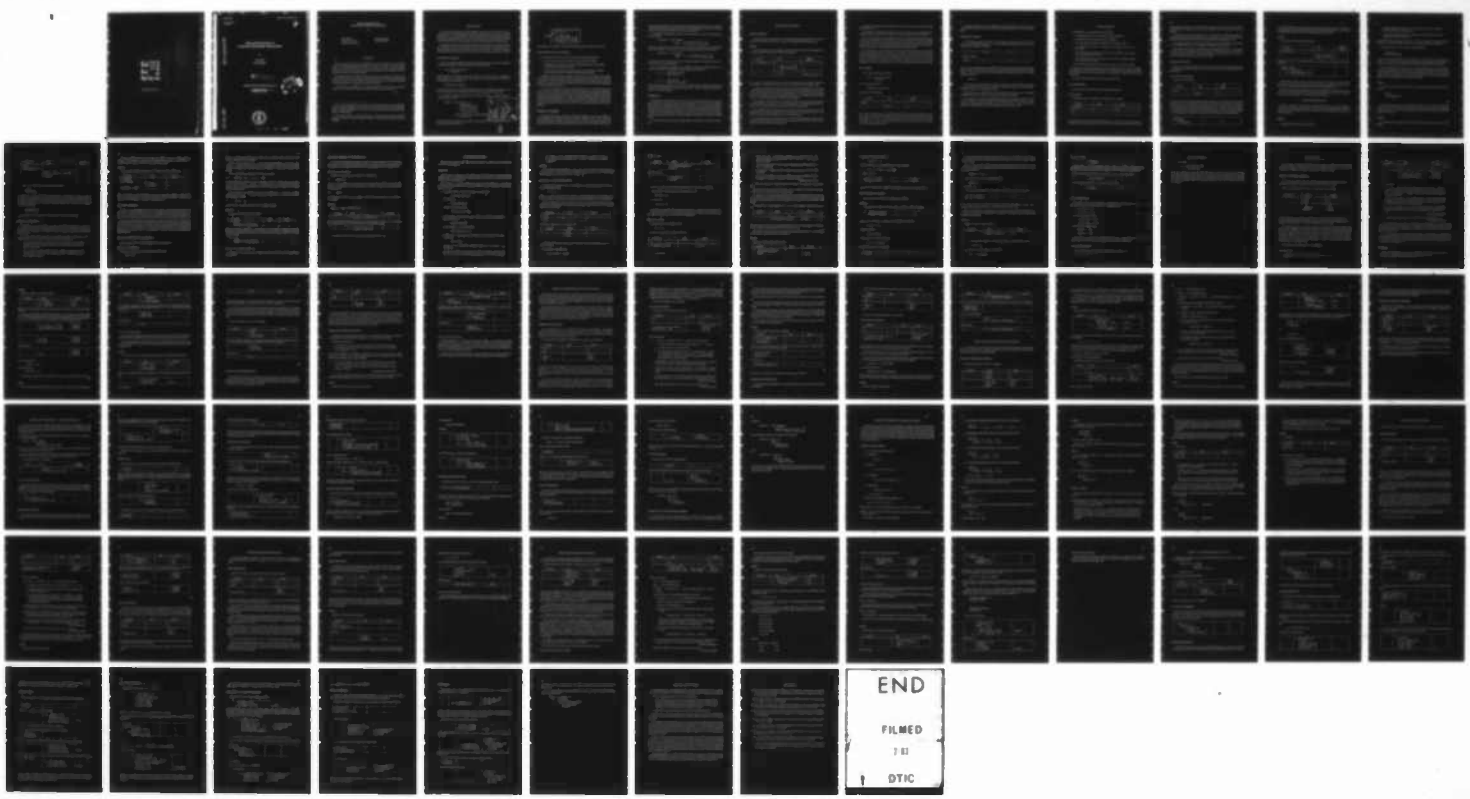
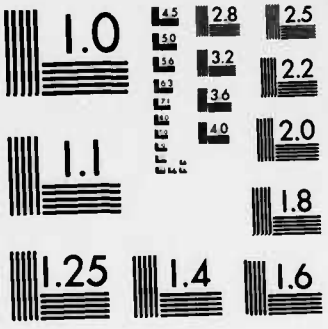
AD-A123 268 SPECIAL RELATIONS IN PROGRAM-SYNTHETIC DEDUCTION(U) 1/1
STANFORD UNIV CA DEPT OF COMPUTER SCIENCE
Z MANNA ET AL. MAR 82 STAN-CS-82-902 N00014-76-C-0687
UNCLASSIFIED F/G 9/2 NL





March 1982

Report. No. STAN-CS-82-902

Also numbered:
AIM-345

Special Relations in Program-Synthetic Deduction

by

Zohar Manna
Richard Waldinger

Department of Computer Science

Stanford University
Stanford, CA 94305

SPECIAL RELATIONS IN PROGRAM-SYNTHETIC DEDUCTION

by

Zohar Manna
Stanford University
and Weizmann Institute

Richard Waldinger
SRI International

ABSTRACT

Program synthesis is the automated derivation of a computer program from a given specification. In the *deductive approach*, the synthesis of a program is regarded as a theorem-proving problem; the desired program is constructed as a by-product of the proof. This paper presents a formal deduction system for program synthesis, with special features for handling equality, the equivalence connective, and ordering relations.

In proving theorems involving the equivalence connective, it is awkward to remove all the quantifiers before attempting the proof. The system therefore deals with *partially skolemized sentences*, in which some of the quantifiers may be left in place. A rule is provided for removing individual quantifiers when required after the proof is under way.

The system is also *nonclausal*; i.e., the theorem does not need to be put into conjunctive normal form. The equivalence, implication, and other connectives may be left intact.

The research was supported in part by the National Science Foundation under Grants MCS-78-02591 and MCS-79-09495, in part by the Office of Naval Research under Contracts N00014-75-C-0816 and N00014-76-C-0687, and in part by the Air Force Office of Scientific Research under Contract AFOSR-81-0014.

The authors' addresses: Z. Manna, Department of Computer Science, Stanford University, Stanford, CA 94305; R. Waldinger, Artificial Intelligence Center, SRI International, Menlo Park, CA 94025.

INTRODUCTION

One of the earliest techniques for program synthesis, the automated construction of a computer program, has been the *deductive approach*, in which the program is developed by proving a theorem corresponding to the given specification. While program synthesis does not typically require the proof of deep mathematical theorems, it does need deductive systems specially designed to handle constructs commonly occurring in specifications, such as equality, equivalence, and orderings.

In this paper, we present a formal system with facilities for dealing with the equality predicate ($=$), the logical equivalence connective ($\equiv$), and the ordering relations. The system allows us to defer skolemization, the removal of quantifiers, when it is inconvenient. The system is machine-oriented and intended for implementation in interactive and automatic program synthesis systems.

The Deductive Approach

In Manna and Waldinger [1980] we presented a deductive system for the synthesis of applicative (side-effect-free) programs. The paper considered specifications of form

$$f(x) \Leftarrow \text{find } z \text{ such that } r(x, z) \text{ where } p(x).$$

In other words, for an arbitrary input x, the program f is to yield an output z satisfying an *output condition* $r(x, z)$, provided that the input satisfies the *input condition* $p(x)$. The theorem corresponding to the specification is

$$(\forall x)[\mathit{if}\ p(x)\ \mathit{then}\ (\exists z) r(x, z)].$$

The proof is restricted to be sufficiently constructive so that, in establishing the existence of an output z satisfying the required relationship, it tells us how to compute such an output.

For example, to specify a program to find the quotient of dividing a nonnegative integer i by a positive integer j, we write

$$\begin{aligned} \mathit{quot}(i, j) \Leftarrow\ & \text{find } z \text{ such that} \\ & \mathit{isinteger}(z)\ \mathit{and} \\ & (\exists y) \begin{bmatrix} \mathit{isinteger}(y)\ \mathit{and} \\ i = z \cdot j + y\ \mathit{and} \\ 0 \le y\ \mathit{and}\ y < j \end{bmatrix} \\ & \text{where } \mathit{isinteger}(i)\ \mathit{and}\ \mathit{isinteger}(j)\ \mathit{and} \\ & i \ge 0\ \mathit{and}\ j > 0. \end{aligned}$$

Here the predicate $\mathit{isinteger}(u)$ is a type predicate expressing that u is an integer. The theorem corresponding to this specification is

$$(\forall i)(\forall j)\left[\begin{array}{l}\textit{if } \textit{isinteger}(i) \textit{ and } \textit{isinteger}(j) \textit{ and} \\ \quad i \geq 0 \textit{ and } j > 0 \\ \textit{then} \\ \quad (\exists z)\left\{\begin{array}{l}\textit{isinteger}(z) \textit{ and} \\ (\exists y)\left[\begin{array}{l}\textit{isinteger}(y) \textit{ and} \\ i = z \cdot j + y \textit{ and} \\ 0 \leq y \textit{ and } y < j\end{array}\right]\end{array}\right.\end{array}\right]$$

(For simplicity, we shall omit the type predicates when the context makes the type clear.)

Design Criteria for a Formal System

A formal system to prove such theorems must have the following capabilities:

- It must prove theorems with both universal and existential quantifiers.
- It must be able to handle theories with mathematical induction, such as nonnegative integers, finite sets, lists, and trees.
- It must be facile in handling the equality predicate, the equivalence connective, and the ordering relations; these appear frequently in specifications.

In addition, we want the proofs to appear natural to people. The advantage of such a quality for an interactive system is self-evident. For an automatic system, our hope is that a natural form will enable us to exploit the heuristics of human intuition. On the other hand, we also want the system to be machine-oriented, in the sense that there should be only a small number of legal next steps to choose from at each stage.

It has long been observed that systems requiring the theorem to be converted into clause form can cause it to explode and lose intuitive content. Such systems are particularly awkward for proving theorems by mathematical induction, because, if the induction hypothesis is propositionally complex, it may be dispersed over several clauses. This makes it difficult to recognize when we have succeeded in reducing the theorem to an instance of the induction hypothesis, since the theorem and the induction hypothesis will be syntactically dissimilar. A *nonclausal* system, which does not require us to transform the theorem to clause form, is thus particularly appropriate for program synthesis.

Equivalence and Equality

Our earlier deductive system (Manna and Waldinger [1980]) and that of Murray [1982], are both nonclausal and are suitable candidates for program synthesis. However, neither system has any special provisions for handling equality, equivalence, or orderings. The equality predicate is of obvious importance in expressing the specifications of programs. Ordering relations not only occur

frequently in specifications, but are also used in the "well-founded induction principle" we employ. The equivalence connective is of special importance in dealing with specifications expressed in terms of the set constructor $\{x : p(x)\}$ ("the set of all x such that $p(x)$").

For example, we might specify a program to find the Cartesian product of two finite sets s_1 and s_2 as follows:

$$cart(s_1, s_2) \Leftarrow \text{find } z \text{ such that } z = \left\{ y : \quad (\exists x_1)(\exists x_2) \begin{bmatrix} y = \langle x_1, x_2 \rangle \ \textit{and} \\ x_1 \in s_1 \ \textit{and} \ x_2 \in s_2 \end{bmatrix} \right\}$$

(Here $\langle x_1, x_2 \rangle$ denotes the pair of elements x_1 and x_2.) Unless the theorem prover deals explicitly with the set constructor, we are likely to rephrase the specification with the circumlocution:

$$cart(s_2, s_2) \Leftarrow \text{find } z \text{ such that } (\forall y) \left\{ y \in z \ \equiv \ (\exists x_1)(\exists x_2) \begin{bmatrix} y = \langle x_1, x_2 \rangle \ \textit{and} \\ x_1 \in s_1 \ \textit{and} \ x_2 \in s_2 \end{bmatrix} \right\}$$

In fact, even if we have the set constructor in our formal language, we are likely to rephrase it in terms of equivalence during the proof.

Now an equivalence has appeared in our specification and the corresponding theorem. Of course, we can remove it by appealing to such rewriting transformations as

$$\mathcal{F} \equiv \mathcal{G} \Rightarrow \begin{bmatrix} (\textit{if } \mathcal{F} \textit{ then } \mathcal{G}) \ \textit{and} \\ (\textit{if } \mathcal{G} \textit{ then } \mathcal{F}) \end{bmatrix}$$

or

$$\mathcal{F} \equiv \mathcal{G} \Rightarrow \begin{bmatrix} (\mathcal{F} \ \textit{and} \ \mathcal{G}) \ \textit{or} \\ ((\textit{not } \mathcal{F}) \ \textit{and} \ (\textit{not } \mathcal{G})) \end{bmatrix}.$$

But decomposing the connective in this way may needlessly multiply the length of the proof and destroy its intuitive content. Instead, we present deduction rules for dealing with equivalence explicitly in a nonclausal setting.

Skolemization

Traditionally, all the quantifiers of a theorem are removed by skolemization before the proof begins. However, if the theorem contains an explicit equivalence, we cannot remove any quantifiers in its scope without removing the equivalence first, as we shall see. Our earlier system and that of Murray deal only with fully skolemized sentences, from which the equivalences have been removed. The rules we present here, on the other hand, can be applied to partially skolemized sentences, in which some of the quantifiers and equivalences may remain intact. We also present rules for removing quantifiers one at a time, as it becomes expedient, at any point in the theorem-proving process.

Our treatment here will be informal; we shall justify only some of the rules, and in an intuitive way.

THE DEDUCTIVE APPROACH

Deductive Tableaus

The basic structure of this approach is the *deductive tableau*, which consists of a set of *rows*; each row contains either an *assertion* or a *goal*, and an optional associated *output entry*.

Example:

The rows below are part of the tableau for the synthesis of the *integer quotient* program; in the actual synthesis, these rows are interspersed with others.

assertions	goals	outputs $quot(i,\ j)$
1. $i \geq 0$ *and* $j > 0$		
	2. $(\exists y)\begin{bmatrix} i = z \cdot j + y \text{ and} \\ 0 \leq y \text{ and } y < j \end{bmatrix}$	z
	3. $i < j$	0
	4. $j \leq i$	$quot(i - j,\ j) + 1$

Here, i and j are constants, and y and z are variables. An instance of a row is obtained by replacing free variables of a row with terms; constants and bound variables cannot be replaced. ∎

The intuitive meaning of the tableau is that if, under any given interpretation, every instance of each of the assertions is true, then some instance of at least one of the goals is true. In this case, we will say that the entire tableau is *valid*. Furthermore, if some instance of one of the goals is true or some instance of one of the assertions is false, then the corresponding instance of the output entry will satisfy the specification for the desired program.

Thus, the goals of the tableau have a tacit disjunction between them, while the assertions have a tacit conjunction. In addition, the free variables of the goals have a tacit existential quantification, while the free variables of the assertions have a tacit universal quantification.

For example, the second row above has a free variable z, which is also the output entry. This means that if, for a given interpretation, there is some value of z for which goal 2 is true, then that value of z will satisfy the specification for the quotient program.

If an assertion has no output entry, we are not concerned with the output in the case in which the assertion is false. For example, assertions that are axioms will have no output entries. Typically, all the goals will have output entries.

A tableau that contains as a goal the proposition *true*, or as an assertion the proposition *false*, will always be valid.

It is possible to use tableaus that contain more than one output column, corresponding to the synthesis of systems of more than one program, but we shall not discuss this extension here.

Note that the distinction between assertions and goals is artificial and does not increase the logical power of the system. In fact, if we delete an assertion from the tableau and add its negation as a new goal, or delete a goal and add its negation as a new assertion, we obtain an equivalent tableau; this is known as the *duality property*. The distinction between assertions and goals does make proofs easier for people to understand and may have strategic import.

The free variables in a row are dummies; they may be systematically replaced by new variables without changing the meaning of the tableau. For simplicity, we assume that the variables are implicitly *standardized apart*, so that the variables of any row are distinct from those of any other row, and the variable bound by one quantifier is distinct both from that bound by any other quantifier and from any free variable. If, in an example, we happen to write a tableau in which this restriction is violated, we may imagine that the variables are distinguished by invisible subscripts.

How to Begin

If we are given a specification of form

$$f(x) \Leftarrow \text{find } z \text{ such that } r(x,z) \text{ where } p(x),$$

the corresponding theorem is

$$(\forall x)[\mathit{if}\ p(x)\ \mathit{then}\ (\exists z) r(x,z)].$$

We construct an initial tableau

assertions	goals	outputs $f(a)$
$p(a)$		
	$r(a,z)$	z

Here a is a constant, obtained by removing the quantifier $(\forall x)$ through skolemization, and z is a free variable. The meaning of the tableau is that if, under any interpretation, $p(a)$ is true, then some instance of $r(a,z)$ is true, and the corresponding instance of z will satisfy the specification. The output entry is a device for ensuring that the proof will be sufficiently constructive and for extracting a program from the proof.

Typically, in addition to the input condition $p(a)$, the initial assertions of the tableau will include axioms for the theory under consideration (e.g., integers, finite sets, etc.) and the underlying logic.

The Deductive Process

In the deductive system we describe, we apply *deduction rules* that add new rows to the tableau without changing its meaning – i.e., so that an equivalent tableau is produced. The process terminates if we develop the final goal

	true	t

or the final assertion

false		t

where t is a term consisting entirely of symbols from the target programming language. Because the deduction rules preserve meaning, obtaining such a goal or assertion will imply that the original tableau is valid. We are also assured that t will satisfy the program's specification. The final program we obtain is

$$f(a) \Leftarrow t.$$

The restriction on the symbols of t will ensure that the proof will be sufficiently constructive to enable us to compute the output; in particular, t will not be allowed to contain quantifiers, untestable predicates, or uncomputable functions.

We assume that the variables of the new rows added by a deduction rule are implicitly standardized apart in the same way the variables of the original tableau are.

At each stage, there may be several deduction rules that can legally be applied, not all of which are helpful in reaching a final program. Also, different choices of deduction rules may lead to different final programs, some of which may be better than others. In this paper, we largely disregard the strategic aspect of making an opportune choice of deduction rules.

DEDUCTION RULES

The deduction rules are divided into several categories:

- The *splitting rules* break a row down into its logical components.
- The *skolemization rules* enable us to remove quantifiers.
- The *transformation rules* replace subsentences by equivalent sentences.
- The *resolution rules* enable us to perform a case analysis on the truth of a subsentence.
- The *substitution rules* enable us to use equivalences, equalities, or other special relations that appear in the tableau.
- The *matching rules* enable us to introduce new equivalences, equalities, or other special relations into a tableau.
- The *mathematical-induction* rule enables us to introduce an induction hypothesis.

The splitting and mathematical-induction rules are basically the same as in Manna and Waldinger [1980] but are outlined here for completeness. The transformation and resolution rules have been generalized to allow for explicit quantifiers. The skolemization, substitution, and matching rules are new.

We first describe the splitting and mathematical-induction rules.

The Splitting Rules

The splitting rules break rows down into their logical components.

Rule (and-split):

The *and-split* rule may be expressed in a tableau notation as follows:

assertions	goals	outputs
$\mathcal{F}$ *and* $\mathcal{G}$		t
$\mathcal{F}$		t
$\mathcal{G}$		t

This means that if a tableau contains an assertion of form $\mathcal{F}$ *and* $\mathcal{G}$, we may add $\mathcal{F}$ and $\mathcal{G}$ to our tableau as two separate assertions. The output entries for the new assertions are inherited from

the original assertion; if there is no output entry in the original assertion, there is none in the new assertion either. The assertion $\mathcal{F}$ *and* $\mathcal{G}$ need not be the last row in the tableau; it may occur anywhere.

In general, the rows above the double line in a rule are the *given* or *original* rows, which are required to be present in the tableau before the rule is applied; the rows below the double line are the *derived* or *new* rows, which are added to the tableau as a result of applying the rule.

The original assertion is not deleted from the tableau when the rule is applied. Although this may be advisable for efficiency, we are disregarding strategic considerations here.

The *or-split* rule is similar to the *and-split* rule and breaks a goal of form $\mathcal{F}$ *or* $\mathcal{G}$ down into two goals $\mathcal{F}$ and $\mathcal{G}$. The *if-split* rule breaks a goal of form *if* $\mathcal{F}$ *then* $\mathcal{G}$ down into a new assertion $\mathcal{F}$ and a new goal $\mathcal{G}$. There are no rules for breaking down an assertion of form $\mathcal{F}$ *or* $\mathcal{G}$, an assertion of form *if* $\mathcal{F}$ *then* $\mathcal{G}$, or a goal of form $\mathcal{F}$ *and* $\mathcal{G}$.

Mathematical Induction

We present here only the simplest case of the induction rule, in which the induction hypothesis is formed directly from the theorem to be proved, rather than from a subsequent goal or a generalization.

Rule (mathematical induction):

Suppose our initial tableau is

assertions	goals	outputs $f(a)$
$p(a)$		
	$r(a, z)$	z

In other words, we are trying to construct a program to produce, for an arbitrary input a, an output z satisfying the output condition $r(a, z)$, provided that the input a satisfies the input condition $p(a)$. Then we may assume inductively that the program f we are trying to construct will produce, for an arbitrary input u, an output $f(u)$ satisfying the output condition $r(u, f(u))$, provided that u satisfies the input condition $p(u)$ and that u is strictly less than a in some well-founded ordering $\prec_w$. In other words, we may add to our tableau as a new assertion the induction hypothesis

if $u \prec_w a$ *then if* $p(u)$ *then* $r(u, f(u))$		

This induction hypothesis states that the program will work properly on all inputs "smaller" than the arbitrary input under consideration. The particular well-founded ordering $\prec_w$ to be used in the proof is left unspecified; it must be discovered during the proof process.

Example:

The initial tableau for the quotient program is

assertions	goals	outputs $quot(i,j)$
$i \geq 0$ *and* $j > 0$		
	$(\exists y)\begin{bmatrix} i = z \cdot j + y \ \textit{and} \\ 0 \leq y \ \textit{and} \ y < j \end{bmatrix}$	z

By the induction rule, we are justified in adding to our tableau, as a new assertion, the induction hypothesis

if $(u,v) \prec_w (i,j)$ *then if* $u \geq 0$ *and* $v > 0$ *then* $(\exists y)\begin{bmatrix} u = quot(u,v) \cdot v + y \\ \textit{and} \ 0 \leq y \ \textit{and} \ y < v \end{bmatrix}$		

This assertion contains instances of the term $quot(u,v)$, where *quot* is the program being constructed. If this assertion is used in the proof, terms of the form $quot(s,t)$ can appear in the output column, corresponding to recursive calls in the final *quot* program. ∎

This is the simplest case of the induction rule; the more general case, not presented here, allows us to form an induction hypothesis from rows other than the initial rows of the tableau. This more general induction rule enables us to construct auxiliary subprograms.

THE SKOLEMIZATION RULES

Before we can introduce the skolemization rules, we must introduce the notion of "polarity" and the associated concept of "quantifier force." Polarity is also of strategic import in controlling the other rules. Murray [1982] used it in his formulation of nonclausal resolution and it was known to logicians earlier.

Polarity

A subsentence of a given sentence is said to be

- Of *positive polarity* in the sentence if it is within the scope of an even number of (explicit or implicit) *not* connectives, and.

- Of *negative polarity* in the sentence if it is within the scope of an odd number of (explicit or implicit) *not* connectives.

In determining polarity, a subsentence of form *if* $\mathcal{P}$ *then* $\mathcal{Q}$ is regarded as an abbreviation for (*not* $\mathcal{P}$) *or* $\mathcal{Q}$, so that $\mathcal{P}$ is within the scope of one more implicit *not* connective than $\mathcal{Q}$.

A sentence of form $\mathcal{P} \equiv \mathcal{Q}$ is regarded as an abbreviation for

($\mathcal{P}$ *and* $\mathcal{Q}$) *or*
((*not* $\mathcal{P}$) *and* (*not* $\mathcal{Q}$)),

in which the second occurrences of $\mathcal{P}$ and $\mathcal{Q}$ are within the scope of one more *not* connective than the first. As a consequence, $\mathcal{P}$ and $\mathcal{Q}$ have both positive and negative polarities in the sentence. A subsentence is said to be of *strict polarity* if it does not have both polarities in the sentence.

Intuitively speaking, the truth of a sentence is directly related to the truth of its positive subsentences, and the falsity of its negative ones. In particular, we might make a sentence become true (or valid) by replacing one of its strictly positive subsentences with *true* or one of its strictly negative subsentences with *false*, but never by replacing one of its strictly negative subsentences with *true* or one of its strictly positive subsentences with *false*.

Example:

The subsentences of the following sentence are annotated according to their polarities in the sentence:

(*if* $p(x)^-$
then $((\exists y)q(y)^+)^+)^+$.

■

We can extend the notion of polarity to apply to a tableau as well as to a sentence. We regard each goal as positive in the tableau. Because, by the duality principle, an assertion $\mathcal{F}$ is equivalent to a goal *not* $\mathcal{F}$, each assertion is within the scope of an implicit *not* connective, and is therefore negative in the tableau.

Example:

The subsentences of the following tableau are annotated according to their polarities in the tableau:

assertions	goals	outputs
$\begin{pmatrix} \textit{if } p(x)^{+} \\ \textit{then } ((\exists y) q(y)^{-})^{-} \end{pmatrix}^{-}$		
	$\left(\begin{pmatrix} p(x)^{\pm} \equiv [q(x)^{\pm} \textit{ or } r(x)^{\pm}]^{\pm} \\ \textit{or } p(a)^{+} \end{pmatrix}^{+}\right)^{+}$	

Note that the subsentence $p(x)$ is negative in the sentence

$$\begin{array}{l} \textit{if } p(x) \\ \textit{then } (\exists y) q(y) \end{array}$$

but positive in the tableau, which contains this sentence as an assertion. Note also that every subsentence of an equivalence has both polarities and the only subsentences of both polarities are subsentences of equivalences. If we wanted to include the connective *if* $\mathcal{P}$ *then* $\mathcal{Q}$ *else* $\mathcal{R}$ in our language, the subsentences of $\mathcal{P}$ would also have both polarities, since this construct is regarded as an abbreviation for

$$\begin{array}{l} (\mathcal{P} \textit{ and } \mathcal{Q}) \textit{ or} \\ ((\textit{not } \mathcal{P}) \textit{ and } \mathcal{R}). \end{array}$$

Henceforth, however, we shall not regard this connective as part of the language.

The Force of Quantifiers

By the well-known duality between the universal and existential quantifiers, the "roles" of the quantifiers are reversed by putting them within the scope of an additional negation sign. Thus, the universal quantifier in $\textit{not } (\forall x) p(x)$ plays the same role as the existential quantifier in $(\exists x)\, [\textit{not } p(x)]$.

With this in mind, we define the *force* of a quantifier $(\forall x)$ or $(\exists x)$ in a subsentence $\mathcal{E}$ of form $(\forall x)\mathcal{F}$ or $(\exists x)\mathcal{F}$ in a sentence (or tableau) according to the following rules:

- The quantifier has *universal force* if it is a universal quantifier and $\mathcal{E}$ is of positive polarity, or if it is an existential quantifier and $\mathcal{E}$ is of negative polarity in the sentence (or tableau).
- The quantifier has *existential force* if it is an existential quantifier and $\mathcal{E}$ is of positive polarity, or if it is a universal quantifier and $\mathcal{E}$ is of negative polarity in the sentence (or tableau).

Because a subsentence may have both positive and negative polarity, a quantifier may be of both positive and negative force; these are the quantifiers within the scope of an equivalence. A quantifier that does not have both forces is said to be of ***strict force***.

Example:

The quantifiers in the following tableau are annotated according to their forces:

assertions	goals	outputs
if $(\exists x)^{\exists} p(x)$ *then* $(\exists y)^{\forall} q(y)$		
	$(\forall z)^{\forall} p(z)$	
$[(\exists u)^{\forall\exists} r(u)] \equiv r(a)$		

Here, the quantifier $(\exists x)$ has existential force because the subsentence $(\exists x)p(x)$ is positive in the tableau; the quantifier $(\exists y)$ has universal force because the subsentence $(\exists y)q(y)$ is negative in the tableau. All the quantifiers are of strict force except $(\exists u)$. ∎

Removal of Quantifiers

Rather than regard quantifier removal as a separate stage, to be done before theorem proving takes place, we allow skolemization to occur at any stage of the theorem-proving process. In practice, we are likely to defer removal of those quantifiers within the scope of an equivalence, because this will require prior removal of the equivalence, with consequent explosion of the theorem.

The skolemization rules permit us to remove any quantifier of strict force from a tableau; the variables bound by the quantifier are replaced by free variables if the quantifier is of existential force, and by "skolem" constants or terms if the quantifier is of universal force. Quantifiers of both forces cannot be removed. (However, if we first remove the enclosing equivalences, a quantifier of both forces will be split into two or more quantifiers of strict force; see the section "**Removal of Equivalences.**")

Removal of Quantifiers of Universal Force

We first deal with the removal of quantifiers of strict universal force.

Rule (universal elimination):

Suppose our tableau contains an assertion (or goal) $\mathcal{F}$ of form

$\mathcal{F}$: $\mathcal{F}_0((\ldots z)^{\forall} p)$.

Here, $(...z)^{\forall}\mathcal{P}$ denotes a subsentence of $\mathcal{F}$, where $(...z)^{\forall}$ is a quantifier, either $(\forall z)$ or $(\exists z)$, that is of strict universal force (in the tableau).

Assume that the variables $x_1, x_2, \ldots, x_m$ are the only free variables in $\mathcal{F}$ and that $(...y_1)^{\exists}$, $(...y_2)^{\exists}$, $\ldots$, $(...y_n)^{\exists}$ are the only quantifiers in $\mathcal{F}$ of existential force that contain the subsentene $(...z)^{\forall}\mathcal{P}$ within their scope. Let f be a new function symbol, i.e., one that occurs nowhere in the tableau.

Then we may add to our tableau the new assertion (or goal)

$\mathcal{F}'$: $\mathcal{F}_0(\mathcal{P} \blacktriangleleft \{z \leftarrow f(x_1, \ldots, x_m, y_1, \ldots, y_n)\})$.

In other words, $\mathcal{F}'$ is formed by removing the quantifier $(...z)^{\forall}$ in $\mathcal{F}$ and replacing every occurrence of z in $\mathcal{P}$ by the term $f(x_1, \ldots, x_m, y_1, \ldots, y_n)$. We shall refer to a term added in this way as a *skolem term*, and to f as a *skolem function*. We will say that we have "replaced" the quantifier with the skolem function.

In the special case in which there are no free variables $x_1, x_2, \ldots, x_m$ and no enclosing quantifiers $(...y_1)^{\exists}, (...y_2)^{\exists}, \ldots, (...y_n)^{\exists}$, we let a be a new constant; then we may add to the tableau the new assertion or goal

$\mathcal{F}'$: $\mathcal{F}_0(\mathcal{P} \blacktriangleleft \{z \leftarrow a\})$.

We will refer to a constant added in this way as a *skolem constant*.

Example:

Suppose our tableau contains the assertion

assertions	goals	outputs
$\mathcal{F}$: $r(x)$ *or* $(\forall y)^{\exists}[q(x,y)$ *and* $(\exists z)^{\forall}p(x,y,z)]$		

Here, x is the only free variable in $\mathcal{F}$ and $(\forall y)^{\exists}$ is the only quantifier of existential force that contains the quantifier $(\exists z)^{\forall}$ within its scope. Therefore, we may remove the quantifier $(\exists z)^{\forall}$ from the assertion by replacing every occurrence of z with the skolem term $f(x,y)$, adding to our tableau the new assertion

$\mathcal{F}'$: $r(x)$ *or* $(\forall y)[q(x,y)$ *and* $p(x,y,f(x,y))]$		

where f is a new function symbol. ∎

Note that the rule enables us to remove single occurrences of quantifiers without altering others in the sentence.

Removal of Quantifiers of Existential Force

The forthcoming existential elimination rule allows us to remove quantifiers of strict existential force. However, the quantifier to be removed must not be within the scope of any quantifiers of universal force; such quantifiers should be removed by prior application of the preceding rule.

Rule (existential elimination):

Suppose our tableau contains an assertion or goal $\mathcal{F}$ of form

$\mathcal{F}$: $\mathcal{F}_0((...z)^{\exists} P)$

where $(...z)^{\exists}$ is a quantifier of strict existential force. Assume that no quantifiers of universal force contain the subsentence $(...z)^{\exists} P$ within their scope. Then we may add to the tableau the new assertion or goal

$\mathcal{F}'$: $\mathcal{F}_0(P)$.

In other words, we may remove the quantifier $(...z)^{\exists}$ so that every occurrence of z in P becomes a free variable.

Example:

Suppose our tableau contains the goal

assertions	goals	outputs
	$\mathcal{F}$: $(\exists z_1)^{\exists}[p(z_1)$ *and* $(\exists z_2)^{\exists} q(z_1, z_2)]$	

Here the quantifier $(\exists z_2)^{\exists}$ is not within the scope of any quantifier of universal force. Therefore, we may remove the quantifier $(\exists z_2)^{\exists}$ by adding to the tableau the new goal

	$\mathcal{F}'$: $(\exists z_1)[p(z_1)$ *and* $q(z_1, z_2)]$	

We could also have used the rule to remove the quantifier $(\exists z_1)$ from $\mathcal{F}$. ∎

TRANSFORMATION RULES

Before we introduce the transformation rules, it is necessary to extend the notion of unification to sentences with quantifiers.

Unification

Unification became widely known through its use in the original resolution principle (Robinson [1965]), in which it was applied only to atomic sentences. The extension to nonatomic sentences with quantifiers is straightforward.

We assume that, in matching subsentences of sentences with quantifiers, the variables that are bound in the surrounding sentence are distinguishable from free variables by some invisible annotation. Then:

- Logical connectives are treated like function symbols. Thus,

  $$\textit{if } p(x) \textit{ then } q(x, f(x))$$

  will unify with

  $$\textit{if } p(a) \textit{ then } q(y, z),$$

  yielding a most-general unifier

  $$\{x \leftarrow a, y \leftarrow a, z \leftarrow f(a)\}.$$

- Bound variables are treated like constants. Thus, we cannot unify the subsentence $p(u)$ of the sentence

  $$(\exists u)[p(u) \textit{ and } q(y)]$$

  and the subsentence $p(z)$ of the sentence

  $$(\forall z)[\textit{if } p(z) \textit{ then } r(u, z)].$$

  However, we can unify either of these subsentences with the subsentence $p(x)$ of the sentence

  $$p(x) \textit{ or } s(x),$$

  in which x is free, yielding the most-general unifiers $\{x \leftarrow u\}$ and $\{x \leftarrow z\}$, respectively.

- To unify two sentences of form $(\forall x)P$ and $(\forall x')P'$, we attempt to unify P and $P' \blacktriangleleft \{x' \leftarrow x\}$, the result of replacing all occurrences of x' in P' with x, treating x

as a constant. If we are successful, obtaining a unifier θ, our result is $\{x' \leftarrow x\} \diamond \theta$, the composition of the substitution $\{x' \leftarrow x\}$ and θ. (Similarly for existential quantifiers.)

Example:

To unify $(\forall x)p(x, a, u)$ and $(\forall y)p(y, v, b)$, where u and v are free variables, we first unify $p(x, a, u)$ and $p(y, v, b) \blacktriangleleft \{y \leftarrow x\}$, that is, $p(x, v, b)$, obtaining a unifier $\theta = \{v \leftarrow a, u \leftarrow b\}$. Our resulting unifier is then $\{y \leftarrow x\} \diamond \theta = \{y \leftarrow x, v \leftarrow a, u \leftarrow b\}$. ∎

Statement of a Transformation Rule

Suppose that any sentence of form $\mathcal{P}$ is equivalent to the corresponding sentence of form $\mathcal{Q}$. Then a transformation rule

$$\mathcal{P} \Rightarrow \mathcal{Q}$$

allows us to replace a subsentence of form $\mathcal{P}$ by the corresponding equivalent subsentence of form $\mathcal{Q}$ in any assertion or goal, yielding a new assertion or goal, respectively, to add to the tableau.

Before we present the precise statement, let us give a rough schematic description of the application of a transformation rule to an assertion in the ground case, where there are no variables and also no output entries:

assertions	goals	outputs
$\mathcal{F}(\mathcal{P})$		
$\mathcal{F}(\mathcal{Q})$		

Similarly, to apply the rule to a goal, we write

assertions	goals	outputs
	$\mathcal{F}(\mathcal{P})$	
	$\mathcal{F}(\mathcal{Q})$	

Here, if $\mathcal{F}(\mathcal{P})$ is a sentence with a subsentence $\mathcal{P}$, $\mathcal{F}(\mathcal{Q})$ is the result of replacing every instance of $\mathcal{P}$ in $\mathcal{F}(\mathcal{P})$ with $\mathcal{Q}$.

For example, the *then-false* rule

if $\mathcal{G}$ *then false* $\Rightarrow$ *not* $\mathcal{G}$

applied to the goal

assertions	goals	outputs
	not $\boxed{\textit{if } p(x) \textit{ then false}}$ *and* *not* $q(x)$	

yields the new goal

	not not $p(x)$ *and* *not* $q(x)$	

We use the box to indicate the subexpression to which the rule is about to be applied.

Other examples of transformation rules are the *not-not* rule

$$\textit{not not } \mathcal{G} \Rightarrow \mathcal{G}$$

and the *or-two* rule

$$\mathcal{G} \textit{ or } \mathcal{G} \Rightarrow \mathcal{G}.$$

To describe the application of these rules more precisely, we regard the script letters $\mathcal{G}$, $\mathcal{H}$, ..., that appear in such rules as free variables that range over sentences, and we attempt to unify the left-hand side of the rule with subsentences of the tableau.

Rule (transformation):

The application of a transformation rule

$$P \Rightarrow Q$$

to an assertion is represented in tableau notation by

assertions	goals	outputs
$\mathcal{F}$		f
$(\mathcal{F} \blacktriangleleft \theta) \blacktriangleleft \{P \blacktriangleleft \theta \leftarrow Q \blacktriangleleft \theta\}$		$f \blacktriangleleft \theta$

Here we assume that

- There is a set $\{P_1, \ldots, P_k\}$ of disjoint subsentences of $\mathcal{F}$ such that $P, P_1, \ldots, P_k$ are unifiable, with most-general unifier θ. Thus $P \blacktriangleleft \theta, P_1 \blacktriangleleft \theta, \ldots, P_k \blacktriangleleft \theta$ are all identical sentences.
- $\mathcal{F} \blacktriangleleft \theta, P \blacktriangleleft \theta, Q \blacktriangleleft \theta$, and $f \blacktriangleleft \theta$ are the results of applying the substitution θ to $\mathcal{F}$, P, Q and f, respectively.
- $(\mathcal{F} \blacktriangleleft \theta) \blacktriangleleft \{P \blacktriangleleft \theta \leftarrow Q \blacktriangleleft \theta\}$ is the result of replacing every occurrence of $P \blacktriangleleft \theta$ in $\mathcal{F} \blacktriangleleft \theta$ with $Q \blacktriangleleft \theta$.
- If x is any free variable in $\mathcal{F}$ that occurs within the scope of a quantifier, θ cannot instantiate x to any term t containing a bound variable of $\mathcal{F}$.

(*dependency restriction*)

If there is no output entry f in the original row, then there is no output entry in the new row either.

In the precise version of the rule, we consider a set of subsentences of $\mathcal{F}$ because these reduce to a single sentence on application of the substitution θ.

We assume that the variables of transformation rules are standardized apart in the same way as the variables of the tableau itself. Thus, the bound and free variables of transformation rules are tacitly renamed so that they are distinct both from one another and from the variables of the tableau.

The application of a transformation rule $P \Rightarrow Q$ to a goal is similar. In tableau notation, we have

assertions	goals	outputs
	$\mathcal{F}$	f
	$(\mathcal{F} \blacktriangleleft \theta) \blacktriangleleft \{P \blacktriangleleft \theta \leftarrow Q \blacktriangleleft \theta\}$	$f \blacktriangleleft \theta$

The same notation and the same restrictions apply as when the rule was applied to an assertion. It is also possible to apply transformation rules to output entries.

We first illustrate the rule with a straightforward example; then we present a counterexample to show that the dependency restriction is necessary.

Example:

Suppose our tableau contains the assertion

assertions	goals	outputs
$\mathcal{F}$: $\boxed{p(a,y) \text{ and } p(x,f(x))}$ or $r(x,y)$		$g(x,y)$

Then we can apply the *and-two* rule

$$\mathcal{G} \text{ and } \mathcal{G} \Rightarrow \mathcal{G}$$

to the subexpression $p(a,y)$ *and* $p(x,f(x))$ of $\mathcal{F}$. The unifier θ is

$$\{x \leftarrow a,\ \ y \leftarrow f(a),\ \ \mathcal{G} \leftarrow p(a,f(a))\}$$

and the new row is

$p(a,f(a))$ *or* $r(a,f(a))$		$g(a,f(a))$

Note that the substitution θ is applied to the output entry as well as to the assertion. ∎

The Dependency Restriction

Let us consider the rationale for the dependency restriction.

Example:

To see why the restriction is required, suppose our tableau contains the assertion

$\mathcal{F}$: $(\exists y)\boxed{p(x,y) \text{ or } p(y,x)}$		

Then, were the restriction not required, we could apply the *or-two* rule

$$\mathcal{G} \text{ or } \mathcal{G} \Rightarrow \mathcal{G}$$

to the subsentence

$$p(x,y) \text{ or } p(y,x)$$

of the assertion. The unifier θ would be

$$\{x \leftarrow y,\ \ \mathcal{G} \leftarrow p(y,y)\}$$

and the (erroneous) new row would be

$(\exists y)p(y,y)$		

This step violates the dependency restriction, because x is a free variable in the assertion, x occurs within the scope of the quantifier $(\exists y)$, and θ instantiates x to the term y, which contains a bound variable.

The new assertion is not a valid conclusion to draw from the given one. For example, in the interpretation whose domain is the set of integers $\{0, 1\}$ and that takes $p(x, y)$ to mean $x < y$, the given assertion means

$$(\exists y)[x < y \ \ or \ \ y < x]$$

for any x, which is true, but the new assertion means

$$(\exists y)[y < y],$$

which is false. ∎

In fact, if we had skolemized the given assertion, we would have obtained an assertion

$p(x, f(x))$ *or* $p(f(x), x)$		

The *or-two* rule cannot be applied to this assertion, because its left-hand side $\mathcal{G}$ *or* $\mathcal{G}$ fails to unify with the assertion; the subterms x and $f(x)$ cannot be unified. ∎

When the application of a transformation rule is blocked by the dependency restriction, it is possible that the rule may be applicable if the quantifier of the offending bound variable is first removed by skolemization.

Example:

Suppose our tableau contains the goal

assertions	goals	outputs
	$(\exists y)(p(x, y)$ *or* $p(y, x))$	

Then if we momentarily disregard the dependency restriction, we can apply the *or-two* rule

$$\mathcal{G} \ \ or \ \ \mathcal{G} \ \Rightarrow \ \mathcal{G}$$

to the subsentence $p(x, y)$ *or* $p(y, x)$. The unifier θ is

$$\{x \leftarrow y, \ \ \mathcal{G} \leftarrow p(y, y)\}$$

and the new goal is

	$(\exists y)p(y, y)$	

Although this is a valid step, which preserves the meaning of the tableau, it does violate the dependency restriction: the free variable x, which is within the scope of the quantifier $(\exists y)$, is instantiated to the bound variable y. Thus, in this case, the restriction is unduly prohibitive.

Had we first removed the quantifier by skolemization, however, obtaining the goal

	$p(x, y)$ *or* $p(y, x)$	

we could indeed have applied the *or-two* rule to obtain the goal

	$p(y, y)$	

The True-False Rules

We assume we have a full complement of *true-false* rules for removing occurrences of the propositions *true* and *false* from sentences, e.g., the *and-true* rules

$$\mathcal{G} \textit{ and } \textit{true} \Rightarrow \mathcal{G}$$

$$\textit{true and } \mathcal{G} \Rightarrow \mathcal{G},$$

the *then-true* and *then-false* rules,

$$\textit{if } \mathcal{G} \textit{ then true} \Rightarrow \textit{true}$$

$$\textit{if } \mathcal{G} \textit{ then false} \Rightarrow \textit{not } \mathcal{G},$$

and the *all-true* and *all-false* rules

$$(\forall x)\textit{true} \Rightarrow \textit{true}$$

$$(\forall x)\textit{false} \Rightarrow \textit{false}.$$

These rules and certain of the other transformation rules are so fundamental that sometimes we will apply them automatically, as a simplification step, without mentioning it.

Removal of Equivalence

We also assume we have the *equivalence elimination* rules, the *iff-or* rule

$$\mathcal{G} \equiv \mathcal{H} \Rightarrow (\mathcal{G} \textit{ and } \mathcal{H}) \textit{ or}$$

$$((\mathit{not}\ \mathcal{G})\ \mathit{and}\ (\mathit{not}\ \mathcal{H}))$$

and the *iff-and* rule

$$\mathcal{G} \equiv \mathcal{H} \Rightarrow (\mathit{if}\ \mathcal{G}\ \mathit{then}\ \mathcal{H})\ \mathit{and}\ (\mathit{if}\ \mathcal{H}\ \mathit{then}\ \mathcal{G}).$$

These rules will enable us to remove equivalences when we cannot prove the theorem otherwise. By repeated application of these rules, we can ensure that a given quantifier has strict force, and then remove it by skolemization. This may be necessary if we fail to apply, say, a transformation rule because a quantified variable has caused the unification to fail or the dependency restriction to be violated.

THE RESOLUTION RULE

The resolution rule performs a case analysis on the truth of a subsentence of the assertions or goals of a tableau. At the same time, the rule instantiates variables and accounts for the introduction of conditional expressions into the program being constructed.

Statement of the Resolution Rule

The rule can be applied to two rows of the tableau, whether these rows contain assertions or goals. We present first the "GG-form" of the rule, which applies to two goals.

The schematic description of the ground version of the rule is as follows.

assertions	goals	outputs
	$\mathcal{F}(P)$	f
	$\mathcal{G}(P)$	g
	$\mathcal{F}(true)$ *and* $\mathcal{G}(false)$	*if* P *then* f *else* g

In other words, we seek a common subsentence P of $\mathcal{F}$ and $\mathcal{G}$, replace all occurrences of P in $\mathcal{F}$ and in $\mathcal{G}$ with *true* and *false*, respectively, and add the conjunction of the resulting sentences as a new goal. The output entry is a conditional expression, with P as its test.

The rationale for this rule is as follows. Consider an interpretation under which the derived goal $\mathcal{F}(true)$ *and* $\mathcal{G}(false)$ is true; we seek to show that one of the two given goals $\mathcal{F}(P)$ or $\mathcal{G}(P)$ is then also true under this interpretation. Because the conjunction is true, both of its conjuncts $\mathcal{F}(true)$ and $\mathcal{G}(false)$ are true. In the case in which P is true under the interpretation, the given goal $\mathcal{F}(P)$ is true; in this case, f is a suitable output. In the case in which P is false, the given goal $\mathcal{G}(P)$ is true; in this case, g is a suitable output. In either case, the conditional expression *if* P *then* f *else* g is a suitable output.

The more precise description of the rule is as follows:

Rule (resolution):

An application of the resolution rule is written in tableau notation by

assertions	goals	outputs
	$\mathcal{F}$	f
	$\mathcal{G}$	g
	$(\mathcal{F} \blacktriangleleft \theta) \blacktriangleleft \{P \blacktriangleleft \theta \leftarrow \mathit{true}\}$ *and* $(\mathcal{G} \blacktriangleleft \theta) \blacktriangleleft \{P \blacktriangleleft \theta \leftarrow \mathit{false}\}$	*if* $P \blacktriangleleft \theta$ *then* $f \blacktriangleleft \theta$ *else* $g \blacktriangleleft \theta$

Here we assume that

- $P = \{P_1, \ldots, P_k\}$ is a set of subsentences of $\mathcal{F}$ and $Q = \{Q_1, \ldots, Q_\ell\}$ is a set of subsentences of $\mathcal{G}$ that are all unifiable with most-general unifier θ. Thus $P_1 \blacktriangleleft \theta, \ldots, P_k \blacktriangleleft \theta, Q_1 \blacktriangleleft \theta, \ldots, Q_\ell \blacktriangleleft \theta$ are all identical sentences, denoted (by abuse of notation) by $P \blacktriangleleft \theta$.

- As before, $(\mathcal{F} \blacktriangleleft \theta) \blacktriangleleft \{P \blacktriangleleft \theta \leftarrow \mathit{true}\}$ and $(\mathcal{G} \blacktriangleleft \theta) \blacktriangleleft \{P \blacktriangleleft \theta \leftarrow \mathit{false}\}$ denote the results of replacing every occurrence of $P \blacktriangleleft \theta$ in $\mathcal{F} \blacktriangleleft \theta$ and $\mathcal{G} \blacktriangleleft \theta$, respectively, with the propositional symbols *true* and *false*, respectively.

- If x is any free variable in $\mathcal{F}$ or in $\mathcal{G}$ that occurs within the scope of a quantifier, then θ cannot instantiate x to any term t containing a bound variable of $\mathcal{F}$ or of $\mathcal{G}$.

  (*dependency restriction*)

- No variable that is bound in $\mathcal{F}$ or in $\mathcal{G}$ may occur free in the new row.

  (*no-escape restriction*)

In the precise version of the rule, we consider a set of subsentences of $\mathcal{F}$ (and of $\mathcal{G}$) because these sentences reduce to a single sentence on application of the substitution θ. Recall we have assumed that the variables of our tableau are standardized apart, so that the variables of $\mathcal{F}$ are distinct from those of $\mathcal{G}$.

Murray's [1982] polarity strategy for resolution allows us to consider only those applications of the rule under which some occurrence of $P \blacktriangleleft \theta$ in $\mathcal{F} \blacktriangleleft \theta$ is positive in the tableau and some occurrence of $P \blacktriangleleft \theta$ in $\mathcal{G} \blacktriangleleft \theta$ is negative in the tableau. This strategy not only preserves completeness, but also rarely blocks a reasonable step.

Examples

We give a straightforward example of the application of the rule and two counterexamples illustrating the necessity for the dependency and no-escape restrictions.

Example:

Suppose our tableau contains the two goals

assertions	goals	outputs
	$\mathcal{F}$: $\boxed{x < a}^{+}$	$f(x)$
	$\mathcal{G}$: $not\ \boxed{b < y}^{-}$	$g(y)$

Here we use the box notation to indicate the subsentences that are about to be matched in applying the rule.

According to the tableau, if we can find x such that $x < a$, then $f(x)$ is a suitable output, and if we can find y such that $not\ (b < y)$, then $g(y)$ is a suitable output. Let P be the subset $\{x < a, b < y\}$ of subsentences of $\mathcal{F}$ and $\mathcal{G}$. Then P is unifiable with most-general unifier $\theta = \{x \leftarrow b, y \leftarrow a\}$, and $P \blacktriangleleft \theta$ is $b < a$. By the resolution rule, we may infer the new goal

	$(b < a) \blacktriangleleft \{(b < a) \leftarrow true\}$ *and* $(not\ (b < a)) \blacktriangleleft \{(b < a) \leftarrow false\}$	*if* $b < a$ *then* $f(b)$ *else* $g(a)$

i.e.,

	true and *(not false)*	*if* $b < a$ *then* $f(b)$ *else* $g(a)$

which reduces to

	true	*if* $b < a$ *then* $f(b)$ *else* $g(a)$

under the *not-false* rule

$$not\ false \Rightarrow true$$

and the *and-true* rule

$$P\ and\ true \Rightarrow P.$$

Note that this application of the resolution rule is in accordance with the polarity strategy ∎

Example:

To see why the dependency restriction is necessary, assume our tableau contains the two goals

assertions	goals	outputs
	$\mathcal{F}:\ (\forall z)\boxed{p(z,u)}^{+}$	
	$\mathcal{G}:\ (\forall y)(not\ \boxed{p(x,y)}^{-})$	

If the dependency restriction were not imposed, we would be able to apply the resolution rule to match $p(z,u)$ against $p(x,y)$, with most-general unifier $\theta = \{x \leftarrow z, u \leftarrow y\}$, obtaining (erroneously) the new row

	$(\forall z)true\ and$ $(\forall y)(not\ false)$	

which reduces to

	$true$	

after true-false transformation.

This step violates the dependency restriction, because the free variables x and u, which occur within the scopes of quantifiers, are instantiated to the bound variables z and y, respectively.

The preceding deduction is not sound, because we can imagine interpretations under which all instances of both goals are false, e.g., if p is the equality predicate and the domain has more than one element. ∎

Example:

To see why the no-escape restriction is necessary, assume our tableau contains the goals

assertions	goals	outputs
	$\boxed{p(z)}^{+}\ and\ q(z)$	$g(z)$
	$(\forall u)(not\ \boxed{p(u)}^{-})$	

Then, if the no-escape condition were not imposed, we would be able (erroneously) to derive the goal

	$(true\ and\ q(u))\ and$ $(\forall u)(not\ false)$	$g(u)$

which reduces to

	$q(u)$	$g(u)$

Here the bound variable u of the second goal has "escaped" and become free, giving it a tacit existential quantification in the new goal it did not have in the original goal.

For instance, in an interpretation over the integers in which $p(z)$ and $q(z)$ denote the conditions that z is even and odd, respectively, both given goals are false: our first goal requires that we find a z that is both even and odd, while our second goal requires us to show that every integer is not even. The derived goal, on the other hand, is true: it requires that we find an integer u that is odd.

Note that, if the tableau contains the two goals

assertions	goals	outputs
	$\boxed{p(z)}^{+}$	
	$(\forall u)(not\ \boxed{p(u)}^{-})$	

then we could apply the resolution rule to match $p(z)$ against $p(u)$, taking the most-general unifier $\theta = \{z \leftarrow u\}$ without violating either restriction. In this case, the new goal is

	true and $(\forall u)(not\ false)$	

which reduces to

	true	

Dual Forms of the Resolution Rule

We have given the GG-form of the resolution rule, which applies to two goals. The AA-, AG-, and GA-forms of the rule, which apply to two assertions, an assertion and a goal, and a goal and an assertion, respectively, may be derived by duality from the GG-form. The schematic version of the GA-form of the rule (ground case) is as follows:

assertions	goals	outputs
	$\mathcal{F}(P)$	f
$\mathcal{G}(P)$		g
	$\mathcal{F}(true)$ *and* *not* $\mathcal{G}(false)$	*if* P *then* f *else* g

The precise description of the rule and its restrictions are analogous to those of the GG-form. The AA-form is phrased to produce a new assertion rather than a new goal. If one of the given rows, say $\mathcal{F}(P)$, has no output entry, the output entry for the new row is simply g (or, in the precise version, $g \blacktriangleleft \theta$) rather than a conditional expression. If neither of the given rows has an output entry, the new row has no output entry either. The polarity strategy for the dual forms of the resolution rule is precisely the same as that for the GG-form.

Relaxing the Dependency Restriction

The dependency restriction for the resolution rule can be relaxed to allow the rule to apply in more situations; the relaxed restriction, however, is more complex than the original.

Recall that the restriction is

> If x is any free variable in the given rows $\mathcal{F}$ or $\mathcal{G}$ that occurs within the scope of a quantifier, then the unifier θ cannot instantiate x to any term t containing a bound variable of $\mathcal{F}$ or of $\mathcal{G}$.

Actually, the restriction can be relaxed by applying it only to free variables that occur within the scope of a quantifier whose variable actually occurs in one of the matched sentences. More precisely, the restriction can be revised as follows:

- If x is any free variable in $\mathcal{F}$ or in $\mathcal{G}$ that occurs within the scope of a quantifier $(\ldots y)$ whose variable y occurs in at least one of the matched sentences $P_1, \ldots, P_k$ or $Q_1, \ldots, Q_\ell$, then θ cannot instantiate x to any term t containing a bound variable of $\mathcal{F}$ or of $\mathcal{G}$.

  (*relaxed dependency restriction*)

Let us look at an (admittedly rare) example of a valid application of the resolution rule that violates the original dependency restriction but not the relaxed dependency restriction.

Example:

Suppose our tableau contains the goal and assertion

assertions	goals	outputs
	$\mathcal{F}:\ (\forall y)(\boxed{p(x)}^{+}\ \textit{and}\ q(y))$	
$\mathcal{G}:\ (\exists z)\begin{pmatrix}\textit{if}\ r(z)\\ \textit{then}\ \boxed{p(z)}^{-}\end{pmatrix}$		

Disregarding both versions of the dependency restriction momentarily, we can apply the GA-resolution rule, taking θ to be $\{x \leftarrow z\}$, to obtain the new goal

	$(\forall y)(\textit{true}\ \textit{and}\ q(y))\ \textit{and}$ $\textit{not}\ (\exists z)\begin{pmatrix}\textit{if}\ r(z)\\ \textit{then}\ \textit{false}\end{pmatrix}$	

which reduces to

	$(\forall y)q(y)\ \textit{and}$ $\textit{not}\ (\exists z)(\textit{not}\ r(z))$	

under true-false transformation.

This step is legitimate — it preserves the meaning of the tableau — but it violates the original dependency restriction. The free variable x in the goal $\mathcal{F}$, which occurs within the scope of the quantifier $(\forall y)$, is instantiated by θ to the bound variable z. On the other hand, the step does not violate the relaxed dependencty restriction, because the variable y of the quantifier $(\forall y)$ does not occur in the matched subsentence $p(x)$. ∎

We did not present the relaxed dependency restriction at first because it is more complex than the original restriction and only permits a few additional applications of the resolution rule.

EQUALITY AND EQUIVALENCE SUBSTITUTION RULES

The equality predicate has long been recognized as meriting special treatment. The use of axioms to represent the properties of the relation lengthens the proof and dramatically explodes the search space. In the resolution framework, special inference rules such as paramodulation (Wos and Robinson [1969]) and E-resolution (Morris [1969]) were soon brought to bear in an attempt to control the proliferation of clauses.

The equivalence connective has not been recognized as such a trouble spot, but, as we have indicated in the introduction, it is common in the specification of programs. Proofs become longer and lose their intuitive motivation when equivalence is paraphrased in terms of other connectives. Furthermore, the techniques that apply to the equality predicate can be easily adapted to the equivalence connective. In this section, we present nonclausal versions of both paramodulation and E-resolution and apply the rules to both equality and equivalence.

Equality Substitution Rule

The "substitution rules" are our nonclausal counterpart of paramodulation. The equality substitution rule allows us to use an equality that occurs in one row of a tableau to replace a subterm with an equal term in another (or even possibly the same) row. We present the AA-form of the rule, which applies between two assertions.

The rough schematic description of the ground version of the rule is as follows:

assertions	goals	outputs
$\mathcal{F}(S = T)$		f
$\mathcal{G}(S)$		g
$\mathcal{F}(\mathit{false})$ *or* $\mathcal{G}(T)$		*if* $S = T$ *then* g *else* f

Here, we seek an explicit equality $S = T$ in $\mathcal{F}$, where S also occurs in $\mathcal{G}$. We replace every occurrence of $S = T$ in $\mathcal{F}$ with *false*, replace some occurrences of S in $\mathcal{G}$ with T, and add their disjunction as a new assertion. The output entry is a conditional expression with $S = T$ as its test. Note that, in an abuse of notation, we do not necessarily replace every occurrence of S in $\mathcal{G}$ with T.

The rationale for this rule is as follows. Consider an interpretation under which both given assertions are true; we seek to show that the derived assertion is also true under this interpretation. Equivalently, we show that if the derived assertion is false, then one or the other of the given assertions is also false. Because the disjunction $\mathcal{F}(\mathit{false})$ *or* $\mathcal{G}(T)$ is false, each of its disjuncts

$\mathcal{F}(\mathit{false})$ and $\mathcal{G}(\mathcal{T})$ is false. In the case in which $\mathcal{S} = \mathcal{T}$ is false, because $\mathcal{F}(\mathit{false})$ is false, we know the given assertion $\mathcal{F}(\mathcal{S} = \mathcal{T})$ is false; in this case, f is a suitable output (i.e., it satisfies the specification for the desired program). In the case in which $\mathcal{S} = \mathcal{T}$ is true, because $\mathcal{G}(\mathcal{T})$ is false, we know the given assertion $\mathcal{G}(\mathcal{S})$ is false; in this case, g is a suitable output. In either case, the conditional expression *if* $\mathcal{S} = \mathcal{T}$ *then* g *else* f is a suitable output.

The precise description of the rule follows:

Rule (equality substitution):

Expressed in our tableau notation, the rule is

assertions	goals	outputs
$\mathcal{F}$		f
$\mathcal{G}$		g
$(\mathcal{F} \blacktriangleleft \theta) \blacktriangleleft \{(\mathcal{S} \blacktriangleleft \theta = \mathcal{T} \blacktriangleleft \theta) \leftarrow \mathit{false}\}$ *or* $(\mathcal{G} \blacktriangleleft \theta) \triangleleft \{\mathcal{S} \blacktriangleleft \theta \leftarrow \mathcal{T} \blacktriangleleft \theta\}$		*if* $\mathcal{S} \blacktriangleleft \theta = \mathcal{T} \blacktriangleleft \theta$ *then* $g \blacktriangleleft \theta$ *else* $f \blacktriangleleft \theta$

Here we assume that

- $\mathcal{S} = \{s_0, s_1, \ldots s_k\}$ and $\mathcal{T} = \{t_1, \ldots, t_k\}$ are sets of terms such that
  - $\mathcal{F}$ contains at least one occurrence of each equality $s_1 = t_1, \ldots, s_k = t_k$;
  - $\mathcal{G}$ contains at least one occurrence of s_0;
  - θ is a most-general unifier of $\mathcal{S}$ and of $\mathcal{T}$: i.e., $s_0 \blacktriangleleft \theta, s_1 \blacktriangleleft \theta, \ldots, s_k \blacktriangleleft \theta$ are identical terms, denoted by $\mathcal{S} \blacktriangleleft \theta$; and $t_1 \blacktriangleleft \theta, \ldots, t_k \blacktriangleleft \theta$ are identical terms, denoted by $\mathcal{T} \blacktriangleleft \theta$; and θ is one of the most-general substitutions that make these expressions identical.
- $(\mathcal{F} \blacktriangleleft \theta) \blacktriangleleft \{(\mathcal{S} \blacktriangleleft \theta = \mathcal{T} \blacktriangleleft \theta) \leftarrow \mathit{false}\}$ denotes the result of replacing every occurrence of the subsentence $\mathcal{S} \blacktriangleleft \theta = \mathcal{T} \blacktriangleleft \theta$ in $\mathcal{F} \blacktriangleleft \theta$ with the proposition *false*.
- The symbol $\triangleleft$ is defined so that $(\mathcal{G} \blacktriangleleft \theta) \triangleleft \{\mathcal{S} \blacktriangleleft \theta \leftarrow \mathcal{T} \blacktriangleleft \theta\}$ denotes the result of replacing one or more (but not necessarily all) occurrences of $\mathcal{S} \blacktriangleleft \theta$ in $\mathcal{G} \blacktriangleleft \theta$ with $\mathcal{T} \blacktriangleleft \theta$.
- If x is any variable in $\mathcal{F}$ or in $\mathcal{G}$ that occurs within the scope of a quantifier, then θ cannot instantiate x to any term t containing a bound variable of $\mathcal{F}$ or of $\mathcal{G}$.
  (*dependency restriction*)
- No variable that is bound in $\mathcal{F}$ or in $\mathcal{G}$ may occur free in the new row.
  (*no-escape restriction*)

If one of the given rows, say $\mathcal{F}$, has no output entry, the output entry for the new row is simply $g \blacktriangleleft \theta$ rather than a conditional expression, as in the resolution rule. Again, if neither of the given rows has an output entry, the new row has no output entry either.

The dependency restriction for this rule can be relaxed in the same way as for the resolution rule.

According to the polarity strategy, we may assume that one occurrence of one of the equalities $s_i = t_i$ in $\mathcal{F}$ is negative in the tableau. We may also require that some element of S not be a variable.

This rule degenerates to paramodulation in the clausal, quantifier-free case. The completeness results of Brand [1975] apply to this rule if the skolemization, splitting, and transformation rules are included in the system, so that we can reduce our theorem to clause form. We assume the identity axiom $x = x$ is included among the assertions.

The motivation for the dependency and the no-escape restrictions of the equality substitution rule is the same as for the resolution rule.

Example:

Assume our tableau contains the two assertions

assertions	goals	outputs
if $q(a)$ *then* $\boxed{f(x,a)} = g(x)$		
$(\exists u)p(\boxed{f(u,v)}, u, v)$		

Then, by the equality substitution rule, taking $S = \{f(u,v), f(x,a)\}$ and $T = \{g(x)\}$, and $\theta = \{x \leftarrow u, v \leftarrow a\}$, we can derive the new assertion

(*if* $q(a)$ *then false*) *or* $(\exists u)p(g(u), u, a)$		

which reduces to

(*not* $q(a)$) *or* $(\exists u)p(g(u), u, a)$		

We again use the box notation to indicate the expressions to be matched.

Equivalence Substitution Rule

This rule is precisely analogous to the equality substitution rule, with equivalence playing the role of equality.

The rough schematic description of the ground version of the rule is as follows:

assertions	goals	outputs
$\mathcal{F}(S \equiv T)$		f
$\mathcal{G}(S)$		g
$\mathcal{F}(\mathit{false})$ *or* $\mathcal{G}(T)$		*if* $S \equiv T$ *then* g *else* f

The more precise description of the rule is as follows:

Rule (equivalence substitution):

assertions	goals	outputs
$\mathcal{F}$		f
$\mathcal{G}$		g
$(\mathcal{F} \blacktriangleleft \theta) \blacktriangleleft \{(P \blacktriangleleft \theta \equiv Q \blacktriangleleft \theta) \leftarrow \mathit{false}\}$ *or* $(\mathcal{G} \blacktriangleleft \theta) \triangleleft \{P \blacktriangleleft \theta \leftarrow Q \blacktriangleleft \theta\}$		*if* $P \blacktriangleleft \theta \equiv Q \blacktriangleleft \theta$ *then* $g \blacktriangleleft \theta$ *else* $f \blacktriangleleft \theta$

The restrictions for the rule are the same as for the equality substitution rule, with equivalence playing the role of equality and sentences playing the role of terms.

We assume that we have among our assertions the *reflexivity axiom* for equivalence $\mathcal{G} \equiv \mathcal{G}$, where $\mathcal{G}$ is a metavariable that can be matched against sentences.

To take full advantage of our ability to leave quantifiers intact, we include among our assertions such familiar equivalences from predicate logic as the *some-or* equivalence

$$(\exists x)[\mathcal{G} \textit{ or } \mathcal{H}] \equiv [(\exists x)\mathcal{G} \textit{ or } (\exists x)\mathcal{H}]$$

and the *all-and* equivalence

$$(\forall x)[\mathcal{G} \textit{ and } \mathcal{H}] \equiv [(\forall x)\mathcal{G} \textit{ and } (\forall x)\mathcal{H}].$$

Such equivalences are redundant in the presence of the skolemization rules, but may shorten deductions dramatically by allowing us to avoid skolemization and the removal of equivalences.

Example:

Suppose our tableau contains a goal

assertions	goals	outputs
	$r(x) \equiv \boxed{(\exists y)[p(x,y) \text{ or } (\forall z)q(y,z)]}$	

Then, by applying the equivalence substitution rule between this goal and the *some-or* equivalence,

$\boxed{(\exists x)[\mathcal{G} \text{ or } \mathcal{H}]} \equiv$ $[(\exists x)\mathcal{G} \text{ or } (\exists x)\mathcal{H}]$		

we can obtain the new goal

	false or $r(x) \equiv [(\exists y)p(x,y) \text{ or } (\exists y)(\forall z)q(y,z)]$	

which reduces to

	$r(x) \equiv [(\exists y)p(x,y) \text{ or } (\exists y)(\forall z)q(y,z)]$	

RESOLUTION AND SUBSTITUTION WITH MATCHING

The matching rules may be regarded as adding a new equality (or equivalence) to a goal when, because of a mismatch, we fail to apply the resolution rule or a substitution rule. We present first the GG-resolution rule with equality matching.

Resolution With Equality Matching

In its rough schematic form, the rule is as follows:

assertions	goals	outputs
	$\mathcal{F}(\mathcal{R}(S))$	f
	$\mathcal{G}(\mathcal{R}(T))$	g
	$S = T$ *and* $\mathcal{F}(true)$ *and* $\mathcal{G}(false)$	*if* $\mathcal{R}(S)$ *then* f *else* g

Here, we assume that S and T are distinct terms. If they were identical, we could apply the resolution rule; in this case, we add the conjunct $S = T$ as an additional condition to be proved.

The rationale for the rule is as follows: for an interpretation under which the derived goal is true, its conjunct $S = T$ is true, and $\mathcal{R}(S)$ and $\mathcal{R}(T)$ are equivalent. The justification for this rule is then the same as for the basic resolution rule, without equality matching. Before we give the precise description of the rule, let us motivate it with an example.

Example:

Suppose our tableau contains the two goals

assertions	goals	outputs
	$\mathcal{F}$: $\boxed{p(x,a,b)}^{+}$ *and* $(\boxed{p(c,z,g(z))}^{+}$ *or* $q(x))$	$f(x)$
	$\mathcal{G}$: *not* (*if* $r(y)$ *then* $\boxed{p(c,y,g(y))}^{-}$)	$g(y)$

In attempting to unify the boxed subsentences of $\mathcal{F}$ and of $\mathcal{G}$, the unification algorithm develops the substitution

$$\theta = \{x \leftarrow c,\ \ y \leftarrow a,\ \ z \leftarrow a\}$$

and then fails because the correponding terms b and $g(a)$ cannot be unified. If we somehow could establish that the mismatched terms b and $g(a)$ were equal, we could apply the resolution rule. This motivates the precise statement. We will return to this example afterwards. ∎

The precise description of the rule is as follows:

Rule (*resolution with equality matching*):

In our tableau notation, the rule is expressed as follows:

assertions	goals	outputs
	$\mathcal{F}$	f
	$\mathcal{G}$	g
	$S\blacktriangleleft\theta = T\blacktriangleleft\theta$ *and* $(\mathcal{F}\blacktriangleleft\theta)\blacktriangleleft\{P_1\blacktriangleleft\theta \leftarrow$ *true*, $\ldots, P_k\blacktriangleleft\theta \leftarrow$ *true*$\}$ *and* $(\mathcal{G}\blacktriangleleft\theta)\blacktriangleleft\{Q_1\blacktriangleleft\theta \leftarrow$ *false*, $\ldots, Q_\ell\blacktriangleleft\theta \leftarrow$ *false*$\}$	*if* $\mathcal{R}$ *then* $f\blacktriangleleft\theta$ *else* $g\blacktriangleleft\theta$

Here we assume the following:

- $P_1, P_2, \ldots, P_k$ are subsentences of $\mathcal{F}$.
- $Q_1, Q_2, \ldots, Q_\ell$ are subsentences of $\mathcal{G}$.
- $S = \{s_1, s_2, \ldots, s_k\}$ *and* $T = \{t_1, t_2, \ldots, t_\ell\}$ are sets of subterms of $P_1, \ldots, P_k, Q_1, \ldots,$ and Q_ℓ.
- $\mathcal{R}$ is a sentence and θ a most-general substitution such that
  - θ unifies S; i.e., $s_1 \blacktriangleleft \theta, s_2 \blacktriangleleft \theta, \ldots,$ and $s_m \blacktriangleleft \theta$ are identical terms, denoted by $S \blacktriangleleft \theta$.
  - θ unifies T; again $T \blacktriangleleft \theta$ denotes the unified term.
  - $S \blacktriangleleft \theta$ and $T \blacktriangleleft \theta$ are distinct terms.
  - $\mathcal{R}$ is "nearly identical" to each of the sentences $P_i \blacktriangleleft \theta$; in other words, for each i between 1 and k,

    $$(P_i \blacktriangleleft \theta) \trianglelefteq \{S \blacktriangleleft \theta \leftarrow T \blacktriangleleft \theta\} \text{ is } \mathcal{R}.$$

    That is, $\mathcal{R}$ can be obtained by replacing in $P_i \blacktriangleleft \theta$ zero, one, or more occurrences of $S \blacktriangleleft \theta$ with $T \blacktriangleleft \theta$.
  - $\mathcal{R}$ is "nearly identical" to each of the sentences $Q_j \blacktriangleleft \theta$; in other words, for each j between 1 and ℓ,

    $$(Q_j \blacktriangleleft \theta) \trianglelefteq \{S \blacktriangleleft \theta \leftarrow T \blacktriangleleft \theta\} \text{ is } \mathcal{R}.$$

- If x is any variable in $\mathcal{F}$ or in $\mathcal{G}$ that occurs within the scope of a quantifier, then θ cannot instantiate x to any term containing a bound variable of $\mathcal{F}$ or of $\mathcal{G}$.
  (*dependency restriction*)
- No bound variable of $\mathcal{F}$ or $\mathcal{G}$ may occur free in the new row.
  (*no-escape restriction*)

The discovery of the sets S and T and the substitution θ is the natural by-product of an attempt to unify the subsentences P_i and Q_j if the unification algorithm returns pairs of mismatched terms when it nearly succeeds. The rule may be generalized to the case in which there are several pairs of mismatched terms. The dependency restriction for this rule may be relaxed in the same way as for the resolution rule.

This rule degenerates to E-resolution (Morris [1969]) in the clausal case.

Example:

In our discussion prior to the statement of the rule, we considered a tableau with the two goals

assertions	goals	outputs
	$\mathcal{F}$: $\boxed{p(x,a,b)}^{+}$ *and* ($\boxed{p(c,z,g(z))}^{+}$ *or* $q(x)$)	$f(x)$
	$\mathcal{G}$: *not* (*if* $r(y)$ *then* $\boxed{p(c,y,g(y))}^{-}$)	$g(y)$

Recall that the boxed subsentences of $\mathcal{F}$ and $\mathcal{G}$ failed to unify because of the mismatched terms b and $g(a)$. However, we can still apply the resolution rule with equality matching, taking

$$\theta = \{x \leftarrow c, y \leftarrow a, z \leftarrow a\},$$
$$S = \{b\},$$
$$T = \{g(z), g(y)\},$$

and

$$\mathcal{R} = p(c,a,b),$$

to add to our tableau the new goal

	$b = g(a)$ *and* (*true and* (*true or* $q(c)$)) *and* *not* (*if* $r(a)$ *then false*)	*if* $p(c,a,b)$ *then* $f(c)$ *else* $g(a)$

which reduces under transformation to

	$b = g(a)$ *and* $r(a)$	*if* $p(c,a,b)$ *then* $f(c)$ *else* $g(a)$

∎

According to the polarity strategy, we may restrict application of the rule to cases in which, for some i, at least one occurrence of $P_i \blacktriangleleft \theta$ in $\mathcal{F} \blacktriangleleft \theta$ is positive, and at least one occurrence of $Q_j \blacktriangleleft \theta$ in $\mathcal{G} \blacktriangleleft \theta$ is negative, in the tableau.

The resolution rule with equivalence matching is identical to the rule with equality matching if we replace the equality predicate with the equivalence connective, and references to terms and subterms with sentences and subsentences, respectively.

Substitution with Equality Matching

We can add a new equality to a row upon failing to apply the equality (or equivalence) substitution rule. We present only the schematic AA-form of the equivalence substitution rule with equality matching.

assertions	goals	outputs
$\mathcal{F}(P(S) \equiv Q)$		f
$\mathcal{G}(P(T))$		g
if $S = T$ *then* $\mathcal{F}(false)$ *or* $\mathcal{G}(Q)$		*if* $P(S) \equiv Q$ *then* g *else* f

Here, if S and T were identical, we could apply the equivalence substitution rule; we therefore add the condition $S = T$ to the assertion as an antecedent. In the GG- and other forms of the rule, the condition $S = T$ is added to the goal as a conjunct.

A similar rule allows us to add a new equivalence (rather than an equality) to a row upon failing to apply the equivalence substitution rule.

Before we introduce the rules for handling special relations other than equality, let us give an extensive example involving equality and equivalence.

EQUALITY AND EQUIVALENCE: A COMPLETE EXAMPLE

In this section we present an example that employs the techniques presented so far. The example is akin to the synthesis of the Cartesian-product program, but is simplified to avoid constructing auxiliary subprograms, which requires the general induction rule, not the special case we have discussed here.

The program to be constructed appends the integer 1 onto every element of a given finite set. Our initial specification is

$$cartone(s) \Leftarrow \text{find } z \text{ such that } (\forall y)\begin{bmatrix} y \in z & \equiv \\ (\exists x)(y = (1,x) & \textit{and } x \in s) \end{bmatrix}$$

Here $(1,x)$ is the pair whose first element is 1 and whose second is x. Note that there is no input condition; the type condition $isset(s)$ is omitted.

In this derivation, we will sometimes simplify new rows automatically with true-false and other fundamental transformation rules, without presenting the intermediate results.

The initial tableau for this specification is

assertions	goals	outputs $cartone(s)$
	1. $(\forall y)^{\forall}\begin{bmatrix} y \in z & \equiv \\ (\exists x)(y = (1,x) & \textit{and } x \in s) \end{bmatrix}$	z

The Induction Hypothesis

By the induction rule, we may consider an arbitrary input set s and assume that the program $cartone(u)$ we are attempting to construct will yield an output that satisfies the given specification, provided that the input u is a set strictly less than s in some well-founded ordering $\prec_w$. Thus, we can add to our assertions the induction hypothesis

2. *if* $u \prec_w s$ *then* $(\forall y)^{\exists}\begin{bmatrix} y \in cartone(u) & \equiv \\ (\exists x)(y = (1,x) & \textit{and } x \in u) \end{bmatrix}$		

Dropping the Quantifiers

As we have indicated by annotation, the quantifier $(\forall y)$ in goal 1 is of universal force while the same quantifier in assertion 2 is of existential force. By the quantifier elimination rules, we

can replace the quantifier with a skolem function g in the goal and with a free variable y in the assertion, thereby obtaining a new goal and assertion

	3. $g(z) \in z \equiv (\exists x)\left(g(z) = \langle 1, x\rangle \text{ and } \boxed{x \in s}^{\pm} \right)$	z
4. *if* $u \prec_w s$ *then* $\left[\begin{array}{l} y \in \mathit{cartone}(u) \equiv \\ (\exists x)(y = \langle 1, x\rangle \text{ and } x \in u) \end{array}\right]$		

We may think of the skolem term $g(z)$ in goal 3 as an arbitrary element.

Note that the subexpression $x \in s$ has both polarities because it is within the scope of an equivalence.

The Base Case

We assume that we have among our assertions the *empty-set membership* axiom

not $\boxed{y \in \{\}}^{+}$		

By the resolution rule with equality matching, we can match the subsentence $y \in \{\}$ in this assertion against the subsentence $x \in s$ in goal 3, taking θ to be $\{y \leftarrow x\}$. As the polarity annotations indicate, this match is in accordance with the polarity strategy. The new row we obtain is

	5. $s = \{\}$ *and* *not not true and* $\left[\begin{array}{l} g(z) \in z \equiv \\ (\exists x)(g(z) = \langle 1, x\rangle \text{ and } \mathit{false}) \end{array}\right]$	z

which reduces (under true-false transformation) to

	6. $s = \{\}$ *and* $\boxed{\mathit{not}\ g(z) \in z}^{+}$	z

Applying the GA-resolution rule between goal 6 and the *empty-set membership* axiom

$\boxed{\mathit{not}\ y \in \{\}}^{-}$		

we obtain the goal (after transformation)

	7. $s = \{\}$	$\{\}$

Note that in this step we have instantiated the output variable z, obtaining a ground term in the output column. This row means that, in the case in which the input s is the empty set, the output can also be taken to be the empty set.

Decomposition of the Goal

Let us turn our attention back to the earlier goal 3, which was formed from the initial goal by removing a quantifier:

	$g(z) \in z \equiv$ $(\exists x)(g(z) = (1, x)$ *and* $\boxed{x \in s})$	z

We assume that we have among our assertions the *nonempty-set membership* axiom:

if not $u = \{\}$ *then* $\left(\begin{array}{l}\boxed{y \in u} \equiv \\ (y = elt(u)\ \ or\ \ y \in rest(u))\end{array}\right)^{-}$		

Here $elt(u)$ is an arbitrary element of the nonempty set u, and $rest(u)$ is the set of all the other elements of u. By the equality substitution rule, taking θ to be $\{y \leftarrow x, u \leftarrow s\}$, we can use this assertion to replace $x \in s$ in the goal with

$$x = elt(s)\ \ or\ \ x \in rest(s)$$

obtaining (after true-false transformation)

	8. $(not\ s = \{\})$ *and* $\left(\begin{array}{l}g(z) \in z \equiv \\ (\exists x)\left[\begin{array}{l}g(z) = (1, x)\ \ and \\ (x = elt(s)\ \ or\ \ x \in rest(s))\end{array}\right]\end{array}\right)$	z

Applying the equivalence substitution rule twice in succession, first to the *and-or* distributive equivalence

$\boxed{(\mathcal{F}\ and\ (\mathcal{G}\ or\ \mathcal{H}))} \equiv$ $(\mathcal{F}\ and\ \mathcal{G})\ or\ (\mathcal{F}\ and\ \mathcal{H})$		

and this goal, and then for the *some-or* equivalence

$\boxed{(\exists x)(\mathcal{G}\ or\ \mathcal{H})} \equiv$ $(\exists x)\mathcal{G}\ or\ (\exists x)\mathcal{H}$		

and the resulting goal, we obtain

	9. $(not\ s = \{\})\ and$ $\left(g(z) \in z \equiv \left[\begin{array}{l}\boxed{(\exists x)(g(z) = \langle 1, x\rangle\ and\ x = elt(s))\ or}\\ (\exists x)(g(z) = \langle 1, x\rangle\ and\ x \in rest(s))\end{array}\right]\right)$	z

By the transformation rule

$$(\exists y)(\mathcal{F}\ and\ y = t) \Rightarrow \mathcal{F} \blacktriangleleft \{y \leftarrow t\}$$

applied to the goal, taking $\theta = \{y \leftarrow x, \mathcal{F} \leftarrow (g(z) = \langle 1, x\rangle), t \leftarrow elt(s)\}$ we obtain

	10. $(not\ s = \{\})\ and$ $g(z) \in z \equiv \left[\begin{array}{l}g(z) = \langle 1, elt(s)\rangle\ or\\ \boxed{(\exists x)(g(z) = \langle 1, x\rangle\ and\ x \in rest(s))}\end{array}\right]$	z

Note that the substitution θ contains a replacement for the bound variable y; this is because we are unifying two quantified sentences.

Using the Induction Hypothesis

Recall we have assumed as our induction hypothesis (after skolemization) the assertion 4,

$if\ u \prec_w s$ $then \left[\begin{array}{l}y \in cartone(u) \equiv\\ \boxed{(\exists x)(y = \langle 1, x\rangle\ and\ x \in u)}\end{array}\right]$		

By the equivalence substitution rule we may use the equivalence of the induction hypothesis (from right to left, where $\theta = \{y \leftarrow g(z), u \leftarrow rest(s)\}$) to replace the subsentence

$$(\exists x)(g(z) = \langle 1, x\rangle\ and\ x \in rest(s))$$

of the goal with

$$g(z) \in cartone(rest(s))$$

obtaining

	11. $\left(not \begin{bmatrix} if\ rest(s) \prec_w s \\ then\ false \end{bmatrix}\right)$ *and* $(not\ s = \{\})$ *and* $g(z) \in z \equiv \begin{bmatrix} g(z) = \langle 1, elt(s)\rangle\ or \\ g(z) \in cartone(rest(s)) \end{bmatrix}$	z

which reduces (under true-false transformation) to

	12. $rest(s) \prec_w s$ *and* $(not\ s = \{\})$ *and* $g(z) \in z \equiv \boxed{\begin{array}{l} g(z) = \langle 1, elt(s)\rangle\ or \\ g(z) \in cartone(rest(s)) \end{array}}$	z

Introducing the Recursive Call

We assume that we have among our assertions the *member-insertion* axiom

$(x \in y \circ u) \equiv \boxed{(x = y\ \ or\ \ x \in u)}$		

(Here $y \circ u$ is the result of adding the element y to the set u.) By the equivalence substitution rule, we may use the axiom (from right to left) to replace the subsentence

$$g(z) = \langle 1, elt(s)\rangle\ \ or$$
$$g(z) \in cartone(rest(s))$$

with the sentence

$$g(z)\ \in\ \langle 1, elt(s)\rangle \circ cartone(rest(s)),$$

obtaining

	13. $rest(s) \prec_w s$ *and* $(not\ s = \{\})$ *and* $\boxed{g(z) \in z \equiv [\, g(z) \in \langle 1, elt(s)\rangle \circ cartone(rest(s))\,]}$	z

Finally, by GA-resolution, matching the subsentence

$$g(z) \in z \equiv g(z) \in \langle 1, elt(s)\rangle \circ cartone(rest(s))$$

against the equivalence reflexivity axiom

$\boxed{\mathcal{F} \equiv \mathcal{F}}^-$		

taking z to be $\langle 1, elt(s)\rangle \circ cartone(rest(s))$, we obtain the goal

	14. $\boxed{rest(s) \prec_w s}$ *and* $(not\ s = \{\})$	$\langle 1, elt(s)\rangle \circ$ $cartone(rest(s))$

Note that at this stage we have discovered another instantiation for the output variable z. The term, which appears as the output entry, contains a recursive call $cartone(rest(s))$. This term is a suitable output in the case that s is a nonempty set, provided we can show that the argument $rest(s)$ is strictly less than s in the ordering $\prec_w$.

Proof of Termination

We have not yet found a well-founded ordering $\prec_w$ to serve as a basis for the induction. We expect to have properties of many standard orderings among our assertions. Assume that we have the *subset-rest* axiom

if not $u = \{\}$ *then* $\boxed{rest(u) \prec_{subset} u}^-$		

where $\prec_{subset}$ is the proper subset ordering over the finite sets. By GA-resolution, we can match the subsentence

$$rest(s) \prec_w s$$

of the goal against the subsentence

$$rest(u) \prec_{subset} u$$

of the assertion, to obtain the goal

	15. *not* $\boxed{s = \{\}}^-$	$\langle 1, elt(s)\rangle \circ$ $cartone(rest(s))$

Note that in this step we have selected the well-founded ordering $\prec_w$ to be the proper subset ordering $\prec_{subset}$.

The Final Program

Recall that we have earlier developed goal 7,

	$\boxed{s = \{\}}^+$	$\{\}$

By GG-resolution between this goal and the new goal 15, we obtain the final goal

	16. *true*	*if* $s = \{\}$ *then* $\{\}$ *else* $\langle 1, elt(s)\rangle \circ$ $cartone(rest(s))$

This step accounts for the introduction of a conditional expression in the output column. The final program we extract from the proof is

$$\begin{aligned} cartone(s) \Leftarrow\ & \textit{if } s = \{\} \\ & \textit{then } \{\} \\ & \textit{else } \langle 1, elt(s)\rangle \circ \\ & \quad cartone(rest(s)) \end{aligned}$$

Synthesis of the Cartesian-Product Program

The above proof is similar to the derivation of the Cartesian product program $cart(s_1, s_2)$, which computes the Cartesian product of two finite sets s_1 and s_2. The specification for that

program is

$$cart(s_1, s_2) \Leftarrow \text{find } z \text{ such that } (\forall y)\left\{\begin{matrix} y \in z \equiv \\ (\exists x_1)(\exists x_2)\left[\begin{matrix} y = \langle x_1, x_2 \rangle \ \textit{and} \\ x_1 \in s_1 \ \textit{and} \ x_2 \in s_2 \end{matrix}\right] \end{matrix}\right\}$$

The final program we obtain is the system of two programs

$$\begin{aligned} cart(s_1, s_2) \Leftarrow\ & \textit{if } s_1 = \{\} \\ & \textit{then } \{\} \\ & \textit{else } carttwo(s_1, s_2) \cup \\ & \quad cart(rest(s_1), s_2), \end{aligned}$$

where

$$\begin{aligned} carttwo(s_1, s_2) \Leftarrow\ & \textit{if } s_2 = \{\} \\ & \textit{then } \{\} \\ & \textit{else } \langle elt(s_1), elt(s_2) \rangle \circ \\ & \quad carttwo(s_1, rest(s_2)). \end{aligned}$$

Here, $\cup$ is the set union function and $carttwo(s_1, s_2)$ is an auxiliary subprogram that computes the Cartesian product of $\{elt(s_1)\}$ and s_2. The auxiliary program appears through the use of the more general induction principle.

POLARITY WITH RESPECT TO SPECIAL RELATIONS

Equality is only one relation that has special importance in program synthesis. The inequalities $<$ and $\leq$ over the integers or reals, and the subset relation $\subseteq$ and the membership relation $\in$ over the sets, are examples of other relations that merit special treatment. In this section we extend the rules we have given for equality to apply to other relations in particular circumstances. This extension is particularly effective for transitive (ordering) relations. But first we must extend the notion of polarity, which we have introduced for subsentences, to apply to terms as well, relative to a particular relation $\prec$.

Relations and Monotonicity

Let $\prec$ be a relation. We shall say that

- $\prec$ is *irreflexive* if

$$not\ x \prec x;$$

- $\prec$ is *total* if

$$x \prec y \ \ or \ \ x = y \ \ or \ \ y \prec x;$$

- $\prec$ is *transitive* if

$$if\ (x \prec y \ \ and \ \ y \prec z)\ then \ \ x \prec z;$$

- $\prec$ is *asymmetric* if

$$not\,(x \prec y \ \ and \ \ y \prec x);$$

for all x, y, and z.

We define the *weak relation* $\preceq$ *associated with* $\prec$ by

$$x \preceq y \ \equiv \ (x \prec y \ \ or \ \ x = y).$$

We shall use $y \succ x$ and $y \succeq x$ synonymously with $x \prec y$ and $x \preceq y$, respectively.

Definition: Let f and p be a function and predicate of arity n and let j be an integer between 1 and n inclusive.

With respect to a relation $\prec$, we shall say that

- f is (*weakly*) *monotonically increasing* in its jth argument provided that

  if $x \prec y$
  then $f(z_1, \ldots, z_{j-1}, x, z_{j+1}, \ldots, z_n) \preceq f(z_1, \ldots, z_{j-1}, y, z_{j+1}, \ldots, z_n)$

- p is (*weakly*) *monotonically increasing* in its jth argument provided that

  if $x \prec y$
  then if $p(z_1, \ldots, z_{j-1}, x, z_{j+1}, \ldots, z_n)$
  then $p(z_1, \ldots, z_{j-1}, y, z_{j+1}, \ldots, z_n)$

- f is (*weakly*) *monotonically decreasing* in its jth argument provided that

  if $y \prec x$
  then $f(z_1, \ldots, z_{j-1}, x, z_{j+1}, \ldots, z_n) \preceq$
  $f(z_1, \ldots, z_{j-1}, y, z_{j+1}, \ldots, z_n)$

- p is (*weakly*) *monotonically decreasing* in its jth argument provided that

  if $y \prec x$
  then if $p(z_1, \ldots, z_{j-1}, x, z_{j+1}, \ldots, z_n)$
  then $p(z_1, \ldots, z_{j-1}, y, z_{j+1}, \ldots, z_n)$

for all x, y, and $z_1, \ldots, z_n$. ∎

Of course, some functions and predicates are neither monotonically increasing nor decreasing in some of their arguments with respect to a given relation $\prec$.

Example:

The minus function $(-)$ is monotonically increasing in its first argument with respect to the $<$ relation; i.e.,

if $x < y$
then $x - z \leq y - z$

for all integers x, y, and z. Furthermore, the minus function is monotonically decreasing in its second argument, i.e.,

if $y < x$
then $z - x \leq z - y$

for all integers x, y, and z. ∎

Example:

The member predicate $\in$ is monotonically increasing in its second argument with respect to the subset relation $\prec_{subset}$; i.e.,

if $x \prec_{subset} y$
then if $z \in x$
then $z \in y$

for all sets x and y and elements z. ∎

Note that $\in$ is neither monotonically increasing nor decreasing in its first argument with respect to $\prec_{subset}$.

Remark:

If $\prec$ is a transitive relation, then $\prec$ is monotonically increasing in its second argument with respect to $\prec$ itself; i.e.,

if $x \prec y$
then if $z \prec x$
then $z \prec y$.

Also, $\prec$ is monotonically decreasing in its first argument with respect to $\prec$ itself; i.e.,

if $y \prec x$
then if $x \prec z$
then $y \prec z$. ∎

Polarity of Terms

We are now ready to extend the notion of polarity to apply to terms, with respect to a given relation $\prec$.

Definition (polarity): The polarity of a subsentence of a given sentence or tableau, as defined in an earlier section, is also its polarity in the sentence or tableau with respect to $\prec$. For terms, we have the following additional rules:

If a subsentence $p(s_1, \ldots, s_{j-1}, t, s_{j+1}, \ldots, s_n)$ occurs in a sentence or tableau, then the polarity of t (with respect to $\prec$) is the same as the polarity of the subsentence if p is monotonically increasing in its jth argument, and the polarity of t (with respect to $\prec$) is opposite to the polarity of the subsentence if p is monotonically decreasing in its jth argument.

Similarly, if a subterm $f(s_1, \ldots, s_{j-1}, t, s_{j+1}, \ldots, s_n)$ occurs in a sentence or tableau, then the polarity of t (with respect to $\prec$) is the same as the polarity of the subterm if f is monotonically increasing in its jth argument, and the polarity of t (with respect to $\prec$) is opposite to the polarity of the subterm if f is monotonically decreasing in its jth argument.

Note that some terms may be neither positive nor negative with respect to a given relation $\prec$, and that some terms may be both positive and negative. We shall say that a term has *strict* positive or negative polarity if it has one but not both of these polarities.

Example:

In the tableau

assertions	goals	outputs
if $x + 1 \leq y$ *then* $x < y$		

- The subsentence $x + 1 \leq y$ is positive in the tableau with respect to $<$ (by the ordinary rules governing polarity).
- Therefore, the term $x + 1$ is negative in the tableau with respect to $<$ (because the $\leq$ predicate is monotonically decreasing in its first argument with respect to $<$).
- Therefore, the first occurrence of the term x is negative in the tableau with respect to $<$ (because the $+$ function is monotonically increasing in its first argument).

The notion of polarity with respect to a relation $\prec$ is important because, roughly speaking, a sentence gets "truer" as its strictly positive subterms get bigger and as its strictly negative subterms get smaller. This observation is made precise in the following proposition.

Proposition (polarity): The notion of polarity with respect to a relation $\prec$ satisfies the following two properties:

if $s \prec t$
then if $\mathcal{E}$
then $\mathcal{E} \lhd \{s^+ \leftarrow t\}$ (*positive part*)

and

if $s \succ t$
then if $\mathcal{E}$
then $\mathcal{E} \lhd \{s^- \leftarrow t\}$ (*negative part*)

for all terms s and t and sentences $\mathcal{E}$, where $\mathcal{E} \triangleleft \{s^+ \leftarrow t\}$ is the result of replacing one or more strictly positive occurrences of s in $\mathcal{E}$ with t, and $\mathcal{E} \triangleleft \{s^- \leftarrow t\}$ is the result of replacing one or more strictly negative occurrences of s in $\mathcal{E}$ with t.

The proof of the proposition is by induction on the structure of the sentence.

Example:

In the tableau

assertions	goals	outputs
if $x + 1 \leq y$ *then* $x < y$		

with respect to the relation $<$:

- The occurrence of $x + 1$ is strictly negative in the sentence $x + 1 \leq y$ (because $\leq$ is monotonically decreasing in its first argument); therefore, replacing this occurrence by something smaller makes this sentence "truer" (by the negative part of the proposition).
- The occurrence of $x + 1$ is strictly positive in the sentence *if* $x + 1 \leq y$ *then* $x < y$; therefore, replacing this occurrence by something bigger makes this sentence "truer" (by the positive part of the proposition).
- The occurrence of x is strictly negative in the sentence $x + 1 \leq y$ (because $+$ is monotonically increasing in its first argument); therefore, replacing this occurrence by something smaller makes this sentence "truer" (by the negative part of the proposition). ∎

RELATION SUBSTITUTION RULE

We are now ready to extend the equality substitution rule to an arbitrary relation $\prec$.

Small-to-Big Version

The rough schematic description of the ground version of the rule (AA-form) is as follows:

assertions	goals	outputs
$\mathcal{F}(S \prec T)$		f
$\mathcal{G}(S^-)$		g
$\mathcal{F}(\mathit{false})$ *or* $\mathcal{G}(T)$		*if* $S \prec T$ *then* g *else* f

Here $\mathcal{F}(S \prec T)$ is an assertion with an occurrence of the subsentence $S \prec T$, where S and T are terms; $\mathcal{G}(S^-)$ is an assertion with an occurrence of S which is strictly negative in the tableau with respect to $\prec$ (or, equivalently, S is strictly positive in $\mathcal{G}(S)$); and $\mathcal{G}(T)$ is the result of replacing that occurrence of S in $\mathcal{G}(S)$ with T.

The rationale for this rule is as follows. Consider an interpretation under which both given assertions are true; we seek to show that the derived assertion is also true under this interpretation. Equivalently, we show that if the derived assertion is false, then one or the other of the given assertions is also false.

Because the disjunction $\mathcal{F}(\mathit{false})$ *or* $\mathcal{G}(T)$ is false, each of its disjuncts is false. In the case in which $S \prec T$ is false, because the disjunct $\mathcal{F}(\mathit{false})$ is false, we know the given assertion $\mathcal{F}(S \prec T)$ is false; in this case, f is a suitable output. In the case in which $S \prec T$ is true, because the disjunct $\mathcal{G}(T)$ is false, and because S is strictly positive in $\mathcal{G}(S)$, we know (by the positive part of the polarity proposition) the goal $\mathcal{G}(S)$ is false; in this case, g is a suitable output. In either case, the conditional expression *if* $S \prec T$ *then* g *else* f is a suitable outpnt.

According to the polarity strategy, we may assume that some occurrence of $S \prec T$ in $\mathcal{F}(S \prec T)$ is negative in the tableau. We may also assume that S is not a free variable.

The precise version of the rule is as follows:

assertions	goals	outputs
$\mathcal{F}$		f
$\mathcal{G}$		g
$(\mathcal{F} \blacktriangleleft \theta) \blacktriangleleft \{(S \blacktriangleleft \theta \prec T \blacktriangleleft \theta) \leftarrow \mathit{false}\}$ *or* $(\mathcal{G} \blacktriangleleft \theta) \triangleleft \{S^- \blacktriangleleft \theta \leftarrow T \blacktriangleleft \theta\}$		*if* $S \blacktriangleleft \theta \prec T \blacktriangleleft \theta$ *then* $g \blacktriangleleft \theta$ *else* $f \blacktriangleleft \theta$

Here we assume that

- $S = \{s_0, s_1, \ldots, s_k\}$ and $T = \{t_1, \ldots, t_k\}$ are sets of terms such that
  - $\mathcal{F}$ contains at least one occurrence of each inequality $s_1 \prec t_1, \ldots, s_k \prec t_k$;
  - $\mathcal{G}$ contains at least one occurrence of s_0 that is strictly negative in the tableau with respect to $\prec$;
  - θ is a most-general unifier of S and of T: i.e., $s_0 \blacktriangleleft \theta, s_1 \blacktriangleleft \theta, \ldots, s_k \blacktriangleleft \theta$ are identical terms, denoted by $S \blacktriangleleft \theta$; and $t_1 \blacktriangleleft \theta, \ldots, t_k \blacktriangleleft \theta$ are identical terms, denoted by $T \blacktriangleleft \theta$; and θ is one of the most-general substitutions that make these expressions identical.
- $(\mathcal{F} \blacktriangleleft \theta) \blacktriangleleft \{(S \blacktriangleleft \theta \prec T \blacktriangleleft \theta) \leftarrow \mathit{false}\}$ denotes the result of replacing every occurrence of the subsentence $S \blacktriangleleft \theta \prec T \blacktriangleleft \theta$ in $\mathcal{F} \blacktriangleleft \theta$ with the proposition *false*.
- $(\mathcal{G} \blacktriangleleft \theta) \triangleleft \{S^- \blacktriangleleft \theta \leftarrow T \blacktriangleleft \theta\}$ denotes the result of replacing one or more (but not necessarily all) occurrences of $S \blacktriangleleft \theta$ in $\mathcal{G} \blacktriangleleft \theta$ with $T \blacktriangleleft \theta$ for which the corresponding element of S is strictly negative in the tableau with respect to $\prec$.
- If x is any variable in $\mathcal{F}$ or in $\mathcal{G}$ that occurs within the scope of a quantifier, then θ cannot instantiate x to any term containing a bound variable of $\mathcal{F}$ or $\mathcal{G}$.
  (*dependency restriction*)
- No variable that is bound in $\mathcal{F}$ or in $\mathcal{G}$ may occur free in the new row.
  (*no-escape restriction*)

The dependency restriction may be relaxed as usual. According to the polarity strategy, we may also assume that at least one occurrence of one of the inequalities $s_i \prec t_i$ in $\mathcal{F}$ is negative in the tableau. We may also require that one of the elements of S not be a free variable.

Example:

Suppose we have the two assertions

assertions	goals	outputs
$\mathcal{F}$: *if* $p(x)$ *then* $(\boxed{h(x,a)} \leq c)^-$		$f(x)$
$\mathcal{G}$: *if* $q(y)$ *then* $\boxed{h(b,y)}^- \geq 0$		$g(y)$

Note that the occurrence of $h(b,y)$ is negative in the tableau with respect to $<$. Applying the relation substitution rule, taking $\theta = \{x \leftarrow b, y \leftarrow a\}$, we can add the new assertion

(*if* $p(b)$ *then false*) *or* (*if* $q(a)$ *then* $c \geq 0$)		*if* $h(b,a) < c$ *then* $g(a)$ *else* $f(b)$

which reduces (under transformation) to

(*not* $p(b)$) *or* (*if* $q(a)$ *then* $c \geq 0$)		*if* $h(b,a) < c$ *then* $g(a)$ *else* $f(b)$

Big-to-Small Version

The preceding rule is the "small-to-big" version; it replaces instances of the "small" $S \blacktriangleleft \theta$ by a "big" $T \blacktriangleleft \theta$, in the case in which s_0 is negative in the tableau; there is also a "big-to-small" version of the rule, which applies in the case in which s_0 is strictly positive in the tableau (and therefore strictly negative in the assertion). In schematic form, the ground version of this rule is as follows:

assertions	goals	outputs
$\mathcal{F}(S \succ T)$		f
$\mathcal{G}(S^+)$		g
$\mathcal{F}(\mathit{false})$ *or* $\mathcal{G}(T)$		*if* $S \succ T$ *then* g *else* f

The rationale for this version is analogous to the rationale for the small-to-big version, and relies on the negative part of the polarity proposition.

The precise version of the rule and its restrictions are analogous to the previous small-to-big version.

TOTAL-RELATION SUBSTITUTION RULE

The above rule applies to any relation $\prec$. If the relation $\prec$ is total, there is an additional rule we can apply. (Recall that a relation $\prec$ is total if $x \prec y$ *or* $x = y$ *or* $y \prec x$, for all elements x and y.)

Small-to-Big Version

Expressed in schematic form, the ground version of the rule is as follows:

assertions	goals	outputs
$\mathcal{F}(S \prec T)$		f
$\mathcal{G}(S^+)$		g
$\mathcal{F}(true)$ *or* $\mathcal{G}(T)$		*if* $S \prec T$ *then* f *else* g

Note that in this rule we require that the occurrence of S be strictly positive in the tableau (or, equivalently, strictly negative in $\mathcal{G}(S)$) with respect to $\prec$.

The rationale for the rule is as follows. Consider an interpretation under which both given assertions are true; we seek to show that the derived assertion is also true under this interpretation. Equivalently, we show that if the derived assertion is false, then one or the other of the given assertions is false.

Because the disjunction $\mathcal{F}(true)$ *or* $\mathcal{G}(T)$ is false, each of its disjuncts is false. In the case in which $S \prec T$ is true, because the disjunct $\mathcal{F}(true)$ is false, we know the given assertion $\mathcal{F}(S \prec T)$ is false; in this case, f is a suitable output. In the case in which $S \prec T$ is false, because $\prec$ is total, we know that $S = T$ or $T \prec S$.

In the case in which $S = T$, because the disjunct $\mathcal{G}(T)$ is false, we know the given assertion $\mathcal{G}(S)$ is false; in this case, g is a suitable output. In the case in which $T \prec S$, because the disjunct $\mathcal{G}(T)$ is false, and because S is strictly negative in $\mathcal{G}(S)$, we know (by the negative part of the polarity proposition) that again the given assertion $\mathcal{G}(S)$ is false; in this case also, g is a suitable output.

In each case, the conditional expression *if* $S \prec T$ *then* f *else* g is a suitable output.

According to the polarity strategy, we need apply the rule only when some occurrence of $S \prec T$ in $\mathcal{F}(S \prec T)$ is positive in the tableau. Thus, we never need to apply both the total-ordering substitution rule and the basic ordering substitution rule in the same situation. We may also require that S not be a free variable.

We omit the precise description for the total-relation substitution rule, because it is analogous to the basic rule.

Big-to-Small Version

The preceding version is "small-to-big"; it replaces the "small" S with the "big" T in $\mathcal{G}(S)$. The corresponding "big-to-small" version of the rule, which replaces a "big" S with a "small" T, is as follows (in schematic form for the ground case):

assertions	goals	outputs
$\mathcal{F}(S \succ T)$		f
$\mathcal{G}(S^-)$		g
$\mathcal{F}(true)$ *or* $\mathcal{G}(T)$		*if* $S \succ T$ *then* f *else* g

Note here that the occurrence of S in $\mathcal{G}(S)$ to be replaced is strictly negative in the tableau, i.e., positive in $\mathcal{G}(S)$. Furthermore, according to the polarity strategy, we need apply the rule only if some occurrence of $S \succ T$ in $\mathcal{F}(S \succ T)$ is positive in the tableau. We may also require that S not be a free variable.

Example:

Suppose our tableau contains the assertion

assertions	goals	outputs
if $p(x)$ *then not* $(\boxed{f(x,d)} < a)^+$		

and the goal

	$q(y)$ *and* $\boxed{f(b,y)}^+ \geq c$	$t(x,y)$

Note that the $<$-relation is total over the integers and the boxed occurrence of $f(b,y)$ in the goal is strictly positive in the tableau with respect to $<$. Applying the AG-form of the total-relation

substitution rule small-to-big, taking θ to be

$$\{x \leftarrow b,\ y \leftarrow d\}$$

we can replace $f(b,d)$ with a in the goal to obtain the new goal

	$not \begin{pmatrix} if\ p(b) \\ then\ not\ true \end{pmatrix}$ and $\begin{pmatrix} q(d)\ and \\ a \geq c \end{pmatrix}$	$t(b,d)$

which reduces to

	$p(b)\ and\ q(d)\ and\ a \geq c$	$t(b,d)$

under true-false transformation.

Note that, because the annotated occurrence of $f(x,d) < a$ in the assertion is positive, this application of the total-relation substitution rule is in accordance with the polarity strategy.

RESOLUTION WITH RELATION MATCHING

The preceding rules adapt the equality substitution rule to arbitrary relations; in this section we adapt the resolution rule with equality matching to use an arbitrary relation, instead of equality.

As usual, we first give the schematic form of the ground version of the rule.

assertions	goals	outputs
	$\mathcal{F}(\boxed{\mathcal{R}(\mathcal{T}^+, \mathcal{S}^-)}^+)$	f
	$\mathcal{G}(\boxed{\mathcal{R}(\mathcal{S}^+, \mathcal{T}^-)}^-)$	g
	$\mathcal{S} \preceq \mathcal{T}$ *and* $\mathcal{F}(\mathit{true})$ *and* $\mathcal{G}(\mathit{false})$	*if* $\mathcal{R}(\mathcal{S}, \mathcal{T})$ *then* f *else* g

Here the notation $\boxed{\mathcal{R}(\mathcal{S}^+, \mathcal{T}^-)}$ means that $\mathcal{S}$ is a strictly positive occurrence of a term, and $\mathcal{T}$ is a strictly negative occurrence of a term, not in the tableau, but in the boxed subsentence $\mathcal{R}(\mathcal{S}, \mathcal{T})$, with respect to the relation $\prec$. Also, $\mathcal{R}(\mathcal{T}, \mathcal{S})$ is the result of replacing $\mathcal{S}$ with $\mathcal{T}$ and $\mathcal{T}$ with $\mathcal{S}$, *simultaneously*, in $\mathcal{R}(\mathcal{S}, \mathcal{T})$. We assume that $\mathcal{S}$ and $\mathcal{T}$ are distinct terms, and admit the special case in which either $\mathcal{S}$ or $\mathcal{T}$ does not actually occur in $\mathcal{R}(\mathcal{S}, \mathcal{T})$.

Note that, if this rule applies, resolution with equality matching also applies. When both rules apply, however, the rule with relation matching is preferable, as the derived goal of this rule is easier to establish than the derived goal of the equality rule. The goal for this rule has a weak inequality $\mathcal{S} \preceq \mathcal{T}$, in place of the full equality $\mathcal{S} = \mathcal{T}$ required by the equality rule.

The rationale for this rule is as follows. Consider an interpretation under which the derived goal is true; we seek to show that one or the other of the two given goals is true.

Because the conjunction $\mathcal{S} \preceq \mathcal{T}$ *and* $\mathcal{F}(\mathit{true})$ *and* $\mathcal{G}(\mathit{false})$ is true, each of its conjuncts is true. In the case in which $\mathcal{R}(\mathcal{S}, \mathcal{T})$ is false, because the conjunct $\mathcal{G}(\mathit{false})$ is true, we know the given goal $\mathcal{G}(\mathcal{R}(\mathcal{S}, \mathcal{T}))$ is also true; in this case, g is a suitable output. In the case in which $\mathcal{R}(\mathcal{S}, \mathcal{T})$ is true, because the conjunct $\mathcal{S} \preceq \mathcal{T}$ is true, and because $\mathcal{S}$ is strictly positive and $\mathcal{T}$ strictly negative in $\mathcal{R}(\mathcal{S}, \mathcal{T})$, we know (by two applications of the polarity proposition) that $\mathcal{R}(\mathcal{T}, \mathcal{S})$ is also true. Therefore, because the conjunct $\mathcal{F}(\mathit{true})$ is true, the given goal $\mathcal{F}(\mathcal{R}(\mathcal{T}, \mathcal{S}))$ is also true; in this case, f is a suitable output. In either case, the conditional expression *if* $\mathcal{R}(\mathcal{S}, \mathcal{T})$ *then* f *else* g is a suitable output.

According to the polarity strategy, we need only apply either case of the rule if $\mathcal{R}(\mathcal{T}, \mathcal{S})$ is positive in the tableau and $\mathcal{R}(\mathcal{S}, \mathcal{T})$ is negative in the tableau.

The precise form of the resolution rule with relation matching is as follows.

Rule (resolution with relation matching):

assertions	goals	outputs
	$\mathcal{F}$	f
	$\mathcal{G}$	g
	$S \blacktriangleleft \theta \preceq T \blacktriangleleft \theta$ *and* $(\mathcal{F} \blacktriangleleft \theta) \blacktriangleleft \{P_1 \blacktriangleleft \theta \leftarrow \mathit{true}, \ldots, P_k \blacktriangleleft \theta \leftarrow \mathit{true}\}$ *and* $(\mathcal{G} \blacktriangleleft \theta) \blacktriangleleft \{Q_1 \blacktriangleleft \theta \leftarrow \mathit{false}, \ldots, Q_\ell \blacktriangleleft \theta \leftarrow \mathit{false}\}$	*if* $\mathcal{R}$ *then* $f \blacktriangleleft \theta$ *else* $g \blacktriangleleft \theta$

Here we assume that

- $P_1, \ldots, P_k$ are subsentences of $\mathcal{F}$.
- $Q_1, \ldots, Q_\ell$ are subsentences of $\mathcal{G}$.
- $S = \{s_1, \ldots, s_m\}$ and $T = \{t_1, \ldots, t_n\}$ are sets of subterms of $P_1, \ldots, P_k, Q_1, \ldots,$ and Q_ℓ.
- $\mathcal{R}$ is a sentence and θ a most-general substitution such that
  - θ unifies S; i.e., $s_1 \blacktriangleleft \theta, \ldots, s_m \blacktriangleleft \theta$ are identical terms, denoted by $S \blacktriangleleft \theta$.
  - θ unifies T; again, $T \blacktriangleleft \theta$ denotes the unified term.
  - $S \blacktriangleleft \theta$ and $T \blacktriangleleft \theta$ are distinct terms.
  - $\mathcal{R}$ is "falser" than all the sentences $P_i \blacktriangleleft \theta$; in other words, for each i between 1 and k,

$$(P_i \blacktriangleleft \theta) \trianglelefteq \{(S \blacktriangleleft \theta)^- \leftarrow T \blacktriangleleft \theta, (T \blacktriangleleft \theta)^+ \leftarrow S \blacktriangleleft \theta\} \text{ is } \mathcal{R}.$$

    That is, $\mathcal{R}$ can be obtained by replacing in $P_i \blacktriangleleft \theta$ zero, one, or more strictly negative occurrences of $S \blacktriangleleft \theta$ with $T \blacktriangleleft \theta$, and zero, one, or more strictly positive occurrences of $T \blacktriangleleft \theta$ with $S \blacktriangleleft \theta$, simultaneously.

  - $\mathcal{R}$ is "truer" than all the sentences $Q_j \blacktriangleleft \theta$; in other words, for each j between 1 and ℓ,

$$(Q_j \blacktriangleleft \theta) \trianglelefteq \{(S \blacktriangleleft \theta)^+ \leftarrow T \blacktriangleleft \theta, (T \blacktriangleleft \theta)^- \leftarrow S \blacktriangleleft \theta\} \text{ is } \mathcal{R}.$$

- If x is any variable in $\mathcal{F}$ or in $\mathcal{G}$ that occurs within the scope of a quantifier, then θ cannot instantiate x to any term containing a bound variable of $\mathcal{F}$ or of $\mathcal{G}$.

  (*dependency restriction*)

- No bound variable of $\mathcal{F}$ or of $\mathcal{G}$ may occur free in the new row.

  (*no-escape restriction*)

The dependency restriction may be relaxed as usual.

The discovery of the sentence $\mathcal{R}$, the sets S and T, and the substitution θ is the by-product of an attempt to unify the subsentences P_i and Q_j if the unification algorithm returns pairs of mismatched terms and their polarities when it nearly succeeds.

Example:

Suppose our tableau contains the two goals

assertions	goals	outputs
	$\mathcal{F}$: $\boxed{c \in t^+}$ *and* $\boxed{c \in s(x)^+}^+$	$f(x)$
	$\mathcal{G}$: *not* $\boxed{y \in s(a)^+}^-$	$g(y)$

We attempt to apply GG-resolution, matching the boxed subsentences. The unification is nearly successful: if we take

θ to be $\{x \leftarrow a, y \leftarrow c\}$,

the only failure is the occurrence of the constant t in $\mathcal{F}$, which will not unify with the corresponding occurrences of $s(x)$ and $s(a)$.

The mismatched terms, however, are strictly positive, not in the tableau, but in the boxed subsentences, with respect to the subset relation $\prec_{subset}$. Therefore, we can apply the resolution rule with $\prec_{subset}$-matching, taking

P_1 to be $c \in t$,

P_2 to be $c \in s(x)$,

Q_1 to be $y \in s(a)$,

$\mathcal{R}$ to be $c \in s(a)$,

S to be $\{s(x), s(a)\}$,

T to be $\{t\}$.

Note that

$(P_1 \blacktriangleleft \theta) \trianglelefteq \{t^+ \leftarrow s(a)\}$ is $\mathcal{R}$

$P_2 \blacktriangleleft \theta$ is $\mathcal{R}$

$Q_1 \blacktriangleleft \theta$ is $\mathcal{R}$.

Therefore, we can add to our tableau the new goal

	$s(a) \preceq_{subset} t$ *and* (*true and true*) *and* *not false*	*if* $c \in s(a)$ *then* $f(a)$ *else* $g(c)$

which reduces to

	$s(a) \preceq_{subset} t$	*if* $c \in s(a)$ *then* $f(a)$ *else* $g(c)$

under true-false transformation.

The above deduction is more complex than a person would usually make in a single step. Let us show that the conclusion in this case is indeed correct.

Suppose that the new goal $s(a) \preceq_{subset} t$ is true; we would like to show that one of the given goals is true. We distinguish between two cases.

Case: $c \in s(a)$ is true.

Then, because $s(a) \preceq_{subset} t$, we know $c \in t$ is also true. Therefore, if x is taken to be a, both conjuncts of the given goal $\mathcal{F}$ are true and, hence, $f(a)$ is a suitable output.

Case: $c \in s(a)$ is false.

Then, taking y to be c, the given goal $\mathcal{G}$ is true, and, hence, $g(c)$ is a suitable output.

In either case, the conditional expression *if* $c \in s(a)$ *then* $f(a)$ *else* $g(c)$ is a suitable output. ∎

Example:

Suppose that our tableau contains the goal

assertions	goals	outputs
	$\mathcal{F}$: $\boxed{p(z, rest(s)) \text{ and } z \in s^+}^+$ and $q(s)$	z

and the assertion

$\mathcal{G}$: *if* $r_1(u)$ *then if not* $r_2(u)$ *then* $\boxed{p(f(u), u)\ \textit{and}\ f(u) \in u^+}$		

We attempt to apply GA-resolution between the goal and assertion, matching the boxed subsentences. The unification is nearly successful: if we take

$$\theta \text{ to be } \{u \leftarrow rest(s), z \leftarrow f(rest(s))\},$$

the only failure is the annotated occurrence of the variable u in $\mathcal{G}$. This variable is instantiated by θ to be $rest(s)$, and therefore will not unify with the corresponding occurrence of the constant s in $\mathcal{F}$.

The mismatched terms, however, are strictly positive, not in the tableau, but in the boxed subsentences, with respect to the subset relation $\prec_{subset}$. Therefore, we can apply the GA-resolution rule with $\prec_{subset}$ matching. The sentence $\mathcal{R}$ can be taken to be

$$p(f(rest(s)), rest(s))\ \textit{and}\ f(rest(s)) \in rest(s)$$

or

$$p(f(rest(s)), rest(s))\ \textit{and}\ f(rest(s)) \in s.$$

The new goal we obtain is

	$rest(s) \preceq_{subset} s$ *and* $\begin{pmatrix} true\ \textit{and} \\ q(s) \end{pmatrix}$ *and* $\textit{not} \begin{pmatrix} \textit{if}\ r_1(rest(s)) \\ \textit{then if not}\ r_2(rest(s)) \\ \textit{then}\ false \end{pmatrix}$	$f(rest(s))$

which reduces to

	$rest(s) \preceq_{subset} s$ *and* $q(s)$ *and* $r_1(rest(s))$ *and* $\textit{not}\ r_2(rest(s))$	$f(rest(s))$

under true-false transformation.

Note that, because the matched subsentence of the given goal is positive, and the matched subsentence of the given assertion is negative, in the tableau, the application of the rule is in accordance with the polarity strategy. ∎

EXAMPLE: THE MAXIMUM ELEMENT OF A SET

The program $max(s)$ to be constructed finds the greatest element of a finite set s of integers. Our initial specification is

$$max(s) \Leftarrow \text{find } z \text{ such that } z \in s \text{ and } (\forall y)[\text{if } y \in s \text{ then } z \geq y]$$

where $not\ s = \{\}$.

The initial tableau for this specification is

assertions	goals	outputs $max(s)$
1. $not\ s = \{\}$		
	2. $z \in s$ *and* $(\forall y)^{\forall}[\text{if } y \in s \text{ then } z \geq y]$	z

where s is a constant and z is a free variable.

The Induction Hypothesis

By the induction rule, we may consider an arbitrary input set s and assume that the program $max(u)$ to be constructed will yield an output that satisfies the given specification, provided that the input u is a set strictly less than s in some well-founded ordering $\prec_w$. Thus, we can add to our assertions the induction hypothesis

3. *if* $u \prec_w s$ *then if not* $u = \{\}$ *then* $max(u) \in u$ *and* $(\forall y)^{\exists}[\text{if } y \in u \text{ then } max(u) \geq y]$		

Dropping the Quantifiers

As we have indicated by annotation, the quantifier $(\forall y)$ in goal 2 is of universal force while the same quantifier in assertion 3 is of existential force. By the skolemization rules, we can replace the

quantifier with a skolem function g in the goal and with a free variable y in the assertion, thereby obtaining a new goal and assertion

	4. $z \in s$ *and* $\begin{bmatrix} \textit{if } \boxed{g(z) \in s} \\ \textit{then } z \geq g(z) \end{bmatrix}$	z
5. $\textit{if } u \prec_w s$ $\textit{then if not } u = \{\}$ $\textit{then } max(u) \in u \textit{ and } \begin{bmatrix} \textit{if } y \in u \\ \textit{then } max(u) \geq y \end{bmatrix}$		

We may think of the skolem term $g(z)$ in goal 4 as an arbitrary element.

Decomposing the Goal

We assume we have among our assertions the *nonempty-set membership* axiom

$\textit{if not } u = \{\}$ $\textit{then } \left(\boxed{y \in u} \equiv \begin{bmatrix} y = elt(u) \textit{ or} \\ y \in rest(u) \end{bmatrix} \right)^-$		

Here $elt(u)$ is an arbitrary element of the nonempty set u, while $rest(u)$ is the set of all the other elements of u.

By the equivalence substitution rule, we can use this assertion to replace $g(z) \in s$ in goal 4 with

$$g(z) = elt(s) \textit{ or } g(z) \in rest(s).$$

obtaining (after transformation)

	6. $\boxed{\textit{not } s = \{\}}^+$ *and* $z \in s$ *and* $\begin{bmatrix} \textit{if } \begin{pmatrix} g(z) = elt(s) \textit{ or} \\ g(z) \in rest(s) \end{pmatrix} \\ \textit{then } z \geq g(z) \end{bmatrix}$	z

(In an alternative derivation, we apply the same axiom to the subsentence $z \in s$ instead.)

The conjunct *not* $s = \{\}$ of the goal may be dropped by GA-resolution against the input condition

$\boxed{not\ s = \{\}}^-$		

(assertion 1), obtaining

	7. $z \in s$ *and* $\boxed{\begin{pmatrix} if & g(z) = elt(s) & or \\ & g(z) \in rest(s) & \end{pmatrix} \\ then\ z \geq g(z)}$	z

Applying the equivalence substitution rule between the goal and the *if-or* distributive equivalence

$\left(\begin{matrix}\boxed{if\ (\mathcal{F}\ or\ \mathcal{G})\ then\ \mathcal{H}} \equiv \\ \left[\begin{matrix}(if\ \mathcal{F}\ then\ \mathcal{H})\ and \\ (if\ \mathcal{G}\ then\ \mathcal{H})\end{matrix}\right]\end{matrix}\right)^-$		

we obtain

	8. $z \in s$ *and* $\begin{pmatrix} if & (\boxed{g(z)} = elt(s))^- & \\ then & z \geq \boxed{g(z)} & \end{pmatrix}$ *and* $\begin{pmatrix} if\ g(z) \in rest(s) \\ then\ z \geq g(z) \end{pmatrix}$	z

At this point we apply the equality substitution rule to the goal and itself (!), using the equality $g(z) = elt(s)$ to replace one instance of $g(z)$ by $elt(s)$, obtaining

	9. $z \in s$ *and* $\begin{pmatrix} if\ g(z) = elt(s) \\ then\ \boxed{z \geq elt(s)}^+ \end{pmatrix}$ *and* $\begin{pmatrix} if\ g(z) \in rest(s) \\ then\ z \geq g(z) \end{pmatrix}$	z

The other instances of $g(z)$ in the goal are allowed to remain. We shall use this goal twice in the derivation, once to give us the base cases and once to give us the recursive call.

The Base Cases

We can now apply the GA-resolution rule between goal 9 and the $\geq$-*reflexivity* axiom

$\boxed{x \geq x}^-$		

taking $\theta = \{x \leftarrow elt(s), z \leftarrow elt(s)\}$, obtaining

	10. $\boxed{elt(s) \in s}^+$ *and* $\begin{pmatrix} \textit{if } g(elt(s)) \in rest(s) \\ \textit{then } elt(s) \geq g(elt(s)) \end{pmatrix}$	$elt(s)$

Note that we have found one instantiation for the output z.

Assume that we have a *member* axiom for the element relation

if $not\ u = \{\}$ *then* $\boxed{elt(u) \in u}^-$		

We can then apply GA-resolution between the goal and the axiom, to obtain the goal

	11. $\boxed{not\ s = \{\}}^+$ *and* $\begin{pmatrix} \textit{if } g(elt(s)) \in rest(s) \\ \textit{then } elt(s) \geq g(elt(s)) \end{pmatrix}$	$elt(s)$

The conjunct $not\ s = \{\}$ can again be dropped by GA-resolution against the input condition $not\ s = \{\}$, yielding

	12. *if* $g(elt(s)) \in rest(s)$ *then* $elt(s) \geq g(elt(s))$	$elt(s)$

In other words, in the case that $elt(s)$ is greater than or equal to any arbitrary element of $rest(s)$, we know $elt(s)$ is a suitable output. We shall use this goal twice in the derivation, to provide an output expression for the program's two base cases.

Introducing the Recursive Call

Recall that we have previously developed a goal 9,

	$\boxed{\begin{pmatrix} z \in s^+ \text{ and} \\ \begin{pmatrix} \text{if } g(z) \in rest(s) \\ \text{then } z \geq g(z) \end{pmatrix} \end{pmatrix}}^+$ and $\begin{pmatrix} \text{if } g(z) = elt(s) \\ \text{then } z \geq elt(s) \end{pmatrix}$	z

(We have commuted the conjuncts in preparation for the next step.)

By GA-resolution with $\prec_{subset}$ matching to this goal and the (skolemized) induction hypothesis (assertion 5)

$\begin{array}{l} \text{if } u \prec_w s \\ \text{then if not } u = \{\} \\ \qquad \text{then } \boxed{\begin{pmatrix} max(u) \in u^+ \text{ and} \\ \begin{pmatrix} \text{if } y \in u \\ \text{then } max(u) \geq y \end{pmatrix} \end{pmatrix}} \end{array}$		

taking

$$\theta = \{u \leftarrow rest(s),\ \ z \leftarrow max(rest(s)),\ \ y \leftarrow g(max(rest(s)))\}$$

we obtain the goal

	13. $rest(s) \preceq_{subset} s$ and $\begin{pmatrix} \text{if } g(max(rest(s))) = elt(s) \\ \text{then } (max(rest(s)) \geq \boxed{elt(s)})^+ \end{pmatrix}$ and $rest(s) \prec_w s$ and not $(rest(s) = \{\})$	$max(rest(s))$

This step was possible because the annotated occurrence of u in the induction hypothesis is strictly positive, not in the tableau, but in the boxed subsentence, with respect to the proper-subset relation $\prec_{subset}$.

At this stage, through the use of the induction hypothesis, a recursive call has appeared in the output column. We shall use the induction hypothesis one more time.

Introducing the Conditional Expression

Recall that we have previously developed a goal 12,

	if $g(elt(s)) \in rest(s)$ *then* $\boxed{elt(s)}^{+} \geq g(elt(s))$	$elt(s)$

The annotated occurrence of $elt(s)$ in this goal is strictly positive with respect to the $<$-relation. Therefore, we can apply the total-relation substitution rule to goal 13 and goal 12 [bearing in mind that $max(rest(s)) \geq elt(s)$ is synonymous with $elt(s) \leq max(rest(s))$] to replace $elt(s)$ with $max(rest(s))$ in goal 12, obtaining the new goal

	14. $rest(s) \preceq_{subset} s$ *and* $rest(s) \prec_{w} s$ *and* *not* $(rest(s) = \{\})$ *and* $\boxed{\begin{array}{l}\textit{if } g(elt(s)) \in rest(s)\\ \textit{then } max(rest(s)) \geq g(elt(s))\end{array}}^{+}$	*if* $max(rest(s)) \geq elt(s)$ *then* $max(rest(s))$ *else* $elt(s)$

Note that at this stage a conditional expression has appeared in the output column.

The last conjunct of the goal can be dropped by GA-resolution against the induction hypothesis

if $u \prec_{w} s$ *then if not* $u = \{\}$ *then* $max(u) \in u$ *and* $\boxed{\begin{array}{l}\textit{if } y \in u\\ \textit{then } max(u) \geq y\end{array}}$		

this time taking

$$\theta = \{u \leftarrow rest(s),\ y \leftarrow g(elt(s))\}.$$

We obtain the new goal

	15. $\boxed{rest(s) \preceq_{subset} s}$ *and* $rest(s) \prec_{w} s$ *and* *not* $(rest(s) = \{\})$	*if* $max(rest(s)) \geq elt(s)$ *then* $max(rest(s))$ *else* $elt(s)$

This completes our use of the induction hypothesis.

Choice of Ordering

Up to now we have not chosen the well-founded ordering $\prec_w$ on which our induction is based. We assume that among our assertions we have the axioms for many orderings.

We apply the equivalence substitution rule to the definition of the weak ordering $\preceq_{subset}$,

$\left(\boxed{u \preceq_{subset} v} \equiv \right.$ $\left.(u \prec_{subset} v \ \ or \ \ u = v)\right)^{-}$		

and the goal, obtaining

	16. $\left(\boxed{rest(s) \prec_{subset} s}^{+}\right.$ $\left. or\ rest(s) = s\right)$ *and* $\boxed{rest(s) \prec_w s}^{+}$ *and* *not* $(rest(s) = \{\})$	*if* $max(rest(s)) \geq elt(s)$ *then* $max(rest(s))$ *else* $elt(s)$

By GA-resolution between the goal and the *subset* axiom

if not $u = \{\}$ *then* $\boxed{rest(u) \prec_{subset} u}^{-}$		

we reduce the goal to

	17. *not* $(rest(s) = \{\})$ *and* $\boxed{not\ s = \{\}}^{+}$	*if* $max(rest(s)) \geq elt(s)$ *then* $max(rest(s))$ *else* $elt(s)$

With this step, the well-founded ordering $\prec_w$ has been chosen to be the proper-subset ordering $\prec_{subset}$ over the finite sets.

Final Stages

The conjunct *not* $s = \{\}$ is again dropped by GA-resolution between the goal and the input condition (goal 1) *not* $s = \{\}$ obtaining

	18. *not* ($\boxed{rest(s)} = \{\}$)$^-$	*if* $max(rest(s)) \geq elt(s)$ *then* $max(rest(s))$ *else* $elt(s)$

In other words, we have determined that, in the case in which $rest(s)$ is not the empty set, a suitable output is given by the conditional expression in the output entry. Henceforth (intuitively speaking), we deal with the case in which $rest(s)$ is the empty set.

Recall that we have already developed a goal 12,

	if $g(elt(s)) \in \boxed{rest(s)}$ *then* $elt(s) \geq g(elt(s))$	$elt(s)$

By equality substitution between goal 18 and goal 12, we can replace $rest(s)$ with $\{\}$ in goal 12, obtaining

	19. *if* $\boxed{g((elt(s)) \in \{\}}^-$ *then* $elt(s) \geq g(elt(s))$	*if* $rest(s) = \{\}$ *then* $elt(s)$ *else if* $max(rest(s)) \geq elt(s)$ *then* $max(rest(s))$ *else* $elt(s)$

Note that at this stage an additional layer of conditional expression has been wrapped around the output entry.

At last, by AG-resolution between the *empty-set membership* axiom

not $\boxed{y \in \{\}}^+$		

and goal 19, we obtain the final goal

	20. *true*	*if* $rest(s) = \{\}$ *then* $elt(s)$ *else if* $max(rest(s)) \geq elt(s)$ *then* $max(rest(s))$ *else* $elt(s)$

Note that by this step an additional layer of conditional expression has been wrapped around the output entry.

Because we have obtained the goal *true* with a primitive output entry, our proof is complete. The final program is thus

$$
\begin{aligned}
max(s) \Leftarrow\ & \textit{if } rest(s) = \{\,\} \\
& \textit{then } elt(s) \\
& \textit{else if } max(rest(s)) \geq elt(s) \\
& \qquad \textit{then } max(rest(s)) \\
& \qquad \textit{else } elt(s).
\end{aligned}
$$

STRATEGY AND DISCUSSION

In this paper we have mainly disregarded the question of strategic guidance. We envision an automatic implementation of our deductive system to be governed by the following crude strategy:

- Remove all quantifiers of strict force by skolemization.
- If a rule fails to apply because of the mismatching of two bound variables or the violation of the dependency or no-escape restrictions, replace the offending bound variables by eliminating their quantifiers, after first getting rid of any surrounding equivalences by the equivalence-removal transformation rules.
- Match larger subexpressions and subterms before matching smaller ones.

In other words, we attempt to complete the proof while leaving the quantifiers and equivalences intact, but we remove them when the presence of bound variables is suspected to interfere with the proof.

The derivations included in this paper are the most concise formal derivations we have seen for these programs. For an interactive system it is clearly better to introduce high-powered rules such as ours, so that deductions will be shorter and closer to a "natural," intuitive argument. For an automatic system, however, it is not necessarily an improvement to introduce such rules, particularly if they duplicate the effects of several lower-level rules and thus lead to redundancy in the search for a proof.

However, the human implementer of an automatic system must be able to read and understand the "trace," i.e., the steps in the search for a proof. When the system is led astray, the synthesis system designer must provide heuristics to guide the search. If the steps of the trace are in terms of low-level rules, the person cannot understand it well enough to supply this heuristic guidance. Our hope is that human-oriented heuristics will be easier to discover if proofs are expressed in higher-level steps. Until we accumulate experimental evidence, we cannot be certain how efficient the implementation will be.

Acknowledgments: The authors would like to thank Ed Ashcroft, Yoni Malachi, Mark Stickel, Mabry Tyson, Pierre Wolper, and Frank Yellin for their suggestions and careful reading; and Evelyn Eldridge-Diaz for preparing the manuscript with the TEX typesetting system.

REFERENCES

Bledsoe, W. W., and L. M. Hines [July 1980], "Variable elimination and chaining in a resolution-based prover for inequalities," *Proceedings of the 5th Conference on Automated Deduction*, Les Arcs, France, pp. 70-87.

Brand, D. [Dec. 1975], "Proving theorems with the modification method," *SIAM Journal of Computing*, Vol. 4, No. 2, pp. 412-430.

Manna, Z. and R. Waldinger [Jan. 1980], "A deductive approach to program synthesis," *ACM Transactions on Programming Languages and Systems*, Vol. 2, No. 1, pp. 92-121.

Morris, J. B. [May 1969], "E-resolution: extension of resolution to include the equality relation," *Proceedings of the International Joint Conference on Artificial Intelligence*, Washington, DC, pp. 287-294.

Murray, N. V. [Jan. 1982], "Completely non-clausal theorem proving," *Artificial Intelligence*, Vol. 18, No. 1, 67-85.

Nevins, A. J. [Oct. 1974], "A human-oriented logic for automatic theorem-proving," *Journal of the ACM*, Vol. 21, No. 4, pp. 606-621.

Robinson, J. A. [Jan. 1965], "A machine-oriented logic based on the resolution principle," *Journal of the ACM*, Vol. 12, No. 1, pp. 23-41.

Slagle, J. B. [Jan. 1972], "Automatic theorem proving with built-in theories including equality, partial ordering, and sets," *Journal of the ACM*, Vol. 19, No. 1, pp. 120-135.

Wos, L. and G. Robinson [1969], "Paramodulation and theorem proving in first order theories with equality," in *Machine Intelligence 4*, (B. Meltzer and D. Michie, editors), American Elsevier, NY, pp. 135-150.

	$\mathcal{F}:$ $\boxed{\begin{array}{l} p(z, rest(s)) \ and \\ z \in s^+ \end{array}}^+$ and $q(s)$	z

and the assertion

The conjunct "*not* $s = \{\}$" can again be dropped by GR resolution against the input condition *not* $s = \{\}$, yielding

	12. *if* $g(elt(s)) \in rest(s)$ *then* $elt(s) \geq g(elt(s))$	$elt(s)$

In other words, in the case that $elt(s)$ is greater than or equal to any arbitrary element of $rest(s)$, we know $elt(s)$ is a suitable output. We shall use this goal twice in the derivation, to provide an output expression for the program's two base cases.

	$\begin{pmatrix} \textit{if } g(max(rest(s))) = elt(s) \\ \textit{then } (max(rest(s)) \geq \boxed{elt(s)})^{+} \end{pmatrix}$ *and* $rest(s) \prec_{w} s$ *and* *not* $(rest(s) = \{\})$	$max(rest(s))$

This step was possible because the annotated occurrence of u in the induction hypothesis is strictly positive, not in the tableau, but in the boxed subsentence, with respect to the proper-subset relation $\prec_{subset}$.

this time taking

$$\theta = \{u \leftarrow rest(s),\ \ y \leftarrow g(elt(s))\}.$$

We obtain the new goal

	15. $\boxed{rest(s) \preceq_{subset} s}$ *and* $rest(s) \prec_w s$ *and* *not* $(rest(s) = \{\})$	*if* $max(rest(s)) \geq elt(s)$ *then* $max(rest(s))$ *else* $elt(s)$

	17. $not\,(rest(s) = \{\})$ *and* $\boxed{not\ s = \{\}}^{+}$	*if* $max(rest(s)) \geq elt(s)$ *then* $max(rest(s))$ *else* $elt(s)$

With this step, the well-founded ordering $\prec_w$ has been chosen to be the proper-subset ordering $\prec_{subset}$ over the finite sets.

not $[y \in \{\}]$		

and goal 19, we obtain the final goal

	20. *true*	*if* $rest(s) = \{\}$ *then* $elt(s)$ *else if* $max(rest(s)) \geq elt(s)$ *then* $max(rest(s))$ *else* $elt(s)$