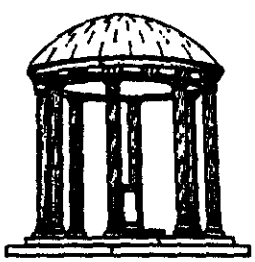
A Semantic Variant of the Modified Problem Reduction Format

TR89-001

January 1989

Xumin Nie
David A. Plaisted



The University of North Carolina at Chapel Hill
Department of Computer Science
CB#3175, Sitterson Hall
Chapel Hill, NC 27599-3175

A Semantic Variant of the Modified Problem Reduction Format *

Xumin Nie and David A. Plaisted
Department of Computer Science
University of North Carolina at Chapel Hill
Chapel Hill, North Carolina 27599
Internet: {nie, plaisted}@cs.unc.edu

Abstract

We generalize Plaisted's modified problem reduction format using arbitrary interpretations. The resulting system has a goal-subgoal structure and supports back chaining. It allows unachievable subgoals to be deleted according to some interpretation. Multiple models can be used for deleting unachievable subgoals. The resulting system is a generalization of Gelernter's system to full first order logic. We also show that our system is compatible with the set of support strategy and some useful techniques, such as guessing variable instantiations using model and the splitting technique, can be incorporated in our system. Several examples conclude the paper.

Key words and phrases. Theorem proving, automated deduction, semantics, problem reduction format, models and interpretations.

Length in words. 5000.

1. Introduction

Several systems have been proposed to use semantics or examples in theorem proving systems. Gelernter's Geometry theorem prover [Gelernter 59] is the earliest system, which proves theorems in plane geometry; It uses back chaining and represents semantic information using diagrams; unachievable subgoals are deleted. The inference system in Gelernter's system is similar to that of Prolog and only complete for Horn clauses. Reiter [Reiter 76] proposes a natural deduction system which uses arbitrary interpretations to delete unachievable subgoals. His system is incomplete. The set of support strategy [Wos 65] is a powerful and completeness preserving restriction strategy for resolution [Chang&Lee 73]. Slagle [Slagle 67] proposes the semantic resolution which is a generalization of hyper-resolution to arbitrary models; His system gives a semantics criterion for restricting which resolutions are performed. Sandford [Sandford 80] proposes Hereditary Lock resolution which combines the lock resolution [Boyer 70] and the model strategy [Luckham 70] which also gives a semantics criterion for restricting resolutions. Wang [Wang 85] proposes the semantically guided hierarchical deduction which uses only false resolvents in some model as goal clauses in the hierarchical deduction procedure.

* This work was supported in part by the National Science Foundation under grant DCR-8516243 and by the Office of Naval Research under grant N00014-86-K-0680.

The system we are to present in this paper is in the spirit of Slagle's system. The system is developed based on the observations that the modified problem reduction format in [Plaisted 88] selects its inference rules according to a particular interpretation which interprets all the positive literal to be true. The system can use an arbitrary interpretation for selecting its inference rules and the resulting inference system is still complete. The system can delete unachievable subgoals according to some interpretation and multiple models can be used to delete unachievable subgoals in some cases. The advantage of our system over Slagle's system is that our system has a natural goal-subgoal structure and supports back chaining. Our system can be regarded as a generalization of Gelernter's system to full first order logic. It is compatible with the set of support strategy and can use the interpretation to suggest instantiations for free variables. Also, a technique similar to the splitting technique in [Bledsoe 71] can be incorporated into our system.

1.1. Terminology

We will define the terminology first. A *term* is a well formed expression composed of variables, constant symbols and function symbols. An *atom* is an expression of the form $p(t_1, \cdots, t_n)$ where t_i $(1 \le i \le n)$ are terms and p is a predicate symbol. A *literal* is an atom or an atom preceded by a negation sign $\neg$. A literal is *positive* if it is an atom, *negative* if it is an atom preceded by $\neg$. For an atom A, $A = \neg\neg A$. A *clause* is a disjunction of literals. A *Horn-like clause* is of the form $L :- L_1, L_2, \cdots, L_n$, which represents the clause $L \vee \neg L_1 \vee \neg L_2 \cdots \neg L_n$, where L is called the *head literal*. $L_1, \cdots, L_n$ constitute the *clause body*. A general clause C is converted into a Horn-like clause HC as follows. One of the positive literal in C is chosen as the head literal of HC and all other literals in C are negated and put in the clause body of HC. If C contains only negative literals, we use the special literal FALSE as the head literal of HC. A clause only containing negative literals is called a *goal clause*.

Given a Horn-like clause $L :- L_1, L_2, \ldots, L_n$ where $L \neq$ FALSE, a *contrapositive* of the clause is a Horn-like clause $L' :- L_1', L_2', \ldots, L_n'$ where $L' = \neg L_i$ $(1 \le i \le n)$ and each L_j' is an L_j $(1 \le j \le n$ and $i \ne j)$ or $\neg L$ where $\neg L$ and each L_i or $\neg L_i$ appears precisely once. For a Horn-like clause $FALSE :- L_1, L_2, \ldots, L_n$, the contrapositives are Horn-like clauses of the form $L' :- L_1', L_2', \ldots, L_{n-1}'$ where $L' = \neg L_i$ $(1 \le i \le n)$ and each L_j' is an L_j $(1 \le j \le n$ and $i \ne j)$ where each L_i or $\neg L_i$ appears precisely once. The contrapositive for a unit clause L is $FALSE :- \neg L$ [Loveland 78]. For example, given a Horn-like clause $P :- Q, \neg R$, one contrapositive would be $\neg Q :- \neg P, \neg R$.

The following definitions are mainly from [Wang 85]. A *simplified first-order formula* is a quantifier-free first order formula containing no logical symbols other than $\neg$, $\wedge$ and $\vee$. Also each negation symbol $\neg$ occurring in the formula must be applied to an atom. A clause is a simplified first-order formula by definition. An *interpretation* I for a simplified first-order formula W consists of a domain D together with

1) For each n-ary predicate symbol P in W, an associated function $P_I: D^n \rightarrow \{T, F\}$.

2) For each n-ary function symbol G in W, an associated function $G_I: D^n \rightarrow D$.

The logical symbols are interpreted in I in the usual way. An *interpretation instance* of a term or a simplified first-order formula W, denoted by W', is obtained from W by substituting all variables occurring in W with some elements in D. For a simplied first-order formula W, we use $I \models_E W$ to denote the fact that there is an interpretation instance W' of W which is interpreted to be T by I; we use $I \models_U W$ to denote the fact that the interpretation I interprets all the interpretation instances W' of W to be T; We use $I \not\models_E W$ to denote $I \models_U \neg W$ and use $I \not\models_U W$ to denote $I \models_E \neg W$. We call an interpretation I a *model* for a simplified first-order formula W if $I \models_U W$. An interpretation I is a model for a set of simplified first-order formulae is it is a model for each formula in S. We use $\vdash$ and $\models$ to denote derivability and validity respectively.

1.2. Modified Problem Reduction Format

The modified problem reduction format [Plaisted 88] is an extension of Prolog-style Horn clauses logic programming to full first order logic. It uses Prolog style back chaining without contrapositives and handles non-Horn clauses using case analysis. The modified problem reduction format accepts Horn-like clauses as input. The modified problem reduction format has an inference rule per input clause plus the *assumption axioms* and the *case analysis rule*. To be specific, assume S is a set of Horn-like clauses. We obtain a set of inference rules from S for the modified problem reduction format as follows: For each Horn-like clause $L :- L_1, L_2, \ldots, L_n$ in S, we have the following *clause rule*. We call the Γ's on the left of the arrow $\rightarrow$ *assumption list*.

Clause Rules

$$\frac{[\Gamma_0 \rightarrow L_1 \Rightarrow \Gamma_1 \rightarrow L_1], [\Gamma_1 \rightarrow L_2 \Rightarrow \Gamma_2 \rightarrow L_2], \ldots, [\Gamma_{n-1} \rightarrow L_n \Rightarrow \Gamma_n \rightarrow L_n]}{\Gamma_0 \rightarrow L \Rightarrow \Gamma_n \rightarrow L}$$

We also have the *assumption axioms* and the *case analysis rule*.

Assumption Axioms

$\Gamma \rightarrow L \Rightarrow \Gamma \rightarrow L$ if $L \in \Gamma$ L is a literal.

$\Gamma \rightarrow \neg L \Rightarrow \Gamma, \neg L \rightarrow \neg L$ L is positive.

Case Analysis Rule

$$\frac{[\Gamma_0 \rightarrow L \Rightarrow \Gamma_1, \neg M \rightarrow L], [\Gamma_1, M \rightarrow L \Rightarrow \Gamma_1, M \rightarrow L] \quad |\Gamma_0| \leq |\Gamma_1|}{\Gamma_0 \rightarrow L \Rightarrow \Gamma_1 \rightarrow L}$$

The readers can refer to [Plaisted 88] for a complete discussion of the inference system. For our discussion, we only give the soundness and completeness theorems. We use $\vdash_S$ to denote derivability using the inferences rules from S.

Soundness Theorem ([Plaisted 88]): If $\vdash_S \Gamma_1 \rightarrow L \Rightarrow \Gamma_2 \rightarrow L$, then Γ_1 is a prefix of Γ_2 and $S \models \Gamma_2 \supset L$.

Completeness Theorem ([Plaisted 88]): If a set of clauses S is unsatisfiable, then $\vdash_S \rightarrow \text{FALSE} \Rightarrow \vdash_S \rightarrow \text{FALSE}$.

2. A Semantics Generalization

The modified problem reduction format can be generalized as follows. Given a set of clauses S and an interpretation M for S which interprets the literal FALSE to be T. For the moment, we will assume that the input clauses are all the contrapositives and all the clauses of the form $\text{FALSE} :- L_1, L_2, \ldots, L_n$ where $\neg L_1 \vee \neg L_2 \vee \cdots \vee \neg L_n$ is a clause in S. We have the following set of inference rules. For each Horn-like clause $L :- L_1, L_2, \ldots, L_n$, the clause rule

$$\frac{[\Gamma_0 \rightarrow L_1 \Rightarrow \Gamma_1 \rightarrow L_1], [\Gamma_1 \rightarrow L_2 \Rightarrow \Gamma_2 \rightarrow L_2], \ldots, [\Gamma_{n-1} \rightarrow L_n \Rightarrow \Gamma_n \rightarrow L_n]}{\Gamma_0 \rightarrow L \Rightarrow \Gamma_n \rightarrow L}$$

is applicable only if $M \models_E L$. The case analysis rule

$$\frac{[\Gamma_0 \rightarrow L \Rightarrow \Gamma_1, \neg N \rightarrow L], [\Gamma_1, N \rightarrow L \Rightarrow \Gamma_1, N \rightarrow L] \quad |\Gamma_0| \leq |\Gamma_1|}{\Gamma_0 \rightarrow L \Rightarrow \Gamma_1 \rightarrow L}$$

is applicable only if $M \models_E L$. The assumption axioms are

$$\Gamma \rightarrow L \Rightarrow \Gamma \rightarrow L \text{ if } L \in \Gamma \quad L \text{ is a literal.}$$

$$\Gamma \rightarrow \neg L \Rightarrow \Gamma, \neg L \rightarrow \neg L \quad L \text{ is positive.}$$

We will prove the soundness and completeness of the system. We only deal with the ground case, which can be lifted to first order logic in the usual way.

Theorem 1 (soundness): If $\vdash_S \Gamma_1 \rightarrow L \Rightarrow \Gamma_2 \rightarrow L$, then Γ_1 is a prefix of Γ_2 and $S \models \Gamma_2 \supset L$.

Proof. We can prove this by the induction of the size of the proof, i.e., the number of times the inference rules are used, making use of the length restriction in the case analysis rule (This theorem is the same as the soundness theorem in [Plaisted 88]). □

Theorem 2 (completeness) If a set of clauses S is unsatisfiable and M is an interpretation for S, then $\vdash_S \rightarrow \text{FALSE} \Rightarrow \vdash_S \rightarrow \text{FALSE}$.

Proof. Let atom(S) denote the set of atoms in S and N-atom(S) denote the set of literals $\neg L$ where $L \in$ atom(S). Let M be the set of literals in S which are interpreted true by M, then $M \subseteq$ atom(S) $\cup$ N-atom(S) $\cup$ {FALSE}. For every literal atom A ($\in$ atom(S)), either $A \in M$ or $\neg A \in M$. Specifically, $\text{FALSE} \in M$. Consider the following set of inferences rules which includes all the clauses rules from S as described above, plus the assumption axioms

$$\Gamma \rightarrow L \Rightarrow \Gamma \rightarrow L \text{ if } L \in \Gamma$$

$$\Gamma \rightarrow L \Rightarrow \Gamma, L \rightarrow L \text{ if } M \models\!\!\!\not\approx_E L$$

and the case analysis rule

$$\frac{[\Gamma_0 \rightarrow L \Rightarrow \Gamma_1, N \rightarrow L],\ [\Gamma_1, \neg N \rightarrow L \Rightarrow \Gamma_1, \neg N \rightarrow L] \quad |\Gamma_0| \leq |\Gamma_1|}{\Gamma_0 \rightarrow L \Rightarrow \Gamma_1 \rightarrow L}$$

where $M \models_E L$ and N is a literal. This system is complete from the proof of theorem 6 in [Plaisted 88], if we regard all the literals in M as positive and all other literals as negative. The completeness of our system follows if we observe that, if we use the assumption axioms and case analysis rule in our method, we are merely fixing the order of the two cases for each application of the case analysis rule. Obviously it does not matter in which order the case analysis is done. □

The modified problem reduction format can be regarded as a special case of the system above where M = atom(S) ∪ {FALSE}. However, this generalization has some advantages over the modified problem reduction format. We will discuss them in the next section.

3. Discussions

The system can be made stronger than presented above in several ways. First, consider the input clause $L :- L_1, L_2, \ldots, L_n$. M is the interpretation used. Suppose a subgoal $\Gamma_0 \rightarrow L$ is being attempted during the proof. The use of the clause rule

$$\frac{[\Gamma_0 \rightarrow L_1 \Rightarrow \Gamma_1 \rightarrow L_1],\ [\Gamma_1 \rightarrow L_2 \Rightarrow \Gamma_2 \rightarrow L_2], \ldots, [\Gamma_{n-1} \rightarrow L_n \Rightarrow \Gamma_n \rightarrow L_n]}{\Gamma_0 \rightarrow L \Rightarrow \Gamma_n \rightarrow L}$$

can be stopped if there exists a literal L_i among $L_1, L_2, \ldots, L_n$ such that L_i is a positive literal and $M \models\!\!\!\not\approx_E L_i$ and $\neg(L_i \in \Gamma_0)$, even if $M \models_E L$. This is because when $\Gamma_{i-1} \rightarrow L_i$ is attempted, the only way to solve it is to use the assumption axiom. But it is impossible to have any extra positive literal in Γ_{i-1} other than those already in Γ_0, since the assumption axiom only adds negative literals and the case analysis rule, although adding positive literals, will remove them itself. Second, a new solution $\Gamma_0 \rightarrow L$ need not be cached, in case caching is performed, if $M \models\!\!\!\not\approx_E L$ since the subgoals of the form $\Gamma \rightarrow L$ will never be attempted except using the assumption axioms. Third, consider the case analysis rule. Suppose the clause set is $S = S_0 \cup \{G\}$ where G is the goal clause. We may have several models $M_1, \cdots, M_k$ for S_0. In this event, the application of the case analysis rule can also be stopped if there exists a M_j among $M_1, \cdots, M_k$ such that $M_j \models\!\!\!\not\approx_E \Gamma_1 \rightarrow L$. This follows from the soundness theorem since $S_1 \models\!\!= \Gamma_1 \rightarrow L$. Finally, consider the clause $L :- L_1, L_2, \ldots, L_n$. If $M \models\!\!\!\not\approx_E L$, this clause will never be used in our method. Thus we need not to include such clauses in our input. In general, we only need to include those clauses whose heads are existentially satisfied by M.

Set of Support Strategy. A powerful and completeness preserving restriction strategy for resolution is the set of support strategy [Wos 65]. In this strategy, the clause set

is divided into two sets A and T where $S = A \cup T$ and $A \cap T = \varnothing$. A usually represents the axioms and the special hypotheses and is satisfiable and T represents the negation of the theorem and is called the *set of support*. A resolution operation is only allowed for two clauses if at least one of them comes from the set of support. The modified problem reduction format is not compatible with the set of support strategy because its goal clauses are always all-negative clauses which do not always come from the negation of the theorem. We will show that our system is compatible with the set of support strategy. In fact, we will show that it is only necessary to have one goal clause according to some model M (A goal clause is a clause whose clause head is the literal FALSE). This means that we need not to include all the clauses of the form FALSE:-$L_1, L_2, \ldots, L_n$ where $\neg L_1 \vee \neg L_2 \vee \cdots \vee \neg L_n$ is a clause, contrary to what we have stated earlier.

Suppose S is an unsatisfiable clause set. Assume that $S = A \cup T$ and $A \cap T = \varnothing$ and A is satisfiable, where A represents the axioms the special hypotheses and T represents the negation of the theorem. Further assume $T = \{C_1, C_2, \cdots, C_n\}$. If there exists an interpretation M such that $M \models_U A$ and $M \models_U C_i$ $(i = 1, \cdots, n\text{-}1)$ and $M \not\models_U C_n$, we can use C_n as the only goal clause according to the interpretation M. If such an interpretation does not exist, then we know $\models (A \cup \{C_1, C_2, \cdots, C_{n-1}\}) \supset C_n$. Thus $A \cup \{C_1, C_2, \cdots, C_{n-1}\} = S_1$ is unsatisfiable since S is. We can apply the same argument to S_1.

Gelernter's method. If the clause set consists of only Horn clauses and the interpretation only interprets positive literals to be true, our method is the same as Gelernter's method. Further more, no contrapositive will be needed. Thus our method is a generalization of Gelernter's method to full first order logic.

Limiting variable instantiations. When a subgoal $\Gamma \rightarrow L$ is attempted, if there are only a few, or possibly one, instantiations for some variables in $\Gamma \rightarrow L$ which make the subgoal true in the interpretation, we can either instantiate these variables in $\Gamma \rightarrow L$ in case there is only one such instantiation or limit all the future instantiations for these variables to those few.

Splitting technique. In [Bledsoe 71], the splitting technique is proposed where, to prove a theorem $A \wedge B$, A and B are proven as separate theorem. Similar technique can be used in our system. The inference rule

$$\frac{\Gamma, \neg L \rightarrow L}{\Gamma \rightarrow L}$$

is sound since it is a special case of the case analysis rule

$$\frac{[\Gamma \rightarrow L \Rightarrow \Gamma, L \rightarrow L] \quad [\Gamma, \neg L \rightarrow L \Rightarrow \Gamma, \neg L \rightarrow L]}{\Gamma \rightarrow L \Rightarrow \Gamma \rightarrow L}$$

To understand the motivation behind it, we note a situation that arises frequently in problem solving where a problem can be decomposed into several subproblems. The splitting technique as presented has some advantages. First, we can make use of the unit fact which is the negation of the subgoal using assumption axiom. This can be regarded

as a limited use of the ancestry resolution as in Model Elimination [Loveland 69] or SL-Resolution [Kowalski&Kuehner 71]. Second, the search can be better directed since we can start a new search on the subproblems, due to the support for back chaining provided by our system. This technique is powerful only if we can effectively identify the important subproblems and is admittedly difficult to control. Some heuristics or user guidance are usually needed.

Contrapositives. The Prolog-style extension to full first order logic (non-Horn clauses) using the Model Elimination procedure [Loveland 69] requires the use of all the contrapositives if the clause set is non-Horn [Stickel 86]. The modified problem reduction format, on the other hand, does not need any contrapositive at all. Using contrapositives sometimes costs efficiency since it effectively increases the number of clauses. More importantly, though, it can lead to unnatural search behavior and makes the search process difficult to control. See [Plaisted 88] for some examples. However, our experience has shown that, for non-Horn problems specially, some contrapositives can significantly help to improve the efficiency of the prover based on the modified problem reduction format.

The problem is to decide which contrapositives to use. As we have noted, given an interpretation M for a set of clause S, we only need to include the Horn-like clauses whose clause head is existentially satisfied by M. Thus our system brings a solution for this problem by making use of problem domain knowledge represented in an interpretation. Essentially, the input clauses in our system represent semantically provable paths in the search space. This point will become clear in the discussion on interpretations.

Interpretations. As we have noted, given a set of clauses S and an interpretation M for S, we only need to have, as input clauses, the clauses whose clauses heads are existentially satisfied by M. Now the question is how to design a suitable interpretation for a given theorem. This is not a trivial matter. It is difficult to automate since problem domain knowledge is usually required and it is hard to give a precise description of what is a suitable interpretation. The difficulty for a human to design an interpretation lies in the interpretation of the skolem functions [Wang 85].

A method for designing interpretations for a set of clauses is proposed in [Wang 85]. This method is a general method and can be slightly modified to select the input clauses for our system. We will briefly discuss this method and the modification. The basic idea of Wang's method is to put together all the clauses containing the same uninterpreted symbol, often skolem function symbols, and use some interpretation rules to interpret the uninterpreted symbol. We will briefly present Wang's method below.

Given a natural interpretation I for a theorem and the natural interpretations of the function symbols and predicate symbols, we need to interpret the uninterpreted symbols. We call a simplified first-order formula an *interpretation normal form* (INF) if it is in the following form:

$$L_1 \vee \cdots \vee L_k \vee [C_1 \wedge \cdots \wedge C_h]$$

where L_i's are literals and C_i's are clauses. Note that, corresponding to each INF $L_1 \vee \cdots \vee L_k \vee [C_1 \wedge \cdots \wedge C_h]$, there is an equivalent set of clauses $L_1 \vee \cdots \vee L_k \vee C_1$, $L_1 \vee \cdots \vee L_k \vee C_2$, ..., $L_1 \vee \cdots \vee L_k \vee C_h$. There are two special cases of INF: a clause $L_1 \vee \cdots \vee L_k$ is in INF form where $h = 1$ and $C_1 = \Box$ and a formula $C_1 \wedge \cdots \wedge C_h$ is in INF form where $k = 1$ and $L_1 = \Box$.

There are two *interpretation rules* for a formula in INF form. These two interpretation rules can be used to select all the necessary contrapositives. For each interpretation instance of an INF form

$$L_1 \vee \cdots \vee L_k \vee [C_1 \wedge \cdots \wedge C_h]$$

R1 if $I \models_E [\neg L_1 \wedge \cdots \wedge \neg L_k]$, then for each i $(1 \le i \le h)$, $I \models_U C_i$, except when C_i is the negation (or part of) of the theorem.

R2 if $I \models_U [L_1 \vee \cdots \vee L_k]$, then there should be at most one i $(1 \le i \le h)$, $I \models_E \neg C_i$.

Let W be $L_1 \vee \cdots \vee L_k \vee [C_1 \wedge \cdots \wedge C_h]$. Rule R1 states that, if $I \models_E [\neg L_1 \wedge \cdots \wedge \neg L_k]$, then for each i $(1 \le i \le h)$, $I \models_U C_i$, except when C_i is the negation (or part of) of the theorem. This is simply stating that the interpretation should satisfy all axioms and hypotheses of the theorem. Note that each C_i $(1 \le i \le h)$ is a clause. Let some C_i be the clause $N_1 \vee N_2 \vee \cdots \vee N_n$. Each N_j $(1 \le j \le n)$ would be the head of some Horn-like clause. However, if for some N_s $(1 \le s \le n)$, $I \not\models_E N_s$, we need not to have the Horn-like clause whose head is N_s for the simple reason that the clause would never be used if it were in the input. If for some j, $I \not\models_E C_j$, we will have a goal clause FALSE :- $\neg L_1$, ..., $\neg L_k$, $\neg C_j$, with a little misuse of notation.

Let W be $L_1 \vee \cdots \vee L_k \vee [C_1 \wedge \cdots \wedge C_h]$. Rule R2 states that, if $I \models_U [L_1 \vee \cdots \vee L_k]$, then there should be at most one i $(1 \le i \le h)$, $I \models_E \neg C_i$. Each L_i $(1 \le i \le h)$ would be the clause head of some Horn-like clauses. For the same reason, we need not to have those Horn-like clauses whose head is L_s $(1 \le s \le h)$ where $I \not\models_E L_s$. If $I \models_E L_m$ $(1 \le m \le h)$, we would have, again, with a little misuse of notation, the following Horn-like clauses

L_m :- $\neg L_1$, ..., $\neg L_k$, $\neg C_1$.
L_m :- $\neg L_1$, ..., $\neg L_k$, $\neg C_2$.
...
L_m :- $\neg L_1$, ..., $\neg L_k$, $\neg C_h$.

Consider when it comes to proving L_m. It is clear now that rule R2 states that there should be only one rule for proving L_m since there is at most one C_j such that $I \models_E \neg C_j$.

4. Some Examples

It requires trivial effort to implement our system, using the existing implementation of the modified problem reduction format [Plaisted 88]. We will discuss several examples and show the experimental results.

Intermediate Value Theorem (IMV). If a function f is continuous in a real closed interval [a, b], where $f(a) \leq 0$ and $f(b) \geq 0$, then $\exists x\ [(a \leq x) \wedge (x \leq b) \wedge (f(x) = 0)]$. The input clauses of IMV can be found in [Wang&Bledsoe 87]. Further discussion on this problem can be found in [Ballantyne&Bledsoe 82, Bledsoe 82]. This problem is interesting for us because several techniques contribute to solving it.

We have designed an interpretation for IMV using the two interpretation rules. At the same time, we select the necessary contrapositives based on the resulting interpretation. Some contrapositives are determined to be unnecessary. For example, one of the clauses is

$$p(f(x),0) \vee \neg p(a,x) \vee \neg p(x,b) \vee \neg p(x,h(x))$$

where $p(x, y) \equiv x \leq y$ and h is a skolem function. Note that any reasonable interpretation will not interpret $\neg p(a,x)$ or $\neg p(x,b)$ to be T. Thus the Horn-like clauses with $\neg p(a,x)$ or $\neg p(x,b)$ being heads are unnecessary.

The top-level goal for IMV is $\exists x\ [f(x) = 0]$, which is instantiated to $f(l) = 0$ using the interpretation. Using the predicate p, we get two subgoals p(f(l), 0) and p(0, f(l)). The splitting technique is used to solve these two subgoals separately, much like a human would do for this problem. The ground subgoals which are interpreted to be F in the interpretation are deleted. We only deal with ground subgoals for the sake of simplicity.

Our prover fails to obtain a proof for IMV without the contrapositives. With the contrapositives and without using the splitting technique, a prover takes about 15,000 seconds on a SUN 3/60 workstation using ALS-Prolog (Version 0.60) from Applied Logic Systems. More than 45,000 inferences are performed. The prover is able to obtain a proof in about 200 seconds using the splitting technique and the contrapositives. However, the prover obtains a proof in about 300 seconds using the splitting technique and the contrapostives, if we delete the ground subgoals interpreted to F in the interpretation.

Schubert's Statement [Walther 84]. This problem has been the subject of much study. We have done several experiments on it. The prover gets a proof for this problem in 748 seconds, without any contrapositive, on a SUN 3/60 workstation using ALS-Prolog. 8921 inferences are performed. With all the contrapositives, the prover takes 730 seconds to get a proof. 7939 inferences are performed. It does not makes a big difference to use an interpretation to delete false subgoals for some reason. We would like to point out that the proof obtained using contrapositives seems to be much smaller than the one without contrapositives.

Attaining Minimum (or Maximum) Theorem (AM8). A continuous function f in a closed real interval [a, b] attains its minimum (or maximum) in this interval [Wang&Bledsoe 87]. Some contrapositives are added when we design an interpretation for it. Without the contrapositives, the prover can not obtain a proof. The prover finds a proof in about 2 hours with the contrapositives. A look at the proof tells why. The added contrapositives reduce the number of case analyses by solving some negative subgoals,

quite a few of them in this problem, directly.

5. Conclusion

We have presented a generalization of the modified problem reduction format. The resulting system supports back chaining and is compatible with the set of support strategy. It allows the deletion of the false subgoals in some interpretation. What is most interesting about our system is probably that it provides an answer to the problem about how contrapositives are handled in similar systems. We need to do more research on the effect of the deletion of unachievable subgoals in our system. We have experienced several occasions where it has adverse effect on the performance of the prover to delete false subgoals in some interpretation.

References

[Ballantyne&Bledsoe 82] Ballantyne, A. and W.W. Bledsoe, "On Generating and Using Examples in Proof Discovery", *Machine Intelligence*, Vol. 10, pp. 3-39, Harwood, Chichester 1982.

[Bledsoe 71] Bledsoe, W.W., "Splitting and Reduction Heuristics in Automatic Theorem Proving", *Artificial Intelligence*, Vol. 2, pp. 55-77, North-Holland Publishing Company, 1971.

[Bledsoe 77] Bledsoe, W.W., "Non-resolution Theorem Proving", *Artificial Intelligence*, Vol. 9, pp. 1-35, North-Holland Publishing Company, 1977.

[Bledsoe 82] Bledsoe, W.W., "Using Examples to Generate Instantiations for Set Variables", Technical Report No. ATP-67, Department of Computer Science, University of Texas at Austin, July 1982.

[Boyer 70] Boyer, R., "Locking: A Restriction of Resolution", Ph.D. dissertation, University of Texas at Austin, Austin, TX, 1970.

[Chang&Lee 73] Chang, C. and R. Lee, "Symbolic Logic and Mechanical Theorem Proving", Academic Press, New York, 1973.

[Gelernter 59] Gelernter, H., "Realization of a Geometry Theorem-Proving Machine", *Proc. ICIP*, pp. 273-282, Paris UNESCO House, 1959.

[Kowalski&Kuehner 71] Kowalski, R. and D. Kuehner, "Linear Resolution with Selection Function", *Artificial Intelligence*, Vol. 2, pp. 227-260, 1971.

[Loveland 69] Loveland, D.W., "A Simplified Format for the Model Elimination Theorem-Proving Procedure", *J. ACM*, Vol. 16, No. 3, pp. 349-363, July 1969.

[Luckham 70] Luckham, D., "Refinement Theorems in Resolution Theory", *Proc. IRIA*

Symp. Automatic Demonstration, Versailles, France, 1968, Springer-Verlag, New York, pp. 163-190, 1970.

[McCharen&al 76] McCharen, J.D., R.A. Overbeek and L.A. Wos, "Problems and Experiments for and with Automated Theorem-Proving Programs", *IEEE Transactions on Computers*, Vol. C-25, No. 8, August 1976.

[Plaisted 88] Plaisted, D.A., "Non-Horn Clause Logic Programming Without Contrapositives", *Journal of Automated Reasoning*, Vol. 4, Nov. 3, September 1988.

[Reiter 76] Reiter, R., "A Semantically Guided Deductive System for Automatic Theorem Proving", *IEEE Transaction on Computers*, Vol. C-25, No. 4, pp. 328-334, April 1976.

[Sandford 80] Sandford, D.M., "Using Sophisticated Models in Resolution Theorem Proving", Lecture Notes in Computer Science, No. 90, G. Goos and J. Hartmanis eds, Springer-Verlag, 1980.

[Slagle 67] Slagle, J.R., "Automatic Theorem Proving with Renamable and Semantics Resolution", *JACM*, Vol. 14, No. 4, pp. 687-697, October 1967.

[Stickel 86] Stickel, M.E., "A PROLOG Technology Theorem Prover: Implementation by an Extended PROLOG Compiler", *Proc. of IJCAI*, pp. 573-587, Oxford, England, July 1986.

[Walther 84] Walther, C., "A Mechanical Solution of Schubert's Steamroller by Many-sorted Resolution", *Proc. 8th AAAI*, pp. 30-334, 1984.

[Wang 85] Wang, T.C., "Designing Examples for Semantically Guided Hierarchical Deduction", *Proc. of IJCAI*, pp. 1201-1207, 1985.

[Wang&Bledsoe 87] Wang, T.C. and W.W. Bledsoe, "Hierarchical Deduction", *Journal of Automated Reasoning*, Vol. 3, No. 1, 1987.

[Wos 65] Wos, L.T., "Efficiency and Completeness of the Set of Support Strategy in Theorem Proving", *J. of ACM*, Vol. 12, No. 4, October 1965.