Solving Practical Reasoning Poblems with Extended Disjunctive Logic Programming

Heinrich Herre * Gerd Wagner †

Abstract We present a definition of stable generated models for extended generalized logic programs (EGLP) which a) subsumes the definition of the answer set semantics for extended normal logic programs [GL91]; and b) does not refer to negation-as-failure by allowing for arbitrary quantifier free formulas in the body and in the head of as rule (i.e. does not depend on the presence of any specific connective, nor any specific syntax of rules).

We show how to solve classical ATP problems in the framework of extended disjunctive logic programming (EDLP) where neither *Contraposition* nor the *Law of the Excluded Middle* are admitted principles of inference. Besides being able to solve classical ATP problems in a monotonic reasoning mode, EDLP can as well treat commonsense reasoning problems by employing its intrinsic nonmonotonic inference capabilities based on stable generated models. EDLP thus proves itself as a general-purpose AI reasoning system.

1 Introduction

A logic program consists of *facts* and *rules*. Facts correspond to sentences of a suitably restricted language, and rules are represented in the sequel by (Gentzen) sequents. A set of facts can be viewed as a database whose semantics is determined by its minimal models. In the case of logic programs, minimal models are not adequate because they are not able to capture 'groundedness', i.e. the directedness of rules. Therefore, *stable* models in the form of certain fixpoints have been proposed by Gelfond and Lifschitz [GL88] as the intended models of normal logic programs. We have generalized this notion in [HW96] by presenting a definition which is neither fixpoint-based nor dependent on any specific rule syntax and allows arbitrary open formulas in the body and the head of a rule. Formalisms admitting arbitrary formulas are more expressive and natural to use since they permit an easier translation from informal specifications.

Logic programs with both strong negation and disjunction are called *extended disjunctive logic programs*(EDLP).[1] Despite their comparable expressiveness they do not lead to classical first order logic (FOL). Our account of EDLPs is based on partial logic where the possibility of two kinds of negation, called *weak* and *strong*, arises

*herre@informatik.uni-leipzig.de, Institut für Informatik, Univ. Leipzig, Augustusplatz 10-11, 04109 Leipzig, Germany

†gw@inf.fu-berlin.de, Institut für Informatik, Univ. Leipzig, Augustusplatz 10-11, 04109 Leipzig, Germany

[1] See, e.g., [GL91], or [MR93].

in a natural way.[2] In EDLPs, neither the *Law of the Excluded Middle* nor the *Contraposition* principle hold. It seems, however, that these classical principles in their full generality are not needed in many cases of natural inferences, cf [HP92]. While classical FOL seems to be appropriate for purely theoretical (notably mathematical) reasoning, EDLP seems to be more appropriate for practical reasoning.

2 Preliminaries

A *signature* $\sigma = \langle Rel, ExRel, Const, Fun \rangle$ consists of a set of relation symbols Rel, a set of exact relation symbols $ExRel \subseteq Rel$, a set of constant symbols, and a set of function symbols. U_σ denotes the set of all ground terms of σ. For a tuple $t_1, \ldots, t_n$ we will also write $\vec{t}$ when its length is of no relevance. The logical functors are $-, \sim, \wedge, \vee, \forall, \exists$; where $-$ and $\sim$ are called *weak* and *strong negation*, respectively. $L(\sigma)$ is the smallest set containing the atomic formulas of σ, and being closed with respect to the following conditions: if $F, G \in L(\sigma)$, then $\{-F, \sim F, F \wedge G, F \vee G, \exists x F, \forall x F\} \subseteq L(\sigma)$.

$L^0(\sigma)$ denotes the corresponding set of sentences (closed formulas), and $QF(\sigma)$ the set of quantifier free sentences. For sublanguages of $L(\sigma)$ formed by means of a subset $\mathcal{F}$ of the logical functors, we write $L(\sigma; \mathcal{F})$. With respect to a signature σ we define the following sublanguages: $\mathrm{At}(\sigma) = L(\sigma; \emptyset)$, the set of all atomic formulas (also called *atoms*); $\mathrm{Lit}(\sigma) = L(\sigma; \sim)$, the set of all *literals*, and $\mathrm{XLit}(\sigma) = L(\sigma; -, \sim)$, the set of all *extended literals*. We introduce the following conventions. When $L \subseteq L(\sigma)$ is some sublanguage, L^0 denotes the corresponding set of sentences. If the signature σ does not matter, we omit it and write, e.g., L instead of $L(\sigma)$. $QF(\sigma) = L^0(\sigma; -, \sim, \vee, \wedge)$ denotes the set of all quantifier free sentences over the signature σ. If $\langle Y, < \rangle$ is a partial order, then $\mathrm{Min}(\langle Y, < \rangle)$ denotes the set of all minimal elements of Y, i.e. $\mathrm{Min}(\langle Y, < \rangle) = \{X \in Y \mid \text{ there is no } X' \in Y \text{s.th. } X' < X\}$. $Pow(X)$ or 2^X denote the power set of the set X.

3 Model Theory

The universe of a *Herbrand interpretation* of the language $L(\sigma)$ is equal to U_σ, and terms are interpreted canonically. We identify Herbrand interpretations over σ with subsets of $\mathrm{Lit}^0(\sigma)$.

Definition 1 (Interpretation) *Let $\sigma = \langle Rel, ExRel, Const, Fun \rangle$ be a signature. A Herbrand σ-interpretation is a set of literals $I \subseteq \mathrm{Lit}^0(\sigma)$. Its* universe *or* domain *is equal to the set of all ground terms U_σ; its canonical interpretation of ground terms is the identity mapping.*

[2] As was already noticed in [Wag91].

The class of all Herbrand σ-interpretations is denoted by $\boldsymbol{I}_4^H(\sigma) = 2^{\mathrm{Lit}^0(\sigma)}$, while the subclass of all *coherent* Herbrand σ-interpretations is defined as

$$\boldsymbol{I}_c^H(\sigma) = \{I \in 2^{\mathrm{Lit}^0(\sigma)} : \text{ if } a \in I, \text{ then } {\sim}a \notin I\}$$

In the sequel we shall simply say 'interpretation' instead of 'Herbrand interpretation'.

An *instantiation* over an interpretation I is a function $\nu : Var \to U_\sigma$, which can be naturally extended to arbitrary terms by setting $\nu(f(t_1, \ldots, t_n)) = f(\nu(t_1), \ldots, \nu(t_n))$. Analogously, an instantiation ν can be canonically extended to arbitrary formulas F, where we write $F\nu$ instead of $\nu(F)$. Note that for a constant c, being a 0-ary function, we have $\nu(c) = c$. The model relation $\models \subseteq \boldsymbol{I}_4^H(\sigma) \times L^0(\sigma)$ between an interpretation and a sentence is defined inductively as follows.

Definition 2 (Model Relation) *1. For literals $l \in \mathrm{Lit}^0(\sigma)$, $I \models l$ iff $l \in I$.*
2. $I \models F \wedge G$ iff $I \models F$ and $I \models G$.
3. $I \models F \vee G$ iff $I \models F$ or $I \models G$.
4. $I \models \exists x F(x)$ iff $I \models F(t)$ for some $t \in U_\sigma$.
5. $I \models \forall x F(x)$ iff $I \models F(t)$ for all $t \in U_\sigma$.
6. $I \models -F$ iff $I \not\models F$.

In addition, we assume DeMorgan-style rewrite rules for handling the combination of $\sim$ with $-, \wedge, \vee, \forall, \exists$.

We obtain the model operator $\mathrm{Mod}_c(X) = \{I \in \boldsymbol{I}_c^H : I \models X\}$, and the corresponding consequence relation defined by $X \models_c F$ iff $\mathrm{Mod}_c(X) \subseteq \mathrm{Mod}_c(F)$.

Definition 3 (Extension of a Model) *Let $I, I' \in \boldsymbol{I}_4^H$ be two interpretations. We say that I'* extends *I, resp. I' is* informationally greater *or equal than I, symbolically $I \leq I'$, if $I \subseteq I'$.*

Definition 4 (Minimal Models) *For $F \in L(\sigma) \supseteq X$, and $* = 4, c$, we define $\mathrm{Mod}_*^m(X) = \mathrm{Min}(\langle \mathrm{Mod}_*(X) \ , \leq \rangle)$, and $X \models_*^m F$ iff $\mathrm{Mod}_*^m(X) \subseteq \mathrm{Mod}_*(F)$.*

4 Extended Disjunctive Logic Programs

In the following we use Gentzen sequents to represent rules of logic programs.

Definition 5 (Sequent) *A* sequent *s is an expression of the form*

$$F_1, \ldots, F_m \ \Rightarrow \ G_1, \ldots, G_n$$

where $F_i, G_j \in L(\sigma)$ for $i = 1, \ldots, m$ and $j = 1, \ldots, n$. The body *of s, denoted by Bs, is given by $\{F_1, \ldots, F_m\}$, and the* head *of s, denoted by Hs, is given by $\{G_1, \ldots, G_n\}$. $\mathrm{Seq}(\sigma)$ denotes the class of all sequents s such that $Hs, Bs \subseteq L(\sigma)$, and for a given set $S \subseteq \mathrm{Seq}(\sigma)$, $[S]$ denotes the set of all ground instances of sequents from S.*

For a sequent $\Rightarrow F$ with empty body we also write more simply F.

Definition 6 (Satisfaction Set) *Let $I \in \boldsymbol{I}_4^H(\sigma)$, and $F \in L(\sigma)$. Then*

$$\mathrm{Sat}_I(F) = \{\nu \in {U_\sigma}^{Var} : I \models F\nu\}$$

Definition 7 (Model of a Sequent) *Let $I \in \boldsymbol{I}_4^H$. Then,*

$$I \models F_1, \ldots, F_m \Rightarrow G_1, \ldots, G_n iff \bigcap_{i \leq m} Sat_I(F_i) \subseteq \bigcup_{j \leq n} Sat_I(G_j).$$

We define the following classes of sequents:

1. $\mathrm{EDLP}^+(\sigma) = \{s \in \mathrm{Seq}(\sigma) \mid Hs, Bs \subseteq \mathrm{Lit}(\sigma)\}$.
2. $\mathrm{EDLP}(\sigma) = \{s \in \mathrm{Seq}(\sigma) \mid Bs \subseteq \mathrm{XLit}(\sigma), Hs \subseteq \mathrm{Lit}(\sigma), Hs \neq \emptyset\}$.
3. $\mathrm{EGLP}(\sigma) = \{s \in \mathrm{Seq}(\sigma) \mid Hs, Bs \subseteq L(\sigma; -, \sim, \wedge, \vee)\}$.

The class EDLP (EDLP^+) corresponds to extended disjunctive logic programs (without negation-as-failure), and EGLP to extended generalized logic programs. For $S \subseteq \mathrm{EDLP}(\sigma)$, we define the model operator

$$\mathrm{Mod}_c(S) \quad = \quad \{I \in \boldsymbol{I}_c^H(\sigma) : I \models s, \text{ for all } s \in S\}$$

and the associated entailment relation

$$S \models F \quad \text{iff} \quad \mathrm{Mod}_c(S) \subseteq \mathrm{Mod}_c(F)$$

where $F \in L(\sigma)$. Instead of sequent notation $B \Rightarrow H$ we shall also use the rule notation $F \leftarrow G$, where $F = \bigvee H$ and $G = \bigwedge B$.

With respect to a class of interpretations $\boldsymbol{K}$, we write $\boldsymbol{K} \models F$ iff $I \models F$ for all $I \in \boldsymbol{K}$. We denote the set of all sequents from a sequent set S which are applicable in $\boldsymbol{K}$ by

$$S_{\mathbf{K}} = \{s \in [S] : \boldsymbol{K} \models Bs\}$$

If $\boldsymbol{K}$ is a singleton, we omit brackets.

5 Stable Models

In the sequel, we need the notion of an 'interval' of interpretations.

Definition 8 $[M_1, M_2] = \{M \in \boldsymbol{I}_c^H : M_1 \leq M \leq M_2\}$

The following definition of a *stable model* of extended logic programs generalizes the *answer set* semantics of [GL91].

Definition 9 (Stable Model) *Let $S \subseteq \mathrm{EGLP}(\sigma)$. $M \in \mathrm{Mod}_c(S)$ is called a* stable generated model *of S, symbolically $M \in \mathrm{Mod}_c^s(S)$, if there is a chain of Herbrand interpretations $I_0 \leq \ldots \leq I_\kappa$ such that $M = I_\kappa$, and*

1. *$I_0 = \emptyset$.*
2. *For successor ordinals α with $0 < \alpha \leq \kappa$, I_α is a minimal extension of $I_{\alpha-1}$ satisfying the heads of all sequents whose bodies hold in $[I_{\alpha-1}, M]$, i.e.*

$$I_\alpha \in \mathrm{Min}\{ \quad I \in \boldsymbol{I}_c^H(\sigma) : I \geq I_{\alpha-1}, \text{ and } I \models \bigvee Hs, \text{ f.a. } s \in S_{[I_{\alpha-1},M]} \}$$

3. *For limit ordinals $\lambda \leq \kappa$, $I_\lambda = \bigcup_{\alpha<\lambda} I_\alpha$.*

We also say that M is generated *by the S-*stable chain *$I_0 \leq \ldots \leq I_\kappa$.*

The stable entailment relation is defined as follows:

$$S \models_s F \quad \text{iff} \quad \mathrm{Mod}_c^s(S) \subseteq \mathrm{Mod}_c(F)$$

where $F \in L(\sigma)$.

Claim 1 *If M is a stable model of $S \subseteq$ EGLP, then there is either a finite S-stable chain, or a S-stable chain of length ω, generating M.*

Proof: along the same lines as in the case of *generalized logic programs* in [HW96].

Claim 2 *In the case of a (non-disjunctive) extended logic program, stable coherent models agree with answer sets as defined in* [GL90].

6 Correct Derivations

In order to define an automated bottom-up inference procedure corresponding to a sound derivablity relation $\vdash \subseteq \models_s$, we establish a number of inference rules which are correct with respect to our stable semantics. Every set S of sequents defines a nonmonotonic inference operation $C_S : 2^{QF} \to 2^{QF}$ as follows: $C_S(X) = \{F : F \in QF, S \cup X \models_s F\}$. An inference operation $C : 2^{QF} \to 2^{QF}$ is said to be correct for C_S iff $C(X) \subseteq C_S(X)$ for every set X of quantifier free sentences.

Definition 10 (Rules) *Let $S \subseteq EGLP$ be a set of sequents, and $F, G \in QF$.*

(Refl) *If $F \in [S]$, then $S \vdash F$.*
(And) *If $S \vdash F$ and $S \vdash G$, then $S \vdash F \wedge G$.*
(Or) *If $S \vdash F$ or $S \vdash G$, then $S \vdash F \vee G$.*
(Det) *If $S \vdash G$, and $F \leftarrow G \in [S]$, then $S \vdash F$.*
(DS) *If $S \vdash F \vee G$, and $S \vdash -F$, then $S \vdash G$.*
(Coh) *If $S \vdash \sim F$, then $S \vdash -F$.*

We define the basic inference *relation $\vdash_0 \subseteq Pow(EGLP) \times QF$ as the smallest relation being closed under Reflexivity* **(Refl)**, *And, Or, Detachment* **(Det)**, *Disjunctive Syllogism* **(DS)**, *and Coherence* **(Coh)**.

Observation 1 *The consequence relation $\models_c$ is correct for $\models_s$, hence, the basic inference relation $\vdash_0$ is correct for $\models_s$, i.e. $S \vdash_0 F$ implies $S \models_s F$.*

6.1 Exact Predicates and Contrapositive Inference

In an EDLP, we distinguish between two kinds of predicates. This distinction reflects the fact that many predicates (especially in empirical domains) have truth value gaps: neither $p(c)$ nor $\sim p(c)$ has to be the case for specific instances of such *inexact* predicates, like color attributes which can in some cases not be determined because of vagueness. Other predicates, e.g. from legal or theoretical domains, are *exact*, and we then have, for instance, $m(S) \vee \sim m(S)$, stating that Susan is either married or unmarried, resp. $odd(0) \vee \sim odd(0)$, stating that 0 is either odd or non-odd. For an exact predicate p and any ground term t from its domain, the *Tertium Non Datur (TND)* follows from S: $S \models p(t) \vee \sim p(t)$. For an EDLP, this can be achieved by adding all suitable instances of the TND: $\{p(t) \vee \sim p(t) \leftarrow : p \in ExRel\}$.

Observation 2 (Collapse) *Weak and strong negation collapse for exact predicates: let $p \in ExRel$, then $S \vdash \sim p(t)$ iff $S \vdash -p(t)$.*

Observation 3 (Weak Contraposition) *If $S \models_c -F$, and $F \leftarrow G \in [S]$, then $S \models_c -G$.*

The Weak Contraposition principle facilitates the efficient bottom-up computation of models. Together, Coherence and Weak Contraposition help to establish weakly negated conclusions in the course of 'model generation' reasoning. Combined with Coherence, the Weak Contraposition principle leads to Mixed Contraposition: If $S \vdash \sim F$, and $F \leftarrow G \in [S]$, then $S \vdash -G$. While in general, a weakly negated conclusion can only be established after computing all stable models, by means of Coherence and Mixed Contraposition, we can establish weakly negated conclusions before finishing the computation of all stable models.

If the Mixed Contraposition principle is combined with Collapse, we obtain:

Observation 4 (Strong Contraposition) *Let $p \in ExRel$. If $S \vdash \sim F$, and $F \leftarrow p(t) \in [S]$, then $S \vdash \sim p(t)$.*

Definition 11 (Contrapositive Inference) *We define the contrapositive inference relation $\vdash'$ as the smallest relation being closed under Reflexivity, And, Or, Detachment, Disjunctive Syllogism, Coherence, Weak Contraposition, and Collapse.*

By applying Collapse and the contraposition principles, we can compile a program S into S' according to the following transformation steps. Firstly, set $S' = S$. Secondly, add a contrapositive $-G \leftarrow -F$ to S' for every $F \leftarrow G \in S$. Thirdly, apply appropriate rewrite rules in order to transform the heads of the added contrapositves into negation normal forms where negations occur only in extended literal form $-p(t)$, $-\sim p(t)$, or $\sim p(t)$. Finally, apply Collapse to the obtained weakly negated literals. In the sequel, we write $\mathrm{CP}(S)$ instead of S' for the compiled progam.

Observation 5 *Contrapositive inference from S is obtained by basic inference from the contrapositive transformation $CP(S)$, i.e. $S \vdash F$ iff $CP(S) \vdash_0 F$.*

In the next section, we apply these inference rules to illustrative examples.

7 Practical Reasoning Problems

In this section we demonstrate by two simple examples how the framework of disjunctive logic programming can be used to formalize and solve practical reasoning problems.

7.1 Monotonic Problems

The following problem is in its original version due to L. Schubert (and borrowed from [Pel86]). Its solution is obtained by using (in addition to the basic principles Detachment and Disjunctive Syllogism) the possibility to declare certain predicates as exact, thus enabling the restricted inference rule of Strong Contraposition.

Example 1: The formalization of the problem is given by the following extended disjunctive logic program **P1** containing the facts and rules $(A1)$ to $(A9)$. Someone who lives in Dreadsbury Mansion, killed aunt Agatha. Agatha, the butler, and Charles live in Dreadsbury Mansion, and are the only people who live therein:[3]

$(A1)$ $k(A,A) \vee k(B,A) \vee k(C,A)$

A killer hates his victim:

$(A2)$ $h(x,y) \leftarrow k(x,y)$

and is never richer than his victim:

$(A3)$ $\sim r(x,y) \leftarrow k(x,y)$

Charles hates no one that aunt Agatha hates:

$(A4)$ $\sim h(C,x) \leftarrow h(A,x)$

Agatha hates everyone except the butler, and no one the butler does not hate:[4]

$(A5)$ $h(A,x) \leftarrow \sim e(x,B)$

$(A6)$ $\sim h(A,x) \leftarrow \sim h(B,x)$

The butler hates everyone not richer than aunt Agatha:

$(A7)$ $h(B,x) \leftarrow \sim r(x,A)$

No one [in Dreadsbury Mansion] hates everyone:

$(A8)$ $H_A \wedge H_B \wedge H_C$

where $H_x :=\sim h(x,A) \vee \sim h(x,B) \vee \sim h(x,C)$, for $x = A,B,C$. Finally, we also know that Agatha is not the butler and Charles is not the butler:

$(A9)$ $\sim e(A,B) \wedge \sim e(C,B)$

Who killed aunt Agatha: $k(x,A)$?

[3] We reduce this information, namely $lDM(A) \wedge lDM(B) \wedge lDM(C) \wedge \exists x : k(x,A) \wedge lDM(x)$, using the Closed-World Assumption for lDM, to the disjunction $k(A,A) \vee k(B,A) \vee k(C,A)$.

[4] Notice that we use e as an equality predicate. In the case of $(A5)$ we use the contraposition of Schubert's original formulation 'The butler hates everyone Agatha hates.'

A realistic representation of the problem should assume that the predicates *hates* and *killed* may be undetermined in certain cases, i.e. that they are not exact. On the other hand, it seems plausible to declare the predicates *richer* and *equals* as exact, i.e. assuming that they do not have truth value gaps, or in other words, that for all ground instances of them it can be determined whether they are true or false: ExRel $= \{r, e\}$.

While the solution can be found by 'brute force' bottom-up computation, we will construct here only the essential reasoning steps using the restricted Strong Contraposition principle. Two arguments can be found why neither Charles nor the butler killed Agatha, and hence, by the Disjunctive Syllogism, she must have killed herself.

From $(A5)$ and $(A9)$ we get immediately that Agatha hates herself, $h(A, A)$, consequently by $(A4)$, Charles does not hate Agatha, $\sim h(C, A)$, and hence by Mixed Contraposition of $(A2)$, it is not the case that he killed her, $-k(C, A)$.

From $h(A, A)$ we obtain with Mixed Contraposition of $(A6)$ that $- \sim h(B, A)$. From $(A9)$ and $(A5)$, we may also conclude that $h(A, C)$, and consequently again with Mixed Contraposition of $(A6)$ that $- \sim h(B, C)$. By applying the disjunctive syllogism to the disjunction H_B from $(A8)$ and the two derived facts $- \sim h(B, A)$ and $- \sim h(B, C)$, we get $\sim h(B, B)$. Since r is assumed to be exact, Strong Contraposition of $(A7)$ yields $r(B, A)$, and finally by Mixed Contraposition of $(A3)$ we conclude that it is also not the case that the butler killed Agatha, $-k(B, A)$.

7.2 Nonmonotonic Problems

Example 2: Climbers and Skiers. The following disjunctive program (inspired by a similar, but monotonic problem in [She88]) is denoted by **P2**.
Tony, Mike, and John are members of an alpine club. Tony likes rain and John dislikes snow:

$(B1)$ $\quad m(T), m(M), m(J), l(T, r), \sim l(J, s).$

All club members are skiers or climbers:

$(B2)$ $\quad c(x) \vee s(x) \leftarrow m(x).$

Climbers normally dislike rain and snow:[5]

$(B3a)$ $\quad \sim l(x, r) \leftarrow c(x) \wedge -l(x, r).$

$(B3b)$ $\quad \sim l(x, s) \leftarrow c(x) \wedge -l(x, s).$

Skiers like snow and dislike rain

$(B4)$ $\quad l(x, s) \wedge \sim l(x, r) \leftarrow s(x).$

Problem: Is there a climber in the club who likes rain but dislikes snow? Yes, this holds for Tom: $c(T) \wedge \sim l(T, s) \wedge l(T, r)$.

The program **P2** has two stable models. Both contain the facts from $(B1)$, and

[5] Notice, that rules of this form correspond exactly to normal default rules in Reiter's default logic.

the following ones:

$$c(T), c(J), \sim l(T,s), \sim l(J,r)$$

One of them contains in addition $c(M), \sim l(M,s)$, $\sim l(M,r)$, and the other one $s(M), l(M,s), \sim l(M,r)$.

The monotonic inference rules defining $\vdash$ are not sufficient to solve this problem, we have to add some non-monotonic rule which is described as follows. Let $S \subseteq EDLP$, and assume that S contains a complete set of rules for the exact predicates. Let $HeadLit(S) = \bigcup\{H : B \Rightarrow H \in [S]\}$. Let $\vdash_1$ be the smallest inference relation being closed under the rules of contrapositive inference (see definition 11) and the following nonmonotonic inference rule.[6]

(NM1): If $l \notin HeadLit(S)$, then $S \vdash_1 -l$.

Observation 6 *The inference relation $\vdash_1$ is correct for the stable semantics.*

Proof (Sketch). Let $l \notin HeadLit(S)$ and let I be a stable model of S. Assume, $l \in I$. By stability of I there exists a stable chain $I_0 \leq I_1 \leq \ldots \leq I_n \leq \ldots$ generating I. This implies the existence of a stage $k \in \omega$ such that $l \in I_{k+1} - I_k$. It is easy to show, that, because of $l \notin HeadLit(S)$, the interpretation $J = I_{k+1} - \{l\}$ is a model of $\{\bigvee Hs : [I_k, I] \models \bigvee Hs\}$. This contradicts the minimality if I_{k+1}.

We now sketch the solution of the problem **P2**. None of the predicates of the problem domain is assumed to be exact, $ExRel = \emptyset$. Therefore, neither (Collapse) nor the (Strong Contraposition) principle may be used.

By **(Coh)** we obtain $- \sim l(T,r)$ from the corresponding fact in (B1). From this we get by weak contraposition of (B4) $-s(T)$. By **(Det)** we obtain $c(T) \vee s(T)$ from (B1) and (B2). Applying **(DS)** to this disjunction and $-s(T)$ yields $c(T)$. Using the non-monotonic inference rule **NM1** we may derive $-l(T,s)$. By applying **(Det)** to (B3a) we finally conclude $\sim l(Ts)$. Finally, using **(And)** we combine the three literals to the conjunction $c(T) \wedge \sim l(T,s) \wedge l(T,r)$.

8 Conclusion

We have shown how the computational framework of EDLP, allowing for truthvalue gaps and nonmonotonic inference, based on stable generated models, can be used to solve practical reasoning problems.

References

[GL88] M. Gelfond and V. Lifschitz: The Stable Model Semantics for Logic Programming, *Proc. ICLP 1988*, MIT Press, 1988.

[6] By a natural extension of this rule we get a sequence of stronger nonmonotonic relations $\vdash_1 \subseteq \vdash_2 \subseteq \ldots \subseteq \vdash_\alpha \ldots$.

[GL90] M. Gelfond and V. Lifschitz: Logic Programs with Classical Negation, *Proc. ICLP 1990*, MIT Press, 1990.

[GL91] M. Gelfond and V. Lifschitz: Classical Negation in Logic Programs and Disjunctive Databases, *J. New Generation Computing* **9** (1991), 365–385.

[HP92] H. Herre and D. Pearce: Disjunctive Logic Programming, Constructivity and Strong Negation; Springer, in: LNAI 633, 391-410

[HW96] H. Herre and G. Wagner: Stable Models are Not Always Minimal, submitted, available from the LPNMR archive.

[MR93] J. Minker and C. Ruiz: On Extended Disjunctive Logic Pograms, *Proc. of ISMIS'93*, Springer LNAI, 1993.

[Pel86] F.J. Pelletier: Seventy-Five Problems for Testing Automatic Theorem Provers, *J. Automated Reasoning* **2** (1986), 191–216.

[She88] J.C. Shepherdson:Introduction to the Theory of Logic Programming, *Proc. of the Logic Coll. 1986*, North-Holland, 1988

[Wag91] G. Wagner: A Database Needs Two Kinds of Negation, in: Proc. Third Int. Symp. on Mathematical Fundamentals of Database and Knowledge Based Systems, Springer LNCS 495 (1991), 357–371.