A Complete Deduction System for Reasoning with Temporary Assumptions

Pierre Ostier
Institut IMAG, Laboratoire Logiciels Systèmes Réseaux,
B.P. 72,
F-38402 Saint Martin d'Hères - cedex, France,
e-mail : Pierre.Ostier@imag.fr

October 1, 1997

Abstract

When facing a failure of the current goal or reaching a depth limit, most goal directed inference engines seriously lack capabilities to dynamically make assumptions and to pursue deductions. This paper describes an inference system for first order logic that includes those features. The novel contribution is to develop a set of sound and complete rules that mimic human-like proofs and that is capable to give useful information on missing hypotheses in case the search of a proof failed.

1 Introduction

The proposed system, called Hypotheses Domains based System (HDS), is based on the concept of hypotheses domains. These domains associated with case analyses simulate hypothetical reasoning. When a formula is not a theorem useful information on missing hypotheses is provided by the system. But, in addition to this, HDS is also capable of giving understandable proofs when there is no missing hypothesis.

The system, inspired from [4], meets all the requirements for a complete and sound theorem prover. This paper concentrates on the definition of the inference system (Section 2). An example is given in section 3. The section 4 is devoted to a comparison with the MPRF system of Plaisted.

2 Hypotheses Domains Inference System

The notations used are the classical ones [1].

The **facts** are formulas which express the user's problem. The class of facts accepted by HDS is a superclass of Horn clauses. They are formulas of the form $L_1 \dots L_n \Rightarrow L$ where L is a positive literal and the L_i's are positive or negative literals. Notice that double negation is not valid syntactically. Semantically, the notation $L_1 \dots L_n \Rightarrow L$ has the same meaning as $\forall x_1, \dots, \forall x_k (L_1 \ \& \ \dots \ \& \ L_n \Rightarrow L)$ where & denotes the conjunction, and where $\{x_1, \dots, x_k\}$ is the set of variables appearing in litterals L_j.

A particular literal, F, is used to express falsity. It is a positive literal representing a contradiction and it allows the user to express facts corresponding to clauses $L_1 \vee \dots \vee L_n$ where all the L_i's are negative. For example : $\neg A \vee \neg B \vee \neg C$ is rewritten as $A \ B \ C \Rightarrow F$.

$\bar{L}$ denotes the opposite of L.

A **node** is a quadruple $<S, L, D_h, D_s>$ where S is a set of facts, L is a literal, D_h is a set of clauses called inherited hypotheses domain. D_s, called synthetized hypothesis is a clause denoting the missing hypothesis. Semantically, the meaning of $<S, L, D_h, L_1 \vee \ldots L_n>$ is $S, D_h, \bar{L}_1, \ldots, \bar{L}_n \models L$. The facts in S are connected conjunctively. D_h is a set of clauses implicitly linked by conjunctions.

Let $\mathcal{C}_1$ and $\mathcal{C}_2$ being two clauses. The notation $\mathcal{C}_1 \bigvee \mathcal{C}_2$ stands for the least clause (for the subsumption) that is subsumed by $\mathcal{C}_1$ and by $\mathcal{C}_2$.

Let D_h being a set of clauses and D_s being a clause. $D'_h = D_h \bigcup \{D_s\}$ stands for the least set of clauses such that D_s is subsumed by D'_h and such that every clause of D_h is subsumed by a clause of D'_h.

The definition of HDS is simple : only four inferences rules are necessary, expressing two concepts, i.e., subgoaling decomposition and case analyses. The rules are presented in Gentzen's form [1]. To illustrate rule usage, the following set S of facts is used : $S = \{A \quad B \Rightarrow C; \neg A \Rightarrow C\}$. The goal is C.

Subgoal decomposition rule (rule $R1$) :

For every fact $L_1 \ldots L_n \Rightarrow L$ of S, the following rule holds :

$$\frac{<S, L_1, D_h, D_{s_1}> \ldots <S, L_n, D_h, D_{s_n}>}{<S, L, D_h, \bigcup_{i=1}^{n} (D_{s_i})>} \qquad (R1)$$

For the above set of facts S, $R1$ becomes :

$$\frac{<S, A, \{A\}, \Box> qquadseqbSB\{A\}\neg B}{<S, C, \{A\}, \neg B>}$$

Therefore, $D_{s_1} = \Box$ and $D_{s_2} = \neg B$. Informally, $(R1)$ states that if there is a fact $A \quad B \Rightarrow C$, the missing hypotheses in proving C is the union of the missing hypotheses in proving A and in proving B. This rule corresponds to the classical subgoaling decomposition.

Hypotheses domain analysis axiom (rule $R2$) :

$$\frac{}{<S, L, D_h \bigcup \{L \vee L_1 \vee \ldots \vee L_s\}, L_1 \vee \ldots \vee L_s>} \qquad (R2)$$

For the above set of facts S, the left hand side of the antecedent part of the example is $<S, A, \{A\}, \Box>$, obtainable by $(R2)$ since A is known ; $R2$ becomes :

$$\frac{}{<S, A, \{A\}, \Box>}$$

Case analysis rule (rule $R3$) :

$$\frac{<S, L, D_h, D_{s_1}> \qquad <S, L, D_h \bigcup \{D_{s_1}\}, D_{s_2}>}{<S, L, D_h, D_{s_2}>} \qquad (R3)$$

For the above set of facts S, rule $R3$ becomes :

$$\frac{<S, C, \{\}, A> \qquad <S, C, \{A\}, \neg B>}{<S, C, \{\}, \neg B>}$$

Informally, the upper-left-hand side of the rule expresses that C holds in the case $\neg A$ holds, and that we do not know the validity of C when A holds. The upper-right-hand side of the rule expresses the proof of C in this case, that is when A holds. The final resulting domain, namely $\neg B$, expresses the cases in which the validity of C is not established.

Hypotheses domain extension axiom (rule $R4$) :

$$\overline{<S, L, D_h, \bar{L}>} \qquad (R4)$$

For the above set of facts S, an instance of the $R4$ rule is

$$\overline{<S, B, \{A\}, \neg B>}$$

Note how both the rules and the contained operations ($\bigvee$ and $\bigcup$) are simple.

This system is proved consistent. It is also proved refutationally complete in the following sens : If S is an inconsistent set of facts, then $<S, F, \{\}, \Box>$ is derivable.

The HDS system provides natural proofs for the conjunction of two reasons. The first reason is that the two main rules (i.e. $(R1)$ and $(R3)$), associated with a concept of hypotheses domains easy to understand, mimic two human reasoning schemes : the goal-to-subgoal reasoning and the case analysis reasoning. The second reason is that the HDS rules respect the form of the user's facts : no contrapositives are used.

3 Examples

We provide two example. The first one shows how $(R3)$ can be used to do case-analysis reasoning. The second one shows that$(R3)$ can also be used to produce missing hypotheses.

example 1.

Let us consider the set of facts :$S = \{ \ \Rightarrow T(a) \ , \ P(x)T(x) \Rightarrow Q(x) \ , \ \neg P(x) \Rightarrow R(x) \ , \ Q(x) \Rightarrow U(x) \ , \ R(x) \Rightarrow U(x)\}$ where a is a constant.

The following is a derivation of $<S, U(a), \{\}, \Box>$:

$$\dfrac{\dfrac{\dfrac{\overline{<S,P(a),\{\},\neg P(a)>}\,(R4) \quad \overline{<S,T(a),\{\},\Box>}\,(R1)}{<S,Q(a),\{\},\neg P(a)>}(R1)}{<S,U(a),\{\},\neg P(a)>}(R1) \qquad \dfrac{\dfrac{\overline{<S,\neg P(a),\{\neg P(a)\},\Box>}\,(R2)}{<S,R(a),\{\neg P(a)\},\Box>}(R1)}{<S,U(a),\{\neg P(a)\},\Box>}(R1)}{<S,U(a),\{\},\Box>}(R3)$$

Rule $(R3)$ is used in the above derivation to do case analysis reasoning. But this rule can be use to do hypothetical reasoning, such as in the example below.

example 2.

Now, let us consider S' constructed from S by removing $\Rightarrow T(a)$. The following is a derivation of $<S', U(x), \{\}, \neg T(x)>$:

$$\dfrac{\dfrac{\dfrac{\overline{<S',P(x),\{\},\neg P(x)>}\,(R4) \quad \overline{<S',T(x),\{\},\neg T(x)>}\,(R4)}{<S',Q(x),\{\},\neg P(x)\vee\neg T(x)>}(R1)}{<S',U(x),\{\},\neg P(x)\vee\neg T(x)>}(R1) \qquad \dfrac{\dfrac{\overline{<S',\neg P(x),\{\neg P(x)\vee\neg T(x)\},\neg T(x)>}\,(R2)}{<S',R(x),\{\neg P(x)\vee\neg T(x)\},\neg T(x)>}(R1)}{<S',U(x),\{\neg P(x)\vee\neg T(x)\},\neg T(x)>}(R1)}{<S',U(x),\{\},\neg T(x)>}(R3)$$

This proof expresses that "when $T(x)$ holds, $U(x)$ holds".

4 Comparison with Modified Problem Reduction Format

Many researchers have dealt with negation in logic programming. Their work has appeared in InH-Prolog [3], Selected Linear Without contrapositive Variants [5], Prolog Technology Theorem Prover [7], and Modified Problem Reduction Format (MPRF) [6] are related to our work. MPRF appears to be the most closely related system to ours in terms of deduction principles, and we chose to compare the two systems.

MPRF and our inference system are similar by the fact they both have an explicit case analysis rule. MPRF system is made efficient without the loss of completeness by distinguishing negative literals from positive ones: the case analysis rule is only applied to negative literals. The approach of building a set of necessary literals to achieve a proof is potentially present in MPRF system, but is not exploited.

A first difference between the present work and MPRF is that the latter is not able to extract missing hypotheses from a set of facts, as our system does.

A second difference is that D_h is a set of clauses (a conjunction of disjunctions). This is a more general and structured information than the MPRF sets of literals, thus making proofs more "natural". For example, if S is the set of hypotheses $\{\neg\ A\ \neg\ B\ \Rightarrow\ F\ ,\ \neg\ C\ \neg\ D\ \Rightarrow\ F\ ,\ A\ C\ \Rightarrow\ F\}$, $<S, F, \{\}, B \vee D>$ can be established, meaning that "$\neg B \wedge \neg D$" is an hypothesis to be added to S to make S unsatisfiable. This problem cannot be formulated in Plaisted's system.

The proof in HDS is :

$$\frac{\mathcal{D} \qquad \mathcal{D}'}{<S, F, \{\}, \{B \vee D\}>}(R3)$$

where $\mathcal{D}$ is

$$\frac{\dfrac{}{<S,\neg A,\{\},\{A\}>}(R4) \qquad \dfrac{}{<S,\neg B,\{\},\{B\}>}(R4)}{<S, F, \{\}, \{A \vee B\}>}(R1)$$

and $\mathcal{D}'$ is

$$\frac{\dfrac{\dfrac{}{<S,\neg C,\{A\vee B\},\{C\}>}(R4) \qquad \dfrac{}{<S,\neg D,\{A\vee B\},\{D\}>}(R4)}{<S,F,\{A\vee B\},\{C\vee D\}>}(R1) \qquad \dfrac{\dfrac{}{<S,A,\{A\vee B,C\vee D\},\{B\}>}(R2) \qquad \dfrac{}{<S,C,\{A\vee B,C\vee D\},\{D\}>}(R2)}{<S,F,\{A\vee B,C\vee D\},\{B\vee D\}>}(R1)}{< S, F, \{A \vee B\}, \{B \vee D\} >}(R3)$$

Note, in $\mathcal{D}'$, how $<S, F, \{A \vee B, C \vee D\}, \{B \vee D\}>$ is derived. Semantically, this goal correspond to the proof of falsity in the case $A \vee B$ and $C \vee D$ hold. The proof uses two rule (R_2) applications that use hypotheses originating from two different proofs. The resulting hypootheses B and D are then composed together to constitute $B \vee D$.

Plaisted's system does not provide such an answer because it is not able to "factorize" case analyses in such a manner.

5 Conclusion

The proposed approach allows the definition of a simple and explainable inference system for first order logic that is based only on two concepts : subgoaling decomposition rule and explicit case analysis rule. A set of additional hypotheses is associated to each goal (the D_h hypotheses domain). Those temporary hypotheses can be used to solve the goal.

It results in the building of an hypothesis (the D_s hypothesis). This hypothesis represent the cases on which the attempt to prove the goal failed. The syntactic operations on the hypotheses domains are simple and easy to implement. The proposed inference system has the advantage of providing natural proofs and information on missing hypotheses when a proof fails. The soundness and refutational completeness of this system have been established.

A prototype using those inference rules has been written in Prolog and tested in the application domain of elementary geometry. A model (a numerical representation of a geometry construction) is used to guide the search proof. Our approach will be included in a multi-paradigm geometric constraint solver.

References

[1] Fitting M., First-Order Logic and Automated Theorem proving, Texts and Monographs in Computer Science, Springer-Verlag, 1990.

[2] Loveland Donald W., A Simplified Format for the Model Elimination Procedure, Journal of the ACM, 16.3, p. 349-363, 1969.

[3] Loveland Donald W., Near-Horn Prolog and Beyond, Journal of Automated Reasoning 4, p. 69-100, 1988.

[4] Ouabdesselam F., Reasoning with Negation and Explaining Various Types of Answers, Cognitiva 85, Paris, 1985.

[5] Pereira Luis Moniz, Caires Luis & Alferes José, SLWV - A Theorem Prover for Logic Programming, ELP'92, LNCS 660, Bologna, 1992.

[6] Plaisted David A., Non-Horn Clause Logic Programming Without Contrapositives, Journal of Automated Reasoning 4, p. 287-325, 1988.

[7] Stickel Mark E., A Prolog Technology Theorem Prover : Implementation by an Extended Prolog Compiler, Journal of Automated Reasoning 4, p. 353-380, 1988.