SIG-congruence relation $\equiv_{SP}$ on $T_{SIG}(X)$, called SP**-equivalence**:

$$t \equiv_{SP} t' \text{ iff } SP \vdash_{cut} t \equiv t'.$$

ITh(SP) = Gen(TH(AX),GSub) is called the **inductive theory of** SP. Inductive validity coincides with validity in the **initial** SP**-structure Ini(SP)** (cf. [33], Section 3.1). $\sigma \in GSub$ is an SP-solution of a goal g if $SP \vdash_{cut} g\sigma$. Then g is called SP**-solvable**. If SP contains the constants $true$ and $false$ and $true \equiv false$ is not an inductive theorem of SP, then we call SP **Boole-consistent**.

An expansion $E = (gs_1, \ldots, gs_n)$ upon $(TH(AX), GSub)$ (cf. 3.5) is called an **inductive expansion upon** SP. If E uses induction hypotheses from a set CS of Gentzen clauses, we write $gs_1 \vdash_{\mathbf{SP,CS}} gs_n$. ❑

Since $SP \vdash_{cut} c$ for all congruence axioms c for equality predicates of SIG, all goals over SIG are (TH, GS)-equivalence compatible (cf. Def. 3.2). Hence the goal sets of an inductive expansion are automatically (TH, GS)-equivalence compatible (cf. 3.5).

Descent functions >> (cf. Section 3) are well-founded iff the interpretation of >> in *some* SP-model A is well-founded (cf. [33], Prop. 5.3). Practically, A is the original – often informal – model of the data type to be specified by SP.

The soundness of inductive expansions, previously proved directly (cf. [33], Thm. 5.5), is a special case of Thm. 3.6(1) and Cor. 3.9, respectively:

Theorem 4.2 (inductive expansions are sound)

(1) Let $c = \{g_1, \ldots, g_m\} \Leftarrow h$ be a Gentzen clause and $CS = \{gs_1 \Leftarrow h_1, \ldots, gs_n \Leftarrow h_n\}$ be a set of Gentzen clauses such that for all $1 \le i \le m$ and $1 \le j \le n$, $h \Leftarrow h_j, gs_j \Leftarrow g_i \in ITh(SP)$. Then $\{g_1, \ldots, g_m\} \vdash_{SP,CS} \{h\}$ implies $\{c\} \cup CS \subseteq ITh(SP)$.

(2) Let $HS = \{g_1 \Leftarrow g : h_1, \ldots, g_n \Leftarrow g : h_n\}$ be a set of guarded clauses such that $\{h_1, \ldots, h_n\}$ is g-minimal w.r.t. $(TH(AX), GSub)$ and for all $1 \le i,j \le n$, $X_i =_{def} fresh(g_i \Leftarrow g \cup h_i)$ and $h_i = h_j$ implies $g_i = g_j$. Let

$$c = \{\exists X_1(g_1 \cup h_1), \ldots, \exists X_n(g_n \cup h_n)\} \Leftarrow g.$$

Then $\{\exists X_1(g_1 \cup h_1), \ldots, \exists X_n(g_n \cup h_n)\} \vdash_{SP,HS} \{g\}$ implies $\{c\} \cup HS \subseteq ITh(SP)$. ❑

Since for all goals g, h and atoms p, the clauses $\{g,h\} \Leftarrow g$ and $p \Leftarrow p$ are inductive SP-theorems, the following **backward** rules are special cases of deductive resolution or paramodulation:

Goal elimination

$$\frac{\{g,h\}}{\{g\}}$$

Atom factoring Let $\sigma \in Sub$ such that $p\sigma = q\sigma$.

$$\frac{\{g \cup \{p,q\}\}}{\{g\sigma \cup \{q\sigma\}\}}$$

Term unification Let $\sigma \in Sub$ such that $t\sigma = u\sigma$.

$$\frac{\{g \cup \{t \equiv u\}\}}{\{g\sigma\}}$$

Term replacement Let $x \in var(g)$.

$$\frac{\{g[t/x] \cup \{t \equiv u\}\}}{\{g[u/x] \cup \{t \equiv u\}\}}$$

Instantiation Let $\sigma \in Sub$.

$$\frac{\{\exists V g\}}{\{g\sigma \cup EQ(\sigma_{var(g)\setminus V})\}}$$

Σ-simplification Let Σ is be an *SP-compatible simplifier*, i.e. a function on the set of all goal sets such that $gs \Leftrightarrow \Sigma(gs) \in ITh(SP)$ (cf. 10.1).

$$\frac{gs}{\Sigma(gs)}$$

Since for all goals g, h and unsolvable goals g', the clauses $g \Leftarrow g \cup h$ and $FALSE \Leftarrow g'$ are inductive SP-theorems, the following rules are special cases of deductive forward resolution:

Forward goal elimination

$$\frac{\{g \cup h\}}{\{g\}}$$

Goal refutation Let g be SP-unsolvable.

$$\frac{\{g\}}{\{FALSE\}}$$

Goal refutation can also be applied in a backward proof where it is a special case of goal elimination (see above). By Thm. 3.11, inductive *forward* resolution and paramodulation are sound w.r.t. $\vdash_{SP,CS}$ if for descent functions $>>$, $\{x >> y \equiv true\} \vee \{x >> y \equiv false\} \in ITh(SP)$. This condition implies that the initial SP-structure interprets $>>$ as a total function. The dual condition is Boole-consistency: $(FALSE \Leftarrow true \equiv false) \in ITh(SP)$ (cf. Def. 4.1). In fact, inductive *backward* resolution and paramodulation are sound w.r.t. $\vdash_{SP,CS}$ only if SP is Boole-consistent. Otherwise $>>$ were not well-founded.

Example 4.3 (`GREATER`) Using Expander syntax (cf. Sect. 1.1) we specify the *greater* relation on natural numbers and prove its transitivity.

```
GREATER
     functs³       true 0 s >> 3
     preds         > 1 2
     infixes       > >>
     vars          x x' y y' z z' ex
```

```
axioms      (1) {s(x) > 0}
            (2) {s(x) > s(y)} <== {x > y}
            (3) {(s(x),z) >> (x,z') = true}
theorems    (1) {x = 0}∨{x = s(ex!)}[4]
            (2) FALSE <== {0 > x}
            (3) {x > y} <== {s(x) > s(y)}
conjects    (1) {x > z} <== {x > y, y > z}
```

Axiom 3 specifies the descent function >> used in the inductive expansion of Conjecture 1 given below. This proof consists of backward and forward deductive resolution steps and a single backward inductive resolution step with Conjecture 1 as induction hypothesis and the descent function as specified by Axiom 3 (cf. (*) below).

Backward steps always modify the actual conclusion, while forward steps are applied to the actual premise. In terms of Sect. 1.1, the lists $front$ and $rear$ have been merged. Each goal set of $front$ (the backward part of the proof) is preceded by `conclusion`, each goal set of $rear$ (the forward part of the proof) is preceded by `premise`. The goals of a goal set are numbered and listed sequentially. Remember that a goal set represents a *disjunction*, while a goal represents a *conjunction*. Theorems 1 to 3 of `GREATER` are used as lemmas.

```
initial conclusion:
(1) { x > z }
initial premise:
(1) { x > y, y > z }
atom 1 in conclusion goal 1 replaced with axiom GREATER1
atom 1 in conclusion goal 1 replaced with axiom GREATER2
conclusion:
(1) { x = s(x1), z = s(y1), x1 > y1 }
(2) { x = s(x1), z = 0 }
atom 3 in conclusion goal 1 replaced with conjecture 1                (*)
conclusion:
(1) { (x,z) >> (x1,y1) = true, x1 > y2, y2 > y1, x = s(x1), z = s(y1) }
(2) { x = s(x1), z = 0 }
term at position 1 1 in conclusion goal 1 replaced with axiom GREATER3
conclusion:
(1) { x = s(x1), x1 > y2, y2 > y1, z = s(y1) }
(2) { x = s(x1), z = 0 }
premise:
(1) { x > y, y > z }
term at position 1 1 in premise goal 1 replaced with theorem 1
premise:
(1) { x = 0, 0 > y, x > z }
(2) { x = s(ex), s(ex) > y, y > z }
atom 2 in premise goal 1 replaced with theorem 2
premise:
(1) { x = s(ex), s(ex) > y, y > z }
terms at positions 2 2 , 3 1 in premise goal 1 replaced with theorem 1
premise:
(1) { y = 0, s(ex) > 0, 0 > z, x = s(ex) }
```

[3]Positive numbers following a function or predicate symbol F denote the axioms that specify F.

[4]A goal set $\{g_1,\dots,g_n\}$ is denoted by $g_1 \vee \dots \vee g_n$. The exclamation mark identifies ex as an existentially quantified variable.

```
(2) { y = s(ex1), s(ex) > s(ex1), s(ex1) > z, x = s(ex) }
atom 3 in premise goal 1 replaced with theorem 2
premise:
(1) { y = s(ex1), s(ex) > s(ex1), s(ex1) > z, x = s(ex) }
atom 2 in premise goal 1 replaced with theorem 3
premise:
(1) { ex > ex1, y = s(ex1), s(ex1) > z, x = s(ex) }
term at position 3 2 in premise goal 1 replaced with theorem 1
premise:
(1) { z = 0, s(ex1) > 0, ex > ex1, y = s(ex1), x = s(ex) }
(2) { z = s(ex2), s(ex1) > s(ex2), ex > ex1, y = s(ex1), x = s(ex) }
atom 2 in premise goal 2 replaced with theorem 3
premise:
(1) { z = 0, s(ex1) > 0, ex > ex1, y = s(ex1), x = s(ex) }
(2) { ex1 > ex2, z = s(ex2), ex > ex1, y = s(ex1), x = s(ex) }
conjecture 1 has been proved
```

The message `conjecture 1 has been proved` responds to the syntactic check whether or not the actual premise *hs subsumes* the actual conclusion *gs*. Subsumption extends matching from terms to goal sets such that, if *hs* subsumes *gs*, then $gs \Leftarrow hs$ is valid. For the details, cf. [34], Sect. 3.5.

We conclude from Theorem 4.2(1) that Conjecture 1 is an inductive `GREATER`-theorem. A more straightforward proof relying on the fact that `GREATER` is *ground confluent* (cf. 5.5) is given by Example 10.12. ❑

Example 4.4 (`DIVISION`) Using Expander syntax we specify the *division-and-remainder* function on natural numbers and prove its correctness:

```
DIVISION
     functs      true 0 s no_pair + - * 1 2 div 3 4 5 >> 6
     preds       < <= > >=
     infixes     * div >>
     vars        x y z q r
     axioms      (1) {0*x = 0}
                 (2) {s(x)*y = (x*y)+y}
                 (3) {x div y = (0,x)} <== {x < y}
                 (4) {x div y = (s(q),r)} <== {x >= y, y > 0, (x-y) div y = (q,r)}
                 (5) {x div 0 = no_pair}
                 (6) {(y,x,q,r) >> (y',x-y,q',r') = true} <== {x >= y, y > 0}
     theorems    (1) {(z+y)+r = x} <== {z+r = x-y, x >= y}
                 (2) {x < y} <== {y > x}
                 (3) {x = q, y = r} <== {(x,y) = (q,r)}
                 (4) FALSE <== {(x,y) = no_pair}
                 (5) {z = no_pair}∨{z = (0,x), x < y}∨
                     {z = (s(q!),r!), x >= y, y > 0, (x-y) div y = (q!,r!)}
                     <== {x div y = z}
     conjects    (1) {x = (q*y)+r, r < y} <== {y > 0, x div y = (q,r)}
```

An inductive expansion of Conjecture 1 upon `DIVISION` is given below. It consists of deductive backward and forward resolution and paramodulation steps as well as an inductive backward resolution step with Conjecture 1 as induction hypothesis and the descent function specified by Axiom 6 (cf. (*) below).

```
initial conclusion:
(1) { x = (q*y)+r, r < y }
initial premise:
(1) { 0 < y, x div y = (q,r) }
term at position 1 2 1 in conclusion goal 1 replaced with axiom DIVISION1
term at position 1 2 1 in conclusion goal 1 replaced with axiom DIVISION2
conclusion:
(1) { q = s(x1), ((x1*y)+y)+r = x, r < y }
(2) { q = 0, r = x, r < y }
atom 2 in conclusion goal 1 replaced with theorem 1
conclusion:
(1) { (x1*y)+r = x-y, x >= y, q = s(x1), r < y }
(2) { q = 0, r = x, r < y }
atoms 1 4 in conclusion goal 1 replaced with conjecture 1                (*)
conclusion:
(1) { (y, x, q, r) >> (y,x-y,x1,r) = true, 0 < y, (x-y) div y = (x1, r),
      y <= x, q = s(x1) }
(2) { q = 0, r = x, r < y }
term at position 1 1 in conclusion goal 1 replaced with axiom DIVISION6
conclusion:
(1) { y <= x, 0 < y, (x-y) div y = (x1,r), q = s(x1) }
(2) { q = 0, r = x, r < y }
premise:
(1) { 0 < y, x div y = (q,r) }
atom 2 in premise goal 1 replaced with theorem 5
premise:
(1) { (q,r) = no_pair, 0 < y }
(2) { (q,r) = (0,x), x < y, 0 < y }
(3) { (q,r) = (s(q1),r1), x >= y, y > 0, (x-y) div y = (q1,r1), 0 < y }
atom 1 in premise goal 1 replaced with theorem 4
premise:
(1) { (q,r) = (0,x), x < y, 0 < y }
(2) { (q,r) = (s(q1),r1), x >= y, y > 0, (x-y) div y = (q1,r1), 0 < y }
atom 3 in premise goal 2 replaced with theorem 2
premise:
(1) { (q,r) = (0,x), x < y, 0 < y }
(2) { (q,r) = (s(q1),r1), x >= y, (x-y) div y = (q1,r1), 0 < y }
atom 1 in premise goal 1 replaced with theorem 3
atom 1 in premise goal 2 replaced with theorem 3
premise:
(1) { q = 0 , r = x, x < y, 0 < y }
(2) { q = s(q1), x >= y, (x-y) div y = (q1,r), 0 < y }
conclusion:
(1) { y <= x, 0 < y, (x-y) div y = (x1,r), q = s(x1) }
(2) { q = 0, r = x, r < y }
term at position 3 1 replaced with equation 2 in conclusion goal 2
conclusion:
(1) { y <= x, 0 < y, (x-y) div y = (x1,r), q = s(x1) }
```

```
(2) { q = 0, r = x, x < y }
conjecture 1 has been proved
```

We conclude from Theorem 4.2(1) that Conjecture 1 is an inductive `DIVISION`-theorem. The proof uses Theorems 1 to 5 as lemmas. 1 to 4 are obvious. 5 is the *only-if-completion* of **`div`**, i.e. the inverse of Axioms 3, 4 and 5, which specify **`div`** (cf. 5.6). Only-if-completions are inductive theorems if the specification is *normal form complete* (cf. 5.5). A more straightforward proof relying on the fact that `DIVISION` is ground confluent will be given by Example 10.13. ❑

Example 4.5 (`DIVISION'`) A further proof of Conjecture 1 is obtained with the help of Theorem 4.2(2) if the premise of Conjecture 1 is split into the guard $g = \{y > 0, x \equiv x\}$ and the g-minimal goal set $\{\{x\ div\ y = (q,r)\}\}$.[5] Instead of expanding Conjecture 1, we expand the Gentzen clause

$$c \quad = \quad \exists\{q,r\}\{x \equiv (q*y)+r,\ r < y,\ x\ div\ y \equiv (q,r)\} \ \Leftarrow\ \{y > 0\}$$

and use Conjecture 1 as induction hypothesis. By Theorem 4.2(2), this expansion implies that both c and Conjecture 1 are `DIVISION`-theorems. Actually, `DIVISION` is modified as follows:

```
DIVISION'
    functs     true 0 s no_pair + - * 1 2 div 3 4 5 >> 6
    preds      < <= > >=
    infixes    + - * div >>
    vars       x y z q r
    axioms     (1) {0*x = 0}
               (2) {s(x)*y = (x*y)+y}
               (3) {x div y = (0,x)} <== {x < y}
               (4) {x div y = (s(q),r)} <== {x >= y, y > 0, (x-y) div y = (q,r)}
               (5) {x div 0 = no_pair}
               (6) {(x,y) >> (x,y-z) = true} <== {y >= z, z > 0}
    theorems   (1) {(z+y)+r = x} <== {z+r = x-y, x >= y}
               (2) {x >= y} <== {y <= x}
               (3) {x div y = (q!,r!)} <== {y > 0}
               (4) {x > y} <== {y < x}
               (5) {x < y}∨{y <= x}
    conjects   (1) {x = (q*y)+r, r < y} <== {y > 0, x div y = (q,r)}
               (2) {x = (q*y)+r, r < y, x div y = (q,r)} <== {y > 0}
```

Conjecture 2 agrees with c. Theorems 1 to 5 are used as lemmas in the - completely backward - proof of c given below. Since c has only two input variables, we also obtain a new descent function when applying Conjecture 1 as induction hypothesis (cf. (*) below).

```
initial conclusion:
(1) { x = (q*y)+r, r < y, x div y = (q,r) }
initial premise:
(1) { 0 < y }
term at position 3 1 in conclusion goal 1 replaced with axiom DIVISION'3
term at position 3 1 in conclusion goal 1 replaced with axiom DIVISION'4
conclusion:
```

[5]The tautology $x \equiv x$ is used only for fixing x as an input variable.

```
(1) { y <= x, 0 < y, (x-y) div y = (q1,r), x = (s(q1)*y)+r, r < y }
(2) { x < y, x = (0*y)+x }
term at position 4 2 1 in conclusion goal 1 replaced with axiom DIVISION'2
conclusion:
(1) { ((q1*y)+y)+r = x, y <= x, 0 < y, (x-y) div y = (q1,r), r < y }
(2) { x < y, x = (0*y)+x }
term at position 2 2 1 in conclusion goal 2 replaced with axiom DIVISION'1
conclusion:
(1) { ((q1*y)+y)+r = x, y <= x, 0 < y, (x-y) div y = (q1,r), r < y }
(2) { x < y }
atom 1 in conclusion goal 1 replaced with theorem 1
conclusion:
(1) { (q1*y)+r = x-y, x >= y, y <= x, 0 < y, (x-y) div y = (q1,r), r < y }
(2) { x < y }
atom 2 in conclusion goal 1 replaced with theorem 2
conclusion:
(1) { (q1*y)+r = x-y, y <= x, 0 < y, (x-y) div y = (q1,r), r < y }
(2) { x < y }
atoms 1 5 in conclusion goal 1 replaced with conjecture 1                 (*)
conclusion:
(1) { (y,x) >> (y,x-y) = true, 0 < y, (x-y) div y = (q1,r), y <= x }
(2) { x < y }
term at position 1 1 in conclusion goal 1 replaced with axiom DIVISION'6
conclusion:
(1) { y <= x, 0 < y, (x-y) div y = (q1,r) }
(2) { x < y }
atom 3 in conclusion goal 1 replaced with theorem 3
conclusion:
(1) { y > 0, y <= x, 0 < y }
(2) { x < y }
atom 1 in conclusion goal 1 replaced with theorem 4
conclusion:
(1) { 0 < y, y <= x }
(2) { x < y }
atoms 1 2 in conclusion goals 2 1 replaced with theorem 5
conclusion:
(1) { 0 < y }
conjecture 1 has been proved
```

On the one hand, this proof does not the use the only-if-completion $ONLY(div)$ of div as the proof of Ex. 4.4 does. On the other hand, here we use the fact that $\{\{x\ div\ y = (q,r)\}\}$ is g-minimal, i.e. the clause

$$\{q \equiv q', r \equiv r'\} \Leftarrow \{x\ div\ y \equiv (q,r),\ x\ div\ y \equiv (q',r')\}$$

is an inductive theorem. Intuitively, this clause says that div has been specified consistently, i.e., as a function with unique values, while both the normal form completeness of `DIVISION` (cf. Ex. 4.4) and Theorem 3 of `DIVISION'` – together with Axiom 5 – express that div has been specified completely, i.e., as a total function. ❏

5 Canonical specifications

Canonicity summarizes three requirements to a Horn clause specification SP: *normal form completeness*, *strong termination* and, most crucial, *ground confluence*. Ground confluence and normal form completeness are discussed in this and the following chapter. Strong termination is the topic of Section 7. Some of the material presented in Sections 5 to 7 was introduced and thoroughly motivated in [30], [32] and [33], especially certain deviations from CTRS theory (cf. Sections 1.2 and 1.4).

One important deviation from other rewriting approaches is that we parameterize the reduction calculus by a set NF of *normal forms*, which depends on the underlying specification SP. NF is not *a priori* the set of non-rewritable terms. This would entail a circular definition because the reduction calculus is defined *with respect to* NF: normal forms are the (only) terms to be substituted for fresh variables in a rewriting step (cf. Section 2). Normal forms are built up of variables and *constructors* and lead us to *constructor-based* specifications and then to canonical specifications.

Assumption 5.1 For notational convenience, we regard in the sequel each logical predicate P as a Boolean function and each logical atom $P(t)$ as the equation $P(t) \equiv true$. Thus logical predicates become partial Boolean functions, and each Horn clause axiom becomes a conditional equation. Conversely, a Boolean function that stems from a predicate only occurs at the outermost term position of an equation $t \equiv true$. Normal form completeness takes this restriction into account: Boolean terms $P(t)$ where P stems from a predicate need *not* have a normal form. We call a term t an **atomic** term if $t = P(u)$ for some predicate P. ❑

Definition 5.2 (constructor-based specification) A Horn clause specification $SP = (SIG, AX)$ is **constructor-based** if each operation of SIG is either a **constructor** or a **defined function**. In accordance with Assumption 5.1 we regard all logical predicates of SIG as defined functions. A SIG-term over X consisting of constructors and variables is called a SIG**-normal form** over X. The S-sorted sets of all SIG-normal forms over X and all ground SIG-normal forms are denoted by $\mathbf{NF_{SIG}(X)}$ and $\mathbf{NF_{SIG}}$, respectively.

SP is **free-constructor-based** if for each $c = (l \equiv r \Leftarrow h) \in AX$, l contains a defined function. ❑

Definition 5.3 (reduction calculus) Let $SP = (SIG, AX)$ be a constructor-based specification. The **reduction calculus for** SP consists of the following inference rules each of which transforms a goal into a goal.

rewriting $\quad \dfrac{g[l\sigma/z]}{g[r\sigma/z] \cup h\sigma} \quad$ if $z \in var(g)$, $c = (l \equiv r \Leftarrow h) \in AX$ and $fresh(c)\sigma \subseteq NF_{SIG}(X)$

reflection $\quad \dfrac{g \cup \{t \equiv t\}}{g}$

A sequence $g_1, \ldots, g_n$ of goals is called a **goal reduction of** g_1 **into** g_n **upon** SP if for all $1 \leq i \leq n$, g_{i+1} is obtained from g_i by a single rewriting or reflection step. The corresponding inference relation is denoted by $\vdash_{SP}$. g is SP**-convergent** if $g \vdash_{SP} \emptyset$. ❑

Note that $g \vdash_{SP} \emptyset$ implies $g\sigma \vdash_{SP} \emptyset$ for all $\sigma \in Sub$. However, for a nonempty goal h, $g \vdash_{SP} h$ implies $g\sigma \vdash_{SP} h\sigma$ only if σ assigns normal forms to all goals of a goal reduction of g into h.

Definition 5.4 (reduction relation) Let $SP = (SIG, AX)$ be a constructor-based specification. The SP**-reduction relation** $\longrightarrow_{SP}$ is a binary relation both on $T_{SIG}(X)$ and on the set of goals over SIG defined as follows:

- $g \longrightarrow_{SP} g' \quad \Longleftrightarrow_{def} \quad g = t[l\sigma/x]$ and $g' = t[r\sigma/x]$ for some term (goal) t, $x \in var(t)$, $c = (l \equiv r \Leftarrow h) \in AX$ and $\sigma \in Sub$ such that $h\sigma$ is SP-convergent and $fresh(c)\sigma \subseteq NF_{SIG}(X)$.

g' is an SP**-reduct** of a term or goal g if $g \longrightarrow^*_{SP} g'$. A substitution τ is an SP**-reduct** of a substitution σ if $x\sigma \longrightarrow^*_{SP} x\tau$ for all $x \in X$. In this case we write $\sigma \longrightarrow^*_{SP} \tau$. g and g' are SP**-joinable**, written: $g \downarrow_{SP} g'$, if g and g' have a common SP-reduct. g is SP**-reducible into** each SP-reduct of g. If g is the only SP-reduct of g, then g is SP**-reduced**. A goal g is **strongly** SP**-convergent** if all SP-reducts of g are SP-convergent. ❑

Note that two terms t and t' are SP-joinable if and only if the equation $t \equiv t'$ is SP-convergent:

$$t \downarrow_{SP} t' \quad \text{iff} \quad t \equiv t' \vdash_{SP} \emptyset.$$

Definition 5.5 (normal form completeness, ground confluence) Let $SP = (SIG, AX)$ be a constructor-based specification. SP is **normal form complete** if each non-atomic ground SIG-term is SP-reducible into a SIG-normal form (cf. 5.1). SP is **ground confluent** if SP-convergence is closed under $\longrightarrow_{SP}$, i.e., if all SP-convergent ground goals are strongly SP-convergent. ❑

SP is normal form complete if and only if for all terms $F(t)$ where F is a defined function and t is a ground normal form there is an axiom $c = (F(u) \equiv v \Leftarrow h)$ such that t matches u, say $t = u\sigma$, and $h\sigma\tau$ is SP-convergent for some ground normal form substitution τ (which instantiates the fresh variables of c). On the one hand, this complies with constructor-basedness, on the other hand partially specified functions must be totalized by adding to the signature of SP constructor constants for expressing undefinedness such as *no_pair* of `DIVISION` (cf. Ex. 4.4) and axioms that define operations on these constants. Consistency problems arising from the addition of undefinedness constants vanish if SP is ground confluent because two different ground normal forms are joinable only if they are joinable by applying only constructor axioms. Since, by Thm. 5.8 below, ground confluence implies that two ground terms are SP-equivalent only if they are joinable, it is quite easy to avoid inconsistencies such as $succ(t) \equiv_{SP} \perp$ or $true \equiv_{SP} false$.

An important consequence of normal form completeness is the validity of the *only-if-completions* of defined functions:

Definition 5.6 (only-if-completion of a defined function) Let SP be a constructor-based specification, $F : w \to s$ be a defined function of SP and

$$\{c_1, \ldots, c_n\} \;=\; \{F(t_1) \equiv u_1 \Leftarrow g_1, \ldots, F(t_n) \equiv u_n \Leftarrow g_n\}$$

be the set of all axioms of SP with leading function symbol F. For all $1 \leq i \leq n$, let $X_i = var(c_i)$, $x \in X_w$ and $y \in X_s$ such that X_i does neither contain y nor components of x. The Gentzen clause

$$\{\exists X_1(\{x \equiv t_1, u_1 \equiv y\} \cup g_1), \ldots, \exists X_n(\{x \equiv t_n, u_n \equiv y\} \cup g_n)\} \;\Leftarrow\; F(x) \equiv y$$

is called the **only-if-completion** $ONLY(F)$ **of** F. ❑

Lemma 5.7 The inductive theory of a normal form complete specification SP includes the only-if-completion $ONLY(F)$ of each defined function F of SP.

Proof. Let $SP = (SIG, AX)$, $F : w \to s$ be a defined function of SP and

$$\{c_1, \ldots, c_n\} \;=\; \{F(t_1) \equiv u_1 \Leftarrow g_1, \ldots, F(t_n) \equiv u_n \Leftarrow g_n\}$$

be the set of all axioms of SP with leading function symbol F. For all $1 \leq i \leq n$, let X_i, x and y be as in Def. 5.6.

Let $\sigma \in GSub$ such that $SP \vdash_{cut} F(x\sigma) \equiv y\sigma$. Since SP is normal form complete, there are ground normal forms t, u with $x\sigma \longrightarrow^*_{SP} t$ and $F(t) \longrightarrow_{SP} u$. Hence $t = t_i\tau$, $u = u_i\tau$, $g_i\tau \vdash_{SP} \emptyset$ and $fresh(c_i)\tau \subseteq NF_{SIG}$ for some $1 \leq i \leq n$ and $\tau \in GSub$. $g_i\tau \vdash_{SP} \emptyset$ implies $SP \vdash_{cut} g_i\tau$. W.l.o.g. $\tau =_{\{x,y\}} \sigma$ because X_i does not contain neither y nor components of x. Hence $SP \vdash_{cut} x\tau = x\sigma \equiv t = t_i\tau$, $SP \vdash_{cut} y\tau = y\sigma \equiv F(x\sigma) \equiv F(t) \equiv u = u_i\tau$ and thus σ solves the conclusion of $ONLY(F)$. ❑

$\vdash_{SP}$ is sound w.r.t. the cut calculus for SP (cf. Section 4), i.e. for all goals g over SIG, $g \vdash_{SP} \emptyset$ implies $SP \vdash_{cut} g$ (cf. [33], Prop. 6.1). The converse coincides with ground confluence:

Theorem 5.8 (Church-Rosser Theorem I) A normal form complete specification $SP = (SIG, AX)$ is ground confluent iff for all ground (!) goals g, $SP \vdash_{cut} g$ implies $g \vdash_{SP} \emptyset$.

Proof. [33], Theorem 6.5 (2). ❑

In Section 1.4 we have listed four proof-theoretical consequences of ground confluence. The first two, *Base consistency* and *Constructors*, reflect the uniqueness of function values that is guaranteed if the specification is ground confluent. The proofs of Examples 4.3, 4.4 and 4.5 already indicate this relationship. If SP is not ground confluent, proofs often depend on particular lemmas that express the uniqueness of function values. If, however, SP is ground confluent, *narrowing-strategy-controlled goal generation* (cf. Section 8) becomes applicable and those lemmas need no longer be used. Section 6 will tell us how ground confluence is proved. In Sections 8 to 10 we discuss the benefits of ground confluence for expansion proofs.

Most design specifications are ground confluent because, in the course of their development, one has certain "canonical" models in mind, with normal forms as the data and with all operations defined uniquely on normal forms. The uniqueness of function values *is* the informal meaning of ground confluence. Formally, suppose that a ground term Ft has two normal forms, say $Ft \equiv_{SP} u$ and $Ft \equiv_{SP} v$. Then $u \equiv_{SP} v$, ground confluence implies that u and v are SP-joinable and thus, if SP is free-constructor-based, u and v are the same normal forms.

The method for proving ground confluence, which we propagate in the sequel, is different from *completing* SP into a confluent specification (cf. Sect. 1.2). We claim that most design specifications are either already confluent or the conditions imposed on them by completion methods are too restrictive. Here the goal is to prove and maintain ground confluence and not to enforce this property.

For this purpose, the *subreductive expansion calculus* was introduced in [33], Section 6.4. Let us reformulate this calculus as a further instance of the generic expansion calculus defined in Section 3.

The first question is: which Gentzen clauses must be *subreductively* valid in order to ensure ground confluence?

Definition 5.9 (critical clause) Let $SP = (SIG, AX)$ be a specification,

$$l \equiv r \Leftarrow g, \quad l' \equiv r' \Leftarrow h \ \in \ AX,$$

$t \in T_{SIG}(X)$, $x \in var(t)$ and σ, τ be minimal substitutions w.r.t. $\geq$ (cf. Sect. 2) such that $l\sigma = t[l'\tau/x]$ and a function symbol occurs in l at the position of x in t. Then $l\sigma$ is an **SP-redex overlay** and the clause

$$cc \quad = \quad l\sigma \equiv r\sigma \equiv t[r'\tau/x] \ \Leftarrow \ g\sigma \cup h\tau$$

is an **SP-critical clause induced at $l\sigma$.** ❑

The redex overlay $l\sigma$ is made part of cc only for technical reasons. In this way the redex overlay can be derived from the critical clause. Moreover, the variables of $l\sigma$ become input variables of cc, which is crucial for defining the parameter GS when, in the next section, generic expansions are specialized to subreductive expansions. Adding $l\sigma$ to cc does not affect the theory of SP: $cc \in ITh(SP)$ iff $r\sigma \equiv t[r'\tau/x] \ \Leftarrow \ g\sigma \cup h\tau \in ITh(SP)$.

In terms of CTRS theory (cf. Sect. 1.2), ground confluence follows from the convergence of all *feasible critical pairs*, which are the critical clauses with convergent premises, provided that SP is strongly terminating:

Definition 5.10 (strong termination) A constructor-based specification $SP = (SIG, AX)$ is **strongly terminating** if there is a transitive and well-founded relation $>_{SP}$ on T_{SIG} such that the constant *true* is minimal w.r.t. $>_{SP}$ (cf. 5.1) and the following two conditions hold true:

rewrite compatibility For all $c = (l \equiv r \Leftarrow h) \in AX$, $t \in T_{SIG}(\{x\})$ and $\sigma \in GSub$ such that $fresh(c)\sigma \subseteq NF_{SIG}$, $h\sigma \vdash_{SP} \emptyset$ implies $t[l\sigma/x] >_{SP} t[r\sigma/x]$ and $l\sigma >_{SP} h\sigma$.[6]

subterm compatibility For all $t \in T_{SIG}$ and all proper subterms u of t, $t >_{SP} u$.

Then $>_{SP}$ is a **reduction ordering for** SP. ❑

Definition 5.11 (canonical specification) A specification SP is **canonical** if SP is ground confluent, strongly terminating and normal form complete. ❑

Besides that fresh variables are admitted, rewriting compatibility is weaker here than in other CTRS approaches insofar as only *convergent* premise instances $h\sigma$ are considered here. This is sufficient for $\longrightarrow^{+}_{SP}$ (but not $\vdash_{SP}$) to be a subrelation of $>_{SP}$. The reduction ordering is a means for deriving ground confluence from convergence of critical clauses. The proof of this result is carried out by induction along $>_{SP}$.

The weakest notion of validity required for critical clauses is defined as follows.

Definition 5.12 (subreductive and reductive validity) Given a specification $SP = (SIG, AX)$, a reduction ordering $>_{SP}$ for SP and a SIG-term t, a Gentzen clause $c = gs \Leftarrow h$ is **sub-t-reductively valid w.r.t.** SP if for all $\sigma \in GSub$ such that $h\sigma$ is SP-convergent and all SP-convergent ground goals $g <_{SP} t\sigma$ are strongly SP-convergent, $g'\tau$ is SP-convergent[7] for some $g' \in gs$ and $\tau \in GSub$ with $\tau =_{in(c)} \sigma$.

c is **reductively valid w.r.t.** SP if for all $\sigma : X \to NF_{SIG}$ such that $h\sigma$ is strongly SP-convergent, $g\tau$ is strongly SP-convergent for some $g \in gs$ and $\tau \in GSub$ with $\tau =_{in(c)} \sigma$. ❑

Theorem 5.13 (Church-Rosser Theorem II) (cf. 5.8) A normal form complete specification SP is ground confluent iff all inductive theorems of SP (cf. 4.1) are reductively valid w.r.t. SP.

Proof. [33], Theorem 6.5(3). ❑

Theorem 5.14 (Superposition Theorem) Let SP be a strongly terminating and normal form complete specification. SP is ground confluent iff for all SP-critical clauses $t \equiv u \equiv v \Leftarrow h$, $u \equiv v \Leftarrow h$ is sub-t-reductively valid w.r.t. SP.

Proof. [33], Theorem 6.10 and Lemma 6.13. ❑

Theorem 5.14 generalizes the *Buchberger-Newman Lemma* and the *Knuth-Bendix Lemma* for the *ground case* (cf. [27], Lemmas 7 and 8) from unconditional rewrite systems to Horn clauses. These central results of rewrite theory comprise the three-step reduction of each proof of (ground) confluence that uses a reduction ordering $>_{SP}$: (1) from confluence to *local* confluence, (2) from local confluence to the convergence of *critical pairs*, (3) from the convergence of a critical pair cp to an equational proof EP of (u, v) such that EP is *smaller* than the *rewriting ambiguity* $u \leftarrow t \rightarrow v$ that induces (u, v). Actually, this means that all terms of EP are smaller w.r.t. $>_{SP}$ than the redex overlay t (cf. Def. 5.9), i.e. EP consists of t-bounded terms. In the general case treated here we have critical clauses instead of critical pairs and expansions consisting of t-bounded goals instead of equational proofs consisting of t-bounded terms.

6 Subreductive expansion

Let $SP = (SIG, AX)$ be a specification. The parameters $TH = GTh(AX)$ and $GS = GSub$ provided the first instance of the generic expansion calculus defined in Section 3. It led to inductive expansions, which were treated in Section 4. The previous section motivates a second instance of the generic expansion calculus. Here

[6] By 5.1, h is a set of equations, say $h = \{t_1 \equiv u_1, \ldots, t_n \equiv u_n\}$. Hence $t\sigma >_{SP} h\sigma$ means $t\sigma >_{SP} t_i$ and $t\sigma >_{SP} u_i$ for all $1 \leq i \leq n$.

[7] [33] requires *strong* convergence, but a short glance at the proof of [33], Thm. 6.10, reveals that convergence is sufficient.

the parameters TH and GS depend on a given **input term**, i.e. a SIG-term $\mathbf{t}$ with $var(t) \subseteq X_{in}$.

Let $>_{SP}$ be a reduction ordering for SP and t be an input term. $\mathbf{RTh(SP)}$ denotes the set of all SP-convergent ground goals and $\mathbf{GSub}(>_{\mathbf{SP}}, \mathbf{t})$ stands for the set of all ground substitutions σ such that all SP-convergent ground goals $g <_{SP} t\sigma$ are strongly SP-convergent.

Let $\mathbf{TH} = \mathbf{RTh(SP)}$ and $\mathbf{GS} = \mathbf{GSub}(>_{\mathbf{SP}}, \mathbf{t})$. Then the conditions on TH and GS given in Section 3 hold true because t is an input term. Moreover, a Gentzen clause c is sub-t-reductively valid w.r.t. SP iff c follows from TH by induction on GS (cf. 3.1).

Definition 6.1 A goal g is t**-bounded** if for all $\sigma \in GSub$, $g\sigma \vdash_{SP} \emptyset$ implies $t\sigma >_{SP} g\sigma$. ❑

Proposition 6.2 t-bounded goals are (TH, GS)-equivalence compatible (cf. 3.2).

Proof. Let $g[u/x]$ be a t-bounded goal and v be a SIG-term. The clause

$$c \quad = \quad g[v/x] \Leftarrow g[u/x] \cup \{u \equiv v\}$$

must follow from TH by induction on GS. Hence suppose that $(g[u/x] \cup \{u \equiv v\})\sigma \in TH$ for some $\sigma \in GS$, i.e. $g[u/x]\sigma$ is SP-convergent, u and v are SP-joinable and all SP-convergent ground goals $h <_{SP} t\sigma$ are strongly SP-convergent. This implies $g[v/x]\sigma \vdash_{SP} \emptyset$ because $g[u/x]$ is t-bounded. We conclude that c follows from TH by induction on GS. ❑

Axioms with t-bounded premises are sub-t-reductively valid if SP is strongly terminating and normal form complete:

Proposition 6.3 Suppose that $SP = (SIG, AX)$ is strongly terminating and normal form complete. $c = (l \equiv r \Leftarrow h) \in AX$ is sub-t-reductively valid w.r.t. SP if h is t-bounded.

Proof. Let $\sigma \in GS$ such that $h\sigma$ is SP-convergent. Since h is t-bounded, $t\sigma >_{SP} h\sigma$. Hence $h\sigma$ is strongly SP-convergent because $\sigma \in GS$. Since SP be normal form complete, there is $\tau : X \to NF_{SIG}$ such that $x\sigma \longrightarrow^*_{SP} x\tau$ for all $x \in var(c)$. Hence $h\tau$ is SP-convergent because $h\sigma$ is strongly SP-convergent. This implies $l\tau \longrightarrow_{SP} r\tau$ by the definition of $\longrightarrow_{SP}$. Hence $l\sigma \equiv r\sigma$ is SP-convergent. We conclude that c is sub-t-reductively valid w.r.t. SP. ❑

In particular, all unconditional axioms are sub-t-reductively valid.

Since GS is a proper subset of $GSub$, the condition

$$\tau \in GS \quad \text{implies} \quad \rho\sigma\tau \in GS$$

of inductive resolution and paramodulation (cf. 3.5) does not hold trivially as in the case of inductive expansions (cf. Sect. 4). But is easy to see that this implication holds true whenever $t\tau >_{SP} t\rho\sigma\tau$. This condition complies well with the descent condition

$$x_{in} >> x_{in}\rho\sigma \equiv true$$

generated by inductive resolution and paramodulation. Also taking into account Prop. 6.2, the rules of sub-t-reductive expansion are thus simplified as follows (cf. Section 3): Let CS be a set of Gentzen clauses and $\rho : X \to X$ be a renaming of X_{in} away from X_{in}.

Backward deductive resolution Let $\sigma \in Sub$ and $\{\exists X_1 h_1\sigma, \ldots, \exists X_n h_n\sigma\} \Leftarrow h \in Gen(TH, GS)$ such that for all $1 \leq i \leq n$, $h_i\sigma$ is t-bounded and $X_i \cap var(g_i\sigma \cup X_{in}\sigma \cup X_{in}) = \emptyset$.

$$\frac{\{g_1 \cup h_1, \ldots, g_n \cup h_n\}}{\{g_1\sigma \cup \ldots \cup g_n\sigma \cup h \cup EQ(\sigma_{in})\}}$$

Backward deductive paramodulation Let $\sigma \in Sub$, $\{\exists X_1(u \equiv t_1)\sigma, \ldots, \exists X_n(u \equiv t_n)\sigma\} \Leftarrow h \in$

$Gen(TH,GS)$ and $x \in var(g_1 \cap \ldots \cap g_n)$ such that for all $1 \le i \le n$, $X_i \cap var(g_i[u/x]\sigma \cup X_{in}\sigma \cup X_{in}) = \emptyset$.

$$\frac{\{g_1[t_1/x], \ldots, g_n[t_n/x]\}}{\{g_1[u/x]\sigma \cup \ldots \cup g_n[u/x]\sigma \cup h \cup EQ(\sigma_{in})\}}$$

Backward inductive resolution Let $c = (\{\exists X_1 h_1, \ldots, \exists X_n h_n\} \Leftarrow h) \in CS$ and $\sigma \in Sub$ be a renaming of $Z = X_1 \cup \ldots \cup X_n$ away from $var(c)$ such that for all $1 \le i \le n$, $f_i\sigma = h_i\rho\sigma$ is t-bounded and $Z\sigma \cap var(g_i\sigma \cup X_{in}\sigma \cup X_{in}) = \emptyset$, and for all $\tau \in GSub$, $t\tau >_{SP} t\rho\sigma\tau$.

$$\frac{\{g_1 \cup f_1, \ldots, g_n \cup f_n\}}{\{g_1\sigma \cup \ldots \cup g_n\sigma \cup h\rho\sigma \cup \{x_{in} >> x_{in}\rho\sigma \equiv true\} \cup EQ(\sigma_{in})\}}$$

Backward inductive paramodulation Let $c = (\{\exists X_1(u \equiv t_1), \ldots, \exists X_n(u \equiv t_n)\} \Leftarrow h) \in CS$, $x \in var(g_1 \cap \ldots \cap g_n)$ and $\sigma \in Sub$ be a renaming of $Z = X_1 \cup \ldots \cup X_n$ away from $var(c)$ such that for all $1 \le i \le n$, $u_i\sigma = t_i\rho\sigma$ and $Z\sigma \cap var(g_i[u\rho/x]\sigma \cup X_{in}\sigma \cup X_{in}) = \emptyset$, and for all $\tau \in GSub$, $t\tau >_{SP} t\rho\sigma\tau$.

$$\frac{\{g_1[u_1/x], \ldots, g_n[u_n/x]\}}{\{g_1[u\rho/x]\sigma \cup \ldots \cup g_n[u\rho/x]\sigma \cup h\rho\sigma \cup \{x_{in} >> x_{in}\rho\sigma \equiv true\} \cup EQ(\sigma_{in})\}}$$

t-boundedness also replaces equivalence compatibility in the definition of a sub-t-reductive expansion:

Definition 6.4 A sequence $E = (gs_1, \ldots, gs_n)$ of goal sets is called a **sub-t-reductive expansion upon** SP if

- for all $1 \le i < n$, gs_{i+1} is obtained from gs_i by a single deductive or inductive resolution or paramodulation step as defined above,
- for all $1 \le i \le n$, gs_i consists of t-bounded goals.

If E uses induction hypotheses from a set CS of Gentzen clause, we write $gs_1 \vdash^{\mathbf{t}}_{\mathbf{SP},\mathbf{CS}} gs_n$. ❑

Analogously to Thm. 4.2, we obtain a special case of Thm. 3.6(1) and Cor. 3.9, respectively:

Theorem 6.5 (subreductive expansions are sound)

(1) Let $c = \{g_1, \ldots, g_m\} \Leftarrow h$ be a Gentzen clause and $CS = \{gs_1 \Leftarrow h_1, \ldots, gs_n \Leftarrow h_n\}$ be a set of Gentzen clauses such that for all $1 \le i \le m$ and $1 \le j \le n$, $h \Leftarrow h_j, gs_j \Leftarrow g_i \in Gen(TH,GS)$. Then $\{g_1, \ldots, g_m\} \vdash^t_{SP,CS} \{h\}$ implies $\{c\} \cup CS \subseteq Gen(TH,GS)$.

(2) Let $HS = \{g_1 \Leftarrow g : h_1, \ldots, g_n \Leftarrow g : h_n\}$ be a set of guarded clauses such that $\{h_1, \ldots, h_n\}$ is g-minimal w.r.t. (TH,GS) and for all $1 \le i,j \le n$, $X_i =_{def} fresh(g_i \Leftarrow g \cup h_i)$ and $h_i = h_j$ implies $g_i = g_j$. Let

$$c \;=\; \{\exists X_1(g_1 \cup h_1), \ldots, \exists X_n(g_n \cup h_n)\} \Leftarrow g.$$

Then $\{\exists X_1(g_1 \cup h_1), \ldots, \exists X_n(g_n \cup h_n)\} \vdash^t_{SP,HS} \{g\}$ implies $\{c\} \cup HS \subseteq Gen(TH,GS)$. ❑

Theorems 5.14 and 6.5 justify the method for proving ground confluence by subreductive expansion:

Theorem 6.6 (criterion for ground confluence) Suppose that SP is strongly terminating and normal form complete. If for all SP-critical clauses $t \equiv u \equiv v \Leftarrow h$,

$$\{u \equiv v\} \;\vdash^t_{SP,\{u \equiv v \Leftarrow h\}}\; h,$$

then SP is ground confluent. ❑

Theorem 6.6 allows us to apply the critical clause c as an induction hypothesis when proving that c is subreductively valid. Moreover, Σ-simplification (cf. Sect. 4) can be applied to a goal set gs of a sub-t-reductive expansion without violating its soundness if $gs \Leftrightarrow \Sigma(gs)$ is sub-t-reductively valid.

In closing this section, we stress the point that, for proving ground confluence, the subreductive validity of a critical clause c or of a lemma used in an expansion of c cannot be concluded from its inductive validity with the help of Theorem 5.13. This theorem *assumes* that SP is ground confluent, while a subreductive expansion is a part of the *proof* that SP is ground confluent.

7 Strong termination and t-boundedness

Theorem 6.6 provides a method for showing ground confluence: construct subreductive expansions of critical clauses. Yet it lacks criteria for the strong termination of SP and the t-boundedness of the goals of a sub-t-reductive expansion (cf. 6.4). Sufficient conditions for both properties are obtained on the basis of a systematic construction of reduction orderings. To this end we have introduced the *path calculus* for deriving valid pairs $t >_{SP} u$ where t and u are terms, equations or multisets of terms and equations (cf. [33], Section 6.2). A reduction ordering $>_{SP}$ extends to multisets of terms and equations as follows:

- $M >_{SP} N$ iff $M \neq N$ and for all $u \in N \setminus M$ there is $t \in M \setminus N$ such that $t >_{SP} u$,
- $M >_{SP} (t \equiv t')$ iff $M >_{SP} \{t, t'\}$,
- $(t \equiv t') >_{SP} M$ iff $\{t, t'\} >_{SP} M$.

We remind the reader of Assumption 5.1 that all logical predicates of SIG are regarded as operations and, consequently, the logical atom $P(t)$ is actually an abbreviation of the equation $P(t) \equiv true$.

Definition 7.1 (path calculus) Suppose that

- $\geq_{SIG}$ is a reflexive and transitive relation on the set of operations and logical predicates of SIG,
- $\succ$ is a set of predicates of SIG with transitive and well-founded interpretations in $Ini(SP)$ (cf. 4.1).

The **$(\geq_{SIG}, \succ)$-path calculus for SP** consists of the following inference rules for deriving goal sets where the atoms are SIG-atoms and expressions of the form $t \gg u$ such that t and u are terms, equations or multisets of terms and equations.

Multiset rules Let $true \neq t \neq \emptyset$, $t' \neq \emptyset \neq u \cup u'$ and $t \cup u \neq t' \cup u'$.

$$\frac{\{(t \cup u) \gg (t' \cup u')\}^8}{\{t \gg t',\ (t \cup u) \gg u'\}} \qquad \frac{\{(t \cup u) \gg u\}}{TRUE} \qquad \frac{\{t \gg u \equiv u'\}}{\{t \gg \{u, u'\}\}} \qquad \frac{\{u \equiv u' \gg t'\}}{\{\{u, u'\} \gg t'\}} \qquad \frac{\{t \gg true\}}{TRUE}$$

Subterm rules Let $F(t_1, \ldots, t_k)$ be a term and $1 \leq i \leq k$.

$$\frac{\{F(t_1, \ldots, t_k) \gg t\}}{\{t_i \gg t\}} \qquad \frac{\{F(t_1, \ldots, t_k) \gg t_i\}}{TRUE}$$

***SIG*-rules** Let t and $F(t_1, \ldots, t_k)$ be terms such that $F \geq_{SIG} G$, but not $G \geq_{SIG} F$.

$$\frac{\{t \gg F(t_1, \ldots, t_k)\}}{\{t \gg \{t_1, \ldots, t_k\}\}}$$

Let $F(t_1,\dots,t_k)$ and $G(u_1,\dots,u_n)$ be terms such that $F \geq_{SIG} G$ and $G \geq_{SIG} F$.

$$\frac{\{F(t_1,\dots,t_k) \gg G(u_1,\dots,u_n)\}}{\begin{array}{l}\{(t_1,\dots,t_k) \succ (u_1,\dots,u_n),\ F(t_1,\dots,t_k) \gg \{u_1,\dots,u_n\}\}\\ \vee\ \{(t_1,\dots,t_k) \equiv (u_1,\dots,u_n),\ \{t_1,\dots,t_k\} \gg \{u_1,\dots,u_n\}\}\end{array}} \quad \text{if } k = n$$

$$\frac{\{F(t_1,\dots,t_k) \gg G(u_1,\dots,u_n)\}}{\{(t_1,\dots,t_k) \succ (u_1,\dots,u_n),\ F(t_1,\dots,t_k) \gg \{u_1,\dots,u_n\}\}} \quad \text{if } k \neq n$$

The corresponding inference relation is denoted by $\vdash^{\mathbf{path}}$.

Given a set Sol of ground substitutions, we add a further rule to the $(\geq_{SIG},\succ)$-path calculus:

Sol-**rule** $$\frac{\{t \gg x\}}{\{t \gg x\sigma \mid \sigma \in Sol\}} \qquad \text{if} \quad x \in X \setminus var(t)$$

The inference relation of the path calculus for $(\geq_{SIG},\succ)$ extended by the *Sol*-rule is denoted by $\vdash^{\mathbf{path}}_{\mathbf{Sol}}$. ❑

A path calculus for SP does not only depend on the syntax of SP via the signature ordering $\geq_{SIG}$ and binary predicates $\succ$, but also takes into account the semantics of SP by the requirement that $\succ$ is interpreted in $Ini(SP)$ as a set of well-founded relations. This is the same condition a descent function $>>$ has to satisfy (cf. Section 3). If $\succ$ stems from descent functions, we might say that the *reduction* ordering $>_{SP}$ defined below is based on *induction* orderings. Indeed, for applying inductive rules in subreductive expansions, $>>$ must be included into $\succ$ (see below).

Theorem 7.2 (reduction ordering defined by a path calculus) Given a path calculus for SP, define a binary relation $>_{SP}$ on T_{SIG} as follows: For all ground terms t, u,

$$t >_{SP} u \quad \text{iff} \quad \{t \gg u\} \vdash^{path} gs \text{ for a goal set } gs \text{ such that } SP \vdash_{cut} g \text{ for some } g \in gs$$

(cf. Sect. 4). $>_{SP}$ is a reduction ordering for SP (cf. 5.10) if for all $c = (l \equiv r \Leftarrow h) \in AX$ the following condition holds true:

(1) $l\sigma >_{SP} \{r\sigma\} \cup h\sigma$ for all $\sigma \in GSub$ such that $h\sigma$ is SP-convergent and $fresh(c)\sigma \subseteq NF_{SIG}(X)$.[9]

Proof. [33], Theorem 6.7. ❑

Condition 7.2(1) can also be proved with the given path calculus for SP, extended by the *Sol*-rule where *Sol* is the set of ground solutions of h:

Corollary 7.3 (criterion for strong termination) Let $>_{SP}$ be defined as in 7.2 and

$$Sol \quad =_{def} \quad \{\sigma \in GSub \mid SP \vdash_{cut} h\sigma,\ fresh(c)\sigma \subseteq NF_{SIG}(X)\}.$$

$>_{SP}$ is a reduction ordering for SP if all $(l \equiv r \Leftarrow h) \in AX$ are SP-**decreasing**, i.e. there is a goal set gs such that

$$\{l \gg \{r\} \cup h\} \ \vdash^{path}_{Sol} \ gs$$

and $gs \Leftarrow h$ is an inductive SP-theorem (cf. 4.1).

Proof. Let $(l \equiv r \Leftarrow h) \in AX$ and $\sigma \in GSub$ such that $h\sigma$ is SP-convergent and $fresh(c)\sigma \subseteq NF_{SIG}(X)$. Hence $SP \vdash_{cut} h\sigma$ and thus $\sigma \in Sol$. Let $\{l \gg \{r\} \cup h\} \vdash^{path}_{Sol} gs$ such that $gs \Leftarrow h \in ITh(SP)$. Then

[8] $\cup$ denotes multiset union.

[9] The extension of $>_{SP}$ to multisets is defined as above.

$\{l\sigma \gg \{r\sigma\} \cup h\sigma\} \vdash^{path} gs\sigma$ and $SP \vdash_{cut} gs\sigma$. The definition of $>_{SP}$ implies $l\sigma >_{SP} \{r\sigma\} \cup h\sigma$. Hence 7.2(1) holds true and we conclude that $>_{SP}$ is a reduction ordering for SP. ❑

With regard to the axiom $l \equiv r \Leftarrow h$, Corollary 7.3 yields a criterion for the l-boundedness of h. The general t-boundedness criterion reads as follows:

Lemma 7.4 (criterion for t-boundedness) Let $>_{SP}$ be defined as in 7.2, t be an input term (cf. Sect. 6), g be a goal and

$$Sol \quad =_{def} \quad \{\sigma \in GSub \mid SP \vdash_{cut} g\sigma\}.$$

g is t-bounded (cf. 6.1) if there is a goal set gs such that

$$\{t \gg g\} \vdash^{path}_{Sol} gs$$

and $gs \Leftarrow g$ is an inductive SP-theorem.

Proof. Let $\sigma \in GSub$ such that $g\sigma$ is SP-convergent. Hence $SP \vdash_{cut} g\sigma$ and thus $\sigma \in Sol$. Let $\{t \gg g\} \vdash^{path}_{Sol} gs$ such that $gs \Leftarrow g \in ITh(SP)$. Then $\{t\sigma \gg g\sigma\} \vdash^{path} gs\sigma$ and $SP \vdash_{cut} gs\sigma$. The definition of $>_{SP}$ implies $t\sigma >_{SP} g\sigma$. ❑

The path calculus defining $>_{SP}$ can also be used to show the condition

(1) $t\tau >_{SP} t\rho\sigma\tau$

on inductive steps of subreductive expansions (cf. 6.4). Since (1) comes together with the descent condition $x_{in} >> x_{in}\rho\sigma \equiv true$, (1) can be guaranteed if $>>$ is included in the parameter $\succ$ of the path calculus defining $>_{SP}$. Both $>>$ and $\succ$ are required to have well-founded interpretations in $Ini(SP)$. If, in addition, $F >_{SIG} G$ for the leftmost symbol F of t and all operations G of $x_{in}\rho\sigma$, then, indeed, (1) can be derived by applying SIG-rules (cf. 7.1). $F >_{SIG} G$ holds true if F is a defined function, G is a constructor (cf. 5.2) and, as usual, defined functions dominate constructors with respect to the symbol ordering $\geq_{SIG}$ of the path calculus.

Example 7.5 (`STACK_AS_MAP`) Using Expander syntax (cf. Section 1.1) we specify finite stacks, finite maps (arrays, RAMs) and an implementation of stacks by pairs of a map and a top index. This example is a popular "benchmark" for formal refinement approaches (cf. e.g. [16], Sect. 4.4; [33], Ex. 7.20; [28], Sect. 4.1). As constructor-based specifications, `STACK` and `MAP` have separate lists for constructors (`consts`) and defined functions (`defs`) (cf. 5.2).

```
STACK
    consts    0 empty push
    defs      pop 1 2 top 3 4
    vars      x st
    axioms    (1)  {pop(empty) = empty}
              (2)  {pop(push(x,st)) = st}
              (3)  {top(empty) = 0}
              (4)  {top(push(x,st)) = x}

MAP
    consts    0 new put >
    defs      <> get 3 4 5
    vars      i j x y f
    axioms    (1)  {put(i,x,put(i,y,f)) = put(i,x,f)}
              (2)  {put(i,x,put(j,y,f)) = put(j,y,put(i,x,f))} <== {j > i}
              (3)  {get(new,i) = 0}
```

```
            (4)  {get(put(i,x,f),i) = x}
            (5)  {get(put(i,x,f),j) = get(f,j)} <== {i <> j}
```

The implementation is given by an extension of `MAP`:

```
STACK_AS_MAP
   base      MAP
   consts    s mkStack
   defs      empty 1 push 2 pop 3 4 top 5
   vars      st
   axioms    (1)  {empty = mkStack(new,0)}
             (2)  {push(x,mkStack(f,i)) = mkStack(put(s(i),x,f),s(i))}
             (3)  {pop(mkStack(f,0)) = mkStack(f,0)}
             (4)  {pop(mkStack(f,s(i))) = mkStack(f,i)}
             (5)  {top(mkStack(f,i)) = get(f,i) }
             (6)  {mkStack(put(i,x,f),j) = mkStack(f,j)} <== {i > j}
```

Axioms 1 to 5 realize the operations of `STACK` in terms of `MAP`. `mkStack` is the *abstraction function* from pairs of a map and an index to stacks. Axiom 6 defines the equivalence relation on stacks induced by `mkStack`. It says that all updates of f at indices i greater than the stacktop index j do not change the stack implemented by f. Without going into the discussion of what makes an implementation correct in general we claim that the following three conditions are reasonable requirements for calling `STACK_AS_MAP` a correct implementation of `STACK` by `MAP`.

(1) The union SP of `STACK` and `STACK_AS_MAP` is ground confluent.

(2) The axioms of `STACK` are inductive `STACK_AS_MAP`-theorems.

(3) Axiom 6 of `STACK_AS_MAP` is an inductive SP'-theorem where $SP' = SP \setminus \{Axiom\ 6\}$.

Proof of (1). In order to apply Thm. 6.6 we first note that SP is constructor-based (cf. 5.2). It might look strange to regard the predicate `>` as a constructor of the subspecification `MAP` of SP. But we are forced to do so because `>` occurs in a constructor axiom of `MAP`. SP is also normal form complete and strongly terminating. For proving the latter we refer to Cor. 7.3. The signature ordering $\geq_{SIG}$ of a suitable path calculus for SP can be defined as the transitive closure of

$$\{pop, top\} \geq_{SIG} \{empty, push\} \geq_{SIG} \{mkStack\} \geq_{SIG} \{get\} \geq_{SIG} \{new, put\} \geq_{SIG} \{0, s, >, <>\}.$$

For ensuring that Axiom 2 of `MAP` is SP-decreasing (cf. 7.3), SP must be extended by predicates $>_3$ and $>_{map}$ such that

```
(i,x,f) >3 (j,y,f)  <==  i > j
(i,x,f) >3 (j,y,g)  <==  f >map g
put(i,x,f) >map put(j,y,g)  <==  (i,x,f) >3 (j,y,g)
{put(i,x,f) >map f}\/{put(i,x,f) = f}
```

are inductive SP-theorems. The predicate `>` of `MAP`, $>_3$ and $>_{map}$ are included in the path calculus parameter $\succ$ (cf. 7.1). Under these assumptions we obtain the following proof that Axiom 2 of `MAP` is SP-decreasing. It involves applications of both the $(\geq_{SIG}, \succ)$-path calculus and the inductive expansion calculus (cf. 4.1).

$$
\begin{array}{ll}
& \{put(i,x,put(j,y,f)) \gg \{put(j,y,put(i,x,f)),\ j>i\}\} \\
\vdash^{path} & \{put(i,x,put(j,y,f)) \gg put(j,y,put(i,x,f))\} \\
\vdash^{path} & \{\{(i,x,put(j,y,f)) >_3 (j,y,put(i,x,f)),\ put(i,x,put(j,y,f)) \gg \{j,y,put(i,x,f)\}\}, \\
& \{(i,x,put(j,y,f)) \equiv (j,y,put(i,x,f)),\ \{i,x,put(j,y,f)\} \gg \{j,y,put(i,x,f)\}\}\} \\
\vdash_{SP,\emptyset} & \{(i,x,put(j,y,f)) >_3 (j,y,put(i,x,f)),\ put(i,x,put(j,y,f)) \gg \{j,y,put(i,x,f)\}\} \\
\vdash_{SP,\emptyset} & \{put(j,y,f) >_{map} put(i,x,f),\ put(i,x,put(j,y,f)) \gg put(i,x,f)\} \\
\vdash_{SP,\emptyset} & \{j>i,\ put(i,x,put(j,y,f)) \gg put(i,x,f)\} \\
\vdash^{path} & \{\{j>i,\ (i,x,put(j,y,f)) >_3 (i,x,f),\ put(i,x,put(j,y,f)) \gg \{i,x,f\}\}, \\
& \{j>i,\ (i,x,put(j,y,f)) \equiv (i,x,f),\ \{i,x,put(j,y,f)\} \gg \{i,x,f\}\}\} \\
\vdash^{path} & \{\{j>i,\ (i,x,put(j,y,f)) >_3 (i,x,f)\}, \\
& \{j>i,\ (i,x,put(j,y,f)) \equiv (i,x,f)\}\} \\
\vdash_{SP,\emptyset} & \{\{j>i,\ put(j,y,f) >_{map} f\}, \\
& \{j>i,\ put(j,y,f) \equiv f\}\} \\
\vdash_{SP,\emptyset} & \{j>i\}
\end{array}
$$

Next we must find out the set CC of SP-critical clauses (cf. 5.9, 6.6). We leave it to the reader to construct (the 8 elements of) the set $CC($`MAP`$)$ of `MAP`-critical clauses. The rest of CC is given by the following clauses:

```
 (1)  pop(empty) = empty = pop(mkStack(new,0))
 (2)  pop(push(x,mkStack(f,i))) = mkStack(f,i) = pop(mkStack(put(s(i),x,f),s(i)))
 (3)  top(empty) = 0 = top(mkStack(new,0))
 (4)  top(push(x,mkStack(f,i))) = x = top(mkStack(put(s(i),x,f),s(i)))
 (5)  push(x,mkStack(put(i,y,f),j)) = mkStack(put(s(j),x,put(i,y,f),n,x),s(j))
        = push(x,mkStack(f,j))  <==  i > j
 (6)  pop(mkStack(put(i,x,f),0)) = mkStack(put(i,x,f),0) = pop(mkStack(f,0))  <==  i > 0
 (7)  pop(mkStack(put(i,x,f),s(j))) = mkStack(put(i,x,f),j) = pop(mkStack(f,s(j)))
        <==  i > s(j)
 (8)  top(mkStack(put(i,x,f),j)) = get(put(i,x,f),j) = top(mkStack(f,j))  <==  i > j
 (9)  mkStack(put(i,x,put(i,y,f)),k) = mkStack(put(i,y,f),k) = mkStack(put(i,x,f),k)
        <==  i > k
(10)  mkStack(put(i,x,put(j,y,f)),k) = mkStack(put(j,y,f),k)
        = mkStack(put(j,y,put(i,x,f)),k)  <==  j > i > k
```

By Thm. 6.6, it remains to show $\{u \equiv v\} \vdash^{t}_{SP,\{u\equiv v \Leftarrow h\}} h$ for all $\{t \equiv u \equiv v\} \Leftarrow h \in CC$. We present subreductive expansions for the above 10 clauses, corresponding ones for $CC($`MAP`$)$ are left to the reader. That each goal of these expansions is t-bounded can be shown easily with the help of Lemma 7.4 based on the above-sketched path calculus.

```
initial conclusion:
 (1) { empty = pop(mkStack(new,0)) }
term at position 1 2 in conclusion goal 1 replaced with axiom STACK_AS_MAP3
conclusion:
 (1) { empty = mkStack(new,0) }
atom 1 in conclusion goal 1 replaced with axiom STACK_AS_MAP1
conclusion:
 (1) TRUE

initial conclusion:
 (1) { mkStack(f,i) = pop(mkStack(put(s(i),x,f),s(i))) }
```

```
term at position 1 2 in conclusion goal 1 replaced with axiom STACK_AS_MAP3
conclusion:
 (1) { mkStack(f,i) = mkStack(put(s(i),x,f),i) }
term at position 1 2 in conclusion goal 1 replaced with axiom STACK_AS_MAP6
conclusion:
 (1) { s(i) > i }
atom 1 in conclusion goal 1 replaced with theorem s(i)>i
conclusion:
 (1) TRUE

initial conclusion:
 (1) { 0 = top(mkStack(new,0)) }
term at position 1 2 in conclusion goal 1 replaced with axiom STACK_AS_MAP5
conclusion:
 (1) { 0 = get(new,0) }
term at position 1 2 in conclusion goal 1 replaced with axiom MAP3
conclusion:
 (1) TRUE

initial conclusion:
 (1) { x = top(mkStack(put(s(i),x,f),s(i))) }
term at position 1 2 in conclusion goal 1 replaced with axiom STACK_AS_MAP5
conclusion:
 (1) { x = get(put(s(i),x,f),s(i)) }
term at position 1 2 in conclusion goal 1 replaced with axiom MAP4
conclusion:
 (1) TRUE

initial conclusion:
 (1) { mkStack(put(s(j),x,put(i,y,f)),s(j)) = push(x,mkStack(f,j)) }
initial premise:
 (1) { i > j }
term at position 2 1 in conclusion goal 1 replaced with axiom STACK_AS_MAP2
conclusion:
 (1) { mkStack(put(s(j),x,put(i,y,f)),s(j)) = mkStack(put(s(j),x,f),s(j)) }
term at position 1 1 1 in conclusion goal 1 replaced with axiom MAP1
conclusion:
 (1) { mkStack(put(s(j),x,f),s(j)) = mkStack(put(s(j),x,f),s(j)), i = s(j) }
 (2) { mkStack(put(s(j),x,put(i,y,f)),s(j)) = mkStack(put(s(j),x,f),s(j)) }
term at position 1 1 1 in conclusion goal 2 replaced with axiom MAP2
conclusion:
 (1) { i = s(j) }
 (2) { mkStack(put(i,y,put(s(j),x,f)),s(j)) = mkStack(put(s(j),x,f),s(j)), i > s(j) }
atom 1 in conclusion goal 2 replaced with axiom STACK_AS_MAP6
conclusion:
 (1) { i = s(j) }
 (2) { i > s(j) }
atoms 1 1 in conclusion goals 1 2 replaced with theorem i=s(j)\/i>s(j) <== i>j
```

```
conclusion:
 (1) { i > j }

initial conclusion:
 (1) { mkStack(put(i,x,f),0) = pop(mkStack(f,0)) }
initial premise:
 (1) { i > 0 }
term at position 2 1 in conclusion goal 1 replaced with axiom STACK_AS_MAP3
conclusion:
 (1) { mkStack(put(i,x,f),0) = mkStack(f,0) }
atom 1 in conclusion goal 1 replaced with axiom STACK_AS_MAP6
conclusion:
 (1) { i > 0 }

initial conclusion:
 (1) { mkStack(put(i,x,f),j) = pop(mkStack(f,s(j))) }
initial premise:
 (1) { i > s(j) }
term at position 2 1 in conclusion goal 1 replaced with axiom ARRAY2STACK4
conclusion:
 (1) { mkStack(put(i,x,f),j) = mkStack(f,j) }
atom 1 in conclusion goal 1 replaced with axiom STACK_AS_MAP6
conclusion:
 (1) { i > j }
atom 1 in conclusion goal 1 replaced with theorem i>j <== i>s(j)
conclusion:
 (1) { i > s(j) }

initial conclusion:
 (1) { get(put(i,x,f),j) = top(mkStack(f,j)) }
initial premise:
 (1) { i > j }
term 1 1 in conclusion goal 1 replaced with axiom MAP5
conclusion:
 (1) { get(f,j) = top(mkStack(f,j)), i <> j }
term 1 2 in conclusion goal 1 replaced with axiom STACK_AS_MAP5
conclusion:
 (1) { i <> j }
atom 1 in conclusion goal 1 replaced with theorem i<>j <== i>j
conclusion:
 (1) { i > j }

initial conclusion:
 (1) { mkStack(put(i,y,f),k) = mkStack(put(i,x,f),k) }
initial premise:
 (1) { i > k }
term 1 1 in conclusion goal 1 replaced with axiom STACK_AS_MAP6
conclusion:
```

```
 (1) { mkStack(f,k) = mkStack(put(i,x,f),k), i > k }
term 2 1 in conclusion goal 1 replaced with axiom STACK_AS_MAP6
conclusion:
 (1) { i > k }

initial conclusion:
 (1) { mkStack(put(j,y,f),k) = mkStack(put(j,y,put(i,x,f)),k) }
initial premise:
 (1) { j > i, i > k }
term at position 1 1 in conclusion goal 1 replaced with axiom STACK_AS_MAP6
conclusion:
 (1) { mkStack(f,k) = mkStack(put(j,y,put(i,x,f)),k), j > k }
term at position 1 2 in conclusion goal 1 replaced with axiom STACK_AS_MAP6
conclusion:
 (1) { mkStack(f,k) = mkStack(put(i,x,f),k), j > k }
term at position 1 2 in conclusion goal 1 replaced with axiom STACK_AS_MAP6
conclusion:
 (1) { j > k, i > k }
atom 1 in conclusion goal 1 replaced with theorem j>k <== j>i>k
conclusion:
 (1) { j > i, i > k }
```

Proof of (2). All axioms of `STACK` are ground reducible w.r.t. `STACK_AS_MAP`:

Definition (ground reducibility) Let $SP = (SIG, AX)$ be a Horn clause specification, C be a set of Horn clauses over SIG and $SP' = (SIG, AX \cup C)$. C is **ground reducible w.r.t.** SP if for all $t \equiv u \Leftarrow h \in C$ and $\sigma \in GSub$ such that $h\sigma$ is SP'-convergent, $t\sigma$ is not SP-reduced. ❑

Since by (1), the union of `STACK` and `STACK_AS_MAP` is ground confluent, the following result immediately implies that all axioms of `STACK` are inductive `STACK_AS_MAP`-theorems.

Ground Reducibility Lemma ([33], Lemma 7.11(1))[10] Let $SP = (SIG, AX)$ be a Horn clause specification, C be a set of Horn clauses over SIG and $SP' = (SIG, AX \cup C)$. If SP' is ground confluent and strongly terminating and C is ground reducible w.r.t. SP, then $C \subseteq ITh(SP)$. ❑

Proof of (3). We extend $SP \setminus \{Axiom\ 6\}$ with a descent function $>>$ used in the inductive expansion of Axiom 6 given below.

```
SP'
    base      STACK  STACK_AS_MAP-{Axiom 6}
    defs      >> 1
    infixes   >>
    vars      i j x f
    axioms    (1)  {(i,x,f,j) >> (i,x,f,s(j)) = true} <== {i > s(j)}
    conjects  (1)  {mkStack(put(i,x,f),j) = mkStack(f,j)} <== {i > j}

initial conclusion:
 (1) { mkStack(put(i,x,f),j) = mkStack(f,j) }
initial premise:
 (1) { i > j }
```

[10] This result generalizes [21], Theorem 1, from unconditional equational specifications to Horn clause specifications.

```
term at position 1 1 in conclusion goal 1 replaced with axiom STACK_AS_MAP4
conclusion:
 (1) { pop(mkStack(put(i,x,f),s(j))) = mkStack(f,j) }
term at position 1 2 in conclusion goal 1 replaced with axiom STACK_AS_MAP4
conclusion:
 (1) { pop(mkStack(put(i,x,f),s(j))) = pop(mkStack(f,s(j))) }
term at position 1 1 1 in conclusion goal 1 replaced with axiom STACK_AS_MAP2
conclusion:
 (1) { i = s(j), pop(push(x,mkStack(f,j))) = pop(mkStack(f,s(j))) }
 (2) { pop(mkStack(put(i,x,f),s(j))) = pop(mkStack(f,s(j))) }
term at position 2 1 in conclusion goal 1 replaced with axiom STACK1
conclusion:
 (1) { i = s(j), mkStack(f,j) = pop(mkStack(f,s(j))) }
 (2) { pop(mkStack(put(i,x,f),s(j))) = pop(mkStack(f,s(j))) }
term at position 2 1 in conclusion goal 1 replaced with axiom STACK_AS_MAP4
conclusion:
 (1) { i = s(j) }
 (2) { pop(mkStack(put(i,x,f),s(j))) = pop(mkStack(f,s(j))) }
term at position 1 1 1 in conclusion goal 2 replaced with conjecture 1
conclusion:
 (1) { i = s(j) }
 (2) { (i,x,f,j) >> (i,x,f,s(j)) = true, i > s(j) }
term at position 1 1 in conclusion goal 2 replaced with axiom SP'1
conclusion:
 (1) { i = s(j) }
 (2) { i > s(j) }
atoms 1 1 in conclusion goals 1 2 replaced with theorem i=s(j)\/i>s(j) <== i>j
conclusion:
 (1) { i > j }
```

❏

One important benefit from proving ground confluence by subreductive expansion instead of *inductive completion* (cf. Sect. 1.2) is the fact that the lemmas applied in a subreductive expansion are not added to the axioms of the specification and thus need neither be decreasing nor be checked for redex overlays among themselves or with the original axioms. By the definition of a sub-t-reductive expansion (6.4), we must only check that all proof goals are t-bounded.

In [33], Sect. 7.4, we have generalized inductive completion from unconditional equations to Horn clauses. Actually, a proof of C by inductive completion is a set of subreductive expansions upon $SP \cup C$ of *proper reducts* of C. In [33], such a proof of C is called a *reductive expansion of* C. In the case of unconditional axioms reductive expansion essentially agrees with *term rewriting induction for orientable equations* (cf. [36], Prop. 13). Here the conjectures of C are not used as induction hypotheses, but as additional axioms. This sounds more liberal, but is in fact more restrictive because the method requires $SP \cup C$ to be strongly terminating so that in fact only SP-decreasing conjectures can be proved (cf. 7.3).

The confinement to t-bounded proof goals and decreasing conjectures remains even if we exploit the Ground Reducibility Lemma (cf. Ex. 7.5) to its full extent. At first, this lemma has an inverse: if

(1) SP is canonical (cf. 5.11),

(2) $SP' = SP \cup C$ is strongly terminating

and $C \subseteq ITh(SP)$, then C is ground reducible w.r.t. C. At second, if (1) and (2) hold true and C is ground reducible, then, for proving that SP' is ground confluent, one need not check all SP'-critical clauses, but only those induced by C *on* SP, and even the condition of subreductive validity these clauses must satisfy can be weakened considerably. Without going into the details of this approach[11] we call the pair (SP, C) *inductively convergent* if these conditions, which are weaker than critical clause convergence of SP', are fulfilled. The underlying weaker notion of subreductive validity reads as follows (cf. 5.12):

Definition 7.6 (subreductive validity w.r.t. (SP,C)) Given a specification $SP = (SIG, AX)$, a set C of Horn clauses over SIG, a reduction ordering $>_{SP}$ for SP and a SIG-equation (!) e, a Gentzen clause $c = gs \Leftarrow h$ is **sub-e-reductively valid w.r.t.** (SP, C) if for all $\sigma \in GSub$ such that $h\sigma$ is SP'-convergent and all SP'-convergent ground goals $g <_{SP} e\sigma$ are strongly SP'-convergent *and* SP*-convergent*, $g'\tau$ is SP'-convergent for some $g' \in gs$ and $\tau \in GSub$ with $\tau =_{in(c)} \sigma$. ❑

This definition gives rise to a further instance of generic expansion: let $TH = RTh(SP')$ as in Section 6 and define GS as the set of all ground substitutions σ such that all SP'-convergent ground goals $g < t\sigma$ are strongly SP'-convergent *and* SP*-convergent*. Section 6 then provides a method for proving inductive convergence, which is even more general than the two criteria for the (SP, C)-subreductive validity of critical clauses given in [35].

Inductive convergence generalizes term rewriting induction somewhat further than reductive expansion, although the basic assumption that there is a reduction ordering for SP' and C is SP-decreasing w.r.t. that ordering is the same. The attractiveness of rewriting induction comes from the above-mentioned invertibility of underlying results. For instance, if (1) holds true, then C consists of inductive theorems of SP *only if* (SP, C) is inductively convergent (cf. [35]). Hence inductive convergence *characterizes* inductive validity under Assumptions (1) and (2). Since a proof of C by inductive convergence reduces C to SP'-critical clauses, it often goes through without explicit induction steps. These two properties of inductive convergence and other variants of rewrite induction or inductive completion cause some people to call them "automatic" or even "inductionless".

Since, roughly said, the reduction ordering for SP' takes over the role of induction orderings, it has also been claimed that rewriting induction avoids hierarchical proofs where lemmas need other induction orderings than the actual conjectures and thus cannot be proved "in the same run" (cf. [36], Ex. 15). We claim that this depends on the range of induction orderings the proof system allows us to use. If the lemmas are specializations of the conjectures, then there is of course an induction ordering where both can be based upon so that they can be proved simultaneously. If, however, some lemma properly generalizes some conjecture, rewriting induction will need a separate proof of this lemma as well as explicit induction will do. For instance, in Reddy's example ([36], Ex. 15), the proof of $x + y \equiv y + x$, the lemmas are specializations and thus need not be proved separately if one uses the following induction ordering (cf. Sect. 3):

$$(x, y) >> (x', y') \quad \text{iff} \quad x + y > x' + y'.$$

$x + y \equiv y + x$ is not decreasing and thus cannot be proved by rewriting induction. For such cases Reddy proposed SP-rewriting *modulo* C instead of SP'-rewriting (cf. [36], Remark 14). We mentioned rewriting modulo a theory in Sect. 1.5 as a kind of "built-in" simplification. Rewriting modulo works well only for theories that are presented by very simple unconditional axioms such as commutativity, associativity or idempotency of a binary function. The equation $x + y \equiv y + x$ falls into this class, but program correctness conditions rarely have such a simple structure. Hence rewriting modulo conjectures of this kind cannot lead to a fairly general proof method.

Another approach to circumvent the requirement that conjectures be decreasing is taken by [26] and [1]. Essentially, their method is a proof that C is reductively valid w.r.t. SP (cf. 5.12), carried out by *explicit*

[11] which are given in [35]

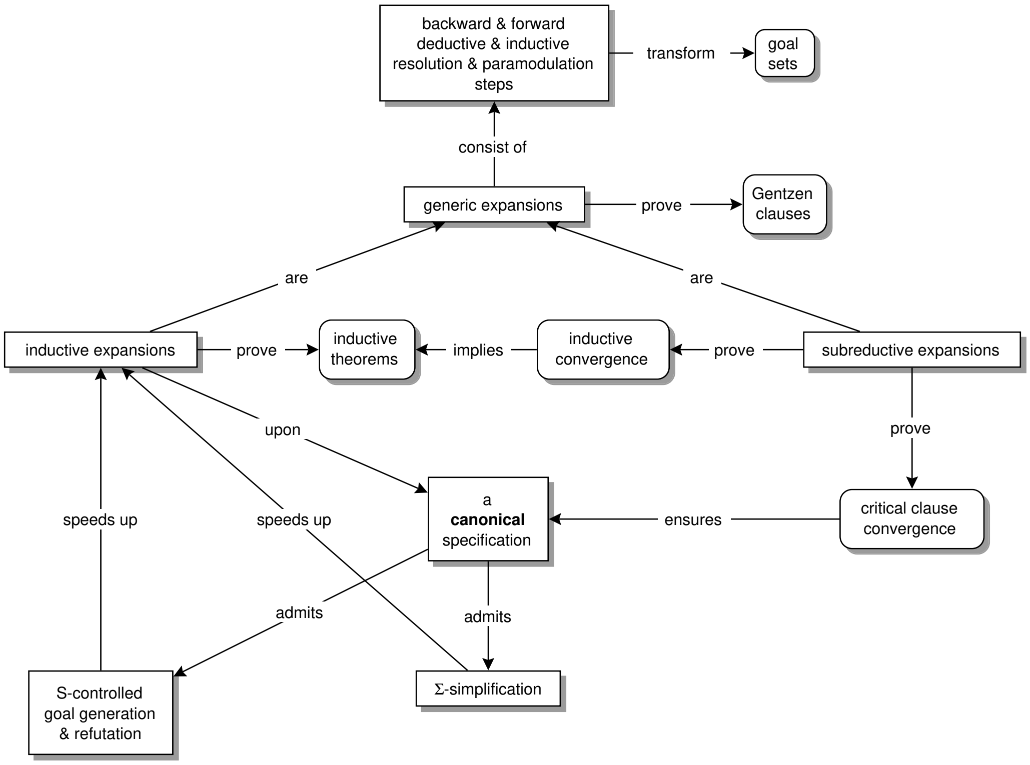

Figure 2. The impact of canonicity

induction along a reduction ordering for SP. They induce on *test set* instances of C. Roughly said, a test set $TS(c)$ for $c \in C$ is a set of substitutions that subsumes all ground solutions of the premise of c. $TS(c)$ corresponds to the set of critical clauses of c on axioms of SP if C were proved by inductive convergence. [35] shows that if the minimal substitutions obtained from the critical clauses of C on SP (cf. 5.9) form test sets for C, then Assumption (2) can indeed be dropped from the above-mentioned characterization of inductive validity by inductive convergence. Only (1) is left, which is also the main assumption to ensure that S-controlled goal generation and refutation are sound deduction rules (cf. Sect. 8).

Both results also yield methods to *refute* inductive theorems: if $c = g \Leftarrow h$ is not provable by inductive convergence, then c is invalid; if $S(g) = \emptyset$, then c is invalid either (cf. 8.5(2)). The following lemma provides another criterion for invalidity under Assumption (1) and thus confirms the thesis of 1.4 that ground confluence is crucial for refutation procedures (cf. [1], Thm. 18[12]).

Lemma 7.7 (reduced theorems are invalid) Let $SP = (SIG, AX)$ be a canonical specification and $c = t \equiv u \Leftarrow g$ be a Horn clause over SIG with $t \neq u$. $c \notin ITh(SP)$ if there is $\sigma \in GSub$ such that $SP \vdash_{cut} g\sigma$ and

- both $t\sigma$ and $u\sigma$ are SP-reduced, or
- $t\sigma$ is SP-reduced and $t\sigma >_{SP} u\sigma$, or
- $u\sigma$ is SP-reduced and $u\sigma >_{SP} t\sigma$.

Proof. Assume that $c \in ITh(SP)$. Then $SP \vdash_{cut} g\sigma$ implies $SP \vdash_{cut} t\sigma \equiv u\sigma$. By Thm. 5.8, $t\sigma \downarrow_{SP} u\sigma$. Since $t \neq u$, $t\sigma$ or $u\sigma$ is SP-reducible. W.l.o.g. suppose that $t\sigma$ is SP-reducible. Then, by assumption, $u\sigma$ is

[12] [1] show a variant of Lemma 7.7 based on a test set and with a very complicated proof. The version given here gets faster to the point.

SP-reduced and $u\sigma >_{SP} t\sigma$. By Thm. 5.8, $SP \vdash_{cut} g\sigma$ implies that $g\sigma$ is SP-convergent. Hence $t\sigma \longrightarrow_{SP} u\sigma$ and thus $t\sigma >_{SP} u\sigma$ because $>_{SP}$ is a reduction ordering for SP. Since $u\sigma >_{SP} t\sigma$, the last conclusion contradicts the well-foundedness of $>_{SP}$. ❑

At least if the critical clauses of C on SP do not form a test set or if C includes non-Horn clauses, an explicit proof by inductive expansion still seems to be the most direct way to show C. Reasonable examples of C, which express correctness conditions on functional-logic programs, are not hard to find.

Inductive convergence, rewrite induction and inductive completion are proof methods that are tailored to Assumption (1). The proof-theoretical implications of ground confluence listed in Sect. 1.4 do not include these methods because here we are mainly interested in the method of inductive expansion and how we can improve it by taking advantage of a ground confluent specification. To this end Assumption (1) brings forth additional inference rules, which significantly reduce the number of lemmas to be applied in expansions and thus enhance the "automation degree" of proofs in another way than by a tailor-made proof method. Such rules are developed in the following sections.

8 Narrowing

In contrast to the expansion rules the rules of the reduction calculus (cf. 5.3) never instantiate the variables of the goal set to be expanded when applying a theorem. If instantiation is admitted, rewriting becomes narrowing and reflection becomes unification. The original version of the narrowing rule stems from [40]. Further basic as well as surveying work on narrowing can be found in, e.g., [19], [24], [30] and [17]. Narrowing has also been called *oriented* or *directed* paramodulation because – as in the case of rewriting – equations are always applied in the same direction. [30] and [33] deal with narrowing and its completeness properties from this point of view.

Definition 8.1 (narrowing calculus) Let $SP = (SIG, AX)$ be a constructor-based specification (cf. 5.2). A pair $\langle g, \tau\rangle$ consisting of a goal g over SIG and a substitution $\tau : X \to NF_{SIG}(X)$ such that $var(g) \cap dom(\tau) = \emptyset$ is called an **SP-narrowing pair**. The **narrowing calculus for SP** consists of the following inference rules each of which transforms an SP-narrowing pair into an SP-narrowing pair. Let $\sigma \in Sub$.

narrowing rule $$\frac{\langle g[t/z], \tau\rangle}{\langle g[v/z]\sigma \cup h\sigma, \tau\sigma'\rangle}$$

if $t \notin X$, $z \in var(g)$, $t\sigma = u\sigma$, $c = (u \equiv v \Leftarrow h) \in AX$, $fresh(c)\sigma \subseteq NF_{SIG}(X)$, $\sigma' = \sigma_{var(g[t/z])}$ and $\langle g[v/z]\sigma \cup h\sigma, \tau\sigma'\rangle$ is an SP-narrowing pair

unification rule $$\frac{\langle g \cup \{t \equiv u\}, \tau\rangle}{\langle g\sigma, \tau\sigma\rangle}$$ if $t\sigma = u\sigma$ and $\langle g\sigma, \tau\sigma\rangle$ is an SP-narrowing pair

A sequence $np_1, \ldots, np_n$ of SP-narrowing pairs is called a **narrowing expansion of np_1 into np_n upon** SP if for all $1 \leq i < n$, np_{i+1} is obtained from np_i by a single narrowing or unification step. The corresponding inference relation is denoted by $\vdash\!\!\surd_{\mathbf{SP}}$.

Let $np = \langle g, \tau\rangle$ be an SP-narrowing pair. g is **SP-narrowable** if g is empty or $np \vdash\!\!\surd_{SP} np'$ for some np'. Otherwise g is **SP-narrowed**. ❑

For ground goals, SP-narrowability coincides with SP-convergence. If the substitution σ in the above rules were confined to the identity substitution id, then the narrowing calculus would agree with the reduction calculus. Conversely, $g \vdash_{SP} \emptyset$ implies $\langle g, id\rangle \vdash\!\!\surd_{SP} \langle\emptyset, id\rangle$. More generally:

Lemma 8.2 (lifting goal reductions to narrowing expansions) Let $SP = (SIG, AX)$ be a free-constructor-based specification (cf. 5.2) and $\tau : X \to NF_{SIG}(X)$. Then

$$g\tau \vdash_{SP} \emptyset \quad \text{implies} \quad \langle g, id\rangle \vdash\!\!\surd_{SP} \langle\emptyset, \tau\rangle. \quad ❑$$

By induction on the lengths of shortest narrowing expansions one obtains:

Lemma 8.3 (flattening narrowing expansions to goal reductions)

$$\langle g_1,\sigma_1\rangle \vdash\!\!\bigvee_{SP}\langle g_2,\sigma_2\rangle \quad \text{implies} \quad g_1\xi \vdash_{SP} g_2 \;\; \text{for } \xi \text{ with } \sigma_1\xi = \sigma_2.$$

In particular,

$$\langle g_1, id\rangle \vdash\!\!\bigvee_{SP}\langle g_2,\sigma_2\rangle \quad \text{implies} \quad g_1\sigma_2 \vdash_{SP} g_2$$

and

$$\langle g_1, id\rangle \vdash\!\!\bigvee_{SP}\langle \emptyset,\sigma_2\rangle \quad \text{implies} \quad g_1\sigma_2 \vdash_{SP} \emptyset.$$

The search space induced by all expansions of a narrowing pair is not tractable. We must look for strategies, which cover all narrowing expansions into narrowed or empty goals.

Definition 8.4 (narrowing strategy) Let S be a function from narrowing pairs to sets of narrowing pairs. A sequence $np_1,\ldots,np_k$ of narrowing pairs is an **S-expansion of np_1 into np_k** if for all $1 \le i < k$, $np_{i+1} \in S(np_i)$. Then we write $np_1 \vdash^{\mathsf{S}} np_k$.

S is a **narrowing strategy** if for all narrowing pairs np and $np' \in S(np)$ there is a narrowing expansion of np into np' upon SP. S is **narrowing complete** if for all goals g and $\tau : X \to NF_{SIG}$ there is a substitution $\rho \le \tau$ (cf. Sect. 2) such that

$$\langle g, id\rangle \vdash\!\!\bigvee_{SP}\langle \emptyset,\tau\rangle \quad \text{implies} \quad \langle g, id\rangle \vdash^{S} \langle \emptyset,\rho\rangle. \;\; ❑$$

Note that a narrowing complete function S need not be a narrowing strategy. In Section 10 we introduce Σ-*narrowing strategies*, which may be narrowing complete, but which usually are not narrowing strategies.

Lemma 8.5 (soundness and completeness of narrowing strategies) Let S be a function from narrowing pairs to sets of narrowing pairs.

(1) Suppose that S is a narrowing strategy. If $\langle g, id\rangle \vdash^{S} \langle \emptyset,\sigma\rangle$ for some $\sigma : X \to NF_{SIG}(X)$, then $SP \vdash_{cut} g\sigma$. Hence, conversely, if g is not SP-solvable (cf. 4.1), then for all $n \ge 0$ and $\langle h,\sigma\rangle \in S^n(\{\langle g, id\rangle\})$, $h \ne \emptyset$.

(2) Suppose that SP is free-constructor-based, normal form complete and ground confluent and S is narrowing complete. Then $SP \vdash_{cut} g\sigma$ for some $\sigma \in GSub$ implies $\langle g, id\rangle \vdash^{S} \langle \emptyset,\rho\rangle$ for some $\rho : X \to NF_{SIG}(X)$. Hence, conversely, if for all $n \ge 0$ and $\langle h,\sigma\rangle \in S^n(\{\langle g, id\rangle\})$, $h \ne \emptyset$, then g is not SP-solvable.

Proof. (1) $\langle g, id\rangle \vdash^{S} \langle \emptyset,\sigma\rangle$ implies $\langle g, id\rangle \vdash\!\!\bigvee_{SP}\langle \emptyset,\sigma\rangle$. Hence by Lemma 8.3, $g\sigma$ is SP-convergent and thus $SP \vdash_{cut} g\sigma$.

(2) Let $SP \vdash_{cut} g\sigma$ for some $\sigma \in GSub$. Since SP is normal form complete, $g\sigma \longrightarrow^{*}_{SP} g\tau$ for some $\tau : X \to NF_{SIG}$. Since SP is ground confluent, $g\sigma$ is strongly SP-convergent and thus $g\tau$ is SP-convergent. Hence by Lemma 8.2, $\langle g, id\rangle \vdash\!\!\bigvee_{SP}\langle \emptyset,\tau\rangle$ and thus $\langle g, id\rangle \vdash^{S} \langle \emptyset,\rho\rangle$ for some $\rho \le \tau$ because S is narrowing complete. ❑

The following lemma is the key result of this section. In the subsequent theorem we use it for augmenting the inductive expansion calculus with a forward rule, which is sound if the assumptions of Lemma 8.6 on S and SP hold true. It is also a generalization of Lemma 5.7.

Lemma 8.6 (narrowing generates case analyses) Suppose that SP is free-constructor-based, normal form complete and ground confluent and S is a narrowing complete narrowing strategy. Let g be a goal, $S(\langle g, id\rangle) = \{\langle g_1,\sigma_1\rangle,\ldots,\langle g_k,\sigma_k\rangle\}$ and for all $1 \le i \le k$, define $X_i = var(dom(\sigma_i)\sigma_i)$ such that $(var(g) \cup dom(\sigma_i)) \cap X_i = \emptyset$. Then

$$c \;=\; \{\exists X_1(g_1 \cup EQ(\sigma_1)),\ldots,\exists X_k(g_k \cup EQ(\sigma_k))\} \;\Leftrightarrow\; g$$

is an inductive SP-theorem.

Proof. By Thm. 5.13, it is sufficient to show that c is reductively valid w.r.t. SP. So let $\tau : X \to NF_{SIG}$ such that $g\tau$ is strongly SP-convergent. Hence by Lemma 8.2, $\langle g, id\rangle \vdash_{\bigvee SP} \langle\emptyset,\tau\rangle$ and thus $\langle g, id\rangle \vdash^S \langle g_i,\sigma_i\rangle \vdash^S \langle\emptyset,\rho\rangle$ for some $1 \le i \le k$ and $\rho \le \tau$ because S is narrowing complete. Since S is a narrowing strategy, $\langle g_i,\sigma_i\rangle \vdash^S \langle\emptyset,\rho\rangle$ implies $\langle g_i,\sigma_i\rangle \vdash_{\bigvee SP}\langle\emptyset,\rho\rangle$ and thus $\langle g_i,\sigma_i\rangle \vdash_{\bigvee SP}\langle\emptyset,\tau\rangle$. Since $\langle\emptyset,\tau\rangle$ is a narrowing pair, Lemma 8.3 implies $g_i\xi \vdash_{SP} \emptyset$ for ξ such that $\sigma_i\xi = \tau$. Since $\langle g_i,\sigma_i\rangle$ is a narrowing pair, $var(g_i) \cap dom(\sigma_i) = \emptyset$. Hence $x\tau = x\sigma_i\xi = x\xi$ for all $x \in var(g_i)$, and thus $g_i\xi \vdash_{SP} \emptyset$ implies $g_i\tau_i \vdash_{SP} \emptyset$ where $\tau_i = \tau_{X\setminus X_i} + \xi_{X_i}$. Since $dom(\sigma_i) \cap X_i = \emptyset$, $x\tau_i = x\tau = x\sigma_i\xi = x\sigma_i\tau_i$ for all $x \in dom(\sigma_i)$. Hence $EQ(\sigma_i)\tau_i$ consists of reflexive equations so that, trivially, $EQ(\sigma_i)\tau_i$ is SP-convergent. Since SP is ground confluent, we conclude that $(g_i \cup EQ(\sigma_i))\tau_i$ is strongly SP-convergent and thus the $\Leftarrow$-part of c is reductively valid w.r.t. SP.

Let $\tau : X \to NF_\Sigma$ such that $g_i\tau$ and $EQ(\sigma_i)\tau$ are strongly SP-convergent for some $1 \le i \le k$. By Lemma 8.2, $\langle g_i, id\rangle \vdash_{\bigvee SP}\langle\emptyset,\tau\rangle$ and thus $\langle g_i, id\rangle \vdash^S \langle\emptyset,\sigma\rangle$ for some $\sigma \le \tau$ because S is narrowing complete. Hence $\langle g, id\rangle \vdash^S \langle g_i,\sigma_i\rangle \vdash^S \langle\emptyset,\sigma_i\sigma\rangle$ and $\sigma_i\sigma \le \sigma_i\tau$. By Lemma 8.3, $g\sigma_i\sigma$ and thus $g\sigma_i\tau$ are SP-convergent. Since $EQ(\sigma_i)\tau$ is SP-convergent, for all $x \in dom(\sigma_i)$, $x\tau$ and $x\sigma_i\tau$ are SP-joinable. Moreover, $g\sigma_i\tau$ is strongly SP-convergent because SP is ground confluent. Hence $g\tau$ is SP-convergent and thus strongly SP-convergent as well, and we conclude that the $\Rightarrow$-part of c is reductively valid w.r.t. SP. ❑

Intuitively, Lemma 8.6 says that the set $S(\langle g, id\rangle) = \{\langle g_1,\sigma_1\rangle,\dots,\langle g_k,\sigma_k\rangle\}$ of narrowing pairs provides a complete case analysis of g. This gives rise to a general forward rule, which splits a goal into subgoals and thus divides a proof into subproofs. In contrast to the forward rules of Def. 3.10, this rule, called S-controlled goal generation, does not apply "user-defined" lemmas at "user-defined" goal positions, but derives the respective case analysis automatically. If S selects a single redex position (cf. Sect. 9), then performing an S-controlled goal generation step means applying the only-if-completion of a defined function (cf. 5.6).

Definition 8.7 (S-controlled rules) Let S be a function from narrowing pairs to sets of narrowing pairs.

Backward S-controlled goal generation Let S be a narrowing strategy, $S(\langle g, id\rangle) = \{\langle g_1,\sigma_1\rangle,\dots,\langle g_k,\sigma_k\rangle\}$ and for all $1 \le i \le k$, $X_i = var(dom(\sigma_i)\sigma_i)$ such that $(var(g) \cup dom(\sigma_i)) \cap X_i = \emptyset$.

$$\frac{\{g\}}{\{\exists X_1(g_1 \cup EQ(\sigma_1)),\dots,\exists X_k(g_k \cup EQ(\sigma_k))\}}$$

Suppose that SP is free-constructor-based, normal form complete and ground confluent and S is narrowing complete.

Forward S-controlled goal generation Let S be a narrowing strategy, $S(\langle g, id\rangle) = \{\langle g_1,\sigma_1\rangle,\dots,\langle g_k,\sigma_k\rangle\}$ and for all $1 \le i \le k$, $X_i = var(dom(\sigma_i)\sigma_i)$ such that $(var(g) \cup dom(\sigma_i)) \cap X_i = \emptyset$.

$$\frac{\{g\}}{\{\exists X_1(g_1 \cup EQ(\sigma_1)),\dots,\exists X_k(g_k \cup EQ(\sigma_k))\}}$$

S-controlled goal refutation Suppose that for all $n \ge 0$ and $\langle h,\sigma\rangle \in S^n(\langle g, id\rangle)$, $h \ne \emptyset$.

$$\frac{\{g\}}{\{FALSE\}} \quad ❑$$

Theorem 8.8 (S-controlled rules are sound) Let a goal set hs be obtained from a goal set gs by a single S-controlled rule step such that the respective assumptions of Def. 8.7 hold true. If a backward rule is applied, then $gs \Leftarrow h \in ITh(SP)$ for all $h \in hs$. If a forward rule is applied, then $hs \Leftarrow g \in ITh(SP)$ for all $g \in gs$.

Proof. In the case of backward goal generation we have to show that for all $1 \le i \le k$, the clause $g \Leftarrow g_i \cup EQ(\sigma_i)$ is an inductive SP-theorem. Since S is a narrowing strategy, $\langle g, id\rangle \vdash_{\bigvee SP}\langle g_i,\sigma_i\rangle$. By Lemma 8.3, $g\sigma_i \vdash_{SP} g_i$. Let $\tau \in GSub$ such that $SP \vdash_{cut} g_i\tau \cup EQ(\sigma_i)\tau$. Hence $g\sigma_i \vdash_{SP} g_i$ implies $SP \vdash_{cut} g\sigma_i\tau$ and thus $SP \vdash_{cut} g\tau$ because $SP \vdash_{cut} EQ(\sigma_i)\tau$.

The correctness of forward goal generation follows directly from Lemma 8.6. For (backward as well as forward) goal refutation we have to show that $FALSE \Leftarrow g$ is an inductive SP-theorem. Since for all $n \geq 0$ and $\langle h, \sigma\rangle \in S^n(\{\langle g, id\rangle\})$, $h \neq \emptyset$, Lemma 8.5(2) implies that g is not SP-solvable. Hence $FALSE \Leftarrow g \in ITh(SP)$. ❑

9 Redex selection

Which narrowing strategies are ground complete (cf. 8.4)? General conditions for ensuring this property are given in [29], [9], [30], [32] and [10]. In particular, [29] introduced the criterion of *uniformity*, which says that, given a goal g and a normal form substitution σ, the selected *redex* of $g\sigma$ agrees with the σ-instance of the selected redex of g.

Definition 9.1 Given a term or equation t and a goal g with a unique occurrence of the variable z[13], the expression $g \bullet t$ is called a **term position** of the goal $g[t/z]$. If $t \in X$, then $g \bullet t$ is a **variable position** of g. For substitutions σ, $(g \bullet t)\sigma =_{def} g\sigma_{X\setminus\{z\}} \bullet t\sigma$.

Let $SP = (SIG, AX)$ be a constructor-based specification, $g \bullet t$ be a non-variable term position and $\sigma : X \to NF_{SIG}(X)$.

- If there are $u \equiv v \Leftarrow h \in AX$ and $u' \leq u$ such that $t\sigma = u'\sigma$, $h\sigma$ is SP-narrowable and $var(g[v/z]\sigma \cup h\sigma) \cap dom(\sigma_{var(g[t/z])}) = \emptyset$, then $g \bullet t$ is a **partial redex** of $g[t/z]$ **with unifier** σ and **σ-reduct** $g[v/z]\sigma \cup h\sigma$. If $u' = u$, then $g \bullet t$ is called a **total redex**.
- If $t = (u \equiv u')$ such that $u\sigma = u'\sigma$ and $var(g\sigma) \cap dom(\sigma) = \emptyset$, then $g \bullet t$ is a **total redex** of $g[t/z]$ **with unifier** σ and **σ-reduct** $g\sigma$. ❑

Definition 9.2 A **redex selector** R is a function from goals to sets of term positions such that for all SP-narrowable non-empty goals g, $R(g)$ is a nonempty set of total redices. R is **uniform** if for all $\sigma : X \to NF_{SIG}$, $g\sigma \vdash_{SP} \emptyset$ implies $R(g\sigma) \subseteq R(g)\sigma$. ❑

We investigate three redex selectors. Let g be a narrowable goal. $\mathbf{IN}(g)$ is the leftmost-innermost[14] total redex of g, $\mathbf{OUT}(g)$ is the leftmost-outermost total redex of g, and

$$\mathbf{NEED}(g) \; =_{def} \; \{h \bullet t \mid (h \bullet t)\sigma = OUT(g\sigma) \text{ for some ground unifier } \sigma \text{ of } POUT(g)\}$$

where $\mathbf{POUT}(g)$ is the leftmost-outermost partial redex of g.[15]

IN is not always uniform. For instance, let SP be a specification of natural numbers with constructors 0 and $succ$, a unary operation f and axioms $x * 0 \equiv 0$ and $f(succ(x)) \equiv f(x)$. Let $e = (f(x) * y \equiv 0)$. Then

$$IN(e) \;=\; (z * y \equiv 0) \bullet f(x),$$

but

$$IN(e[0/x]) \;=\; IN(f(0) * y \equiv 0) \;=\; (z \equiv 0) \bullet (f(0) * y).$$

The reason for non-uniformity is the fact that $f(0)$ is not reducible and thus SP is not normal form complete.

Lemma 9.3 If SP is free-constructor-based and normal form complete, then IN is a uniform redex selector.

Proof. Let g be an SP-narrowable goal and $\sigma : X \to NF_{SIG}$ such that $g\sigma$ is SP-convergent. Let $h \bullet t = IN(g\sigma)$, i.e. $h \bullet t$ is the leftmost-innermost total redex of $g\sigma$. Since SP is free-constructor-based, σ assigns only

[13] z stands for a term or an atom.
[14] with respect to the root position of t within g (cf. 9.1)
[15] $NEED$ generalizes the redex selector that underlies the *needed narrowing strategy* of [3].

SP-reduced terms to the variables of g. Hence $h[t/z] = g\sigma$ implies $h \bullet t = (h' \bullet t')\sigma$ for some term position $h' \bullet t'$ of g, and $h' \bullet t'$ is a total redex of g. But is it a *leftmost-innermost* one? Let $h'' \bullet t''$ be the leftmost-innermost total redex of g, i.e. $IN(g) = h'' \bullet t''$. Then there are $u \equiv v \Leftarrow h \in AX$ and $\tau \in Sub$ such that $u\tau = t''\tau$. Since SP is constructor-based, u contains a defined function F. Since SP is normal form complete and $h'' \bullet t''$ is a leftmost-innermost redex, F is the root of u and thus of t'' because $t'' \notin X$. Hence $t''\sigma \longrightarrow_{SP} u'$ for some u' because SP is normal form complete. Hence $(h'' \bullet t'')\sigma$ is a total redex of $g\sigma$, which must agree with $h \bullet t$ because $h \bullet t$ is the leftmost-innermost total redex of $g\sigma$. This implies uniformity: $IN(g\sigma) = h \bullet t = (h'' \bullet t'')\sigma = IN(g)\sigma$. ❑

OUT is not always uniform either, even if – in contrast to the previous example – SP is normal form complete. For instance, let SP be a specification of natural numbers with constructors 0 and $succ$, a binary operation g and axioms $g(0,0) \equiv 0$, $g(succ(x),0) \equiv 1$ and $g(x, succ(y)) \equiv 2$.[16] Let $e = (g(g(x,x'),y) \equiv 0)$. Then

$$OUT(e) \;=\; (z \equiv 0) \bullet g(g(x,x'),y),$$

but

$$OUT(e[0/y]) \;=\; OUT(g(g(x,x'),0) \equiv 0) \;=\; (g(z,0) \equiv 0) \bullet f(x,x').$$

In fact, the solution $[0/x, 0/x', 0/y]$ of the equation e cannot be achieved by OUT because $y\sigma \neq 0$ for all unifiers σ of $OUT(e)$. However, $OUT(e)$ agrees with $POUT(e)$ and thus

$$NEED(g) \;=\; \{(z \equiv 0) \bullet g(g(x,x'),y),\; (g(z,y) \equiv 0) \bullet g(x,x')\}.$$

Lemma 9.4 If SP is free-constructor-based, then $NEED$ is a uniform redex selector.

Proof. Let g be an SP-narrowable goal and $\sigma : X \to NF_{SIG}$ such that $g\sigma$ is SP-convergent. Let $h \bullet t \in NEED(g\sigma)$. Since σ is ground, $h \bullet t = OUT(g\sigma)$. Since SP is free-constructor-based, σ assigns only SP-reduced terms to the variables of g. Hence $h[t/z] = g\sigma$ implies $h \bullet t = (h' \bullet t')\sigma$ for some term position $h' \bullet t'$ of g, and $h' \bullet t'$ is a total redex of g. Since SP is constructor-based, σ is a unifier of $POUT(g)$. Hence $h' \bullet t' \in NEED(g)$ and thus $h \bullet t = (h' \bullet t')\sigma \in NEED(g)\sigma$. ❑

For implementing $NEED(g)$ one may replace the set $Uni(g)$ of all ground unifiers of $POUT(g)$ by a finite set Φ of unifiers of $POUT(g)$ such that each $\sigma \in Uni(g)$ is subsumed by some $\tau \in \Phi$, say $\tau\rho = \sigma$, and $OUT(g\tau)\rho = OUT(g\sigma)$. Then

$$NEED(g) \;=\; \{h \bullet t \mid (h \bullet t)\tau = OUT(g\tau) \text{ for some } \tau \in \Phi\}.$$

Expander (cf. Section 1.1) constructs the reducts of $NEED(g)$ in several steps. In general, Φ includes unifiers of total as well as only partial redices of g. For the first ones reducts according to Def. 9.1 can be derived directly. For a unifier τ of a properly partial redex $h \bullet t$, however, only the instance $g\tau$ is returned at first, while the reduct of $OUT(g\tau)$ will be obtained automatically by a subsequent outermost narrowing step. In effect, properly partial redices induce applications of the instantiation rule (cf. Section 4), total redices induce applications of the narrowing rule, and in both cases the leftmost-outermost redex is selected.

In general, the narrowing strategy (cf. Sect. 8) induced by a redex selector is defined as follows.

Definition 9.5 Let R be a redex selector. Then the **narrowing strategy $\mathbf{S}_R$ induced by** R is defined as follows: Let $\langle g, \tau\rangle$ be an SP-narrowing pair such that g is SP-narrowable.

$$S_R(\langle g,\tau\rangle) =_{def} \{\langle h, \tau\sigma_{var(g)}\rangle \mid \sigma \text{ is a minimal unifier of a total redex in } R(g) \text{ with } \sigma\text{-reduct } h\}. \text{ ❑}$$

Of course, S_R is a narrowing strategy. Hence for all ground goals g, $\langle g, id\rangle \vdash_R^S \langle \emptyset, id\rangle$ and thus by Lemma 8.3, $g \vdash_{SP} \emptyset$. Conversely, we have:

[16] [9], Example 1

Lemma 9.6 (completeness of S_R-expansions) Suppose that SP is ground confluent and strongly terminating and R is a redex selector. Then for all ground goals g,

$$g \vdash_{SP} \emptyset \text{ implies } \langle g, id\rangle \vdash^{S_R} \langle \emptyset, id\rangle.$$

Proof. Let g be SP-convergent. We show $\langle g, id\rangle \vdash^{S_R} \langle \emptyset, id\rangle$ by Noetherian induction on g along the reduction ordering $>_{SP}$, which exists by assumption. If g is SP-reduced, then $g = \emptyset$ because $g \vdash_{SP} \emptyset$. So let g be be nonempty. Since g is SP-convergent, g is SP-narrowable. Hence $R(g)$ is defined, i.e. there is a total redex $h \bullet t \in R(g)$. Since g is ground, t is ground and thus there are $(u \equiv v \Leftarrow d) \in AX$ and $\sigma : X \to NF_{SIG}$ such that $t = u\sigma$, $d\sigma$ is SP-narrowable and $\langle h[v\sigma/z] \cup d\sigma, id\rangle \in S_R(\langle g, id\rangle)$. Hence $d\sigma$ is ground and thus SP-convergent so that $g \to_{SP} h[v\sigma/x]$. Since SP is ground confluent and g is SP-convergent, $h[v\sigma/x]$ is also SP-convergent. Since $>_{SP}$ is a reduction ordering for SP, we have $g >_{SP} h[v\sigma/x]$ and $g \geq_{SP} t = u\sigma >_{SP} d\sigma$. $g' =_{def} h[v\sigma/z] \cup d\sigma$ is an SP-convergent ground goal with $g >_{SP} g'$. Hence by induction hypothesis, $\langle g', id\rangle \vdash^{S_R} \langle \emptyset, id\rangle$. Since $\langle g', id\rangle \in S_R(\langle g, id\rangle)$ implies $\langle g, id\rangle \vdash^{S_R} \langle g', id\rangle$, we conclude $\langle g, id\rangle \vdash^{S_R} \langle \emptyset, id\rangle$. ❑

Lemma 9.7 (lifting S_R-expansions) Let R be a uniform redex selector. Then for all goals g and $\tau : X \to NF_{SIG}$,

$$\langle g\tau, id\rangle \vdash^{S_R} \langle \emptyset, id\rangle \quad \text{implies} \quad \langle g, id\rangle \vdash^{S_R} \langle \emptyset, \rho\rangle$$

for some $\rho \leq \tau$.

Proof by induction on the length n of a shortest S_R-expansion of $\langle g\tau, id\rangle$ into $\langle \emptyset, id\rangle$. If $n = 1$, then $g\tau$ and thus g are empty. Hence we obtain the result for $\rho = id$. So let g be nonempty. $\langle g\tau, id\rangle \vdash^{S_R} \langle \emptyset, id\rangle$ implies $g\tau \vdash_{SP} \emptyset$. Hence $g\tau$ is SP-narrowable and thus $R(g\tau)$ is nonempty. Let $h' \bullet t' \in R(g\tau)$. Since R is uniform, $h' \bullet t' = (h \bullet t)\tau$ for some $h \bullet t \in R(g)$. Hence there are $(u \equiv v \Leftarrow d) \in AX$ and $\sigma : X \to NF_{SIG}$ such that $t\tau = u\sigma$, $d\sigma$ is SP-narrowable, $\langle h[v\sigma/z] \cup d\sigma, \sigma_{var(g)}\rangle \in S_R(g)$ and there is a shorter S_R-expansion from $\langle h[v\sigma/z] \cup d\sigma, id\rangle$ into $\langle \emptyset, id\rangle$ than from $\langle g\tau, id\rangle$ into $\langle \emptyset, id\rangle$.

W.l.o.g. $var(c) \cap var(g) = \emptyset$ and $var(g) = dom(\tau)$. Hence t and u have a most general unifier $\rho \leq \sigma + \tau$ such that ρ is a minimal unifier of $h \bullet t$ and $\langle h[v\rho/z] \cup d\rho, \rho_{var(g)}\rangle \in S_R(\langle g, id\rangle)$. Let ξ be the ground substitution with $\rho_{dom(\tau)}\xi = \tau$. Let $g' = h[v\rho/z] \cup d\rho$. Since $g'\xi = h[v\sigma/z] \cup d\sigma$, the induction hypothesis implies $\langle g', id\rangle \vdash^{S_R} \langle \emptyset, \phi\rangle$ for some $\phi \leq \xi$. $\langle g', \rho_{var(g)}\rangle \in S_R(\langle g, id\rangle)$ implies $\langle g, id\rangle \vdash^{S_R} \langle g', \rho_{var(g)}\rangle$, and we conclude $\langle g, id\rangle \vdash^{S_R} \langle \emptyset, \rho_{var(g)}\phi\rangle$. The proof is complete because $\rho_{var(g)}\phi \leq \rho_{var(g)}\xi = \rho_{dom(\tau)}\xi = \tau$. ❑

10 Simplification

The narrowing strategy induced by a redex selector combines redex selection with most general unification. Yet each expansion step is a single application of a rule of the narrowing calculus. Narrowing expansions can be "sped up" significantly if each reduct is simplified by equivalence transformations before it is subjected to further narrowing steps.

Definition 10.1 An **SP-compatible simplifier** Σ is a function between goals such that for each goal g, $g \Leftrightarrow \Sigma(g)$ is an inductive SP-theorem.[17] Σ is extended to goal sets gs as follows:

$$\Sigma(gs) =_{def} \{\Sigma(g) \mid g \in gs\}. ❑$$

SP-compatible simplifiers often occur implicitly as refinements of the narrowing rule, such as *reduced* or *normalizing* and *optimized narrowing* (cf. [30], Sections 8.7 to 8.10; [17], Sections 2.2 and 2.3). Normalizing narrowing combines the narrowing rule with a particular simplifier that assigns to each goal g a reduced reduct of g. Optimized narrowing simplifies equations by applying particular lemmas, which hold true whenever

[17] This includes the case that g is not SP-solvable (cf. 4.1) and $\Sigma(g) = FALSE$.

SP is canonical and free-constructor-based (cf. 5.2). Further rule-based simplifications are given by theory resolution (cf. [41]), rewriting modulo equational theories (cf. [20]) and rewriting modulo algebras (cf. [2]). In contrast to these approaches we distinguish between the rule of **Σ-simplification** (cf. Sect. 4) and more "sensitive" inferences such as induction steps, which produce descent conditions (cf. Sect. 3), or S-controlled goal generation, which produces a case analysis (cf. Sect. 8). Σ-simplification is an equivalence transformation and can thus be used in a forward as well as in a backward proof, while inductive rules and S-controlled goal generation only establish an implication, *either* from the antecedent to the succedent *or* from the succedent to the antecedent of the rule. Hence they can only be used either in a forward or in a backward proof (cf. Sect. 3).

Σ-simplification Let Σ is be an SP-compatible simplifier.

$$\frac{gs}{\Sigma(gs)}$$

Goal refutations (cf. Sect. 4 and 8.7) are equivalence transformations and thus good candidates for a simplifier. But simplifications are supposed to be performed automatically on all goals of a proof. Hence goal refutation should be made into a simplification step only if certain unsolvability *criteria* hold true, which can be checked fast. For instance, checking the applicability condition of S-controlled goal refutation (cf. 8.7) involves the detection of circular S-expansions, which is a rather time-consuming procedure. However, confined to the test of $S(\{\langle g, id\rangle\})$ for emptiness, which obviously *implies* the applicability condition of S-controlled goal refutation, this rule does not slow down, but speed up expansions if it becomes part of Σ.

This instance of S-controlled goal refutation reduces solvability to narrowability. Remember that its soundness depends on properties of SP among which free-constructor-basedness is the most restrictive one (cf. 8.7). If SP is free-constructor-based, then all normal forms are SP-reduced, and this condition is essential for lifting goal reductions to narrowing expansions (cf. 8.2). It is also crucial for the soundness of the following two equivalence rules, which provide almost indispensible simplifications:

constructor decomposition $\quad \dfrac{\{\{F(t_1,\dots,t_k)\equiv F(u_1,\dots,u_k)\}\cup g\}}{\{\{t_1\equiv u_1,\dots,t_k\equiv u_k\}\cup g\}}$ $\quad$ if F is a constructor

constructor clash $\quad \dfrac{\{\{F(t)\equiv G(u)\}\cup g\}}{\{FALSE\}}$ $\quad$ if F and G are two different constructors

The *rewriting modulo* approach (cf. 1.5) was introduced for dealing with specifications that are not free-constructor-based, such as sets, bags and maps. *Constructor axioms* define the equality predicate for all sorts s with non-free constructors. One may avoid constructor axioms by including an interpreter for s-equations into the simplifier. This is the way Expander (cf. 1.1) handles s-equations. A more general approach to get rid of constructor axioms is to present sets, bags, maps, etc. as *action types* with *terminal semantics* and transform these into their *Horn clause completions*, which are free-constructor-based (cf. [35]).

Let us now combine simplification with narrowing.

Definition 10.2 Given an SP-compatible simplifier Σ, a sequence $np_1,\dots,np_n$ of SP-narrowing pairs is called a **Σ-narrowing expansion of np_1 into np_n upon SP** if for all $1\le i<n$, np_{i+1} is obtained from np_i by a single narrowing, unification or Σ-simplification step (cf. 8.1). The corresponding inference relation is denoted by $\vdash\!\!\surd_{\mathbf{SP},\mathbf{\Sigma}}$. ❑

Definition 10.3 Let Σ be an SP-compatible simplifier and S be a narrowing strategy (cf. 8.4). Then for all narrowing pairs np,

$$S^\Sigma(np) \;=_{def}\; \{\langle\Sigma(g),\tau\rangle \mid \langle g,\tau\rangle\in S(np)\}.$$

S^Σ is called the **Σ-narrowing strategy associated with S.** ❑

The question arises whether Lemma 8.6 remains valid for S^Σ instead of S. Let us inspect the proof of that lemma. The crucial step is where Lemma 8.3 is applied in order to conclude $g_i\xi \vdash_{SP} \emptyset$ for ξ with $\sigma_i\xi = \tau$ from $\langle g_i, \sigma_i\rangle \vdash\!\!\bigvee_{SP}\langle\emptyset, \tau\rangle$, which will now read as $\langle g_i, \sigma_i\rangle \vdash\!\!\bigvee_{SP,\Sigma}\langle\emptyset, \tau\rangle$. In fact, Lemma 8.3 does not hold for Σ-narrowing expansions, at least not in its full generality. However, we use it in the proof of the following lemma, which justifies the step from $\langle g_i, \sigma_i\rangle \vdash\!\!\bigvee_{SP,\Sigma}\langle\emptyset, \tau\rangle$ to $g_i\xi \vdash_{SP} \emptyset$ under additional assumptions.

Lemma 10.4 (soundness of Σ-narrowing expansions) Suppose that SP is normal form complete and ground confluent and Σ is an SP-compatible simplifier. Let $\langle g, \sigma\rangle$ be a narrowing pair and $\tau : X \to NF_{SIG}(X)$. Then

$$\langle g, \sigma\rangle \vdash\!\!\bigvee_{SP,\Sigma}\langle\emptyset, \tau\rangle \quad \text{implies} \quad g\xi \vdash_{SP} \emptyset \ \text{ for } \xi \text{ with } \sigma\xi = \tau.$$

Proof by induction on the length of a shortest Σ-narrowing expansion E of $\langle g, \sigma\rangle$ into $\langle\emptyset, \tau\rangle$. If E does not include simplification steps, then the result follows directly from Lemma 8.3. Otherwise E can be splitted into a narrowing expansion E_1 of $\langle g, \sigma\rangle$ into a narrowing pair $np_1 =_{def} \langle g_1, \sigma_1\rangle$, a simplification step E_2 from np_1 to $np_2 = \langle\Sigma(g_1), \sigma_1\rangle$ and a Σ-narrowing expansion E_3 of np_2 into $\langle\emptyset, \tau\rangle$.

Hence $\langle g, \sigma\rangle \vdash\!\!\bigvee_{SP} np_1$ and $np_2 \vdash\!\!\bigvee_{SP,\Sigma}\langle\emptyset, \tau\rangle$. By induction hypothesis, $\Sigma(g_1)\xi \vdash_{SP} \emptyset$ for ξ with $\sigma_1\xi = \tau$. Since Σ is SP-compatible, $g_1 \Leftarrow \Sigma(g_1)$ is an inductive SP-theorem. Since SP is normal form complete and ground confluent, Thm. 5.13 implies that $g_1 \Leftarrow \Sigma(g_1)$ is reductively valid w.r.t. SP. Hence $\Sigma(g_1)\xi \vdash_{SP} \emptyset$ implies $g_1\xi \vdash_{SP} \emptyset$. By Lemma 8.3, $\langle g, \sigma\rangle \vdash\!\!\bigvee_{SP} np_1$ implies $g\phi \vdash_{SP} g_1$ for ϕ with $\sigma\phi = \sigma_1$. Since ξ replaces all variables by normal forms, $g\phi \vdash_{SP} g_1$ implies $g\phi\xi \vdash_{SP} g_1\xi$ and thus $g\phi\xi \vdash_{SP} \emptyset$ because $g_1\xi$ is SP-convergent. Moreover, $\sigma\phi\xi = \sigma_1\xi = \tau$. ❑

Lemma 10.4 leads to the Σ-narrowing version of Lemma 8.6:

Lemma 10.5 (Σ-narrowing generates case analyses) Suppose that SP is free-constructor-based, normal form complete and ground confluent, Σ is an SP-compatible simplifier and S is a narrowing strategy such that S^Σ is narrowing complete. Let g be a goal, $S^\Sigma(\langle g, id\rangle) = \{\langle g_1, \sigma_1\rangle, \ldots, \langle g_k, \sigma_k\rangle\}$ and for all $1 \leq i \leq k$, $X_i =_{def} var(dom(\sigma_i)\sigma_i)$ such that $(var(g) \cup dom(\sigma_i)) \cap X_i = \emptyset$. Then

$$c \quad = \quad \{\exists X_1(g_1 \cup EQ(\sigma_1)), \ldots, \exists X_k(g_k \cup EQ(\sigma_k))\} \ \Leftrightarrow \ g$$

is an inductive SP-theorem.

Proof. The same as the proof of Lemma 8.6 except that S is replaced by S^Σ and thus we apply Lemma 10.4 instead of Lemma 8.3. ❑

Thm. 8.8 must be adapted to Σ-narrowing strategies:

Theorem 10.6 (S^Σ-controlled rules are sound) Let a goal set hs be obtained from a goal set gs by a single S^Σ-controlled rule step such that the respective assumptions of Def. 8.7 hold true. If a backward rule is applied, then $gs \Leftarrow h \in ITh(SP)$ for all $h \in hs$. If a forward rule is applied, then $hs \Leftarrow g \in ITh(SP)$ for all $g \in gs$.

Proof. In the case of backward goal generation we have to show that for all $1 \leq i \leq k$, the clause $g \Leftarrow g_i \cup EQ(\sigma_i)$ is an inductive SP-theorem. Since S is a narrowing strategy, $\langle g, id\rangle \vdash\!\!\bigvee_{SP,\Sigma}\langle g_i, \sigma_i\rangle$. By Lemma 10.4, $g\sigma_i \vdash_{SP} g_i$. Let $\tau \in GSub$ such that $SP \vdash_{cut} g_i\tau \cup EQ(\sigma_i)\tau$. Hence $g\sigma_i \vdash_{SP} g_i$ implies $SP \vdash_{cut} g\sigma_i\tau$ and thus $SP \vdash_{cut} g\tau$ because $SP \vdash_{cut} EQ(\sigma_i)\tau$.

The correctness of forward goal generation follows directly from Lemma 10.5. For (backward as well as forward) goal refutation we have to show that $FALSE \Leftarrow g$ is an inductive SP-theorem. Since for all $n \geq 0$ and $\langle h, \sigma\rangle \in (S^\Sigma)^n(\langle g, id\rangle)$, $h \neq \emptyset$, Lemma 8.5(2) implies that g is not SP-solvable. Hence $FALSE \Leftarrow g \in ITh(SP)$.❑

Consequently, we reformulate Thm. 4.2:

Theorem 10.7 (inductive expansions with S^Σ-controlled goal generation and refutation are sound) Suppose that SP is free-constructor-based, normal form complete and ground confluent, Σ is an SP-compatible simplifier and S is a narrowing strategy such that S^Σ is narrowing complete.

Extend the inference relation $\vdash_{SP,CS}$ (cf. 4.1) by S^Σ-controlled goal generation and refutation.

(1) Let $c = \{g_1, \ldots, g_m\} \Leftarrow h$ be a Gentzen clause and $CS = \{gs_1 \Leftarrow h_1, \ldots, gs_n \Leftarrow h_n\}$ be a set of Gentzen clauses such that for all $1 \leq i \leq m$ and $1 \leq j \leq n$, $h \Leftarrow h_j, gs_j \Leftarrow g_i \in ITh(SP)$. Then $\{g_1, \ldots, g_m\} \vdash_{SP,CS} \{h\}$ implies $\{c\} \cup CS \subseteq ITh(SP)$.

(2) Let $HS = \{g_1 \Leftarrow g : h_1, \ldots, g_n \Leftarrow g : h_n\}$ be a set of guarded clauses such that $\{h_1, \ldots, h_n\}$ is g-minimal w.r.t. $(TH(AX), GSub)$ and for all $1 \leq i, j \leq n$, $X_i =_{def} fresh(g_i \Leftarrow g \cup h_i)$ and $h_i = h_j$ implies $g_i = g_j$. Let

$$c \;=\; \{\exists X_1(g_1 \cup h_1), \ldots, \exists X_n(g_n \cup h_n)\} \Leftarrow g.$$

Then $\{\exists X_1(g_1 \cup h_1), \ldots, \exists X_n(g_n \cup h_n)\} \vdash_{SP,HS} \{g\}$ implies $\{c\} \cup HS \subseteq ITh(SP)$.

Proof. Follows directly from Thm. 3.6(1) and Cor. 3.9 if, at the place where the proof of Thm. 3.6(1) refers to Thm. 3.4 (deductive resolution and paramodulation are sound), we also use Thm. 10.6, which implies that the additional rules are sound as well. ❏

Definition 10.8 Let $>$ be a reduction ordering for SP (cf. 5.10) and Σ be an SP-compatible simplifier. If for all goals g and $\sigma : X \to NF_{SIG}$, $g\sigma \geq \Sigma(g)\sigma$, then Σ is **$>$-reductive** and the pair (SP, Σ) is **strongly terminating**. ❏

If (SP, Σ) is strongly terminating, the combination of Lemmata 9.6 and 9.7 can be generalized from S_R- to S_R^Σ-expansions (cf. 9.2 and 10.3):

Lemma 10.9 (completeness of S_R^Σ-expansions) Suppose that SP is normal form complete and ground confluent, (SP, Σ) is strongly terminating and R is a uniform redex selector (cf. 9.2). Then for all goals g and $\tau : X \to NF_{SIG}$,

$$g\tau \vdash_{SP} \emptyset \quad \text{implies} \quad \langle g, id\rangle \vdash^{S_R^\Sigma} \langle\emptyset, \rho\rangle$$

for some $\rho \leq \tau$ (cf. 9.5 and 10.3).

Proof by Noetherian induction on $g\tau$ along the reduction ordering $>_{SP}$, which exists by assumption. By Lemmata 9.6 and 9.7, there is an S_R-expansion E of $\langle g, id\rangle$ into a narrowing pair $\langle\emptyset, \sigma\rangle$ such that $\sigma \leq \tau$. If $g = \emptyset$, then the proof is complete with $\rho =_{def} id$. Otherwise E can be splitted into a one-step expansion of $\langle g, id\rangle$ into some $\langle g_1, \sigma_1\rangle \in S_R(\langle g, id\rangle)$ and an expansion of $\langle g_1, \sigma_1\rangle$ into $\langle\emptyset, \sigma\rangle$. By Lemma 8.3, $g\sigma_1 \vdash_{SP} g_1$ and $g_1\xi \vdash_{SP} \emptyset$ for ξ with $\sigma_1\xi = \sigma$. Since $\sigma \leq \tau$, there is $\gamma \in GSub$ with $\sigma\gamma = \tau$. Hence γ replaces all variables by normal forms and thus $g_1\xi \vdash_{SP} \emptyset$ implies $g_1\xi\gamma \vdash_{SP} \emptyset$. Let $\phi = \xi\gamma$. Then $\phi \in GSub$ and $\sigma_1\phi = \sigma_1\xi\gamma = \sigma\gamma = \tau$.

Since Σ is SP-compatible, $\Sigma(g_1) \Leftarrow g_1$ is an inductive SP-theorem. Since SP is normal form complete and ground confluent, Thm. 5.13 implies that $\Sigma(g_1) \Leftarrow g_1$ is reductively valid w.r.t. SP. Hence $g_1\phi \vdash_{SP} \emptyset$ implies $\Sigma(g_1)\phi \vdash_{SP} \emptyset$.

Since ϕ replaces all variables by normal forms, $g\sigma_1 \vdash_{SP} g_1$ implies $g\sigma_1\phi \vdash_{SP} g_1\phi$. Since $\langle g, id\rangle \vdash_{\bigvee SP} \langle g_1, \sigma_1\rangle$, $g\sigma_1\phi \neq g_1\phi$. Hence $g\sigma_1\phi \vdash_{SP} g_1\phi \vdash_{SP} \emptyset$ implies $g\sigma_1\phi >_{SP} g_1\phi$ and thus $g\tau = g\sigma_1\phi >_{SP} g_1\phi \geq_{SP} \Sigma(g_1)\phi$ because Σ is $>_{SP}$-reductive. Since ϕ is ground and replaces all variables by normal forms, $\Sigma(g_1)\phi \vdash_{SP} \emptyset$ implies

$$\langle\Sigma(g_1), id\rangle \;\vdash^{S_R^\Sigma}\; \langle\emptyset, \psi\rangle$$

for some $\psi \leq \phi$ by induction hypothesis. Hence by the definition of S_R (cf. 9.5),

$$\langle\Sigma(g_1), \sigma_1\rangle \;\vdash^{S_R^\Sigma}\; \langle\emptyset, \sigma_1\psi\rangle$$

and thus $\langle g, id\rangle \vdash^{S_R^\Sigma} \langle\emptyset, \sigma_1\psi\rangle$ because $\langle g_1, \sigma_1\rangle \in S_R(\langle g, id\rangle)$. Moreover, $\sigma_1\psi \leq \sigma_1\phi = \tau$. Hence the proof is complete with $\rho =_{def} \sigma_1\psi$. ❑

Given a uniform redex selector R, the following lemma extends our previous completeness result for S_R-expansions ([30], Thm. 8.4.9) to S_R^Σ-expansions. Since the only assumptions on Σ are SP-compatibility and $>_{SP}$-reductiveness, Lemma 10.10 also generalizes our completeness result for optimized narrowing ([30], Thm. 8.9.3) and those for normalizing narrowing (see above), even with "inductive consequences" (cf. [17], Sects. 2.2 and 2.3).

Lemma 10.10 (uniform redex selectors induce narrowing complete Σ-narrowing strategies) Suppose that SP is normal form complete and ground confluent, (SP, Σ) is strongly terminating and R is a uniform redex selector. Then S_R^Σ is narrowing complete (cf. 8.4).

Proof. Let g be a goal and $\tau : X \to NF_{SIG}$ such that $\langle g, id\rangle \vdash_{\bigvee SP} \langle\emptyset, \tau\rangle$. By Lemma 8.3, $g\tau$ is SP-convergent. Hence by Lemma 10.9, $\langle g, id\rangle \vdash^{S_R^\Sigma} \langle\emptyset, \rho\rangle$ for some $\rho \leq \tau$. ❑

Lemma 10.10 immediately implies the following corollary of Thm. 10.7:

Corollary 10.11 (inductive expansions with S_R^Σ-controlled goal generation and refutation are sound) Suppose that SP is free-constructor-based, normal form complete and ground confluent, (SP, Σ) is strongly terminating and R is a uniform redex selector. If the inference relation $\vdash_{SP,CS}$ (cf. 4.2) is extended by S_R^Σ-controlled goal generation and refutation, then Thm. 10.7 holds true. ❑

Expander (cf. Sect. 1.1) has a built-in reductive simplifier Σ, which partially evaluates equations and inequations and a number of functions and predicates on standard types such as natural numbers, lists, bags, sets and maps (cf. [34], Sect. 4). Higher-order transformations such as β-reduction of λ-expressions and continuation passing are also performed by Σ. Goals are simplified automatically at the beginning of a proof and after each deduction step.[18] Moreover, Expander provides a `Solve` command that performs a given number of S_R^Σ-controlled goal generation steps where R is either IN or $NEED$ (cf. Sect. 9). The combination of strategy-controlled goal generation and automatic simplification minimizes the number and the complexity of generated subgoals. For instance, a conjecture $g \Leftarrow h$ where g and h have only finitely many solutions can often be proved by applying `Solve` simultaneously to g and h and checking whether the solutions of h subsume those of g (cf. 1.1). In other words, if induction steps are not needed, the proof goes through almost automatically.

More precisely, `Solve` tries to solve a goal g by starting out from the narrowing pair $\langle g, id\rangle$ and performing a sequence of S_R^Σ-controlled goal generation steps. `Solve` stops after a given number of steps with a set nps of narrowing pairs. If $\langle\emptyset, \sigma\rangle \in nps$, then σ solves g. Narrowing pairs $\langle h, \tau\rangle \in nps$ such that $R(h) = \emptyset$ are unsolvable and are thus be deleted, provided that the specification is free-constructor-based, normal form complete and ground confluent (cf. Lemma 8.5(2)). Then `Solve` replaces g by the goal set

$$gs \;=\; \{EQ(\sigma) \mid \langle\emptyset, \sigma\rangle \in nps\} \cup \{h \cup EQ(\sigma) \mid \langle h, \sigma\rangle \in nps,\; R(h) \neq \emptyset\}.$$

Hence the step from g to gs is a sequence of S_R^Σ-controlled goal generation and refutation steps with a weakened applicability condition for goal refutation (cf. 8.7). For getting closer to the full applicability condition, `Solve` may be called with an option to detect and eliminate cycles among the S_R^Σ-expansions produced by `Solve`.

Example 10.12 (`GREATER`) If $\vdash_{SP,CS}$ is extended by S_{IN}^Σ-controlled goal generation, we obtain the following inductive expansion of $CS = \{\text{Conjecture 1}\}$ upon $SP =$ `GREATER` (cf. Ex. 4.3).

```
initial conclusion:
(1)  x > z
initial premise:
```

[18] except when a lemma is applied because some lemmas are to establish a specific goal structure – e.g. for achieving an induction hypothesis –, which might be changed by the simplifier.

```
(1)  x > y, y > z
atom 1 in conclusion goal 1 replaced with axiom GREATER1
atom 1 in conclusion goal 1 replaced with axiom GREATER2
```
conclusion:
```
(1)  x = s(x1), z = s(y1), x1 > y1
(2)  x = s(x1), z = 0
atom 3 in conclusion goal 1 replaced with conjecture 1
```
conclusion:
```
(1)  (x,z) >> (x1,y1) = true, x1 > y2, y2 > y1, x = s(x1), z = s(y1)
(2)  x = s(x1), z = 0
term at position 1 1 in conclusion goal 1 replaced with axiom GREATER3
```
conclusion:
```
(1)  x = s(x1), x1 > y2, y2 > y1, z = s(y1)
(2)  x = s(x1), z = 0
```
premise:
```
(1)  x > y, y > z
atom 1 in premise goal 1 replaced with axiom GREATER1
atom 1 in premise goal 1 replaced with axiom GREATER2
```
premise:
```
(1)  x = s(x1), y = s(y1), x1 > y1, s(y1) > z
(2)  x = s(x1), y = 0, 0 > z
atom 3 in premise goal 2 replaced with theorem 2
```
premise:
```
(1)  x = s(x1), y = s(y1), x1 > y1, s(y1) > z
atom 4 in premise goal 1 replaced with axiom GREATER1
atom 4 in premise goal 1 replaced with axiom GREATER2
```
premise:
```
(1)  z = s(y2), y1 > y2, x = s(x1), y = s(y1), x1 > y1
(2)  z = 0, x = s(x1), y = s(y1), x1 > y1
conjecture 1 has been proved
```

In contrast to the proof given in Ex. 4.3, the above expansion contains only one application of a lemma (Theorem 2 of `GREATER`). ❏

Example 10.13 (`DIVISION`) If $\vdash_{SP,CS}$ is extended by S^{Σ}_{IN}-controlled goal generation, we obtain the following inductive expansion of $CS = \{\text{Conjecture 1}\}$ upon $SP =$ `DIVISION` (cf. Ex. 4.4).

initial **conclusion**:
```
(1)  x = (q*y)+r, r < y
```
initial **premise**:
```
(1)  0 < y, x div y = (q,r)
term at position 1 2 1 in conclusion goal 1 replaced with axiom DIVISION1
term at position 1 2 1 in conclusion goal 1 replaced with axiom DIVISION2
```
conclusion:
```
(1)  q = s(x1), ((x1*y)+y)+r = x, r < y
(2)  q = 0, r = x, r < y
atom 2 in conclusion goal 1 replaced with theorem 1
```
conclusion:
```
(1)  (x1*y)+r = x-y, x >= y, q = s(x1), r < y
```

```
(2)  q = 0, r = x, r < y
atoms 1 4 in conclusion goal 1 replaced with conjecture 1
conclusion:
(1) { (y, x, q, r) >> (y,x-y,x1,r) = true, 0 < y, (x-y) div y = (x1, r),
      y <= x, q = s(x1) }
(2)  q = 0, r = x, r < y
term at position 1 1 in conclusion goal 1 replaced with axiom DIVISION6
conclusion:
(1)  y <= x, 0 < y, (x-y) div y = (x1,r), q = s(x1)
(2)  q = 0, r = x, r < y
premise:
(1)  0 < y, x div y = (q,r)
term at position 2 1 in premise goal 1 replaced with axiom DIVISION3
term at position 2 1 in premise goal 1 replaced with axiom DIVISION4
premise:
(1)  y <= x, 0 < y, (x-y) div y = (q1,r), s(q1) = q
(2)  x < y, 0 = q, x = r, 0 < y
term at position 1 1 replaced with equation 3 in premise goal 2
premise:
(1)  y <= x, 0 < y, (x-y) div y = (q1,r), s(q1) = q
(2)  r < y, 0 = q, x = r, 0 < y
conjecture 1 has been proved
```

In contrast to the proof given in Ex. 4.4, the above expansion contains only one application of a lemma (Theorem 1 of DIVISION). ❑

Inductive expansions of program correctness conditions occurring in different application areas can be found in [34], Sect. 7. Even the simple ones given here reveal the power of simplification and strategy-controlled goal generation. Many lemmas, in particular those derived from only-if-completions (cf. 5.6), are no longer needed. Proofs evolve more automatically. By Thm. 10.7, this effect is guaranteed if the underlying specification is free-constructor-based and canonical. We conclude that the proof-theoretical benefit from canonicity, which is well-known in pure equational reasoning, is passed on to the higher level of inductive reasoning with Gentzen clauses.

References

[1] A. Bouhoula, E. Kounalis, M. Rusinowitch, *Automated Mathematical Induction*, Report INRIA Lorraine No. 1636 (1992)

[2] J. Avenhaus, K. Becker, *Conditional Rewriting modulo a Built-in Algebra,* SEKI Report SR-92-11, Universität Kaiserslautern (1992)

[3] S. Antoy, R. Echahed, M. Hanus, *A Needed Narrowing Strategy,* Proc. 21st ACM Symp. on Principles of Programming Languages (1994) 268-279

[4] H. Bertling, H. Ganzinger, *Completion-Time Optimization of Rewrite-Time Goal Solving,* Proc. RTA '89, Springer LNCS 355 (1989) 45-58

[5] N. Dershowitz, J.-P. Jouannaud, *Rewrite Systems,* in: J. van Leeuwen, ed., Handbook of Theoretical Computer Science, Elsevier (1990) 243-320

[6] N. Dershowitz, M. Okada, G. Sivakumar, *Confluence of Conditional Rewrite Systems,* Proc. CTRS '87, Springer LNCS 308 (1988) 31-44

[7] N. Dershowitz, M. Okada, G. Sivakumar, *Canonical Conditional Rewrite Systems,* Proc. CADE '88, Springer LNCS 310 (1988) 538-549

[8] D. Duffy, *Principles of Automated Theorem Proving,* John Wiley & Sons 1991

[9] R. Echahed, *On Completeness of Narrowing Strategies,* Proc. CAAP '88, Springer LNCS 299 (1988) 89-101

[10] R. Echahed, *Uniform Narrowing Strategies,* Proc. 3rd ALP, Springer LNCS 632 (1992) 259-275

[11] H. Ehrig, B. Mahr, *Fundamentals of Algebraic Specification 1,* Springer 1985

[12] M. Fay, *First Order Unification in an Equational Theory,* Proc. 4th Workshop on Automated Deduction, Academic Press (1979) 161-167

[13] L. Fribourg, *Handling Function Definitions through Innermost Superposition and Rewriting,* Proc. RTA '85, Springer LNCS 202 (1985) 325-344

[14] A. Geser, *Relative Termination,* Ph.D. thesis, Report 91-03, FB Informatik, Universität Ulm (1991)

[15] J.A. Goguen, J.W. Thatcher, E.G. Wagner, *An Initial Algebra Approach to the Specification, Correctness and Implementation of Abstract Data Types,* in: R. Yeh, ed., Current Trends in Programming Methodology 4, Prentice-Hall (1978) 80-149

[16] J. Guttag, E. Horowitz, D.R. Musser *Abstract Data Types and Software Validation*, Report ISI/RR-76-48, University of Southern California 1976

[17] M. Hanus, *The Integration of Functions into Logic Programming: From Theory to Practice,* J. Logic Programming 19&20 (1994) 583-628

[18] S. Hölldobler, *Foundations of Equational Logic Programming,* Springer (1989)

[19] J.M. Hullot, *Canonical Forms and Unification,* Proc. 5th CADE, Springer LNCS 87 (1980) 318-334

[20] J.-P. Jouannaud, H. Kirchner, *Completion of a Set of Rules Modulo a Set of Equations,* SIAM Journal of Computing 15 (1986) 1155-1194

[21] J.-P. Jouannaud, E. Kounalis, *Automatic Proofs by Induction in Equational Theories without Constructors,* Proc. LICS '86 (1986) 358-366

[22] J-P. Jouannaud, B. Waldmann, *Reductive Conditional Term Rewriting Systems,* Proc. Conf. Formal Description of Programming Concepts III, North-Holland (1986) 223-244

[23] S. Kaplan, *Conditional Rewrite Rules,* Theoretical Computer Science 33 (1984) 175-194

[24] S. Kaplan, *Simplifying Conditional Term Rewriting Systems: Unification, Termination and Confluence,* J. Symbolic Computation 4 (1987) 295-334

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

[26] E. Kounalis, M. Rusinowitch, *Mechanizing Inductive Reasoning*, EATCS Bulletin 41 (1990) 216-226

[27] W. Küchlin, *Inductive Completion by Ground Proof Transformation*, in: H. Ait-Kaci, M. Nivat, eds., Resolution of Equations in Algebraic Structures, Vol. 2, Academic Press (1989) 211-244

[28] G. Malcolm, J.A. Goguen, *Proving Correctness of Refinement and Implementation*, Technical Monograph PRG-114, Oxford University Computing Lab 1994

[29] P. Padawitz, *Strategy-Controlled Reduction and Narrowing,* Proc. RTA '87, Springer LNCS 256 (1987) 242-255

[30] P. Padawitz, *Computing in Horn Clause Theories,* Springer 1988

[31] P. Padawitz, *Inductive Expansion: A Calculus for Verifying and Synthesizing Functional and Logic Programs,* J. Automated Reasoning 7 (1991) 27-103

[32] P. Padawitz, *Reduction and Narrowing for Horn Clause Theories,* The Computer J. 34 (1991) 42-51

[33] P. Padawitz, *Deduction and Declarative Programming,* Cambridge University Press 1992

[34] P. Padawitz, *Expander: A System for Testing and Verifying Functional-Logic Programs,* Report No. 522/1994, FB Informatik, Universität Dortmund

[35] P. Padawitz, *Swinging Data Types: The Dielectic between Actions and Constructors,* Report, FB Informatik, Universität Dortmund

[36] U. Reddy, *Term Rewriting Induction*, Proc. CADE 10, Springer LNCS 449 (1990) 162-177

[37] R. Reiter, *On Closed World Data Bases,* Proc. Logic and Data Bases, Plenum Press (1978)

[38] P. Réty, *Improving Basic Narrowing Techniques,* Proc. RTA '87, Springer LNCS 256 (1987) 228-241

[39] E. Shapiro, *The Family of Concurrent Logic Programming Languages,* ACM Computing Surveys 21 (1989) 413-510

[40] J.R. Slagle, Automated Theorem-Proving for Theories with Simplifiers, Commutativity and Associativity, Journal ACM 21 (1974) 622-642

[41] M. Stickel, *Automated Deduction by Theory Resolution,* J. Automated Reasoning 1 (1985) 333-356

[42] M. Wirsing, *Algebraic Specification,* in: J. van Leeuwen, ed., Handbook of Theoretical Computer Science, Elsevier (1990) 675-788

[43] H. Zhang, J.-L. Remy, *Contextual Rewriting,* Proc. RTA '85, Springer LNCS 202 (1985) 46-62