Lecture Notes in Artificial Intelligence 624

Subseries of Lecture Notes in Computer Science
Edited by J. Siekmann

Lecture Notes in Computer Science
Edited by G. Goos and J. Hartmanis

A. Voronkov (Ed.)

Logic Programming and Automated Reasoning

International Conference LPAR '92
St. Petersburg, Russia, July 15-20, 1992
Proceedings

Springer-Verlag
Berlin Heidelberg New York
London Paris Tokyo
Hong Kong Barcelona
Budapest

Computing Induction Axioms

Christoph Walther
Technische Hochschule Darmstadt [1]

Abstract The computation of induction axioms in the explicit induction paradigm is investigated. A simple notion with a well-defined semantics, called a *relation description*, is proposed as the elementary building block for automated reasoning on induction axioms. It is demonstrated how relation descriptions can be created, manipulated and compared by machine so that useful and strong induction axioms can be derived from them. For each of these operations the semantics of their effects and a precise semantical justification for their application is given. It is shown how the proposed framework can be used to describe the methods implemented in *Boyer* and *Moore's* NQTHM system in an abstract setting with a well-defined semantics. NQTHM's merging and subsumption heuristics for combining and comparing induction schemas are analysed as an example, how a rigorous formal approach may uncover implicit assumptions and hidden flaws. A *containment* test then is proposed as a powerful, non-heuristic, and completeness preserving operation to select among competing induction schemas. The motivation for this test evolves straightforwardly in the given framework by recognizing the semantics of the intended effect.

1 *Introduction*

Research on automated induction these days is based on two competing paradigms: *inductive completion* (also termed *inductionless induction* or, less confusingly, *proof by consistency*) evolved from the Knuth-Bendix Completion Procedure (KBCP). It is based on the observation, that for certain theories, the notions of truth and satisfaction coincide, so that the truth of a conjecture φ relative to some set of axioms Φ can be shown by proving that $\Phi \cup \{\varphi\}$ is consistent using (a modification of) the KBCP. Inductive completion is investigated intensively and meanwhile there is a tremendous list of relevant publications (see e.g. [Kapur and Musser, 1987; Reddy, 1990] for theoretical foundations and further references). The other research paradigm which will be addressed in this paper, here called *explicit induction*, resembles the more familiar idea of induction theorem proving: to verify a conjecture $\forall n \in \mathcal{N}.\ \varphi(n)$, prove the *induction base* $\varphi(0)$, and then the *induction step* $[\forall n \in \mathcal{N}.\ \varphi(n) \rightarrow \varphi(n+1)]$. This approach follows *Peano's* induction principle for natural numbers which, using a *well-founded set* $(M, <_M)$, can be generalized yielding the *Generalized Principle of Noetherian Induction*: $[\forall m \in M.\ [\forall k \in M.\ k <_M m \rightarrow \varphi(k)] \rightarrow \varphi(m)] \rightarrow \forall m \in M.\ \varphi(m)$. By the generalized induction principle we are neither restricted to inferences from *n* to *n+1* nor to the set of natural numbers when proving statements by induction.

The investigation of automated theorem proving based on explicit induction dates back more than 20 years, cf. [Burstall, 1969; Boyer and Moore, 1975; Aubin, 1976] among others. Compared with inductive completion, work based on this paradigm is much less intensive as revealed by an inspection of the relevant journals and conference proceedings. Most work is related to the methods implemented in the NQTHM system developed by *Boyer* and *Moore* [Boyer and Moore, 1979; 1988] which presently can be viewed as the most successful endeavor in induction theorem proving (also compared with systems based on the completion approach). But unfortunately, the methods developed by Boyer and Moore *to guide the search* for an induction proof lack a thorough formal foundation. The system and the methods it implements are documented mostly informal or by examples

[1] Author's address: Technische Hochschule Darmstadt, Fachbereich Informatik, Alexanderstr. 10, D 6100 Darmstadt, Germany. E-mail: walther@inferenzsysteme.informatik.th-darmstadt.de

convincing a reader *that* and *how* it works, but providing no insight in the *principles* underlying the methods. Also the original publications resemble more a detailed program documentation which uses a bulk of non-standard program-technical terms so that they are very difficult to understand and they scarcely provide general insights beyond the documented system. These drawbacks are well recognized in the community, e.g. others feel motivated to undertake a "rational reconstruction" of the Boyer-Moore techniques, cf. [Bundy et. al., 1989; Stevens, 1989]. They also provide a motivation for the work to be presented here.

2 *Relation Descriptions*

One crucial point in proving theorems by the explicit induction approach is to find an induction axiom for a given conjecture φ as the *right* instance of the (generalized) induction principle. The invention of a "successful" well-founded relation $<_M$ for a statement constitutes the creativity of a human, and also of an automated expert in induction theorem proving. This problem has two aspects: a *proof strategic* aspect which means that the relation $<_M$ induced upon *allows to prove* $[\forall m \in M.[\forall k \in M.\ k <_M m \rightarrow \varphi(k)] \rightarrow \varphi(m)]$, and a *soundness* aspect, which means that relation $<_M$ induced upon must be *well-founded* indeed.

In order to reason about well-founded relations by machine, we develop a representation for them within the framework of first-order logic. We call a substitution σ a *substitution for* x^* iff $x^* \in \mathcal{V}_w$ is a $|w|$-tuple of distinct variables and for each substitution pair y/t in σ, all variables in t as well as y are members of x^*. As a non-standard feature we also allow substitution pairs of form y/y and we define (as usual) the *domain* Dom(σ) of a substitution σ as the set of all left-hand sides of the substitution pairs in σ.

We now define so-called *atomic relation descriptions* (or *atomic r-descriptions* for short) as the elementary building blocks to represent well-founded relations. To do so we assume a signature $\Sigma = \Sigma^d \cup \Sigma^c$ for function symbols, where Σ^d contains function symbols, as e.g. plus, times, append, ..., *defined* by some algorithms and Σ^c contains *constructor* functions, as e.g. 0, succ, nil, cons, ... We also assume a *standard interpretation* M for the functions in Σ, accepting some ground term from $\mathcal{T}(\Sigma)$ as input, e.g. plus(succ(0) succ(0)), and *always* yields a *constructor* ground term from $\mathcal{T}(\Sigma^c)$ as output, e.g. succ(succ(0)). Also M(q)=q may be assumed for all constructor ground terms q. If the standard interpretation M satisfies a formula ψ, then ψ is an *inductive truth*, which intuitively means that all quantifiers in ψ are meant to range over constructor ground terms only.[2] To test whether a formula ψ is a member of the set $\mathrm{Th}_{\mathrm{Ind}}$ of all *inductive* true theorems is that what an induction theorem prover is good for.[3]

Definition 2.1 An *atomic r-description* C for $x^* \in \mathcal{V}_w$ is a pair (φ, Δ) where φ, called the *range formula* of C, is a quantifier-free formula with at most the variables in x^* as free variables and Δ is a finite and non-empty set of substitutions $\delta_1, \delta_2, \ldots, \delta_n$ for x^*, called the *domain substitutions* of C.

The set *rV*(C) of all *relevant* variables of an atomic r-description C is the set of all variables mentioned in φ or in Δ. A relevant variable *x* of C also is called an *induction variable* of C iff $x \in \mathrm{Dom}(\delta)$ for some domain substitution $\delta \in \Delta$. The set *iV*(C) of all *induction variables* of C is given as Dom(Δ), where $\mathrm{Dom}(\Delta) = \mathrm{Dom}(\delta_1) \cup \ldots \cup \mathrm{Dom}(\delta_n)$.

An atomic r-description C for $x_1 \ldots x_n$ defines a relation $\rightarrow_C$ on the cartesian product $\mathcal{T}(\Sigma^c)_w$ of constructor ground terms by: $r_1 \ldots r_n \rightarrow_C q_1 \ldots q_n$ iff $\mathrm{M}[\![x_1 \ldots x_n / r_1 \ldots r_n]\!] \models \varphi$ and some $\delta \in \Delta$ exists such that $q_i = \mathrm{M}[\![x_1 \ldots x_n / r_1 \ldots r_n]\!] (\delta(x_i))$ for all $x_i \in \mathrm{Dom}(\delta)$. ■

[2] We confine ourselves here with this handwaving formulation of "inductive truth" for sake of brevity and refer to [Walther, 1991a] for a precise definition.

[3] Of course, an induction theorem prover only provides a *sufficient* requirement for $\psi \in \mathrm{Th}_{\mathrm{Ind}}$, because due to *Gödel's First Incompleteness Theorem* neither $\mathrm{Th}_{\mathrm{Ind}}$ is decidable nor is semi-decidable.

For each atomic r-description (φ, Δ), the range formula φ defines the *range* of $\rightarrow_C$, i.e. the set of all $|w|$-tuples in $\mathcal{T}(\Sigma^c)_w$ which posses a "successor" wrt. $\rightarrow_C$, and the domain substitutions in Δ define how such "successors" look like. For instance, C_1 = (x>y, {{x/x, y/y+1}}) is an atomic r-description for $xy\in \mathcal{V}_{\text{number,number}}$ defining the relation $\rightarrow_{C_1}$ as $(n,m) \rightarrow_{C_1} (n,m+1)$ iff $n>m$, cf. Figure 1(i), and (5,3) $\rightarrow_{C_1}$ (5,4) $\rightarrow_{C_1}$ (5,5) is a chain in $\mathcal{T}(\Sigma^c)_{\text{number,number}}$ wrt. the relation $\rightarrow_{C_1}$.[4] The unshaded area in the diagram denotes the range of the relation, i.e. the set of all pairs possessing a "successor" wrt. $\rightarrow_{C_1}$.

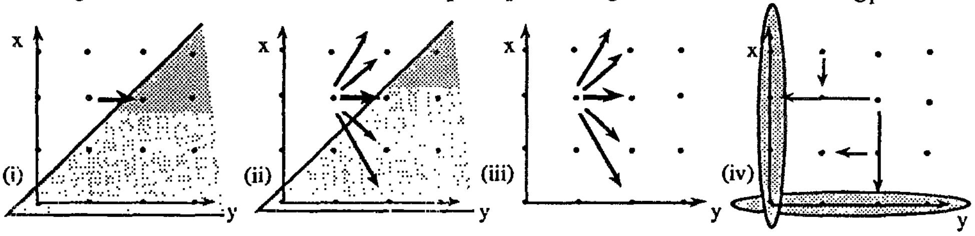

Figure 1

Another atomic r-description for $xy\in \mathcal{V}_{\text{number,number}}$ is C_2 = (x>y, {{y/y+1}}), cf. Fig. 1(ii), and (5,3) $\rightarrow_{C_2}$ (17,4) $\rightarrow_{C_2}$ (8,5) $\rightarrow_{C_2}$ ... is a chain in $\mathcal{T}(\Sigma^c)_{\text{number,number}}$ wrt. the relation $\rightarrow_{C_2}$ defined by C_2. Note that y is the only *induction variable* of C_2 and therefore the x-components in the pairs of this $\rightarrow_{C_2}$-chain, viz. 5, 17, 8, ... , are arbitrarily chosen. Since C_2 possesses no substitution pair of form x/t, nothing is demanded by C_2 for the x-components of $\rightarrow_{C_2}$. The only difference between C_1 and C_2 is the substitution pair x/x in the domain substitution of C_1 and its omission alters the defined relation significantly. Both r-descriptions reveal why we use substitution pairs of form x/x because otherwise a relation such as $\rightarrow_{C_2}$ would be identified with $\rightarrow_{C_1}$ and therefore could not be defined.

Using irrelevant variables, we are able to compare the relations defined by atomic r-descriptions with set-theoretic inclusion even if they are defined on *different spaces* (for this reason we insist on their existence). For instance, C_3 = {(y=y, {{y/y+1}}), cf. Fig. 1(iii), also is an atomic r-description for $xy\in \mathcal{V}_{\text{number,number}}$ but with x being *irrelevant* for C_3. However, C_3 is not an atomic r-description only for x, because y is used in the range formula and also in the domain substitution of C_3. We find $\rightarrow_{C_1} \subset \rightarrow_{C_2} \subset \rightarrow_{C_3}$, if C_1, C_2, and C_3 all are considered as atomic r-descriptions for $xy\in \mathcal{V}_{\text{number,number}}$, cf. Fig. 1(i, ii, iii).

Using atomic r-descriptions, so-called (*composed*) *relation descriptions* (or (*composed*) *r-descriptions* for short) are defined to represent well-founded relations:

Definition 2.2 An *r-description* D for $x^*\in \mathcal{V}_w$ is a finite and non-empty set $\{C_1, \dots, C_k\}$ of atomic r-descriptions for x^*. The set rV(D) of *relevant variables* of D is given as $rV(D) := rV(C_1) \cup \dots \cup rV(C_k)$, and $iV(D) := iV(C_1) \cup \dots \cup iV(C_k)$ is the set of *induction variables* of D. Each r-description D defines a relation $\rightarrow_D$ on $\mathcal{T}(\Sigma^c)_w$ by $\rightarrow_D := \rightarrow_{C_1} \cup \dots \cup \rightarrow_{C_k}$. An r-description D is called *well-founded* iff $\rightarrow_D$ is a well-founded (or *noetherian*) relation, and if so $(\mathcal{T}(\Sigma^c)_w, \rightarrow_D)$ is a well-founded set. ■

For instance, D_1 = {(x>y, {{x/x, y/y+1}})}, cf. Figure 1(i), and D_2 = {(x>y, {{y/y+1}})}, cf. Figure 1(ii), are r-descriptions for $xy\in \mathcal{V}_{\text{number,number}}$, and D_1 is well-founded, whereas D_2 is not.

In conclusion, we use relation descriptions to *define* relations of $\mathcal{T}(\Sigma^c)_w$. A relation description D is a *syntactical* (and finite) object, which, for instance, can be stored in or modified by a computer, whereas the relation $\rightarrow_D$ defined by D is a *semantical* (and

[4] To ease readability we shall use the usual mathematical notation, as e.g. $x>y$, $x-y$, $x+1$, $x-1$, 1 etc., instead of the terms formally required, as e.g. gt(x y)=true, minus(x y), succ(x), pred(x), succ(0) etc.

generally infinite) object not explicitly given. This means that $\rightarrow_D$ defines the *semantics* of D. We may use (inspect, modify, etc.) relation descriptions D to reason about relations $\rightarrow_D$, as e.g. proving that $\rightarrow_D$ is well-founded, that $\rightarrow_D$ is contained in some other relation $\rightarrow_{D'}$ etc., like first-order formulas are used (i.e. inspected, modified, etc.) to reason about truth.

3 Induction Axioms Defined by Relation Descriptions

Since each well-founded relation description D represents a well-founded relation and likewise a well-founded set, we may uniformly associate an induction axiom with D. Given a formula ψ with free variables y* and an r-description D = $\{(\varphi_1, \Delta_1), \ldots, (\varphi_k, \Delta_k)\}$ for x* $\in \mathcal{V}_w$, D is said to be in the *scope* of ψ iff x* $\subset$ y*. If so, D is associated with $k+1$ so-called *induction formulas*, the *base case* ψ_0 and the *step cases* $\psi_1, \ldots, \psi_k$. The range formulas in the atomic r-descriptions of D define the cases of the induction steps and the domain substitutions are used to form the induction hypotheses. The base case is obtained as the complement of all the range formulas in D.

Consider, for instance, some formula ψ[x,y,z] with free variables *x,y,z* and an r-description D = {(x > y, {{x/x, y/y+1}, {x/x−1}})} for *xy*, cf. Figure 2(i). Since D is in the scope of ψ, the induction formulas to prove $\forall$x,y,z. ψ[x,y,z] are computed as:

$\psi_0 = \forall x,y.\ x \le y \rightarrow \forall z.\ \psi[x,y,z]$ and

$\psi_1 = \forall x,y.\ x > y \wedge \forall z.\ \psi[x/x, y/y+1, z] \wedge \forall z,u.\ \psi[x/x\text{-}1, y/u, z] \rightarrow \forall z.\ \psi[x,y,z]$.

Since the variable z from ψ is not a relevant variable of D, *z* remains universally quantified in the induction hypotheses $\forall$z. ψ[x/x, y/y+1, z] and $\forall$z,u. ψ[x/x-1, y/u, z] as well as in the induction conclusion $\forall$z. ψ[x,y,z]. Generally, the smaller *rV*(D) is, the more variables remain universally quantified in the induction hypotheses (and conclusions), and consequently the stronger the induction hypotheses are. So we should do our very best to obtain r-descriptions D with *rV*(D) as small as possible, cf. Sections 6 and 8. The variable *y* in our example is bound as an induction variable in the first induction hypothesis, but is *universally quantified* (after renamed to *u*) in the second one. This is, because we find for all *n*>*m* that $(n,m) \rightarrow_D (n,m+1)$ by the first domain substitution of D but with y$\notin$ Dom({x/x−1}), the second substitution yields $(n,m) \rightarrow_D (n-1, \ldots)$, where ... stands for *any value*, cf. Figure 2(i). So we should do our very best to obtain domain substitutions δ with Dom(δ) as small as possible, cf. Section 6, to obtain hypotheses as strong as possible.

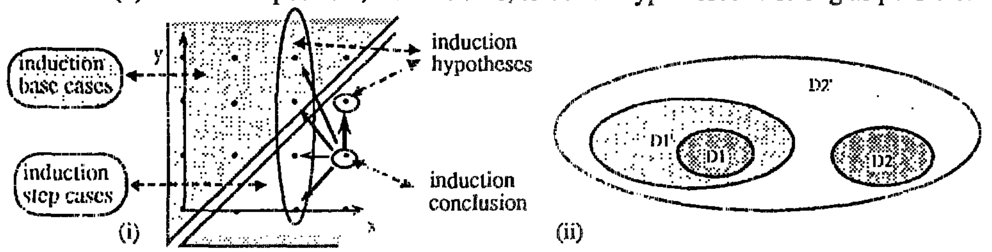

Figure 2

Based on our definition, for *each* formula ψ and *each* well-founded relation description D, an induction axiom [$\psi_0 \wedge \ldots \wedge \psi_k \rightarrow \forall y^*\ \psi$] can be computed, provided D is in the scope of ψ. Then the formula $\forall$y* ψ can be proved by verification of [$\psi_0 \wedge \ldots \wedge \psi_k$]$\in$ Th_{Ind}. The crucial point is however (1) to provide an induction theorem proving system with a set of well-founded relation descriptions, and (2) to provide the system with a facility to select those relation descriptions from the set, which are likely to be successful for an induction proof. We shall present solutions for both problems in the remainder of this paper.

4 R-Descriptions Obtained from Data Structures and Algorithms

Having developed a uniform mechanism to obtain induction axioms from well-founded relation descriptions, one may ask where these relation descriptions actually come from. The answer is quite simple: some of them are computed from the data structures and the (terminating) algorithms defined by the user of an induction theorem prover, and the remaining ones are computed from the r-descriptions already known to the system. We will begin with the first case and continue with the second one in Sections 6, 8 and 9.

Each data structure s, as e.g. *structure* empty add(head:number tail:list):list, defines an r-description D_s for $x \in \mathcal{V}_s$, where the domain substitutions are built with the reflexive selectors of s.[5] For instance, D_{list} = {(x=add(head(x) tail(x)), {{x/tail(x)}})} is obtained from list and D_{sexpr} = {(x=cons(car(x) cdr(x)), {{x/car(x)}, {x/cdr(x)}})} is obtained from the data structure *structure* atom(..) nil cons(car:sexpr cdr:sexpr):sexpr. The relation $\rightarrow_{D_s}$ (or $\rightarrow_s$ for short) defined by D_s on the set $\mathcal{T}(\Sigma^c)_s$ of constructor ground terms of sort s is given as: r $\rightarrow_s$ q iff r=*cons*(r_1 ... r_n) for some constructor *cons* of s and q = r_i for some i. In other words, r $\rightarrow_s$ q iff r is built with a reflexive constructor and q is a direct subterm of r. The relation $\rightarrow_s$ is known as the *structural order* of s and is well-founded, because no term has infinitely many proper subterms. Consequently, D_s is a well-founded r-description.

We find, for instance, add(0 add(1 empty)) $\rightarrow_{list}$ add(1 empty) $\rightarrow_{list}$ empty, but not add(0 add(1 empty)) $\rightarrow_{list}$ add(0 empty), because add(0 empty) is not a subterm of add(0 add(1 empty)). We also find [[a.b].c] $\rightarrow_{sexpr}$ [a.b] $\rightarrow_{sexpr}$ a, [a.b] $\rightarrow_{sexpr}$ b and [[a.b].c] $\rightarrow_{sexpr}$ c as chains in $\mathcal{T}(\Sigma^c)_{sexpr}$ wrt. $\rightarrow_{sexpr}$.[6] Since each D_s is a well-founded r-description, we may e.g. prove a statement $\forall$ x:sexpr ψ[x] by structural induction of sexpr yielding the step case $\forall$ x:sexpr x=cons(car(x) cdr(x)) $\wedge$ ψ[x/car(x)] $\wedge$ ψ[x/cdr(x)] $\rightarrow$ ψ[x] and the base case $\forall$ x:sexpr x$\neq$cons(car(x) cdr(x)) $\rightarrow$ ψ[x].

In a similar way r-descriptions are obtained from algorithms: For each terminating algorithm f we assume a relation description D_f which contains an atomic r-description for each recursive case of f. The conditions for the parameters define the range formulas and the assignments of the formal parameters to their actual parameters in the recursive calls define the domain substitutions of D_f. For instance, we obtain from the algorithms

```
function quotient(x,y:number):number ⇐        function gcd(x,y:number):number ⇐
  if y = 0 then 0                               if x = 0 ∨ y=0 then max(x y)
  if x < y then 0                               if x ≥ y ∧ y≠0 then gcd(x−y y)
  if x ≥ y ∧ y≠0 then 1+quotient(x−y y)         if x ≤ y ∧ x≠0 then gcd(x y−x)
```

the r-descriptions $D_{quotient}$ = {(x $\geq$ y $\wedge$ y$\neq$0, {{x/x−y, y/y}})} and D_{gcd} = {(x $\geq$ y $\wedge$ y$\neq$0, {{x/x−y, y/y}}), (x $\leq$ y $\wedge$ x$\neq$0, {{x/x, y/y−x}})}. The relation $\rightarrow_{D_f}$ (or $\rightarrow_f$ for short) defined by D_f is given as: r* $\rightarrow_f$ q* iff there is some case "*if* φ_i *then* r_i" in f and some term $f(t^*)$ in φ_i or r_i such that $M[\![x^*/r^*]\!] \models \varphi_i$ and q*= $M[\![x^*/r^*]\!]$ (t*). The relation $\rightarrow_f$, called the *computation order* of f, is well-founded because f terminates. Consequently, D_f is a well-founded relation description.

For instance, D_{gcd} is a well-founded r-description for $xy \in \mathcal{V}_{number,number}$, cf. Fig. 1(iv), and an example of a chain in $\mathcal{T}(\Sigma^c)_{number,number}$ wrt. $\rightarrow_{gcd}$ is (3,5) $\rightarrow_{gcd}$ (3,2) $\rightarrow_{gcd}$ (1,2) $\rightarrow_{gcd}$ (1,1) $\rightarrow_{gcd}$ (0,1). Consequently we may prove a statement $\forall$ x,y:number ψ[x,y] by induction on the computation order of gcd yielding the step cases $\forall$ x,y:number x $\geq$ y $\wedge$ y$\neq$0 $\wedge$ ψ[x/x−y, y/y] $\rightarrow$ ψ[x,y] and $\forall$ x,y:number x $\leq$ y $\wedge$ x$\neq$0 $\wedge$ ψ[x/x, y/y−x] $\rightarrow$ ψ[x,y].

[5] A function symbol is called *reflexive* iff its rangesort also appears as one of its domainsorts, and is *irreflexive* otherwise. For instance, add and tail are reflexive, and empty and head are irreflexive.
[6] [a.b] is a shorthand notation for the dotted pair cons(a b).

5 *The Induction Heuristic*

The r-descriptions given by the data structures and the algorithms provide the raw material to compute an useful induction axiom for a formula ψ. If ψ has as free variables y*, we may use any r-description for x* which is in the scope of ψ, i.e. $x^* \subset y^*$, and compute an induction axiom to prove $\forall y^*\ \psi$ (see Section 3). However, albeit this approach is sound, selecting an r-description only because it is in the scope of ψ would be as reasonable as tossing up a coin. As a *proof strategic* requirement, we should use an r-description which provides us with the induction hypotheses necessary to prove the induction conclusions. We therefore formulate the so-called *induction heuristic* to compute - this is to be hoped - such an r-description. To do so, we need to rename the relevant variables of an r-description, yielding a *renamed variant*, and we also allow that non-induction variables may be instantiated, yielding an *instantiated variant* of an r-description:

Definition 5.1 Given an r-description D for $x^* \in \mathcal{V}_w$ and a variable renaming ν for the variables in x*, the r-description $\nu(D)$ for $\nu(x^*) \in \mathcal{V}_w$ is called a *renamed variant* of D, where $\nu(D)$ contains an atomic r-description $(\nu(\varphi), \{\nu(\delta_1), \nu(\delta_2), \dots, \nu(\delta_n)\})$ for each atomic r-description $(\varphi, \{\delta_1, \delta_2, \dots, \delta_n\}) \in D$ and each $\nu(\delta_i)$ is obtained from δ_i by replacing each substitution component x/t in δ_i with $\nu(x)/\nu(t)$.

The *instantiated variant* $\theta(D)$ of D *exists* for some substitution θ iff $\text{Dom}(\theta) \cap iV(D) = \emptyset$ (i.e. θ replaces no variable some domain substitution in D also replaces). If so, $\theta(D)$ contains an atomic r-description $(\theta(\varphi), \{\theta(\delta_1), \theta(\delta_2), \dots, \theta(\delta_n)\})$ for each atomic r-description $(\varphi, \{\delta_1, \delta_2, \dots, \delta_n\}) \in D$ and each $\theta(\delta_i)$ is obtained from δ_i by replacing each substitution component x/t in δ_i with $x/\theta(t)$. The instantiated variant $\theta(D)$ is an r-description for z* which is any list of all distinct variables contained in $\theta(x^*)$. ■

Since the relevant variables of an r-description D only act as *names* of the components of the |w|-tuples related by $\rightarrow_D$, we may rename the variables in x* using any variable renaming ν without altering the defined relation, i.e. $\rightarrow_{\nu(D)} = \rightarrow_D$. For instance, $\nu(D) = \{(u>v, \{\{u/u, v/v+1\}\})\}$ is an r-description for $uv \in \mathcal{V}_{\text{number,number}}$ obtained as a renamed variant of $D = \{(x>y, \{\{x/x, y/y+1\}\})\}$ for $xy \in \mathcal{V}_{\text{number,number}}$. We obtain an *instantiated variant* $\theta(D)$ of an r-description D, if we replace some non-induction variables y of D by some terms t. For instance, $\theta(D) = \{(x \geq u \times v \wedge u \times v \neq 0, \{\{x/x-u \times v\}\})\}$ is an r-description for xuv obtained from $D = \{(x \geq y \wedge y \neq 0, \{\{ x/x-y \}\})\}$ by replacing y with $u \times v$. The relation $\rightarrow_{\theta(D)}$ defined by an instantiated variant $\theta(D)$ of D is well-founded if $\rightarrow_D$ is.

Using renamed and instantiated variants, the induction heuristic now can be formulated. Given a formula ψ with free variables y*, we scan ψ looking for "calls" $f(t_1 \dots t_n)$ of some algorithms f in ψ such that (*) for all induction variables x_i of the r-description D_f, the corresponding arguments t_i in the call all are *distinct variables*. We call the r-description D_f *effective* for $f(t_1 \dots t_n)$ (and in turn for ψ) if the above requirement (*) is satisfied. Then, each such call $f(t_1 \dots t_n)$ suggests an induction axiom built with the r-description $\theta(\nu(D_f))$. Here, $\nu(D_f)$ is a *renamed variant* of D_f, where ν is the substitution containing each substitution pair x_i/t_i and with D_f being effective for $f(t_1 \dots t_n)$, ν is a variable renaming. The r-description $\theta(\nu(D_f))$ is an *instantiated variant* of $\nu(D_f)$, where θ is the substitution containing each substitution pair y_j/t_j such that y_j is a relevant but a non-induction variable of D_f (and consequently of $\nu(D_f)$) and t_j is the corresponding argument in the call $f(t_1 \dots t_n)$.[7]

We know that $\theta(\nu(D_f))$ is well-founded because D_f is well-founded and $\theta(\nu(D_f))$ obviously is in the scope of ψ because $rV(\theta(\nu(D_f))) \subset y^*$ with the definition of ν and θ. This

[7] Here we treat selectors *sel* and equality = like algorithms, i.e. if *sel*(x) or x=t (where $x \in \mathcal{V}_s$) is "called" in a statement ψ, then the r-description $\nu(D_s)$ is suggested for ψ, i.e. *structural induction* is used.

justifies the soundness the approach. But using $\theta(\nu(D_f))$ also is advantageous in terms of provability, because the induction hypotheses obtained are built exactly from the arguments of the recursive calls in f. Consider, for instance, the algorithm *function* half(x:number):number $\Leftarrow$ *if* (x−1)=0 *then* 0; *if* (x−1)≠0 *then* 1+half((x−1)−1) and the statement $\forall$z:number $\psi[z]$ = $\forall$z:number quotient(z 2) = half(z). The induction heuristic suggests $\theta(\nu(D_{half})) = \{(z-1\neq 0, \{\{z/(z-1)-1\}\})\}$ yielding the induction step $\forall$z:number z−1≠0 $\wedge$ ψ[z/(z−1)−1] $\rightarrow$ ψ[z]. It also suggests $\theta(\nu(D_{quotient'})) = \{(z \geq 2 \wedge 2\neq 0, \{\{z/z-2\}\})\}$ yielding the induction step $\forall$z:number z ≥ 2 $\wedge$ 2≠0 $\wedge$ ψ[z/z−2] $\rightarrow$ ψ[z]. Here the r-description $\theta(\nu(D_{quotient'}))$ is computed from the r-description $D_{quotient'} = \{(x\geq y \wedge y\neq 0, \{\{x/x-y\}\})\}$ for $xy\in \mathcal{V}_{number,number}$ obtained from $D_{quotient}$ (cf. Section 4) by a *domain generalization*, cf. Section 6. Note that no induction axiom can be obtained from $D_{quotient}$ directly since y is an induction variable of $D_{quotient}$ and consequently $D_{quotient}$ is not effective for quotient(z 2).

Our *induction heuristic* is directly derived from the *induction schemas suggested by recursive functions* as defined in [Boyer and Moore, 1979], where however the treatment of variables for computing the induction axiom is made explicit by using *renamed* and *instantiated variants*, but the treatment of *unchangeable variables* is omitted here.

6 Domain Generalization

Unfortunately, the approach of selecting relation descriptions by the induction heuristic only is successful for simple cases so far. Quite often the computation order $\rightarrow_f$ of some algorithm f is not large enough to carry the induction, and then the induction heuristic fails to suggest an r-description. For instance, $D_{quotient}$ was not suggested in the example at the end of the last section for precisely this reason. So if we extend $\rightarrow_f$ by relating some more constructor ground terms, we will obtain additional induction hypotheses. Then an r-description will be suggested because an induction step now becomes provable. Technically this means that we must eliminate some of the induction variables y of an r-description D.

Definition 6.1 An r-description D' for y* is called a *domain generalization* of an r-description D for x* upon the induction variable y of D iff D' is obtained by removing each substitution pair y/t from each domain substitution in each atomic r-description of D. ■

The computation of a *domain generalization* resembles the computation of a *measured subset*, cf. [Boyer and Moore, 1979]. Obviously $iV(D')\subset iV(D)$, therefore D' is suggested whenever D is. Also obviously $iV(D')\neq iV(D)$, therefore D' is *more often* suggested than D, viz. when some non-variable term corresponding to y is used in a call, and y is an induction variable in D, but is a non-induction variable in D'. E.g., $D_{quotient}$ was not suggested in the example of the last section, whereas its domain generalization $D_{quotient'}$ was.

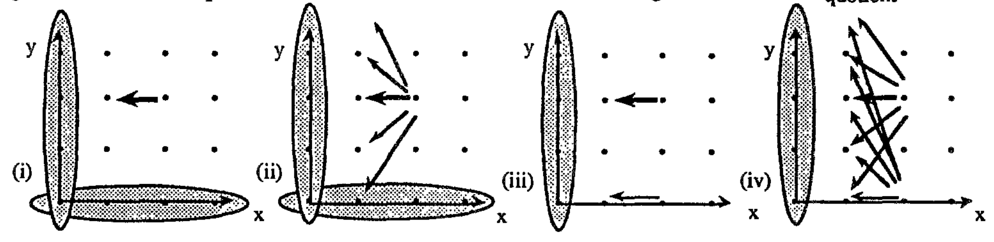

Figure 3

Moreover, the step formulas computed with D' are "more likely to prove" compared to those computed with D because y (after renaming) appears *universally quantified* in the induction hypotheses, cf. Section 3. Consequently, an induction theorem proving system should compute as much domain generalizations as possible in order to obtain $iV(D')$ as small as possible.[8] Figure 3 illustrates the effect of domain generalization for $D = \{(x\neq 0 \wedge y\neq 0, \{\{x/x-1, y/y\}\})\}$, cf. Figure 3(i), yielding $D' = \{(x\neq 0 \wedge y\neq 0, \{\{x/x-1\}\})\}$, cf. Figure 3(ii). The unshaded areas coincide in both diagrams, i.e. the ranges of both relations are the same, but D' relates *more* pairs than D, i.e. an arrow in Figure 3(i) also can be found in Figure 3(ii) (but not vice versa), because the domain of $\rightarrow_D$ is extended.[9]

7 Comparing Induction Axioms

Having generalized the relation descriptions D_f such that the induction heuristic suggests them whenever reasonable, we are now faced with the converse problem: Which relation description shall we actually use when many of them are suggested? Suppose, for instance, the r-descriptions $D_g = \{(x\neq 0, \{\{x/x-1\}\})\}$ and $D_h = \{(x\neq 0 \wedge y=0, \{\{x/x-1\}\}), (x\neq 0 \wedge y\neq 0, \{\{x/x-1\}, \{x/x, y/y-1\}\})\}$ are suggested by the induction heuristic for proving a statement ψ by induction. Viewed semantically, we find $\rightarrow_g \subset \rightarrow_h$ if we consider both D_g and D_h as r-descriptions for xy, cf. Fig. 4(i, ii), and we therefore should use D_h to compute the induction formulas: Since $\rightarrow_g \subset \rightarrow_h$, the induction hypotheses provided by D_g also are provided by D_h, and so D_h must work whenever D_g does (perhaps providing *more* induction hypotheses than necessary). But it may happen that *all* induction hypotheses provided by D_h are needed for a proof of ψ, so that an induction proof using D_g must fail by lack of the induction hypotheses needed. Consequently we always are on the safe side when D_h is used.

So the problem generally is to find out whether $\rightarrow_{D_1} \subset \rightarrow_{D_2}$ holds for a pair of r-descriptions D_1 for $x^* \in \mathcal{V}_u$ and D_2 for $y^* \in \mathcal{V}_v$, where however both are considered as r-descriptions for some $z^* \in \mathcal{V}_w$. We need to consider both D_1 and D_2 as r-descriptions for a *common* $|w|$-tuple z^* of variables, because relations only can be compared by set-theoretic inclusion if they are defined on *identical* spaces. The variable $|w|$-tuple z^* is obtained by inserting variables into x^* or into y^*, which are *irrelevant* for one of the r-descriptions D_1 and D_2. For instance, D_g is an r-description for $x \in \mathcal{V}_{\text{number}}$ but it also can be considered as an r-description for $xy \in \mathcal{V}_{\text{number,number}}$, where y is an irrelevant variable of D_g. But now $\rightarrow_g$ can be compared with $\rightarrow_h$, because D_h also is an r-description for $xy \in \mathcal{V}_{\text{number,number}}$. Note that without using the irrelevant variable y for D_g, we cannot compare $\rightarrow_g$ with $\rightarrow_h$, because $\rightarrow_g$ then is defined on $\mathcal{T}(\Sigma^c)_{\text{number}}$ whereas $\rightarrow_h$ is defined on $\mathcal{T}(\Sigma^c)_{\text{number,number}}$.

To get rid of the necessity to say for which cartesian product $\mathcal{T}(\Sigma^c)_w$ the relation $\rightarrow_D$ of an r-description D at the moment is considered for, we shall compare the relations defined by r-descriptions by so-called *containment* $\sqsubset$ (instead of inclusion $\subset$).

Definition 7.1 A relation $\rightarrow_{D_1}$ of $\mathcal{T}(\Sigma^c)_u$ defined by an r-description D_1 for $x^* \in \mathcal{V}_u$ is *contained* in a relation $\rightarrow_{D_2}$ of $\mathcal{T}(\Sigma^c)_v$ defined by an r-description D_2 for $y^* \in \mathcal{V}_v$, abbreviated $\rightarrow_{D_1} \sqsubset \rightarrow_{D_2}$, iff $\rightarrow_{D_1'} \subset \rightarrow_{D_2'}$, where $\rightarrow_{D_1'}$ and $\rightarrow_{D_2'}$ are the relations of $\mathcal{T}(\Sigma^c)_w$ defined by D_1 and D_2 considered as r-descriptions for $z^* \in \mathcal{V}_w$. Here z^* is some $|w|$-tuple of *distinct* variables such that each member of x^* and y^* appears in z^*. ■

For instance, we find $\rightarrow_g \sqsubset \rightarrow_h$ for the example above. Obviously, $\subset$ entails $\sqsubset$, and $\sqsubset$ is a *reflexive partial order* of relations defined on any cartesian products $\mathcal{T}(\Sigma^c)_w$.

Since the relations $\rightarrow_{D_1}$ and $\rightarrow_{D_2}$ to be compared by containment $\sqsubset$ only are *implicitly* given with the r-descriptions D_1 and D_2, we test for containment by inspecting D_1 and D_2 and formulate a so-called *containment requirement* for D_1 and D_2, which provides a sufficient syntactical criterion for $\rightarrow_{D_1} \sqsubset \rightarrow_{D_2}$ to hold:

[8] However, the well-foundedness of each domain generalization D' has to be verified before it is used.

[9] Since $\rightarrow_D$ is a relation in the sense of $<$, i.e. $a < b$ iff $b \rightarrow_D a$, the *domain* of $\rightarrow_D$ is defined as the set of all elements to the right of $\rightarrow_D$ and the *range* of $\rightarrow_D$ then is the set of all elements to the left of $\rightarrow_D$.

Definition 7.2 An r-description D_1 for $x^* \in \mathcal{V}_u$ is *contained* in an r-description D_2 for $y^* \in \mathcal{V}_v$, in symbols $D_1 \sqsubset D_2$, iff for all $(\varphi, \Delta) \in D_1$ and all $\delta \in \Delta$
(*) $[\forall x^*{:}u\ \forall y^*{:}v\ \ \varphi \rightarrow \bigvee_{(\psi,\Theta) \in D_2} (\psi \wedge \bigvee_{\theta \in \Theta} (\bigwedge_{y \in Dom(\theta)} \delta(y) = \theta(y)))] \in Th_{Ind}$.[10] ■

To verify $D_1 \sqsubset D_2$ means that the so-called *containment formula* (*) has to be verified (perhaps by induction) for all $(\varphi, \Delta) \in D_1$ and all $\delta \in \Delta$, and $D_1 \sqsubset D_2$ entails $\rightarrow_{D_1} \sqsubseteq \rightarrow_{D_2}$ as it can be shown, cf. [Walther, 1991a].[11] Given, for instance, the above r-descriptions D_g and D_h, we test for $D_g \sqsubset D_h$ by proving the trivial containment formula
$\forall x,y{:}number\ x \neq 0 \rightarrow x \neq 0 \wedge y = 0 \wedge x-1 = x-1 \vee x \neq 0 \wedge y \neq 0 \wedge (x-1 = x-1 \vee x-1 = x \wedge \omega = y-1)$.

Figure 4 illustrates the example. The containment formula expresses that (a) the range of the contained relation, as e.g. $\rightarrow_g$ in Figure 4(i), is a subset of the range of the containing relation, as e.g. $\rightarrow_h$ in Figure 4(ii) (so all pairs in the dark shaded area of 4(ii) also are in the dark shaded area of 4(i)), and that (b) the domain of the contained relation is a subset of the domain of the containing relation (so each pair related by $\rightarrow_g$ also is related by $\rightarrow_h$ and each arrow in Figure 4(i) also can be found in Figure 4(ii)). Note that the arrows in the light shaded area of 4(ii) stem from the first atomic r-description of D_h, where the other atomic r-description provides the boldfaced arrow with {x/x, y/y–1} and the remaining arrows with {x/x–1}). Therefore, not a single of the atomic r-descriptions of D_h relates *all* pairs related by the atomic r-description of D_g.

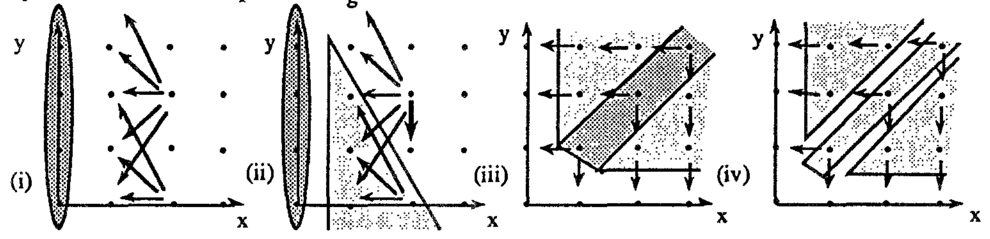

Figure 4

In conclusion, we have defined a proof-technical requirement, viz. $D_1 \sqsubset D_2$, which is sufficient for $\rightarrow_{D_1} \sqsubseteq \rightarrow_{D_2}$ to hold, thus providing in turn a proof-technical requirement to verify when an induction axiom *safely* can be rejected in favour of another one. Hence, we demand that all r-descriptions which are suggested by the induction heuristic be compared with $\sqsubset$ (using the full power of an induction theorem prover to verify the containment formulas) and we demand that those which are contained in another one are disregarded.[12]

8 Range Generalization

Comparing relation descriptions by containment is a powerful tool to reject r-descriptions in favour of other ones *without loosing provability* of the induction formulas. So, it seems worthwhile to have relations descriptions D with $\rightarrow_D$ as large as possible wrt. the $\sqsubseteq$-order. It may happen for relation descriptions D_1 and D_2 that neither $\rightarrow_{D_1} \sqsubseteq \rightarrow_{D_2}$ nor $\rightarrow_{D_2} \sqsubseteq \rightarrow_{D_1}$, but D_1 and D_2 can be generalized yielding D_1' and D_2' such that $\underline{\rightarrow_{D_1} \sqsubseteq \rightarrow_{D_1'}, \rightarrow_{D_2} \sqsubseteq \rightarrow_{D_2'}}$ *and* $\rightarrow_{D_1'} \sqsubseteq \rightarrow_{D_2'}$. In this case, all relation descriptions except D_2'

[10] The symbols $\bigwedge$ and $\bigvee$ denote the universal and the existential quantification over a *finite* domain and therefore can be thought of as an abbreviation of a conjunction or a disjunction respectively.
[11] The falsification of containment formulas (when non-containing relations are compared) usually is trivial, at least if a facility looking for counterexamples for conjectures is applied, cf. [Protzen, 1992].
[12] Note that $D \sqsubset D'$ for each domain generalization D' of an r-description D. Consequently each r-description is disregarded a priori by the induction heuristic in favour of its domain generalization.

can be ignored for computing an induction axiom. Figure 2(ii) illustrates the case.

Since we demand that *all* domain generalizations D_f' of the r-description D_f of an algorithm f are computed, we have already provided for a mechanism supporting the desired effect. But domain generalization does not cover all cases of concern. It may happen that the containment test simply fails because none of the ranges of the compared relations is a subset of the other one. So, an obvious remedy is to extend the range of a relation as much as possible, in order to eventually cover the range of some other relation when the containment test is performed. Technically, this is achieved by replacing a range formula in an atomic r-description by some weaker formula.

Definition 8.1 An r-description D' for $x^* \in \mathcal{V}_w$ is called a ***range generalization*** of an r-description D for x^* iff D' is obtained from D by replacing some atomic r-description (φ, Δ) in D with an atomic r-description (φ', Δ), such that $[\forall x^*{:}w\ \varphi \rightarrow \varphi'] \in Th_{Ind}$. ■

Since we have $\rightarrow_D \sqsubseteq \rightarrow_{D'}$ for each range generalization D' of some r-description D, an induction upon $\rightarrow_{D'}$ always is successful, whenever an induction upon $\rightarrow_D$ is.[13] Figure 3 illustrates the effect of a range generalization for $D = \{(x\neq0 \wedge y\neq0, \{\{x/x-1, y/y\}\})\}$, cf. Figure 3(i), yielding $D' = \{(x\neq0, \{\{x/x-1, y/y\}\})\}$, cf. Figure 3(iii). The unshaded area in Figure 3(i) is also unshaded in Figure 3(iii), i.e. the range of $\rightarrow_D$ is a subset of the range of $\rightarrow_{D'}$, and therefore $\rightarrow_{D'}$ relates *more* pairs than $\rightarrow_D$ relates. Therefore range generalization also increases the provability of induction formulas, because base cases obtained by D, as e.g. $x\neq0 \wedge y=0$, may become step cases when the D' is used. Fig. 3(iv) illustrates the combined effect of domain and range generalization applied to D yielding $D'' = \{(x\neq0,\{\{x/x-1\}\})\}$. The computation of a range generalization corresponds to *weeding out irrelevant tests* yielding a *revised machine* as defined in [Boyer and Moore,1979].[14] An r-description obtained from the original algorithm by domain *and* range generalization corresponds to an *induction template*.

9 *Separated Relation Descriptions*

Unfortunately, range generalization may also support the computation of *unprovable* step formulas. This is because a range generalization may drop conditions from the range formulas which are relevant for providing the "right" induction hypotheses. Then we may attempt to prove an induction conclusion not covered by the induction hypotheses of the induction formula under consideration. Consider, for instance, the r-description $D = \{(y\neq0 \wedge x\geq y, \{\{x/x, y/y-1\}\}), (x\neq0 \wedge x\leq y, \{\{x/x-1, y/y\}\})\}$. It may happen, that the step case $\forall x,y.\ y\neq0 \wedge x\geq y \wedge \psi[x/x, y/y-1] \rightarrow \psi[x,y]$ cannot be proved, because $\psi[x/x-1, y/y]$ is also required as an induction hypothesis for the subcase $x=y\neq0$, or the proof of the other step case fails, because the induction hypothesis $\psi[x/x, y/y-1]$ is not provided there. But obviously this effect cannot occur when the relations defined by the atomic r-descriptions of D are *range disjoint*, i.e. if the range formulas of the atomic r-descriptions exclude each other:

Definition 9.1 A relation description $D = \{C_1,\ldots,C_k\}$ for x^* with $C_i = (\varphi_i, \Delta_i)$ is called *range disjoint* iff $[\forall x^* \neg(\varphi_i \wedge \varphi_j)] \in Th_{Ind}$ for all φ_i, φ_j with $i\neq j$. The *separation* D^* of D is the smallest r-description for x^* such that $(\varphi'_1 \wedge\ldots\wedge \varphi'_k, \Delta'_1 \cup\ldots\cup \Delta'_k) \in D^*$, where $\varphi'_i = \varphi_i$ and $\Delta_i' = \Delta_i$ or else $\varphi'_i = \neg\varphi_i$ and $\Delta_i' = \varnothing$ for all $i \in \{1,\ldots,k\}$. ■

[13] Note that $D \sqsubseteq D'$ for each range generalization D' of an r-description D. Consequently each r-description is disregarded a priori by the induction heuristic in favour of its range generalization.

[14] Additional support is required to *find* range generalizations, cf. the usage of user provided "induction lemmata" in [Boyer and Moore, 1979] or an automated synthetization of weaker range formulas using "difference algorithms" in [Walther, 1988]. Also the well-foundedness of each range generalization D' has to be verified before it is used.

It can be shown that each separation D^* of an r-description D yields a *range disjoint* r-description such that $\rightarrow_{D^*} = \rightarrow_D$, cf. [Walther, 1991a], and consequently range disjointness defines a *normal form* for r-descriptions. Figure 4 illustrates the effect of separating the above r-description D, cf. Fig. 4(iii), yielding $D^* = \{ (y\neq0 \wedge x>y, \{\{x/x, y/y-1\}\}), (x=y\neq0, \{\{x/x, y/y-1\}, \{x/x-1, y/y\}\}), (x\neq0 \wedge x<y, \{\{x/x-1, y/y\}\}) \}$, cf. Figure 4(iv). Both light shaded areas in 4(iii) indicate the ranges of both atomic r-descriptions of D and the dark shaded area shows where they intersect. The three light shaded areas in Fig. 4(iv) indicate the ranges of the atomic r-descriptions of D^*. The computation of a *separation* resembles the notion of *superimposing the machine* as defined in [Boyer and Moore, 1979].

10 An Analysis of the Boyer-Moore Heuristics

Our work originally was motivated by an attempt to understand the semantics of the developments presented in [Boyer and Moore, 1979] and implemented in the NQTHM system. In this system, *induction schemas* are compared by *subsumption*, and those schemas which are *subsumed* are disregarded. The remaining schemas are *merged* and, among the resulting schemas, those which are *flawed* are disregarded.[15]

Subsumption and merging are the main heuristics of the NQTHM for manipulating induction schemas and we give a brief analysis of both heuristics using our notational framework: Subsumption corresponds in its role to our *containment test* for r-descriptions. However, the semantics of both techniques differ significantly, and we went astray for a while when investigating the semantics of subsumption. The reason for our confusion was that subsumption treats *two* completely different phenomena by *one* operation: (1) containment of relations and (2) comparing transitive closures of relations. We feel therefore, that subsumption provides a "heuristic" comparison which lacks a semantic justification, whereas the containment test is a non-heuristic completeness preserving reduction rule. Restated in our framework, a relation description D_1 is subsumed by an r-description D_2 iff for all $(\varphi,\Delta)\in D_1$ and all $\delta\in\Delta$ some $(\psi,\Theta)\in D_2$ and some $\theta\in\Theta$ exists such that (1) $\models [\forall z^*{:}w\ \psi\rightarrow\varphi]$ and (2) for all $\{x/t\}\in\delta$ some $\{x/t'\}\in\theta$ exists such that t is a subterm of t'. If we replace "$\psi \rightarrow \varphi$" with "$\varphi \rightarrow \psi$" in the above requirement (1) and also demand t=t' in requirement (2), then this modification yields a proper subcase of our containment test. But containment uses the full power of an induction prover whereas subsumption only checks for substitution pairs and that φ (written as a set of literals) is a subset of the literals in ψ, thus trivializing the implication test. Therefore, subsumption fails, for instance, to select between both r-descriptions suggested for $\forall$ z:number quotient(z 2)=half(z), whereas containment disregards one of them after verifying $[\forall z{:}\text{number}\ z\geq2 \wedge 2\neq0 \rightarrow z-1\neq0 \wedge (z-2)=(z-1)-1]\in Th_{Ind}$.

Nevertheless, with the original definition subsumption may also disregard the *wrong* r-description, whereas containment *always* selects the right one. Assume, for instance, the r-descriptions $D_1 = \{(x\neq0,\{\{x/x-1\}\})\}$ and $D_2 = \{(x\neq0\wedge P(x), \{\{x/x-1\}\})\}$ are suggested by the induction heuristic. Since (1) $\models [\forall x{:}\text{number}\ x\neq0 \wedge P(x) \rightarrow x\neq0]$ and (2) x–1 is a subterm of x–1, D_1 is subsumed by D_2 and therefore D_2 is used to compute an induction axiom. Computing the same formula and verifying x–1=x–1, containment demands to disregard D_2 in favour of D_1, yielding a stronger induction than with D_2. So one may wonder why subsumption is so successful as claimed by the authors, despite the fact that antecedent and succedent are exchanged in the above implication. The answer is quite simple: in many cases φ and ψ are *identical* (or *equivalent* at least) and the direction of the implication sign does not matter then. But why then is implication not demanded in the right direction? Here, the

[15] We provide no notion corresponding to *flawed* schemas, and we leave an investigation of their role in our framework as a future work.

answer is that another phenomenon completely different from containment, viz. coping with *nested recursions*, also is treated by subsumption, see [Stevens, 1989; Walther, 1991a].

The *merge* of induction schemas performed in the NQTHM system corresponds in its role to our notion of a *separated union*, cf. [Walther, 1991b]. However, the semantics of both techniques differ significantly. Merging essentially tests, whether the intersection of relations (implicitly given by the induction schemas) is not empty, and if so, the *intersection* is computed as the result of the merge, whereas we test for *commutation* of relations, and, if successful, the *union* of relations is computed, cf. [Walther, 1991b]. However, using the intersection of relations induction hypotheses not provided by *both* inductions get lost. So one may wonder why merging is so successful as claimed by the inventors, despite the fact that it eliminates induction hypotheses provided by the constituting relations. And more confusingly, one may wonder what *sense* it makes to use the merged relation $\rightarrow_1 \cap \rightarrow_2$, since with both $\rightarrow_1$ and $\rightarrow_2$ containing $\rightarrow_1 \cap \rightarrow_2$, an induction upon $\rightarrow_1$ and also upon $\rightarrow_2$ *must be* successful, whenever an induction proof using $\rightarrow_1 \cap \rightarrow_2$ is!

The answer is surprisingly simple: merging might be designed to cope with an inherent weakness of the Boyer-Moore logic (and here it actually makes a sense). This logic does not allow universal quantifiers in the induction hypotheses (as opposed to e.g. the systems described in [Aubin, 1979; Biundo et. al, 1986]). Therefore the instantiations for the universally quantified variables computed during a proof of an induction formula have to be computed by the NQTHM system *prior* to this proof. Obviously, the right instantiations only can be found by *proving* the induction formulas. So some heuristic support is necessary to compensate for the absence of the relevant information and this seems to be the role of the merging heuristic. The standard example used in [Boyer and Moore, 1979] to motivate merging is the transitivity statement for <, i.e. $\forall x,y,z{:}\text{number } x<y \wedge y<z \rightarrow x<z$. The induction heuristic suggests $D_1 = \{(x \neq 0, \{\{x/x-1\}\})\}$, $D_2 = \{(y \neq 0, \{\{y/y-1\}\})\}$ and $D_3 = \{(z \neq 0, \{\{z/z-1\}\})\}$, which are merged yielding $D = \{(x \neq 0 \wedge y \neq 0 \wedge z \neq 0, \{\{x/x-1, y/y-1, z/z-1\}\})\}$. The induction formulas derived from D are easily proven. Since the containment tests must fail for all combinations of the suggested r-descriptions, we are left with all three suggestions when working with our proposal. So we arbitrarily choose one of them. Regardless whether we decide to induce according to D_1, D_2 or to D_3, any induction formula obtained is easily proven by the presence of the universal quantifiers in the induction hypotheses, which are instantiated with exactly the same terms as provided by the merged schema.

11 References

Aubin, R. *Mechanizing Structural Induction* . Ph.D. thesis, Univ. of Edinburgh, 1976.
Aubin, R. *Mechanizing Structural Induction* . Theoretical Computer Science, vol 9, 1979.
Biundo, S., Hummel, B., Hutter, D. and Walther, C. *The Karlsruhe Induction Theorem Proving System*. Proc. 8th CADE, Lecture Notes Comp. Sc., vol. 230, 1986.
Boyer, R.S. and J S. Moore *Proving Theorems about LISP functions* . J.ACM 22, 1975.
Boyer, R.S. and J S. Moore *A Computational Logic* . Academic Press, 1979.
Boyer, R.S. and J S. Moore *A Computational Logic Handbook* . Academic Press, 1988.
Bundy, A., et. al. *A Rational Reconstruction and Extension of Recursion Analysis* . Proc. Intern. Joint. Conf. on Artif. Intelligence., Detroit, 1989.
Burstall, R.M. *Proving Properties of Programs by Structural Induction* . Comput. J. 12(1), 1969.
Kapur, D. and Musser, D.R. *Proof by Consistency*. Artificial. Intelligence, vol 31, no 2, 1987.
Protzen, M. *Disproving Conjectures* . Proc. 11th CADE, Saratoga Springs, 1992
Reddy, U. S. *Term Rewriting Induction* . Proc. 10th CADE, Kaiserslautern, 1990.
Stevens, A. *An improved method for the mechanization of inductive proof* . Ph.D Thesis, University of Edinburgh, 1989.
Walther, C. *Argument-Bounded Algorithms as a Basis for Automated Termination Proofs* . Proceedings 9th CADE, Lecture Notes Comp. Sc., vol. 310, 1988.
Walther, C. *Computing Induction Axioms - Methods and Formal Bases* . Research Memo, Fachbereich Informatik, Technische Hochschule Darmstadt, 1991a.
Walther, C. *Combining Induction Axioms by Machine* . Research Memo, Fachbereich Informatik, Technische Hochschule Darmstadt, 1991b.