Stable Generated Models, Partial Temporal Logic and Disjunctive Defaults

Joeri Engelfriet* Heinrich Herre†

Abstract

Stable generated models for extended generalized logic programs with two kinds of negation provide a semantics for logic programming in its (up to now) most general form. We show that stable generated semantics for extended logic programs coincides with the semantics obtained by translating programs into a minimal partial temporal logic. Then we introduce several new versions of (disjunctive) default logic and show that there are strong (semantical) connections between default logic and logic programming with stable generated models.

Key words: Logic programming, default logic, partial temporal logic, stable generated semantics, disjunctive defaults.

1 Introduction

In knowledge representation, two different notions of falsity arise in a natural way. Certain facts are *implicitly false by default* by not being verified in any intended model of the knowledge base. Others are *explicitly false* by virtue of a direct proof of their falsity, corresponding to their falsification in all intended models. These two kinds of falsity in knowledge representation are captured by the two negations, called *weak* and *strong*, of partial logic. Logic programs can be viewed as knowledge bases. A logic program consists of *facts* and *deduction rules*. Facts correspond to sentences of a suitably restricted language, and deduction rules correspond to non-schematic (Gentzen) sequents. While facts express extensional knowledge, rules express intensional knowledge. 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. In [HW97], this notion was generalized by presenting a definition which is neither fixpoint-based nor dependent on any specific rule syntax. In the present paper we relate this definition to the minimal models of systems of temporal partial logic. Furthermore, we investigate the relation between

*Faculty of Mathematics and Computer Science, Vrije Universiteit Amsterdam, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands, joeri@cs.vu.nl

†Institut für Informatik, University of Leipzig, Augustusplatz 10-11, 04109 Leipzig, Germany, herre@informatik.uni-leipzig.de

stable generated models and (disjunctive) default logic. A preliminary version of this paper appeared in [EH97].

The paper has the following structure. After introducing some basic notation in section 2, we recall the notion of a stable generated model in section 3. Section 4 describes the partial temporal logic and its link with stable generated models. In section 5, we introduce the notion of a generated extension of a disjunctive default theory and investigate the connections with stable generated models.

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 $Const$, and a set of function symbols Fun. If $Rel = ExRel$ we say that the signature is classical. U_σ denotes the set of all ground terms of σ. For a tuple $t_1, \dots, t_n$ we will also write $\vec{t}$ when its length is of no relevance. The logical functors are $-, \sim, \neg, \wedge, \vee, \supset, \forall, \exists$, where $-$, $\sim$, and $\neg$ are called *weak*, *strong negation* and *classical negation* and $\supset$ is called *material implication*. The language of partial logic $L(\sigma)$ 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 $\{-F, \sim F, F \wedge G, F \vee G, F \supset G, \exists x F, \forall x F\} \subseteq L(\sigma)$. The language over a classical signature σ of classical logic is denoted by $L_c(\sigma)$; it is the smallest set containing the atomic formulas over σ, and being closed with respect to the following condition: if $F, G \in L_c(\sigma)$, then $\{\neg F, F \wedge G, F \vee G, F \supset G, \exists x F, \forall x F\} \subseteq L_c(\sigma)$.

$L^0(\sigma)$ (and analogous $L^0_c(\sigma)$) denotes the corresponding set of sentences (closed formulas). 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*); $\mathrm{Lit}(\sigma) = \mathrm{At}(\sigma) \cup \{\sim a : a \in At(\sigma)\}$, the set of all *literals*, and $\mathrm{XLit}(\sigma) = \mathrm{Lit}(\sigma) \cup \{-l : l \in \mathrm{Lit}(\sigma)\}$, 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 $L(\sigma)$. $QF(\sigma) = L^0(\sigma; -, \sim, \vee, \wedge, \supset)$ denotes the set of all quantifier free sentences over the signature σ. If $\langle Y, < \rangle$ is a partial order, then $\mathrm{Min}(Y, <)$ denotes the set of all minimal elements of Y, i.e. $\mathrm{Min}(Y, <) = \{x \in Y \mid \text{there is no } x' \in Y \text{ such that } x' < x\}$. $Pow(X)$ or 2^X denotes the power set of the set X.

Let $\sigma = \langle Rel, ExRel, Const, Fun \rangle$ be a signature. A *Herbrand σ-interpretation* of $L(\sigma)$ 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. An intepretation I assigns two relations r^I and $\tilde{r}^I$, which can be viewed as the truth and falsity extension, to every relation symbol $r \in Rel$: $r^I = \{\vec{t} \in U_\sigma^{a(r)} : r(\vec{t}) \in I\}$, and $\tilde{r}^I = \{\vec{t} \in U_\sigma^{a(r)} : \sim r(\vec{t}) \in I\}$, where $a(r)$ denotes the arity of r, such that for all exact predicates $r \in ExRel$, $r^I \cup \tilde{r}^I = U_\sigma^{a(r)}$. Herbrand interpretations of $L(\sigma)$ are called partial interpretations. An Herbrand interpretation of $L_c(\sigma)$ is a subset of $\mathrm{At}^0(\sigma)$, and it is called a positive interpretation.

The class of all partial 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 'partial Herbrand interpretation'. An *instantiation* is a function $\nu : Var \rightarrow U_\sigma$, 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 1 (Model Relation) *Let $I \in \boldsymbol{I}_4^H(\sigma)$.*

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 De Morgan-style rewrite rules for handling the combination of $\sim$ with $-, \wedge, \vee, \forall, \exists$:

$$\begin{array}{rcl}
\sim (F \wedge G) & \longrightarrow & \sim F \vee \sim G \\
\sim (F \vee G) & \longrightarrow & \sim F \wedge \sim G \\
\sim \exists x F(x) & \longrightarrow & \forall x \sim F(x) \\
\sim \forall x F(x) & \longrightarrow & \exists x \sim F(x) \\
\sim\sim F & \longrightarrow & F \\
\sim -F & \longrightarrow & F
\end{array}$$

For formulas F, define $I \models F$ iff for every instantiation ν it holds that $I \models F\nu$. An interpretation I is a model for a set X of formulas if for every formula $F \in X$ the condition $I \models F$ is satisfied.

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 $\mathrm{Mod}_c(X) \subseteq \mathrm{Mod}_c(F)$. Furthermore, denote $C_0(X) = \{F \mid X \models_c F$ and F does not contain weak negation $\}$. 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, if $I \subseteq I'$.

Definition 2 (Minimal Models) *For $F \in X \subseteq L(\sigma)$, we define $\mathrm{Mod}_c^m(X) = \mathrm{Min}(\mathrm{Mod}_c(X)\ , \subseteq)$, and $X \models_c^m F$ iff $\mathrm{Mod}_c^m(X) \subseteq \mathrm{Mod}_c(F)$.*

Program rules are presented in the following by sequents.

Definition 3 (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 4 (Satisfaction Set)

1. *Let $I \in \boldsymbol{I}_4^H(\sigma)$, and $F \in L(\sigma)$.*
   Then $\mathrm{Sat}_I(F) = \{\nu : Var \to U_\sigma \mid I \models F\nu\}$
2. *Two formulas F and G from $L(\sigma)$ are logically equivalent, denoted by $F \equiv G$ iff $Sat_I(F) = Sat_I(G)$ for all $I \in \boldsymbol{I}_4^H(\sigma)$.*

Definition 5 (Model of a Sequent) *Let $I \in \boldsymbol{I}_4^H$. Then,*

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

We define the following classes of sequents:

1. $\mathrm{EDLP}(\sigma) = \{s \in \mathrm{Seq}(\sigma) \mid Hs \subseteq \mathrm{Lit}(\sigma), Bs \subseteq \mathrm{XLit}(\sigma), H(s) \neq \emptyset\}$.
2. $\mathrm{EGLP}(\sigma) = \{s \in \mathrm{Seq}(\sigma) \mid Hs, Bs \subseteq L(\sigma; -, \sim, \wedge, \vee, \supset)\}$.
3. $\mathrm{GLP}(\sigma) = \{s \in \mathrm{Seq}(\sigma) \mid Bs, Hs \subseteq L(\sigma; -, \wedge, \vee, \supset)\}$
4. $\mathrm{SLP}(\sigma) = \{s \in \mathrm{Seq}(\sigma) | B(s) \subseteq \mathrm{At}(\sigma) \cup \{-K | K \in L(\sigma; \wedge)\}, H(s) \subseteq \mathrm{At}(\sigma), H(s) \neq \emptyset\}$.

The class EDLP corresponds to extended disjunctive logic programs, the subsets of EGLP represent the extended generalized logic programs containing the class of generalized logic programs wich are based on GLP. Subsets of SLP are called *super logic programs* and were introduced in [Prz96]. For $S \subseteq \mathrm{EGLP}(\sigma)$, 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

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

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

In sequel we assume that all considered interpretations are coherent (or sometimes positive). Coherent entailment $\models_c$ satisfies the following property.

Proposition 1 (Coherence Principles) *Let be X a set of formulas from $L(\sigma)$, and F a σ-sentence. Then $X \models_c \sim F$ implies $X \models_c -F$, and $X \models_c F$ implies $X \models - \sim F$.*

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 $S_{\mathbf{K}}$ 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.

Observation 1 *Let $S \subseteq \mathrm{EGLP}$ be an extended generalized logic program. Then,*

$$\mathrm{Mod}_c(S) = \mathrm{Mod}_c([S])$$

where $[S]$ is the Herbrand instantiation of S.

A semantics for sequents is given by a preferred model operator $\Phi : 2^{Seq} \to 2^{\boldsymbol{I}_c^H}$, satisfying the condition $\Phi(S) \subseteq \mathrm{Mod}_c(S)$, and defining the preferential entailment relation

$$S \models_\Phi F \;\text{ iff }\; \Phi(S) \subseteq \mathrm{Mod}_c(F)$$

3 Stable Generated Models

We consider a model of a set S of sequents as intended if it can be generated bottom-up starting from zero information by an iterated application of the sequents $s \in [S]$. This intention is captured by the notion of a stable generated model. The subsequent definition is a generalization of the notion discussed in [HW97]. This notion generalizes the answer set semantics of [GL90] and was investigated in [HW97a].

In the sequel, we need the notion of an 'interval' of interpretations: for $M_1, M_2 \in \boldsymbol{I}_c^H$, define $[M_1, M_2] = \{M \in \boldsymbol{I}_c^H : M_1 \subseteq M \subseteq M_2\}$.

Definition 6 (Stable Generated 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 coherent Herbrand interpretations $I_0 \subseteq \ldots \subseteq I_\kappa$ 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.*

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

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 *$I_0 \subseteq \ldots \subseteq I_\kappa$.*

The stable entailment relation is defined as follows:

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

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

Claim 2 ([HW97]) *For every stable generated model, there is a stable chain of length ω generating it.*

In [HW97] it was shown that stable generated models of disjunctive programs are not always minimal and that there are stable generated models which are also minimal models, but which are not answer sets (see [GL90]).

Claim 3 ([HW97]) *In the case of (non-disjunctive) extended logic programs, stable generated models agree with answer sets as defined in [GL90].*

For disjunctive programs, this is not the case.

Example 1 *Let $S = \{\Rightarrow a, b;\ a \Rightarrow b;\ -a \Rightarrow a\}$. We show that $\{a, b\}$ is a stable generated model of S by giving a stable chain for it. $I_0 = \emptyset$, $S_{[\emptyset,\{a,b\}]} = \{\Rightarrow a, b\}$. A minimal extension of $\emptyset$ is either $\{a\}$ or $\{b\}$. Take $I_1 = \{a\}$. Then $S_{[\{a\},\{a,b\}]} = \{a \Rightarrow b; \Rightarrow a, b\}$, and a minimal extension of $\{a\}$ satisfying b gives finally $I_2 = \{a, b\}$. On the other hand, $\{a, b\}$ is not a minimal model of the Gelfond-Lifschitz reduct of S with respect to $\{a, b\}$, which is $\{\Rightarrow a, b; a \Rightarrow b\}$, since $\{b\}$ is a model of it. In fact, S has no answer sets.*

Two programs P and Q are said to be *stable equivalent*, denoted by $P \equiv_{st} Q$, if P and Q have the same stable generated models. A rule $s \in \mathrm{EGLP}$ has *normal form* if $s := F_1, \ldots, F_m \Rightarrow G_1, \ldots, G_n$, where every F_i is a disjunction of extended literals, and every G_j is an extended literal. The rule s has *special normal* form if s has normal form and every F_i is a extended literal. Let $Fin(Seq) = \{X : X \subseteq Seq$ and X is finite $\}$.

Definition 7 (Transformation Rule) *A relation $r \subseteq Fin(Seq) \times Fin(Seq)$ is said to be an admissable transformation rule if r is decidable and for every program $P \subseteq \mathrm{EGLP}$, $s \in P$, and $X, Y \in Fin(Seq)$ such that $r(X, Y)$, the condition $P \equiv_{st} (P - X) \cup Y$ is satisfied. Let $R = \{r_1, ..., r_m\}$ be a finite set of admissable transformation rules. The relation $\longrightarrow_R$ between programs is defined as follows: $P \longrightarrow_R Q$ iff there is a rule $r \in R$, $Y \subseteq P$, $X, Y \in Fin(Seq)$ such that $r(X, Y)$ and $Q = (P - X) \cup Y$. Let $\longrightarrow_R^\star$ be the transitive closure of $\longrightarrow_R$. P can be transformed into Q by the rule system R if $P \longrightarrow_R^\star Q$.*

Lemma 4 *Let $P \subseteq \mathrm{EGLP}$, and $s \in P$, $F \equiv \bigwedge Bs$, and $G \equiv \bigvee Hs$. Then $P \equiv_{st} (P - \{s\}) \cup \{Bs \Rightarrow G\} \equiv_{st} (P - \{s\}) \cup \{F \Rightarrow Hs\}$.*

Proof: Let $s_1 := Bs \Rightarrow G$, and $Q := (P - \{s\}) \cup \{Bs \Rightarrow G\}$; we show that $P \equiv_{st} Q$. Let I be a stable generated model of P and $(I_n)_{n<\kappa}$, $\kappa \leq \omega$, a stable chain for I. It is sufficient to show that I_{n+1} is a minimal model of $Hd = I_n \cup \{Hr : r \in Q_{[I_n,I]}\}$. It is easy to show that that Hd has the same Herbrand models than the set $Hd_1 = I_n \cup \{Hr : r \in P_{[I_n,I]}\}$, and from this follows that Hd and Hd_1 have the same minimal models. The other case is proved analogously. □

Lemma 5 *Let $P \subseteq$ EGLP, and $s \in P$, $s := Bs \Rightarrow F_1 \wedge F_2$, with F_1, F_2 formulas. Then $P \equiv_{st} (P - \{s\}) \cup \{Bs \Rightarrow F_1, Bs \Rightarrow F_2\}$.*

Proof: Let $Q := (P - \{s\}) \cup \{Bs \Rightarrow F_1, Bs \Rightarrow F_2\}$, and $tr(s) = \{Bs \Rightarrow F_1, Bs \Rightarrow F_2\}$. We show that $Mod_s(P) = Mod_s(Q)$.
1) $Mod_s(P) \subseteq Mod_s(Q)$.
Let I be a stable amodel of P and $(I_n)_{n<\omega}$ a stable chain for I. We show that $(I_n)_{n<\omega}$ is Q-stable chain for I. It suffices to prove that I_{n+1} is a minimal model of $I_n \cup \{Hr : r \in Q_{[I_n,I]}\}$. This is implied by the fact that the sets $H_n(Q) = \{Hr : r \in Q_{[I_n,I]}\}$ and $H_n(P) = \{Hr : r \in P_{[I_n,I]}\}$ have the same Herbrand models. Let for $r \in [\{s\}]$ denote $r := Br \Rightarrow F_1^r \wedge F_2^r$. We show the following condition $(\star)$:

For all $r \in [\{s\}]$: $F_1^r \wedge F_2^r \in H_n(P)$ if and only if $\{F_1^r, F_2^r\} \subseteq H_n(Q)$.

If $F_1^r \wedge F_2^r \in H_n(P)$ then $[I_n, I] \models Br$, and hence $\{F_1^r, F_2^r\} \subseteq H_n(Q)$. If $\{F_1^r, F_2^r\} \subseteq H_n(Q)$, then again $[I_n, I] \models Br$ and this implies $F_1^r \wedge F_2^r \in H_n(P)$. The condition $(\star)$ implies that the sets $H_n(P)$ and $H_n(Q)$ have the same Herbrand models and hence the same minimal models.
2) $Mod_s(Q) \subseteq Mod_s(P)$. This condition is proved analogously as 1). □

Corollary 6 *There is a finite set R of admissable transformation rules such that for every finite extended general logic program P there exists a program Q in normal form such that $P \longrightarrow_R^\star Q$.*

Proof Let P be a generalized program, $s \in P$. We carry out the following steps. $s \longrightarrow \{F \Rightarrow G\}$, where $F := \bigwedge Bs$, $G := \bigvee Hs$, then F is transformed into a logically equivalent conjunctive normal form F_1, and G into a logically equivalent conjunctive normal form $G_1 = C_1 \wedge \ldots C_n$. This can be done by a finite set of admissable transformation rules. By lemma 4 $P \equiv_{st} (P - \{s\}) \cup \{F_1 \Rightarrow G_1\}$. By repeated use of the rule described in lemma 5 we get finally a program Q in normal form being stable equivalent to P. □

Corollary 7 *There is a finite set R of admissable transformation rules such that for every extended general logic program P there exists a program Q in normal form such that $P \equiv_{st} Q$. If P is recursively enumerable then Q can be chosen to be recursively enumerable.*

Remark: The following rule $\{F_1 \vee F_2 \Rightarrow H\} \longrightarrow \{F_1 \Rightarrow H, F_2 \Rightarrow H\}$ cannot be used because this rule, though logically true, does not preserve stability (see, however, Proposition 11).

Example 2 *$P = \{b \vee -b \Rightarrow b\}$, and $Q = \{b \Rightarrow b; -b \Rightarrow b\}$. While P has the stable model $\{b\}$ Q has no stable model.*

In example 2 we could replace P by $\{\Rightarrow b\}$. This rule has the form $\{(Bs \Rightarrow Hs, \{\Rightarrow Hs\}) : \bigwedge Bs$ is a tautology $\}$. But also this is rule is not admissable because it is not decidable. Thus we may summarize the results on normal forms by the following proposition.

Proposition 8 *Every program in* EGLP *can be transformed into an equivalent program in which each rule has the form $F_1, \ldots, F_n \Rightarrow G_1, \ldots G_m$ in which each F_i is a disjunction of literals from $XLit$, and each G_i is in $XLit$. We say such a rule (and the program) is in normal form.*

4 Minimal Partial Temporal Logic

In this section we will describe a partial temporal semantics for logic programs. The semantics is obtained by translating a logic program into a nonmonotonic partial temporal logic. This logic was first used to give a temporal semantics to default logic ([ET93]). The idea behind the temporal framework is that a temporal theory describes the reasoning process of a (nonmonotonic) agent. A number of different forms of reasoning were treated in this fashion in [ET94]. Although the full temporal logic is more general (it uses S5 models instead of partial models, and temporal operators may be nested), we will describe a simpler variant here that is sufficient to give semantics to logic programming. First we will formally introduce minimal partial temporal logic.

Definition 8 (Partial model)

1. *A* partial model *M for the signature σ is an assignment of a truth value from $\{0, 1, u\}$ to each of the atoms of σ. This assignment can be extended to arbitrary closed formulas according to the following tables:*

$\sim$	
0	1
1	0
u	u

$\wedge$	0	1	u
0	0	0	0
1	0	1	u
u	0	u	u

$\vee$	0	1	u
0	0	1	u
1	1	1	1
u	u	1	u

2. *The ordering of truth values is defined by $u \leq 0$, $u \leq 1$, $u \leq u$, $1 \leq 1$, $0 \leq 0$. This is extended to partial models by: $M \leq N$ if for all atoms a we have $M(a) \leq N(a)$.*

A partial model describes which formulas the agent knows (or, has derived) at any point in time. To give an account of the total reasoning process of an agent, we have to describe what the agent knows (has derived) at all points in time. The reasoning of the agent is assumed to start at some point, and continues in a discrete manner: based on what the agent knows, it may apply some (nonmonotonic) inference steps to arrive at a new state of knowledge (described by a partial model), from which it may again apply inference steps. We will take the natural numbers (ω) as our flow of time.

Definition 9 (Partial temporal model) *A partial temporal model $\mathcal{M}$ is a sequence $\{\mathcal{M}_t\}_{t\in\omega}$ of partial models. We will assume all partial temporal models are* conservative*, which means that for all t, we have $\mathcal{M}_t \leq \mathcal{M}_{t+1}$. The ordering $\leq$ is extended to partial temporal models by*

$$\mathcal{M} \leq \mathcal{N} \Leftrightarrow \mathcal{M}_t \leq \mathcal{N}_t \ \forall t \in \omega$$

In conservative models, the knowledge of the agent may only increase over time: the agent does not forget and does not revise its knowledge. Whenever we have $\mathcal{M} \leq \mathcal{N}$, then this means that $\mathcal{M}$ contains less (or equal) knowledge than $\mathcal{N}$, *at each time point*.

The temporal language in which the reasoning of an agent can be described, is a restricted version of the language introduced in [ET93]. For each closed formula $\alpha \in L_0(\sigma)$, there are four basic expressions in the language that describe the truth of α in time: $C\alpha$, which states that the agent *currently* knows α, $F\alpha$, which states that the agent will know α *sometimes in the future*, $G\alpha$, stating that the agent knows α *always in the future*, and $X\alpha$, stating that the agent will know α at the *next moment in time*; these basic expressions are called *temporal atoms*. The set of temporal formulas is the smallest set of expressions containing the temporal atoms and closed with respect to the propositional connectives $\{\wedge, \neg, \vee, \rightarrow\}$.

Definition 10 (Semantics) *Let $\mathcal{M}$ be a (conservative) partial temporal model, and let $t \in \omega$.*

1. *For a closed formula $\alpha \in L_0(\sigma)$:*
   - $(\mathcal{M}, t) \models C\alpha \Leftrightarrow \mathcal{M}_t(\alpha) = 1$
   - $(\mathcal{M}, t) \models F\alpha \Leftrightarrow \mathcal{M}_s(\alpha) = 1$ *for some* $s > t$
   - $(\mathcal{M}, t) \models G\alpha \Leftrightarrow \mathcal{M}_s(\alpha) = 1$ *for all* $s > t$
   - $(\mathcal{M}, t) \models X\alpha \Leftrightarrow \mathcal{M}_{t+1}(\alpha) = 1$
2. *For two temporal formulas φ, ψ:*
   - $(\mathcal{M}, t) \models \varphi \wedge \psi \Leftrightarrow (\mathcal{M}, t) \models \varphi$ *and* $(\mathcal{M}, t) \models \psi$
   - $(\mathcal{M}, t) \models \neg\varphi \Leftrightarrow (\mathcal{M}, t) \not\models \varphi$
3. *Implication ($\rightarrow$) and disjunction ($\vee$) are defined as usual from conjunction and negation.*
4. *A formula φ is true in $\mathcal{M}$, denoted $\mathcal{M} \models \varphi$, if $(\mathcal{M}, t) \models \varphi$ for all $t \in \omega$. A set of formulas Th is true in a model $\mathcal{M}$, denoted $\mathcal{M} \models Th$, if $\mathcal{M} \models \varphi$ for all $\varphi \in Th$. Two temporal formulas φ, ψ are equivalent if $(\mathcal{M}, t) \models \varphi \Leftrightarrow (\mathcal{M}, t) \models \psi$ for all partial temporal models $\mathcal{M}$ and $t \in \omega$.*

Even though we want to describe the reasoning process of an agent, we are of course interested in the *final outcomes* of this reasoning process. This is expressed by the *limit* of a partial temporal model.

Definition 11 (Limit model) *Let $\mathcal{M}$ be a (conservative) partial temporal model. Then the limit of $\mathcal{M}$, denoted lim $\mathcal{M}$, is the partial model defined by:*

$$lim\ \mathcal{M}(a) = \begin{cases} 1 \text{ if there exists } t \in \omega \text{ such that } \mathcal{M}_t(a) = 1 \\ 0 \text{ if there exists } t \in \omega \text{ such that } \mathcal{M}_t(a) = 0 \\ u \text{ otherwise} \end{cases}$$

Using the temporal language, we can describe the reasoning behavior of the agent. The temporal formulas prescribe when the agent should make an inference. But we also want the agent to know (or derive) nothing more than that (we do not want to describe explicitly what it should *not* derive. So we want to make sure that the knowledge of the agent over time satisfies the temporal formulas, but otherwise is *minimal.* The following definition formalizes this.

Definition 12 (Minimal model) *Let Th be a temporal theory and let $\mathcal{M}$ be a partial temporal model. Then $\mathcal{M}$ is a* minimal model *of Th, denoted $\mathcal{M} \models_{min} Th$, if*

- $\mathcal{M} \models Th$, *and*
- *for any partial temporal model $\mathcal{N}$, if $\mathcal{N} \models Th$ and $\mathcal{N} \leq \mathcal{M}$, then $\mathcal{N} = \mathcal{M}$.*

The minimal temporal models of a theory describe the intended possible behavior of the agent over time.

We are interested in the connections between logic programs (with the semantics of stable generated models) and theories in partial temporal logic. It turns out that there is a faithful translation of logic programs into formulas of minimal partial temporal logic. The translation will take rules in normal form to temporal formulas. We will make one additional assumption on rules in normal form.

Assumption 1 *Let $F_1, \ldots, F_n \Rightarrow G_1, \ldots G_m$ be a ground rule in normal form. From now on, we will assume that none of the disjunctions F_i contains a complementary literal pair $\{l, -l\}$ with $l \in Lit^0$.*

This is not a true limitation: when a rule $s \in [P]$ in normal form is given, we can delete any formula F_i that contains a complementary literal pair from the body. This yields a stable equivalent program.

Lemma 9 *For a ground rule $F_1, \ldots, F_n \Rightarrow G_1, \ldots G_m$ in normal form, and interpretations I and M, we have:*
$[I, M] \models F_1, \ldots, F_n \Leftrightarrow$ *for all $1 \leq i \leq n$ there is $l_i \in F_i : l_i \in Lit^0$ and $l_i \in I$, or $l_i = -k$ for some $k \in Lit^0$ and $k \notin M$.*

Proof: "$\Leftarrow$": Let J be an interpretation such that $I \subseteq J \subseteq M$, and let $1 \leq i \leq n$. Suppose there is an l_i in F_i with $l_i \in Lit^0$ and $l_i \in I$. Then $l_i \in J$ so $J \models l_i$. Otherwise there is an l_i in F_i with $l_i = -b$ (and $b \in Lit^0$) and $b \notin M$. But then $b \notin J$ so $J \models l_i$. This means that $J \models F_i$. We conclude that $[I, M] \models F_1, \ldots, F_n$.

"$\Rightarrow$": Suppose there is an F_i with $\forall l \in F_i$: if $l \in Lit^0$ then $l \notin I$ and if $l = -b$ with $b \in Lit^0$ then $b \in M$. Define $J = I \cup \{b \mid -b \in F_i \text{ and } b \in Lit^0\}$. It is easy to see that J is a coherent interpretation with $I \subseteq J \subseteq M$. Take $l \in F_i$. If $l \in Lit^0$ then $l \notin I$ so $l \notin J$ (this uses the assumption that F_i does not contain a complementary literal pair) so $J \not\models l$. If $l = -b$ for $b \in Lit^0$ then $b \in J$ so $J \not\models l$. We conclude that $J \not\models F_i$ so $[I, M] \not\models F_1, \ldots, F_n$. □

Definition 13 *Let $s \in EGLP$ be in normal form, say $s = F_1, \ldots, F_n \Rightarrow G_1, \ldots, G_m$, with $F_i = b_1^i \vee \ldots \vee b_{l(i)}^i \vee -c_1^i \vee \ldots \vee -c_{m(i)}^i$ and $G_j = d_j$ for $1 \leq j \leq j_0$ for a j_0 with $0 \leq j_0 \leq m$ and $G_j = -e_j$ for $j_0 < j \leq m$, where each b_i, c_i, d_i, e_i is a member of Lit^0 and $l(i), m(i), k, j \geq 0$. Define the function trans as follows*

$$trans(F_i) = Cb_1^i \vee \ldots \vee Cb_{l(i)}^i \vee \neg Fc_1^i \vee \ldots \vee \neg Fc_{m(i)}^i$$

$$trans(d_i) = Xd_i$$

$$trans(-e_i) = \neg Xe_i$$

$$trans(s) = trans(F_1) \wedge \ldots \wedge trans(F_n) \rightarrow trans(G_1) \vee \ldots \vee trans(G_m)$$

For $P \subseteq EGLP$ in normal form, $trans(P) = \{trans(s) \mid s \in [P]\}$.

The following proposition shows that the translation is faithful (e.g., it preserves semantics). For a partial model M, define $Lit(M) = \{l \in Lit^0 \mid M(l) = 1\}$.

Proposition 10 *Let $P \subseteq EGLP$ be in normal form, then*

1. *For every stable generated model I of P, generated by the ω-chain $I_0 \subseteq I_1 \subseteq \ldots$ there exists a minimal partial temporal model $(M_t)_{t \in \omega}$ of $trans(P)$ such that $Lit(M_t) = I_t$ for all $t \in \omega$ and $Lit(\lim M) = I$.*

2. *For every minimal partial temporal model $(M_t)_{t \in \omega}$ of $trans(P)$ there exists a stable generated model I, generated by an ω-chain $I_0 \leq I_1 \leq \ldots$ such that $Lit(M_t) = I_t$ for all $t \in \omega$ and $Lit(\lim M) = I$.*

Proof: 1. Let $I_0 \leq I_1 \leq \ldots$ be a P-stable ω-chain generating the stable generated model I. Define the temporal partial model $(\mathcal{M}_t)_{t \in \omega}$ by:

$$\mathcal{M}_t(a) = \begin{cases} 1 & \text{if } a \in I_t \\ 0 & \text{if } \sim a \in I_t \\ u & \text{otherwise} \end{cases}$$

Then this model is indeed conservative and it is easy to see that

$$\lim \mathcal{M}(a) = \begin{cases} 1 & \text{if } a \in I \\ 0 & \text{if } \sim a \in I \\ u & \text{otherwise} \end{cases}$$

• $\mathcal{M} \models trans(P)$: Let $F_1, \ldots F_n \Rightarrow G_1, \ldots, G_m$ be a rule from $[P]$ and let $t \in \omega$. Suppose $(\mathcal{M}, t) \models trans(F_1) \wedge \ldots \wedge trans(F_n)$. Take an F_i and suppose $F_i = l_1 \vee \ldots \vee l_k \vee -m_1 \vee \ldots \vee -m_j$ where $l_i, m_i \in Lit^0$. Then $(\mathcal{M}, t) \models Cl_1 \vee \ldots \vee Cl_k \vee \neg Fm_1 \vee \ldots \vee \neg Fm_j$. If $(\mathcal{M}, t) \models Cl_i$ for some i, then $\mathcal{M}_t(l_i) = 1$ so $l_i \in I_t$. If not, then $(\mathcal{M}, t) \models \neg Fm_i$ for some i, so $\lim \mathcal{M}(m_i) \neq 1$, which means that $m_i \notin I$. With Lemma 9 it follows that $[I_t, \mathcal{M}] \models F_1, \ldots, F_n$ so $I_{t+1} \models G_1 \vee \ldots \vee G_m$, so for some i, we have $I_{t+1} \models G_i$. If $G_i \in Lit^0$ then $G_i \in I_{t+1}$ so $\mathcal{M}_{t+1}(G_i) = 1$ so $(\mathcal{M}, t) \models XG_i$. If $G_i = -b$ where $b \in Lit^0$, then $b \notin I_{t+1}$ so $(\mathcal{M}, t) \models \neg Xb$. We have proved that $(\mathcal{M}, t) \models trans(G_1) \vee \ldots \vee trans(G_m)$. We conclude that $\mathcal{M} \models trans(P)$.
• $\mathcal{M} \models_{min} trans(P)$: Suppose N is a partial temporal model such that $\mathcal{N} < \mathcal{M}$ and $\mathcal{N} \models trans(P)$. Let t_0 be the smallest index for which $\mathcal{N}_{t_0} < \mathcal{M}_{t_0}$ and let $J = Lit(\mathcal{N}_{t_0})$. Note first of all that $t_0 > 0$ (since $\mathcal{M}_0(a) = u$ for all atoms a). As $\mathcal{N}$ is conservative we have $I_{t_0-1} \subseteq J \subseteq I_{t_0}$ and $J \neq I_{t_0}$. Let $F_1, \ldots, F_n \Rightarrow G_1, \ldots, G_m$ be a rule from $[P]$ and suppose $[I_{t_0-1}, I] \models F_1, \ldots, F_n$. Choose an F_i and suppose there is an l in F_i with $l \in Lit^0$ such that $l \in I_{t_0-1}$. Then $(\mathcal{N}, t_0 - 1) \models trans(F_i)$. If not, then by Lemma 9 there is an $l \in F_i$ with $l = -b$, where $b \in Lit^0$ such that $b \notin I$. Then we have that $(\mathcal{M}, t_0 - 1) \models \neg Fb$ and as $\mathcal{N} < \mathcal{M}$ it follows that $(\mathcal{N}, t_0 - 1) \models \neg Fb$. As $(\mathcal{N}, t_0 - 1) \models trans(F_1) \wedge \ldots \wedge trans(F_n)$ and $\mathcal{N} \models trans(P)$ it must be the case that $(\mathcal{N}, t_0 - 1) \models trans(G_1) \vee \ldots \vee trans(G_m)$ so for some i we have $(\mathcal{N}, t_0 - 1) \models trans(G_i)$. If $G_i = b$ where $b \in Lit^0$, then $(\mathcal{N}, t_0 - 1) \models Xb$ so $\mathcal{N}_{t_0}(b) = 1$ so $b \in J$ so $J \models G_i$. Otherwise, if $G_i = -b$ ($b \in Lit^0$) then $(\mathcal{N}, t_0 - 1) \models \neg Xb$ so $\mathcal{N}_{t_0}(b) \neq 1$ so $b \notin J$ so $J \models G_i$. In both cases we have that $J \models G_1 \vee \ldots \vee G_m$. This means that J is an extension of I_{t_0-1} satisfying the heads of the clauses whose body is satisfied in $[I_{t_0-1}, I]$, smaller than I_{t_0} which contradicts the definition of a stable chain. Therefore, such a model $\mathcal{N}$ can not exist, so $\mathcal{M} \models_{min} trans(P)$.

2. Suppose $\mathcal{M} \models_{min} trans(P)$. Define $I = Lit(\lim \mathcal{M})$ and $I_t = Lit(\mathcal{M}_t)$ for $t \in \omega$. Since $\mathcal{M}$ is conservative, we have $I_0 \subseteq I_1 \subseteq \ldots \leq I$ and $I = \bigcup_{t=0}^{\infty} I_t$. We will show that $I_0, I_1, \ldots$ is a P-stable chain.
• First of all, $I_0 = \emptyset$. For if not, we can define a partial model $\mathcal{N}$ by $\mathcal{N}_0(a) = u$ for all atoms a and $\mathcal{N}_t = \mathcal{M}_t$ for $t > 0$. It is easy to see that $\mathcal{N} < \mathcal{M}$ and $\mathcal{N} \models trans(P)$, which contradicts the minimality of $\mathcal{M}$.
• Let $t \in \omega$; we will show that I_{t+1} satisfies the heads of all clauses whose body is satisfied in $[I_t, I]$. Let $F_1, \ldots, F_n \Rightarrow G_1, \ldots G_m$ be a rule in P, and suppose $[I_t, I] \models F_1, \ldots, F_n$. Take an F_i; if there is a $b \in Lit^0$ in F_i such that $b \in I_t$, then $(\mathcal{M}, t) \models Cb$. If not, then by Lemma 9 there must be a $b \in Lit^0$ such that $-b \in F_i$ and $b \notin I$. This means that $\lim \mathcal{M}(b) \neq 1$, so $(\mathcal{M}, t) \models \neg Fb$. In both cases we have that $(\mathcal{M}, t) \models trans(F_1) \wedge \ldots \wedge trans(F_n)$ and as $\mathcal{M} \models trans(P)$ it follows that $(\mathcal{M}, t) \models trans(G_1) \vee \ldots \vee trans(G_m)$. So, for some i it must be the case that $(\mathcal{M}, t) \models trans(G_i)$. If $G_i = b \in Lit^0$, then $(\mathcal{M}, t) \models Xb$, so $\mathcal{M}_{t+1}(b) = 1$ so $b \in I_{t+1}$. If $G_i = -b$ for a $b \in Lit^0$, then $(\mathcal{M}, t) \models \neg Xb$ so $\mathcal{M}_{t+1}(b) \neq 1$ so $b \notin I_{t+1}$. In both cases we have that $I_{t+1} \models G_i$ so $I_{t+1} \models G_1 \vee \ldots \vee G_m$.
• Now suppose J is an extension of I_t satisfying the heads of all applicable clauses and

$J \subseteq I_{t+1}$ but $J \neq I_{t+1}$. Define the partial temporal model $\mathcal{N}$ as follows:

$$\mathcal{N}_{t+1}(a) = \begin{cases} 1 & \text{if } a \in J \\ 0 & \text{if } \sim a \in J \\ u & \text{otherwise} \end{cases}$$

and $\mathcal{N}_s = \mathcal{M}_s$ for $s \neq t+1$. It can easily be checked that $\mathcal{N}$ is a conservative model and that $\mathcal{N} < \mathcal{M}$. Now let $s \in \omega$ and let $F_1, \ldots, F_n \Rightarrow G_1, \ldots, G_m$ be a rule from P. Suppose that $(\mathcal{N}, s) \models trans(F_1) \wedge \ldots \wedge trans(F_n)$. First we will treat the case when $s \neq t$. Since $\mathcal{N} < \mathcal{M}$ we have that $(\mathcal{M}, s) \models trans(F_1) \wedge \ldots \wedge trans(F_n)$ ($\mathcal{N}_s \leq \mathcal{M}_s$ and $\lim \mathcal{N} = \lim \mathcal{M}$). As $\mathcal{M} \models trans(P)$ we get $(\mathcal{M}, s) \models trans(G_1) \vee \ldots \vee trans(G_m)$ and as $\mathcal{M}_{s+1} = \mathcal{N}_{s+1}$ it follows that $(\mathcal{N}, s) \models trans(G_1) \vee \ldots \vee trans(G_m)$. Now suppose $s = t$. Consider an F_i, then $(\mathcal{N}, s) \models trans(F_i)$. If there is a $b \in Lit^0$ in F_i with $(\mathcal{N}, t) \models Cb$ then $(\mathcal{M}, t) \models Cb$ ($\mathcal{N}_t = \mathcal{M}_t$) so $b \in I_t$. Otherwise there must be a $b \in Lit^0$ such that $-b \in F_i$ and $(\mathcal{N}, t) \models \neg Fb$. But then we have $(\mathcal{M}, t) \models \neg Fb$ so $b \notin I$. From Lemma 9 we conclude that $[I_t, I] \models F_1, \ldots, F_n$ so $J \models G_1 \vee \ldots \vee G_m$. Therefore, for some i we have $J \models G_i$. If $G_i = b \in Lit^0$, then $b \in J$, so $\mathcal{N}_{t+1}(b) = 1$ so $(\mathcal{N}, t) \models Xb$. If $G_i = -b$ for a $b \in Lit^0$, then $b \notin J$ so $(\mathcal{N}, t) \models \neg Xb$. In both cases we have that $(\mathcal{N}, t) \models trans(G_i)$ so $(\mathcal{N}, t) \models trans(G_1) \vee \ldots \vee trans(G_m)$. This means that $\mathcal{N} \models trans(P)$ which contradicts the assumption that $\mathcal{M}$ is a minimal model of $trans(P)$. Therefore, such an interpretation J can not exist, and I_{t+1} is a minimal extension of I_t satisfying the heads of applicable clauses. We have proved that $I_0, I_1, \ldots, I$ is a P-stable chain generating I. From the definition of stable chain it follows that I is a model of P. □

We will give an example.

Example 3 (Continued Example) *The temporal translation of Example 1 is:*

$$\{Ga \vee Gb, Ca \rightarrow Gb, \neg Fa \rightarrow Ga\}$$

This theory has only one minimal model:

		time				
		0	1	2	3	...
atoms	a	u	1	1	1	...
	b	u	u	1	1	...

It is easy to see that this model corresponds to the stable generated model of the original program and to the stable chain generating it.

The translation into temporal logic simplifies the proof of the following proposition, which continues the discussion on normal forms of section 3.

Proposition 11 *For every extended general logic program P, there exists a stable equivalent program Q which is in special normal form, i.e., all its rules are of the form $F_1, \ldots, F_m \Rightarrow G_1, \ldots, G_n$ where the F_i and G_j are in $XLit$.*

Proof: Let $P \subseteq$ EGLP. By Proposition 8, P can be transformed into a stable equivalent program P' in normal form. Then we can take $[P']$, the Herbrand instantiation of P', and delete, for every rule, any F_i in the body which contains a complementary literal pair $\{l, -l\}$ with $l \in Lit^0$. Let the result be P''. Now consider any rule $s \in P''$, say $s = F_1, \ldots, F_m \Rightarrow G_1, \ldots, G_n$, and suppose that $F_k = b_1 \vee \ldots \vee b_{l(k)} \vee -c_1 \vee \ldots \vee -c_{m(k)}$ for some $1 \leq k \leq m$. Then $trans(s) = trans(F_1) \wedge \ldots \wedge trans(F_m) \rightarrow trans(G_1) \vee \ldots \vee trans(G_n)$ and $trans(F_k) = Cb_1 \vee \ldots \vee Cb_{l(k)} \vee \neg Fc_1 \vee \ldots \vee \neg Fc_{m(k)}$. But in partial temporal logic, the connectives behave classically: for any temporal model $\mathcal{M}$ and $t \in \omega$ it holds that $(\mathcal{M}, t) \models \alpha \wedge (\beta \vee \gamma) \rightarrow \delta$ if and only if $(\mathcal{M}, t) \models \alpha \wedge \beta \rightarrow \delta$ and $(\mathcal{M}, t) \models \alpha \wedge \gamma \rightarrow \delta$. This means that we may replace $trans(s)$ by the rules

$$\begin{gathered} trans(F_1) \wedge \ldots \wedge Cb_1 \wedge \ldots \wedge \rightarrow trans(G_1) \vee \ldots \vee trans(G_n) \\ \vdots \\ trans(F_1) \wedge \ldots \wedge Cb_{l(k)} \wedge \ldots \wedge \rightarrow trans(G_1) \vee \ldots \vee trans(G_n) \\ trans(F_1) \wedge \ldots \wedge \neg Fc_1 \wedge \ldots \wedge \rightarrow trans(G_1) \vee \ldots \vee trans(G_n) \\ \vdots \\ trans(F_1) \wedge \ldots \wedge \neg Fc_{m(k)} \wedge \ldots \wedge \rightarrow trans(G_1) \vee \ldots \vee trans(G_n) \end{gathered}$$

Let T be the temporal theory obtained from $trans(P'')$ by performing the above operation until there are no more disjunctions on the left hand side of any implication. Then T contains only formulas of the form $Ca_1 \wedge \ldots \wedge Ca_n \wedge \neg Fb_1 \wedge \ldots \wedge \neg Fb_m \rightarrow Xd_1 \vee \ldots \vee Xd_k \vee \neg Xe_1 \vee \ldots \vee \neg Xe_l$. But such a formula is the translation of the rule $a_1, \ldots, a_n, -b_1, \ldots, -b_m \Rightarrow d_1, \ldots, d_k, -e_1, \ldots, -e_l$. This means we can find a program Q such that $trans(Q)$ is exactly T. This Q has special normal form. Furthermore, P is stable equivalent to P', which is equivalent (stable, but also classically) to $[P']$, which is in turn equivalent to P''. Stable models of P'' correspond to minimal temporal models of $trans(P'')$, which is equivalent to T (and therefore has the same minimal temporal models). But $T = trans(Q)$, so the minimal temporal models of T correspond to stable generated models of Q. We conclude that Q is stable equivalent to P. $\Box$

We have shown that logic programs can be translated (via a normal form) to formulas of minimal partial temporal logic in a modular way (the translation of a program is the union of the translations of its sequents). The translation preserves the semantics (in the sense of Proposition 10). But how about the other way, i.e., is there a modular translation of minimal partial temporal logic into logic programs (endowed with the stable generated semantics)? For this question to be answered positively, it is sufficient (and necessary) that temporal formulas in a normal form can be translated. Essentially, the temporal language is a propositional language using the temporal atoms as its propositional atoms. Therefore, any temporal formula is equivalent to a formula in conjunctive normal form, and we may focus on the disjuncts. As only conservative models are considered, it is the case that $G\alpha$ is equivalent to $X\alpha$. We may thus concentrate on disjunctions containing atoms $C\alpha$, $\neg C\alpha$, $F\alpha$, $\neg F\alpha$, $X\alpha$, and $\neg X\alpha$. It is easy to see that temporal operators distribute over conjunction and disjunction, e.g. $C(\alpha \wedge \beta)$ is equivalent to $C\alpha \wedge C\beta$, and $C(\alpha \vee \beta)$ is equivalent to $C\alpha \vee C\beta$. This means that an atom $C\alpha$ is equivalent to $C\alpha'$, where α' is a conjunctive normal form (or disjunctive normal form) of α. We may then distribute the C operator over the conjunctions and disjunctions. So we can even restrict ourselves to disjunctions of temporal atoms over propositional literals. Any disjunction not containing atoms of the form Cl or $\neg Fl$ can be translated faithfully to a sequent. Consider a disjunction of the following form (for clarity, we include only one of each sort of temporal atom):

$$\neg Ca \vee Fb \vee Xc \vee \neg Xd$$

where $a, b, c, d \in Lit^0$. This is equivalent to the implication

$$Ca \wedge \neg Fb \longrightarrow Xc \vee \neg Xd$$

This is just the translation of the sequent

$$a, -b \Rightarrow c, d$$

So how about the other two sorts of atoms? An easy example of a formula that can not be translated is Ca. Its only minimal partial temporal model assigns 1 to a and u to any other atoms, for every timepoint. However, any stable chain starts with the empty set. So, no logic program has a stable chain equivalent to this minimal partial temporal model. One might think that the only formulas that can not be translated, are formulas that have a minimal partial temporal model $\mathcal{M}$ for which $\mathcal{M}_0$ does not correspond to the empty set. This not true, however, as witnessed by the following set of formulas:

$$\begin{array}{ll}
(1) & Xa \\
(2) & Ca \rightarrow Xb \\
(3) & Ca \wedge Cb \rightarrow Xc \\
(4) & Ca \wedge \neg Cb \rightarrow Xd \\
(5) & Ca \wedge Cb \wedge Cc \rightarrow Xd
\end{array}$$

The conjunction of these formulas has two minimal models:

$$\mathcal{M}: \begin{array}{c|ccccc}
 & 0 & 1 & 2 & 3 & \dots \\
\hline
a & u & 1 & 1 & 1 & \dots \\
b & u & u & 1 & 1 & \dots \\
c & u & u & u & 1 & \dots \\
d & u & u & 1 & 1 & \dots
\end{array} \qquad \mathcal{N}: \begin{array}{c|ccccc}
 & 0 & 1 & 2 & 3 & \dots \\
\hline
a & u & 1 & 1 & 1 & \dots \\
b & u & 1 & 1 & 1 & \dots \\
c & u & u & 1 & 1 & \dots \\
d & u & u & u & 1 & \dots
\end{array}$$

These models can not both correspond to stable chains of the same program. Since they have the same limit model, the same sequents are applicable with respect to $[\emptyset, Lit(\lim \mathcal{M})]$ as with respect to $[\emptyset, Lit(\lim \mathcal{N})]$. Since $\{a\}$ must be an extension of the empty set satisfying the heads of clauses whose bodies is satisfied in $[\emptyset, Lit(\lim \mathcal{M})]$, it is impossible that $\{a, b\}$ is a minimal such extension. Both of these minimal models start with a partial model corresponding to the empty set. Note that formula (4) above uses an atom $\neg Cb$ on the left of the implication (corresponding to Cb in the disjunction).

5 Disjunctive Defaults

In this section we introduce several versions of disjunctive default logic. Disjunctive default logic is a modification of Reiter's default logic ([Re80]), meant to deal more aptly with disjunctive information. By a first-order default d we understand an expression of the form $d := (\ \alpha : \beta_1, \ldots, \beta_n\ /\ \gamma)$, where α, β_i, γ are first-order formulas of the language $L(\sigma; \neg, \wedge, \vee, \forall, \exists)$. Classical negation is denoted by $\neg$, and $Cn(X)$ denotes the classical deductive closure of a set X of first-order sentences. A default d is said to be open if every of the formulas in d is quantifier-free. Let us denote $pr(d) := \alpha$ (prerequisite of d), $just(d) = \{\beta_1, \ldots, \beta_n\ \}$ the justification of d, and $cons(d) = \gamma$ the consequence of d. In the following we consider only open defaults and their closed instantiations. Then, we may assume that all formulas under consideration are propositional.

When disjunctive information is added, a commitment has to be made to one of the disjuncts. A disjunctive default is an expression of the form $(\ \alpha : \beta_1,\ \ldots, \beta_n\ /\ \gamma_1 | \ldots |\ \gamma_n\ )$, where α, β_i and γ_j are formulas. Disjunctive defaults were introduced in [GLPT91], from which we repeat the following definition.

Definition 14 *[GLPT91] Let D be a set of disjunctive defaults and E a set of sentences. $Min_E(D)$ is the set of all minimal deductively closed sets M satisfying the following condition for every $(\alpha : \beta_1, \ldots, \beta_m/\gamma_1|\ldots|\gamma_n) \in D$: if $\alpha \in M$, and $\neg\beta_1, \ldots, \neg\beta_m \notin E$, then $\{\gamma_1, \ldots, \gamma_n\} \cap M \neq \emptyset$. E is an extension of D if $E \in Min_E(D)$.*

This definition is clearly a generalization of the original fix-point definition of an extension. However, there is also a semi-constructive definition, which is equivalent in the non-disjunctive case. It turns out that if we generalize this semi-constructive definition (which is more in the spirit of stable generated models and minimal temporal models), we get a somewhat different notion (see [EH97]).

Definition 15 *Let D be a set of disjunctive defaults. A deductively closed set E of sentences is a* generated extension *of D if there is a sequence $E_0 \subseteq E_1 \subseteq \ldots \subseteq E_n \subseteq \ldots$ of deductively closed sets of sentences such that*

1. *$E_0 = Cn(\emptyset)$;*
2. *E_{n+1} is a minimal extension of E_n satisfying the following closure condition $Cl_{gen}(E)$: if $(\ \alpha : \beta_1, \ldots, \beta_k\ /\ \gamma_1|\ldots|\gamma_l\ ) \in D$, $\alpha \in E_n$, and $\neg\beta_1, \ldots, \neg\beta_k \notin E$ then $\{\gamma_1, \ldots, \gamma_l\} \cap E_{n+1} \neq \emptyset$.*

and $E = \bigcup_{n<\omega} E_n$.

In the case of non-disjunctive defaults, both definitions above coincide with the original definition of [Re80]. In [EH97] it was shown that extensions of a disjunctive default theory in the sense of [GLPT91] are generated extensions. The converse, however, is not true.

Example 4 *This corresponds to the earlier example, $D = \{(:\ /\ a|b)\ (a :\ /\ b)\ (\ : \neg a\ /\ a)\}$. Then $E = Cn(\{a, b\})$ is a generated extension of D: $E_0 = Cn(\emptyset)$, $E_1 = Cn(\{a\})$, $E_2 = Cn(\{a, b\})$. But E is not an extension. It does satisfy the closure condition with respect to itself, but $Cn(\{b\})$ also satisfies it, so $Min_E(D) = \{Cn(\{b\})\}$.*

In the remaining part of this section we investigate relations between default logic and extended generalized logic programs. There is a simple relation between normal logic programs and a class of special default theories (see [GL90]). In [EH97] it is shown that every super logic program P (see [Prz96]) can be translated into a default theory $tr(P)$ such that the generated extensions of $tr(P)$ and the stable generated models of P coincide in a well-defined sense. We recall this result. Let $r := a_1, \ldots, a_l, \neg K_1, \ldots, \neg K_m \Rightarrow c_1, \ldots, c_n$ be a super rule; where $K_j = b_1^j \wedge \ldots \wedge b_{p_j}^j$, for $1 \leq j \leq m$. We translate such a rule to the following disjunctive default rule: $tr(r) := (a_1 \wedge \ldots \wedge a_l : \neg K_1, \ldots, \neg K_m \ / \ c_1 | \ldots | c_n)$. Let $At(E) = \{a : a$ is a ground atom with $E \models a\}$.

Proposition 12 *Let P be a super logic program, $tr(P)$ the default theory given by the above translation of P.*
If a set I of ground atoms is a stable generated model of P then there is a generated extension E of $tr(P)$ such that $I = At(E)$. If E is a generated extension of $tr(P)$, then $At(E)$ is a stable generated model of P.

Obviously, every disjunctive default theory of a special form can be translated faithfully into a super logic program.

We now consider more complex logic programs and defaults and use for this purpose the framework of partial logic with two negations. We want to embed default logic in stable generated semantics of extended generalized logic programs. In contrast to default logic, logic programming on the one hand is based on partial logic, and on the other hand commits to disjunctions. Therefore, we need a new version of the notion of an extension of a default theory. We use partial logic with one negation ($\neg$) over coherent partial interpretations whose closure operation we have denoted by C_0 (see section 2). A set X is said to be *constructively closed* if $C_0(X) = X$.

Lemma 13 *Let X be a set of literals and α a propositional sentence from $L(\vee, \wedge, \neg)$. Then: $\alpha \in C_0(X) \Leftrightarrow X \models \alpha$.*

Proof: The claim of this lemma follows from the fact that X is a (partial) model of $C_0(X)$. [1]. □

For a set X of sentences from $L(\vee, \wedge, \neg)$, let $Lit(X)$ be the set of literals which are members of X.

Definition 16 *Let D be a set of disjunctive defaults. A constructively closed set E of sentences is a* constructively generated extension *of D if there is a sequence $E_0 \subseteq E_1 \subseteq \ldots \subseteq E_n \subseteq \ldots$ of constructively closed sets of sentences such that*

1. *$E_0 = C_0(\emptyset)$;*

2. *E_{n+1} is a minimal constructively closed extension of E_n satisfying the following closure condition $Cl_{cgen}(E)$: if $(\ \alpha : \beta_1, \ldots, \beta_k \ / \ \gamma_1 \mid \ldots \mid \gamma_l) \in D$, $\alpha \in E_n$, and $\neg\beta_1, \ldots, \neg\beta_k \notin E$ then $\gamma_i \in C_0(Lit(E_{n+1}))$ for some $1 \leq i \leq l$.*

[1] This is not true for Cn(X).

3. $E = \bigcup_{n<\omega} E_n$.

The language in which defaults are expressed, contains classical negation ($\neg$), whereas logic programs contain weak ($-$) and strong ($\sim$) negation. Semantically, classical negation should correspond to strong negation. When translating sentences from the classical language to the language of logic programming, all classical negations should therefore be replaced by strong negation (and vice versa when translating the other way). To avoid notational clutter, we will not explicitly show this translation. The reader is instructed to change classical negation signs to strong negation when moving from the default language to logic programming (and vice versa).

Let $d := (\alpha : \beta_1, \ldots, \beta_n / \gamma_1 \mid \ldots \mid \gamma_l)$ be a default rule. Define

$$lp(d) = \alpha, - \sim \beta_1, \ldots, - \sim \beta_n \Rightarrow \gamma_1, \ldots, \gamma_l$$

and $lp(D) = \{lp(d) | d \in D\}$. We will show that this provides a faithfull embedding of default logic based on constructively generated extensions, into logic programming based on stable generated models. We first prove a lemma.

Lemma 14 *Let $I_0 \subseteq I_1 \subseteq \ldots$ be a sequence of Herbrand interpretations with $I = \bigcup_{n<\omega} I_n$, and let $E_0 \subseteq E_1 \subseteq \ldots$ be a sequence of constructively closed sets of sentences with $E = \bigcup_{n<\omega} E_n$. For a set of (disjunctive) defaults D, define*

$$\begin{aligned} \Omega_n &= \{s \in lp(D) \mid [I_n, I] \models Bs\} \\ \Delta_n &= \{d \in D \mid \textit{ if } d = (\alpha : \beta_1, \ldots, \beta_k / \gamma_1 \mid \ldots \mid \gamma_l) \textit{ then } \alpha \in E_n \textit{ and } \neg\beta_1, \ldots, \neg\beta_k \notin E\} \end{aligned}$$

If $E_n = C_0(I_n)$ for all $n < \omega$, then

$$\Delta_n = \{d \mid lp(d) \in \Omega_n\}, n = 1, 2, 3 \ldots$$

Proof: Suppose $d \in \Delta_n$, with $d = (\alpha : \beta_1, \ldots, \beta_k / \gamma_1 \mid \ldots \mid \gamma_l)$. Then $\alpha \in E_n = C_0(I_n)$, so $I_n \models \alpha$ which implies $[I_n, I] \models \alpha$ (since α does not contain weak negation). Furthermore, $\neg\beta_1, \ldots, \neg\beta_k \notin E = C_0(I)$ (this last equality is easily checked). Thus, we have $I \not\models \neg\beta_1, \ldots, \neg\beta_k$ so $I \models - \sim \beta_1, \ldots, - \sim \beta_k$ from which it follows that $[I_n, I] \models - \sim \beta_1, \ldots, - \sim \beta_k$. We have proved that $[I_n, I] \models B(lp(d))$, so $lp(d) \in \Omega_n$.

Conversely, suppose that $lp(d) \in \Omega_n$. If, again, $d = (\alpha : \beta_1, \ldots, \beta_k / \gamma_1 \mid \ldots \mid \gamma_l)$, then $[I_n, I] \models \alpha, - \sim \beta_1, \ldots, - \sim \beta_k$. It follows that $I_n \models \alpha$, so $\alpha \in E_n$. As $[I_n, I] \models - \sim \beta_1, \ldots, - \sim \beta_k$, we have that $I \not\models \neg\beta_1, \ldots, \neg\beta_k$. But then $\neg\beta_1, \ldots, \neg\beta_k \notin E$, which implies $d \in \Delta_n$. $\square$

Now we are ready to state and prove the main result.

Proposition 15 *Let D be a (disjunctive) default theory.*

1. *For every constructively generated extension E of D there is a stable generated model I of $lp(D)$ such that $I = Lit(E)$. Furthermore, if $(E_n)_{n<\omega}$ is the sequence of Definition 16 leading to E, then $(I_n)_{n<\omega}$ defined by $I_n = Lit(E_n)$ is an $lp(D)$-stable chain generating I.*

2. *For every stable generated model I of $lp(D)$ there is a constructively generated extension E of D such that $Lit(E) = I$. Furthermore, if $(I_n)_{n<\omega}$ is an $lp(D)$-stable chain generating I, then the sequence $(E_n)_{n<\omega}$ defined by $E_n = C_0(I_n)$ is a sequence leading to E in the sense of Definition 16.*

Proof: 1. Let E be a constructively generated extension of $lp(D)$ then there is a sequence $E_0 \subseteq E_1 \subseteq \ldots \subseteq E_n \subseteq \ldots$, $E = \bigcup_{i<\omega} E_i$ satisfying the conditions of Definition 16. Let $I_n = Lit(E_n)$. We show that $(I_n)_{n<\omega}$ is a stable chain for $lp(D)$ generating $I = Lit(E)$. Obviously, $I_0 = \emptyset$, since $Lit(E_0) = \emptyset$. It is easy to check that $E_n = C_0(I_n)$ so we may apply Lemma 14.

Suppose $[I_n, I] \models B(lp(d))$, for $d = (\alpha : \beta_1, \ldots, \beta_n/\gamma_1 \mid \ldots \mid \gamma_l)$. Then $lp(d) \in \Omega_{n+1}$, so $d \in \Delta_{n+1}$. As E is a constructively generated extension, $\gamma_i \in E_{n+1}$ for some $1 \leq i \leq l$. But this implies that $Lit(E_{n+1}) \models \gamma_i$, so $I_{n+1} \models \gamma_i$, yielding $I_{n+1} \models \bigvee H(lp(d))$.

I_{n+1} is also minimal with this property: assume, $I_n \subseteq J \subseteq I_{n+1}$ and J satisfies the heads of the sequents in the set Ω_{n+1}. Then, $C_0(J)$ satisfies the condition $Cl_{cgen}(E)$, and since E_{n+1} is by assumption minimal, it follows $J = I_{n+1}$.

2. Let I be stable generated model of $lp(D)$, $I = \bigcup_{n<\omega} I_n$, $(I_n)_{n<\omega}$ a stable chain. Let $E_n = C_0(I_n)$; we show that $E = \bigcup_{n<\omega} E_n$ is a constructively generated extension of D. Obviously, $E_0 = C_0(\emptyset)$. We have to show: E_{n+1} is a minimal extension of E_n satisfying the closure condition $Cl_{cgen}(E)$. Let $d \in D$, $d := (\alpha : \beta_1, \ldots, \beta_m/\gamma_1 \mid \ldots \mid \gamma_l)$, $\alpha \in C_0(E_n)$, and $\neg\beta_1, \ldots, \neg\beta_m \notin E$. Using Lemma 14, $d \in \Delta_n$ so $lp(D) \in \Omega_n$. As I is a stable generated model, $I_{n+1} \models \gamma_i$ for some $1 \leq i \leq l$. From this one concludes $\gamma_i \in C_0(I_{n+1})$, so $\gamma_i \in E_{n+1}$. E_{n+1} is a minimal extension with the closure property $Cl_{cgen}(E)$. Assume, $E_n \subseteq E' \subseteq E_{n+1}$, E' satisfies $Cl_{cgen}(E)$. Then it holds $I_n \subseteq Lit(E') \subseteq I_{n+1}$. Since $(I_n)_{n<\omega}$ is a stable chain it follows $Lit(E') = I_{n+1}$, hence $E' = C_0(Lit(E')) = E_{n+1}$. □.

It is not clear whether there exists a natural translation of arbitrary extended logic programs into default logic with constructively generated extensions; it seems that this default mechanism is more restricted than extended generalized logic programs. But, for a special class of extended logic programs this can be done. Let $r = F_1, \ldots, F_m, -\sim G_1, -\sim G_n \Rightarrow H_1, \ldots, H_s$ be a program rule such that the formulas F_i, G_j, H_l do not contain weak negation. Programs consisting of rules of this kind are called extended normal disjunctive logic programs. Note that this is a real restriction. All sequents of $EGLP$ have the normal form of Proposition 8. In a sequent in normal form, there may be weakly negated literals in the head, and in the body, literals from Lit^0 and weakly negated literals may simultaneously occur in one of the conjuncts. In extended normal disjunctive logic programs, both of these possibilities may not occur.

We define the following translation into default logic:

$$def(r) = F_1 \wedge \ldots \wedge F_m : G_1, \ldots, G_n/H_1 \vee \ldots \vee H_k$$

For a set P of program rules of this form let $def(P) = \{def(r); r \in P\}$.

Proposition 16 *Let P be a normal extended disjunctive logic program and $def(P)$ its default translation. If the set E of sentences is a constructively generated extension* *of $def(P)$ then $Lit(E)$ is a stable generated model of P. Conversely, for every stable generated model I of P there is a strongly generated extension E of $def(P)$ such that $Lit(E) = I$. In both cases, the sequences generating I, respectively E, are pointwise corresponding in the sense of Proposition 15.*

Proof: From Proposition 15, we know that constructively generated extensions of $def(P)$ correspond to the stable generated models of $lp(def(P))$. By Lemma 4 we may transform $lp(def(P)) = F_1 \wedge \ldots \wedge F_m, - \sim G_1, \ldots, - \sim G_n \Rightarrow H_1 \vee \ldots \vee H_k$ into the rule $F_1, \ldots, F_m, - \sim G_1, - \sim G_n \Rightarrow H_1, \ldots, H_s$ without changing the stable generated semantics. But this latter rule is just the original rule r! We have proved that the constructively generated extensions of $def(P)$ correspond to the stable generated models of P. □

For any set D of disjunctive defaults, $lp(D)$ is a normal disjunctive logic program with corresponding semantics. But by the above proposition, $def(lp(D))$ is a set of defaults whose semantics corresponds to $lp(D)$. This means that D and $def(lp(D))$ have the same constructively generated extensions. But they are not the same: a default $d = (\alpha : \beta_1, \ldots \beta_n/\gamma_1 \mid \ldots \mid \gamma_l$ in D corresponds to the default $(\alpha : \beta_1, \ldots \beta_n/\gamma_1 \vee \ldots \vee \gamma_l$ in $def(lp(D))$. This means that for constructively generated extensions, we do not need disjunctive defaults. This result should not come as a surprise, since constructively closed sets must commit to one of the disjuncts.

6 Conclusion

Generalizing the approach of [EH97] we investigated the relations between stable generated models and minimal partial temporal models. A translation was given from programs into theories in partial temporal logic, which preserves semantics. Furthermore, we introduced the notions of (constructively) generated extensions of a disjunctive default theory and studied embedddings of default theories in logic programming and vice versa.

This established a close relation between three different approaches to non-monotonic reasoning: (disjunctive) default logic, stable generated models of logic programs and minimal temporal partial models. In all three approaches there is a notion of generating conclusions starting from scratch, on the basis of the final outcomes. This connection adds further evidence to the naturalness of stable generated models as a possible semantics for generalized logic programs.

References

[EH97] J. Engelfriet and H. Herre. Generated Models and Extensions of Nonmonotonic Systems. In J. Małuszyński, editor, *Logic Programming, Proceedings of the 1997 International Symposium*, MIT Press, 1997, pp. 85–99.

[En96] J. Engelfriet. Minimal Temporal Epistemic Logic. *Notre Dame Journal of Formal Logic* **37**(2):233–259, 1996.

[ET93] J. Engelfriet and J. Treur. A Temporal Model Theory for Default Logic. In M. Clarke, R. Kruse, S. Moral, editors, *Proceedings ECSQARU'93*, Lecture Notes in Computer Science **747**, Springer-Verlag, 1993, pp. 91–96. Extended and revised version to appear as "An Interpretation of Default Logic in Minimal Temporal Epistemic Logic" in *Journal of Logic, Language and Information.*

[ET94] J. Engelfriet and J. Treur. Temporal Theories of Reasoning. In C. MacNish, D. Pearce, L.M. Pereira, editors, *Proceedings JELIA'94*, Lecture Notes in Artificial Intelligence **838**, Springer-Verlag, 1994, pp. 279–299; Also in *Journal of Applied Non-Classical Logics* **5**(2):239–261, 1995.

[GL88] M. Gelfond and V. Lifschitz. The Stable Model Semantics for Logic Programming. In R.A. Kowalski and K.A. Bowen, editors, *Proceedings ILPS'88*, MIT Press, 1988, pp. 1070–1080.

[GL90] M. Gelfond and V. Lifschitz. Logic Programs with Classical Negation. In D. Warren and P. Szeredi, editors, *Proceedings ICLP'90*, MIT Press, 1990, pp. 579–597.

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

[GLPT91] M. Gelfond, V. Lifschitz , H. Przymusinska and M. Truszczynski. Disjunctive Defaults. In J. Allen, R. Fikes, and E. Sandewall, editors, *Proceedings KR'91*, Morgan Kaufmann, 1991, pp. 230–237.

[HJW95] H. Herre, J. Jaspars and G. Wagner. Partial Logics with two Kinds of Negation as a Foundation for Knowledge-based Reasoning. Report Nr. 12 (1995). Institut für Informatik, Universtät Leipzig. To appear in D. Gabbay and H. Wansing, editors, *What is negation?*, Oxford University Press, 1997.

[HW97] H. Herre and G. Wagner. Stable Models are Generated by a Stable Chain. *Journal of Logic Programming* **30** (2):165–177, 1997.

[HW97a] H. Herre and G. Wagner. Semantics for Extended Generalized Logic Programs. Report. Nr. 7 (1997). Institut für Informatik, Universität Leipzig, ISSN 1430-3701

[LW92] V. Lifschitz and T.C. Woo. Answer Sets in General Nonmonotonic Reasoning. In B. Nebel, C. Rich, and W. Swartout, editors, *Proceedings KR'92*, Morgan Kaufmann, 1992, pp. 603–614.

[Prz91] T.C. Przymusinski. Stable Semantics for Disjunctive Programs. *New Generation Computing* **9**:401–424, 1991.

[Prz96] T.C. Przymusinski. Super Logic Programs and Negation as Belief. In R. Dyckhoff, H. Herre, P. Schroeder-Heister, editors, *Proceedings of the 5th International Workshop on Extensions of Logic Programming*, Lecture Notes in Artificial Intelligence **1050**, Springer-Verlag, pp. 229–236.

[Re80] R. Reiter. A Logic for Default Reasoning. *Artificial Intelligence* **13**:81–132, 1980.