

ON THE COMPUTATION OF THE DISJUNCTIVE WELL-FOUNDED SEMANTICS

C. A. JOHNSON
Computer Science Department
University of Keele
Staffordshire, ST5 5BG
England
email : chrisj@cs.keele.ac.uk

Abstract. A characterisation of the disjunctive well-founded semantics (DWFS) is given in terms of the Gelfond-Lifschitz transformation. This characterisation is used to develop a top-down method of testing DWFS membership, employing a hyperresolution-like operator and quasi cyclic trees to handle minimal model processing. A flexible bottom-up method of computing the DWFS is also given which admits the use of a powerful reduction operator. For finite propositional databases, all of our methods run in polynomial space.

Keywords. Disjunctive logic programs, disjunctive deductive databases, disjunctive well-founded semantics, top-down computation, bottom-up computation.

Introduction

Over the last few years there has been a great deal of interest in the study of declarative semantics for deductive databases and logic programs. In the disjunctive case, semantics have for example been based upon minimal models for positive databases [Gr86], perfect models for stratified databases [Pr88], and for unstratified databases we have disjunctive stable models [Pr91], DWFS [Bra94], WF^3 [Ba91], GDWFS [Ba92] and others. Surveys of these semantics are to be found in [Lb92, Di95].

The disjunctive well-founded semantics (DWFS) was introduced by Brass and Dix [Ar97, Bra94, Bra95, Bra98, Bra98a, Bra99] as the weakest semantics which satisfies certain desirable properties, including the generalised principle of partial evaluation. In [Bra98, Bra98a], they gave a bottom-up characterisation of DWFS using an extension of the Gelfond-Lifschitz transformation. This bottom-up characterisation is employed in the DISLOP project [Ar97] to develop a bottom-up method of computing DWFS using the (bottom-up) methods of [Ni96] for handling minimal model reasoning.

In this paper we present the following results concerning DWFS. Firstly in Section 2 we define DWFS, and present an iterative formulation of DWFS using only Gelfond-Lifschitz transformations of the initial database. This provides a simpler formulation of DWFS, and allows us to show that DWFS can be characterised as the (unique) minimal set satisfying a closure property defined in terms of the Gelfond-Lifschitz transformation.

We are also able to easily show that DWFS is strictly weaker than the disjunctive stationary model semantics.

In Section 3 we focus on positive databases, and (re)introduce the concept of a deduction tree [Jo93, Jo98a] which is based on a hyperresolution-like operator, and facilitates top-down query processing in such databases. In Section 4 we introduce quasi cyclic trees. These are variants of cyclic trees [Jo96, Jo98, Jo99], and enable us to perform top-down minimal model reasoning in databases resulting from applications of the Gelfond-Lifschitz transformation. In Section 5 we combine both deduction and quasi cyclic trees to develop our top-down method for testing DWFS membership. In Sections 6 and 7 we present a bottom-up method of computing the DWFS, which is more flexible than the construction employed in [Bra98, Bra98a], and in particular which admits a very powerful reduction operator. For finite propositional databases, all of the methods developed operate in space that is polynomial in the size of the underlying language.

In Section 1 we introduce the basic terminology and prove some preliminaries. A first order program/database represents the set of its ground instances, and thus for the most part we restrict our attention to propositional programs in a countable language. In Section 8, we briefly discuss the problems in lifting our results to the first order level. Our results are applicable to disjunctive logic programs and deductive databases, although our top-down method requires an assumption to be made that would be more natural in the deductive database setting. We will thus throughout refer to deductive databases rather than logic programs.

§1. Preliminaries

Throughout $\mathcal{L} = \{P_0, P_1, \dots\}$ denotes a countable propositional language. P, Q, \dots (possibly with subscripts) will denote arbitrary predicates in \mathcal{L} , and $\mathcal{P}, \mathcal{Q}, \dots$ will denote finite subsets of \mathcal{L} . A *literal* is a predicate or its negation. If I is a set of literals, then $\bar{I} = \{\neg K : K \in I\}$. A rule is a formula C of the form $A_1 \wedge A_2 \wedge \dots \wedge A_n \wedge \neg A_{n+1} \wedge \dots \wedge \neg A_{n+h} \rightarrow B_1 \vee B_2 \vee \dots \vee B_r$, where each $A_i, B_j \in \mathcal{L}$ and $r > 0$. $\text{antec}(C) = \{A_1, A_2, \dots, A_n\}$, $\mathcal{N}(C) = \{A_{n+1}, A_{n+2}, \dots, A_{n+h}\}$ and $\text{conseq}(C) = \{B_1, B_2, \dots, B_r\}$.

A (deductive) database is a set of rules, which throughout will be denoted by T . T is said to be *positive* iff $\mathcal{N}(C) = \emptyset$ for each $C \in T$.

The following two definitions capture the idea of reducing a database using information that is already known to be true. The first is a special case of the second.

1.1 Definition [Ge88]. If C is a rule, then $\text{pos}(C) = \bigwedge \text{antec}(C) \rightarrow \bigvee \text{conseq}(C)$.

Given $N \subseteq \mathcal{L}$, let $T|_g N$ denote the *Gelfond-Lifschitz transformation*,

$$T|_g N = \{\text{pos}(C) : C \in T, \mathcal{N}(C) \cap N = \emptyset\}.$$

1.2 Definition [Bra98a]. Let $D = D^+ \cup D^-$, where D^+ is a set of non-empty finite disjuncts of predicates, and D^- is a set of negative literals. Let T/D be formed from T by

- (i) removing any rule C for which $D^+ \models \bigvee \mathcal{N}(C)$, and
- (ii) for each (remaining) rule C , replacing $\mathcal{N}(C)$ by $\mathcal{N}(C) - \overline{D^-}$.

Note that if $N \subseteq \mathcal{L}$, then $T|_g N = T/Dis_N$, where $Dis_N = N \cup \{\neg A : A \in \mathcal{L} - N\}$ [Bra98a].

1.3 Definition. Let $D = D^+ \cup D^-$ be of the form given in Definition 1.2, then D is said to be *closed* iff whenever $P_1 \vee P_2 \vee \dots \vee P_k \in D^+$ and $\neg P_1 \in D^-$, then $P_2 \vee \dots \vee P_k \in D^+$.

Note that if D is closed, then the order in which the operations in Definition 1.2 are carried out is immaterial. Notice also that if D is closed, then D is consistent (since D is assumed not to contain the empty disjunct).

1.4 Definition [Bra98a]. Let $Dis(T) = Dis^+(T) \cup Dis^-(T)$ where

$$Dis^+(T) = \{\bigvee \mathcal{P} : T/\mathcal{L} \models \bigvee \mathcal{P}\}, \text{ and}$$

$$Dis^-(T) = \{\neg P : (\forall N \subseteq \mathcal{L})(N \models Dis^+(T) \implies T/Dis_N \models_{min} \neg P)\}.$$

Notice that $T/\mathcal{L} = \{C \in T : \mathcal{N}(C) = \emptyset\}$, and $Dis(T)$ does not contain the empty disjunct. Note that in the definition of $Dis^+(T)$, \mathcal{P} ranges over finite subsets of \mathcal{L} . This results in no loss of information, since for any $\mathcal{I} \subseteq \mathcal{L}$, if $T/\mathcal{L} \models \bigvee \mathcal{I}$, then there must be some finite subset $\mathcal{I}' \subseteq \mathcal{I}$ such that $T/\mathcal{L} \models \bigvee \mathcal{I}'$.

\models_{min} refers to minimal model reasoning, thus $T/Dis_N \models_{min} \neg P$ iff P is false in all minimal models of T/Dis_N .

The above definition can be viewed as follows. Firstly if $\bigvee \mathcal{P} \in Dis^+(T)$, then $\bigvee \mathcal{P}$ is certainly true in T , whence the models of T form a subset of those of $Dis^+(T)$. Secondly if, for any model N of $Dis^+(T)$ (and hence of T), P is false in all (minimal) models of the reduced database, then it is safe to assume $\neg P$.

1.5 Lemma. $Dis(T)$ is closed.

Proof. In the definition of $Dis^-(T)$, take $N = \mathcal{L}$, then $T/Dis_N = T/\mathcal{L}$. If $T/\mathcal{L} \models P_1 \vee P_2 \vee \dots \vee P_k$ and $T/\mathcal{L} \models_{min} \neg P_1$, then trivially $T/\mathcal{L} \models P_2 \vee \dots \vee P_k$. ■

The following technical lemma is vital to the results of later sections.

1.6 Lemma. Let $N \subseteq \mathcal{L}$, and suppose that $D = D^+ \cup D^-$ is of the form given in Definition 1.2, and is closed.

(a) If $N \cap \overline{D^-} = \emptyset$, then

$$(T/D)/Dis_N = \{pos(C) : C \in T, D^+ \not\models \bigvee \mathcal{N}(C), \mathcal{N}(C) \cap N = \emptyset\}.$$

(b) If $N' = N - \overline{D^-}$, then $(T/D)/Dis_N = (T/D)/Dis_{N'}$.

(c) If $N \cap \overline{D^-} = \emptyset$ and $N \models D^+$, then $(T/D)/Dis_N = T/Dis_N$.

Proof. Note first that

$$(T/D)/Dis_N = \{pos(C) : C \in T, D^+ \not\models \bigvee \mathcal{N}(C), \text{ and } (\mathcal{N}(C) - \overline{D^-}) \cap N = \emptyset\}.$$

Let $C \in T$.

(a). Since $N \cap \overline{D^-} = \emptyset$, we have that $(\mathcal{N}(C) - \overline{D^-}) \cap N = \emptyset$ iff $\mathcal{N}(C) \cap N = \emptyset$.

(b). Clearly $(\mathcal{N}(C) - \overline{D^-}) \cap N' \subseteq (\mathcal{N}(C) - \overline{D^-}) \cap N \subseteq (\mathcal{N}(C) - \overline{D^-}) \cap (N' \cup \overline{D^-}) \subseteq (\mathcal{N}(C) - \overline{D^-}) \cap N'$.

(c). By part (a), $pos(C) \in (T/D)/Dis_N$ iff $D^+ \not\models \bigvee \mathcal{N}(C)$ and $\mathcal{N}(C) \cap N = \emptyset$. Now $N \models D^+$, thus if $N \cap \mathcal{N}(C) = \emptyset$ then $D^+ \not\models \bigvee \mathcal{N}(C)$. Thus $pos(C) \in (T/D)/Dis_N$ iff $\mathcal{N}(C) \cap N = \emptyset$ iff $pos(C) \in T/Dis_N$. ■

§2. The disjunctive well-founded semantics

In this section we define the DWFS (in terms of the bottom-up formulation of [Bra98, Bra98a]). We then (Theorems 2.4 and 2.5) use Lemma 1.6 to form a characterisation of DWFS directly in terms of Gelfond-Lifschitz transformations of T . This will prove central to the results of later sections. We also show (Theorems 2.6 and 2.7) that DWFS can be viewed as the unique minimal set satisfying a certain closure operator, again defined in terms of the Gelfond-Lifschitz transformation. Finally (Theorem 2.9) we employ this characterisation to show that DWFS is strictly weaker than the disjunctive stationary semantics.

2.1 Definition [Bra98a]. Let $D_0 = \emptyset$, $D_{\alpha+1} = Dis(T/D_\alpha)$ and for limit ordinals γ , $D_\gamma = \bigcup_{\alpha < \gamma} D_\alpha$. Then the *disjunctive well-founded semantics*, $DWFS = \bigcup_\alpha D_\alpha$.

Notice that each $D_{\alpha+1}$ is closed and consistent (by Lemma 1.5). Henceforth we will refer to the above as the “ D_α construction” of DWFS.

2.2 Proposition [Bra98a]. If $\rho \leq \nu$, then $D_\rho \subseteq D_\nu$.

This monotonicity of the sequence $(D_\alpha : \alpha = 0, 1, 2, \dots)$ and the fact that each $D_{\alpha+1}$ is closed immediately yields the following corollary.

2.3 Corollary. Each D_α (and in particular DWFS) is closed and consistent.

The definition of DWFS given above is based upon computing a set of disjuncts and negative literals, using these to reduce the database with the $/$ -operator, and then repeating the process. This is ideal for a bottom-up computation of DWFS. However for a top-down approach, it is essential to express each of the iterative steps that constructs DWFS directly in terms of T rather than T/D_α . This also gives a simpler characterisation.

2.4 Theorem. $D_{\alpha+1}^+ = \{\bigvee \mathcal{P} : T|_g(\mathcal{L} - \overline{D_\alpha^-}) \models \bigvee \mathcal{P}\}$.

Proof. It suffices to show that $(T/D_\alpha)/\mathcal{L} = T|_g(\mathcal{L} - \overline{D_\alpha^-})$.

For $C \in T$, $\text{pos}(C) \in (T/D_\alpha)/\mathcal{L}$ iff $D_\alpha^+ \not\models \bigvee \mathcal{N}(C)$ and $\mathcal{N}(C) \subseteq \overline{D_\alpha^-}$. But since D_α is closed and consistent, if $\mathcal{N}(C) \subseteq \overline{D_\alpha^-}$, then $D_\alpha^+ \not\models \bigvee \mathcal{N}(C)$. ■

2.5 Theorem. $D_{\alpha+1}^- = \{\neg Q : (\forall N \subseteq \mathcal{L} - \overline{D_\alpha^-})(N \models D_{\alpha+1}^+ \implies T|_g N \models_{\min} \neg Q)\}$.

Proof. Suppose that $\neg Q \in D_{\alpha+1}^-$ and $N \subseteq \mathcal{L} - \overline{D_\alpha^-}$ with $N \models D_{\alpha+1}^+ \supseteq D_\alpha^+$. By Lemma 1.6(c) we have that $T|_g N = (T/D_\alpha)/\text{Dis}_N$, whence by the definition of $D_{\alpha+1}^-$ we have $T|_g N \models_{\min} \neg Q$.

For the converse, suppose that $\neg Q \notin D_{\alpha+1}^-$, then we may find an $N^* \models D_{\alpha+1}^+$ with $(T/D_\alpha)/\text{Dis}_{N^*} \not\models_{\min} \neg Q$. Let $N = N^* - \overline{D_\alpha^-}$, then since $D_\alpha \subseteq D_{\alpha+1}$ and $D_{\alpha+1}$ is closed we have that $N \models D_{\alpha+1}^+ \supseteq D_\alpha^+$, whence by Lemma 1.6(b/c), we have that $(T/D_\alpha)/\text{Dis}_{N^*} = (T/D_\alpha)/\text{Dis}_N = T|_g N$. Thus N contradicts the properties of the right hand side. ■

Theorems 2.6 and 2.7 below show that DWFS is in fact the unique minimal set satisfying the closure operator suggested by the above two results.

2.6 Theorem. $\text{DWFS}^+ = \{\bigvee \mathcal{P} : T|_g(\mathcal{L} - \overline{\text{DWFS}^-}) \models \bigvee \mathcal{P}\}$, and $\text{DWFS}^- = \{\neg Q : (\forall N \subseteq \mathcal{L} - \overline{\text{DWFS}^-})(N \models \text{DWFS}^+ \implies T|_g N \models_{\min} \neg Q)\}$.

Proof. This is immediate from the fact that $\text{DWFS} = D_\alpha = D_{\alpha+1}$ for some α . ■

2.7 Theorem. Suppose that $D = D^+ \cup D^-$ is of the form given in Definition 1.2, and is closed. Suppose also that $D^+ \supseteq \{\bigvee \mathcal{P} : T|_g(\mathcal{L} - \overline{D^-}) \models \bigvee \mathcal{P}\}$, and $D^- \supseteq \{\neg Q : (\forall N \subseteq \mathcal{L} - \overline{D^-})(N \models D^+ \implies T|_g N \models_{\min} \neg Q)\}$.

Then $D \supseteq \text{DWFS}$.

Proof. We show by induction that $D \supseteq D_\alpha$ for each α .

Suppose that $D \supseteq D_\beta$, and that $T|_g(\mathcal{L} - \overline{D^-}) \models \bigvee \mathcal{P}$. Since $\overline{D^-} \supseteq \overline{D_\beta^-}$, we have

$T|_g(\mathcal{L} - \overline{D_\beta^-}) \subseteq T|_g(\mathcal{L} - \overline{D^-}) \models \bigvee \mathcal{P}$. Thus $\bigvee \mathcal{P} \in D^+$.

Suppose now that $D \supseteq D_\beta \cup D_{\beta+1}^+$, and that $(\forall N \subseteq \mathcal{L} - \overline{D_\beta^-})(N \models D_{\beta+1}^+ \implies T|_g N \models_{\min} \neg Q)$. Let $N \subseteq \mathcal{L} - \overline{D^-}$ with $N \models D^+$, then $N \subseteq \mathcal{L} - \overline{D_\beta^-}$ and $N \models D_{\beta+1}^+$, whence $T|_g N \models_{\min} \neg Q$. Thus $\neg Q \in D^-$. ■

We can use the above result to show that DWFS is strictly weaker than the disjunctive stationary model semantics.

2.8 Definition [Pr91]. Let I be a consistent set of literals, $I^+ = \{P \in \mathcal{L} : P \in I\}$, and $I^- = \{\neg P : P \in \mathcal{L}, \neg P \in I\}$.

I is said to be a *disjunctive stationary model* of T iff I^+ is a minimal model of $T|_g(\mathcal{L} - \overline{I^-})$ and $\mathcal{L} - \overline{I^-}$ is a minimal model of $T|_g I^+$.

Let $D_I = \{\bigvee \mathcal{P} : I^+ \models \bigvee \mathcal{P}\} \cup I^-$.

The above characterisation of disjunctive stationary models is shown in [Pr91, Remark 3.1] to be equivalent to the definition given in [Pr91].

2.9 Theorem (a) If I is a disjunctive stationary model of T , then D_I satisfies the conditions of Theorem 2.7.

(b) $\bigcap \{D_I : I \text{ is a disjunctive stationary model of } T\}$ satisfies the conditions of Theorem 2.7.

Proof (a). Suppose that $T|_g(\mathcal{L} - \overline{D_I^-}) \models \bigvee \mathcal{P}$, then since $I^+ \models T|_g(\mathcal{L} - \overline{I^-}) = T|_g(\mathcal{L} - \overline{D_I^-})$ we have that $\bigvee \mathcal{P} \in D_I^+$.

Suppose that $(\forall N \subseteq \mathcal{L} - \overline{D_I^-})(N \models D_I^+ \implies T|_g N \models_{\min} \neg Q)$. But now $I^+ \subseteq \mathcal{L} - \overline{I^-} = \mathcal{L} - \overline{D_I^-}$ and $I^+ \models D_I^+$, whence $T|_g I^+ \models_{\min} \neg Q$. However, $\mathcal{L} - \overline{I^-}$ is a minimal model of $T|_g I^+$, whence $\neg Q \in I^- = D_I^-$.

(b) This follows immediately from part (a) and the (trivial) fact that the property defined in the conditions of Theorem 2.7 is preserved under intersection. ■

Theorem 2.9 shows that DWFS is weaker than the disjunctive stationary model semantics. Notice that DWFS is strictly weaker. For example if $T = \{\neg A \rightarrow P \vee Q, B \rightarrow Q, A \vee B\}$, then T is stratified, whence the disjunctive stationary models are total and coincide with the perfect and disjunctive stable models [Pr91], and are $\{A, \neg P, \neg Q, \neg B\}$ and $\{B, Q, \neg A, \neg P\}$. In particular P is false in all such models, whereas $\neg P \notin \text{DWFS} = \{A \vee B, A \vee Q\}$.

Theorem 2.9 naturally suggests the following (open) question. “If $N \subseteq \mathcal{L}$ is a minimal model of DWFS^+ , can $N \cup \text{DWFS}^-$ always be extended to a disjunctive stationary model of T ?”

§3. Computation in positive databases

In Theorem 2.4 we saw that the computation of $D_{\alpha+1}^+$ employs the positive database $T|_g(\mathcal{L} - \overline{D_\alpha})$. Derivability in such databases can be characterised using deduction trees [Jo93] (Definition 3.5 below) which in turn are based upon the hyperresolution-like operator suggested in part (b) of the following lemma.

3.1 Lemma [Jo93]. Suppose that T is positive and $\mathcal{P} \subseteq \mathcal{L}$.

- (a) If $T \models \bigvee \mathcal{P}$, then we may find a $C \in T$ with $\text{conseq}(C) \subseteq \mathcal{P}$ and $\text{antec}(C) \cap \mathcal{P} = \emptyset$.
- (b) Suppose that $C \in T$ with $\text{conseq}(C) \subseteq \mathcal{P}$ and $\text{antec}(C) \cap \mathcal{P} = \emptyset$. Then $T \models \bigvee \mathcal{P}$ iff $T \models A \vee \bigvee \mathcal{P}$ for each $A \in \text{antec}(C)$.

3.2 Corollary. If $\bigvee \mathcal{P} \in D_{\alpha+1}^+$, then we may find a $C \in T$ such that $\text{conseq}(C) \subseteq \mathcal{P}$, $\text{antec}(C) \cap \mathcal{P} = \emptyset$ and $\mathcal{N}(C) \subseteq \overline{D_\alpha}$.

3.3 Corollary. Suppose that $\mathcal{P} \subseteq \mathcal{L}$ and $C \in T$ with $\text{conseq}(C) \subseteq \mathcal{P}$, $\text{antec}(C) \cap \mathcal{P} = \emptyset$ and $\mathcal{N}(C) \subseteq \overline{D_\alpha}$.

If $\beta > \alpha$, then $\bigvee \mathcal{P} \in D_\beta^+$ iff for each $A \in \text{antec}(C)$, $A \vee \bigvee \mathcal{P} \in D_\beta^+$.

Testing membership of $D_{\alpha+1}^+$ requires testing derivability in the positive database $T|_g(\mathcal{L} - \overline{D_\alpha})$, which in turn can be achieved by iterating Lemma 3.1(b). This iteration can best be viewed as a tree structure, as is illustrated by the following example.

3.4 Example. Suppose that T contains the following rules:

- | | |
|---|----------------------|
| 1. $D \wedge E \wedge F \rightarrow A \vee B$ | 4. $G \vee B$ |
| 2. $H \rightarrow B \vee C \vee E$ | 5. $C \vee D$ |
| 3. $G \rightarrow F$ | 6. $C \vee E \vee H$ |

and we wish to decide whether $T \models \bigvee \mathcal{P}$, where $\mathcal{P} = \{A, B, C\}$. The consequent of rule 1 is contained in \mathcal{P} , thus by Lemma 3.1(b), $T \models \bigvee \mathcal{P}$ iff $T \models D \vee \bigvee \mathcal{P}$, $T \models E \vee \bigvee \mathcal{P}$ and $T \models F \vee \bigvee \mathcal{P}$. This is depicted using the tree \mathcal{T}_1 in Figure 3.4(i).

The root node is the goal to be proven, the “rule node” indicates that rule 1 has been applied, and the lower nodes indicate the subgoals which result from the application of the rule. For instance the node D indicates that we need to show that $T \models D \vee \bigvee \mathcal{P}$.

The subgoal $D \vee \bigvee \mathcal{P}$ is “immediately solved” since T contains $C \vee D$ (rule 5), thus we move on to consider $E \vee \bigvee \mathcal{P}$. Here we can apply rule 2, and thus again, $T \models E \vee \bigvee \mathcal{P}$ iff $T \models H \vee E \vee \bigvee \mathcal{P}$, which in turn is represented by \mathcal{T}_2 in Figure 3.4(i).



Figure 3.4(i).

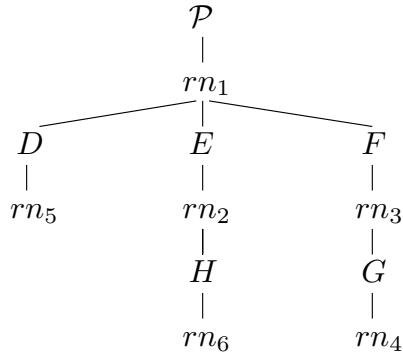


Figure 3.4(ii).

Again the rule nodes rn_5 and rn_2 denote applications of these rules. rn_5 has no child nodes precisely because rule 5 has an empty antecedent.

The lower node labelled H indicates that we need to show that $T \models H \vee E \vee \bigvee \mathcal{P}$, which is trivially true by virtue of rule 6. The subgoal $F \vee \bigvee \mathcal{P}$ is solved by applying rule 3 and then rule 4, yielding the final tree depicted in Figure 3.4(ii). This then shows that $T \models \bigvee \mathcal{P}$.

3.5 Definition [Jo93, Jo98a]. Suppose that T is positive and that $\mathcal{P} \subseteq \mathcal{L}$, then a *deduction tree* for \mathcal{P} in T is a finite tree containing goal nodes and rule nodes satisfying the following conditions.

- (i) Each goal node has the form $?\bigvee \mathcal{Q}$, where $\mathcal{Q} \subseteq \mathcal{L}$. $?\bigvee \mathcal{Q}$ has a single child node which is a rule node. If $?\bigvee \mathcal{Q}$ is not the root node, then its parent node is a rule node.

The root node (at the top of the tree) is a goal node labelled with $?\bigvee \mathcal{P}$.

- (ii) Each rule node is labelled with a rule $C \in T$ (written rn_C). The parent of rn_C is a goal node of the form $?\bigvee \mathcal{Q}$, where $\text{conseq}(C) \subseteq \mathcal{Q}$ and $\text{antec}(C) \cap \mathcal{Q} = \emptyset$. For each $K \in \text{antec}(C)$, rn_C has a child goal node of the form $?\bigvee(\{K\} \cup \mathcal{Q})$.

(iii) Each leaf node is a rule node.

Notice that condition (iii) follows from condition (i), and is included simply for clarity.

3.6 Theorem. If T is positive and $\mathcal{P} \subseteq \mathcal{L}$, then $T \models \bigvee \mathcal{P}$ iff \mathcal{P} has a deduction tree in T .

Proof. The implication from right to left is trivial using Lemma 3.1(b).

Suppose that $T \models \bigvee \mathcal{P}$. Let $\text{EXT}(T) = \{C \in T : \text{antec}(C) = \emptyset\}$, and $\text{INT}(T) = T - \text{EXT}(T)$. Since \mathcal{L} is countable we may assume an enumeration of $\text{INT}(T)$ of the form $\text{INT}(T) = (C_i : i = 0, 1, 2, \dots)$. We construct the deduction tree \mathcal{T} iteratively. Initially set \mathcal{T} to consist simply of the (root) goal node $?\bigvee \mathcal{P}$.

Suppose that \mathcal{T} is given and that $?\bigvee \mathcal{Q}$ is a goal leaf node in \mathcal{T} . If $\text{EXT}(T) \models \bigvee \mathcal{Q}$, then pick $C \in \text{EXT}(T)$ with $C \subseteq \mathcal{Q}$, and terminate the branch by appending rn_C . If $\text{EXT}(T) \not\models \bigvee \mathcal{Q}$, then pick i such that (i) $\text{conseq}(C_i) \subseteq \mathcal{Q}$ and $\text{antec}(C_i) \cap \mathcal{Q} = \emptyset$, and (ii) for each $j < i$, C_j does not satisfy condition (i). Such a rule must exist by Lemma 3.1(a), and the fact that $T \models \bigvee \mathcal{Q}$ (since $\mathcal{Q} \supseteq \mathcal{P}$) and $\text{EXT}(T) \not\models \bigvee \mathcal{Q}$.

Suppose that the above construction does not terminate, then an infinite branch \mathcal{B} must be generated. Let $\mathcal{C} = \bigcup \{\mathcal{Q} : ?\bigvee \mathcal{Q} \text{ occurs on } \mathcal{B}\}$, then $\text{EXT}(T) \not\models \bigvee \mathcal{C}$, and for each $C \in \text{INT}(T)$, if $\text{conseq}(C) \subseteq \mathcal{C}$ then $\text{antec}(C) \cap \mathcal{C} \neq \emptyset$. This then shows that $\mathcal{L} - \mathcal{C} \models T \wedge \neg \bigvee \mathcal{P}$, thus contradicting the fact that $T \models \bigvee \mathcal{P}$. ■

A top-down method of testing derivability in positive databases using deduction trees is given in [Jo93, Jo98a]. This method is “branch at a time”, and therefore for finite propositional databases can be made to operate in space which is linear in the size of the underlying language (as a result of the fact that duplicate predicates are not added along a given branch).

§4. Quasi cyclic trees

Employing Theorem 2.5 to test membership of D_{α}^- requires us to apply minimal model reasoning to positive databases of the form $T|_g N$. In this section we define quasi cyclic trees, which provide a characterisation of minimal models of such databases. Quasi cyclic trees are variants of cyclic trees introduced in [Jo96] as a means of testing minimal model membership in positive databases.

We first provide an example to illustrate the structure of quasi cyclic trees.

4.1 Example. Suppose that T consists of the following rules

- | | | |
|--|--|--|
| 1. $Q_2 \wedge Q_3 \wedge \neg R_1 \rightarrow Q_1 \vee Q_5$ | 2. $Q_1 \wedge \neg R_2 \rightarrow Q_2$ | 3. $S_2 \wedge \neg R_3 \rightarrow Q_3$ |
| 4. $R_1 \rightarrow Q_1 \vee Q_2 \vee Q_4$ | 5. $S_2 \vee R_5$ | 6. $S_1 \rightarrow Q_3 \vee Q_2$ |
| 7. $R_1 \vee R_7$ | 8. $Q_5 \rightarrow Q_2$. | 9. $Q_2 \wedge \neg S_1 \rightarrow Q_1$ |

and suppose that we wished to test whether Q_1 lies in some minimal model M of the database $T|_gN$, where $N = \{S_1\}$. If so, then we may find a rule $C \in T$ with $\mathcal{N}(C) \cap N = \emptyset$ and such that $M - \{Q_1\} \not\models \text{pos}(C)$. The only possibilities for C are rules 1 and 4. Suppose that C is rule 1, then $\{Q_1, Q_2, Q_3\} \subseteq M \subseteq \mathcal{L} - \{Q_5\}$. This is depicted in the first level of the tree \mathcal{T}_1 (Figure 4.1).

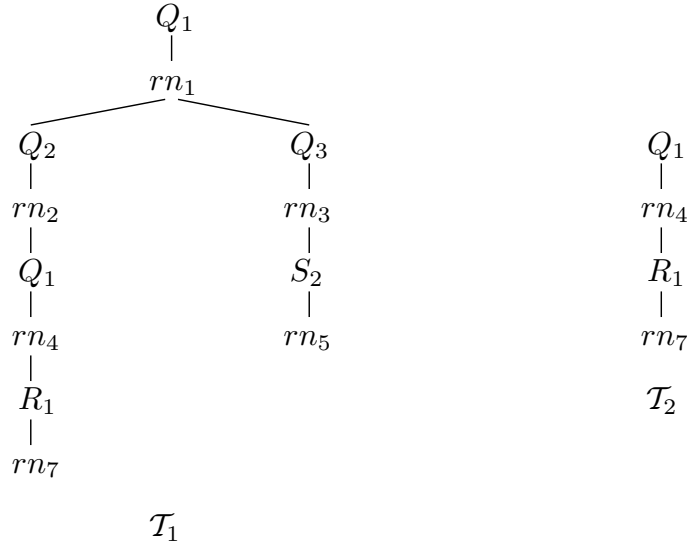


Figure 4.1.

We can then apply the same reasoning to find rules which witness that $M - \{Q_2\} \not\models T|_gN$ and $M - \{Q_3\} \not\models T|_gN$, and in this case the only possibilities are rules 2 and 3 respectively, thus yielding the next level in the tree \mathcal{T}_1 . This then tells us that $\{Q_1, Q_2, Q_3, S_2\} \subseteq M \subseteq \mathcal{L} - \{Q_5\}$. When we come to look at the cycle along the left hand branch there is no point in looking for a rule that witnesses that $M - \{Q_1\} \not\models T|_gN$ (since we have already done so), thus we look for a rule that witnesses that $M - \{Q_1, Q_2\} \not\models T|_gN$. In this case the only possibility is rule 4, and rules 7 and 5 can then terminate the branches yielding the final constraint $\{Q_1, Q_2, Q_3, S_2, R_1\} \subseteq M \subseteq \mathcal{L} - \{Q_5, R_5, Q_4, R_7\}$.

But then note finally that using the rules in the tree we can easily show that if $M^* \models T|_gN$ with $M^* \cap \{Q_5, R_5, Q_4, R_7\} = \emptyset$, then $M^* \supseteq \{Q_1, Q_2, Q_3, S_2, R_1\}$.

Applying rule 4 (instead of rule 1) at the first step would have yielded the tree \mathcal{T}_2 and the constraint $\{Q_1, R_1\} \subseteq M \subseteq \mathcal{L} - \{Q_2, Q_4, R_7\}$. Again if $M^* \models T|_gN$ with $M^* \cap \{Q_2, Q_4, R_7\} = \emptyset$, then $\{Q_1, R_1\} \subseteq M^*$.

Thus Q_1 belongs to some minimal model of $T|_gN$ iff $T|_gN \not\models \bigvee\{Q_5, R_5, Q_4, R_7\}$ or $T|_gN \not\models \bigvee\{Q_2, Q_4, R_7\}$.

Notice the features of the above construction. We are assuming that the predicates within the tree are taken from our intended minimal model M of $T|_gN$. At each stage we take the cycle CYC along the current branch, and look for a rule $C \in T$ such that $N \cap \mathcal{N}(C) = \emptyset$ and $M - CYC \not\models pos(C)$, ie., $antec(C) \subseteq M - CYC$ and $conseq(C) \cap (M - CYC) = \emptyset$. Since $M \models pos(C)$ we must also have that $CYC \cap conseq(C) \neq \emptyset$. These features are captured in the following definition.

4.2 Definition [Jo96]. Let \mathcal{T} be a finite tree containing predicate nodes and rule nodes satisfying the following conditions.

- (i) The root node (at the top of \mathcal{T}) is a predicate node.
- (ii) If n is a predicate node then n is labelled with a predicate, denoted by $lab(n)$. n has a single child node which is a rule node. If n is not the root node, then its parent node is a rule node.
- (iii) If rn is a rule node, then rn is labelled with a rule $C \in T$ (written rn_C). For each $K \in antec(C)$, rn_C has a (predicate) child node labelled with K .

Then we make the following definitions.

- (1) Suppose that n is a predicate node in \mathcal{T} . Let

$$CYC(n) = \{lab(n') \mid n' \text{ is a predicate node, } n' \geq n, \text{ and} \\ \exists n'' \geq n', n'' \text{ is a predicate node, } lab(n'') = lab(n)\}.$$

If the child node of n is rn_C , then define $\mathcal{O}(rn_C) = conseq(C) - CYC(n)$.

- (2) Let $\mathcal{O}(\mathcal{T}) = \bigcup\{\mathcal{O}(rn_C) \mid rn_C \text{ is a rule node in } \mathcal{T}\}$,
 $\mathcal{N}(\mathcal{T}) = \bigcup\{\mathcal{N}(C) \mid rn_C \text{ is a rule node in } \mathcal{T}\}$, and
 $Pred(\mathcal{T}) = \{lab(n) \mid n \text{ is a predicate node in } \mathcal{T}\}$.
- (3) Let $P \in \mathcal{L}$ and \mathcal{T} be a tree satisfying the conditions above, then \mathcal{T} is said to be a *quasi cyclic tree* for P in T iff
 - (a) The root node is labelled with P ,
 - (b) $Pred(\mathcal{T}) \cap \mathcal{O}(\mathcal{T}) = \emptyset$,
 - (c) whenever rn_C is a rule node in \mathcal{T} with parent n , then
 - (i) $conseq(C) \cap CYC(n) \neq \emptyset$,
 - (ii) $antec(C) \cap CYC(n) = \emptyset$, and
 - (d) \mathcal{T} has no predicate leaf node.

Again condition (d) follows from condition (ii) and is included simply for clarity. Notice that condition (c) is inherently top-down. In [Jo96, Theorem 5.1.14] it is shown that in finite propositional databases, the length of any branch through a (quasi) cyclic tree is bounded by $|\mathcal{L}| * (|\mathcal{L}| + 1) / 2$ (where \mathcal{L} is the underlying language). In [Jo96, Jo98], top-down methods for traversing cyclic (and hence quasi cyclic) trees are presented. These methods are “branch at a time”, and hence for finite propositional databases operate in space which is quadratic in the size of the underlying language.

Theorems 4.3 and 4.6 show that quasi cyclic trees capture minimal model membership in $T|_gN$.

4.3 Theorem [Jo96, Theorem 5.1.10]. Suppose that \mathcal{T} is a quasi cyclic tree in T , $\mathcal{N}(\mathcal{T}) \cap N = \emptyset$, and $M \models T|_gN$ with $M \cap \mathcal{O}(T) = \emptyset$. Then $Pred(\mathcal{T}) \subseteq M$.

In order to employ quasi cyclic trees in query processing we will, in an infinite language, need to ensure that each predicate has only a finite number of such trees, and that such trees can be constructed finitely. With this in mind we introduce the following concepts.

4.4 Definition. Let $P, Q \in \mathcal{L}$, then we write $P \preceq Q$ iff there is a rule $C \in T$ such that $P \in \text{antec}(C)$ and $Q \in \text{conseq}(C)$.

4.5 Definition (a) T is *well-founded* iff

- (i) for each $P \in \mathcal{L}$ there are only a finite number of rules $C \in T$ for which $P \in \text{conseq}(C)$, and
 - (ii) there is no infinite sequence $(P_i : i = 0, 1, 2, \dots)$ of distinct predicates in \mathcal{L} such that for each i , $P_i \succeq P_{i+1}$.
- (b) T is *cyclic-finite* iff each predicate $P \in \mathcal{L}$ has only a finite number of quasi cyclic trees in T .

If T satisfies condition (a)(i) of the above definition, then it is easy to give other equivalent characterisations of property (a)(ii).

4.6 Lemma. Suppose that T satisfies condition (a)(i) of Definition 4.5, then T is well-founded iff there there is no infinite sequence $(P_i : i = 0, 1, 2, \dots)$ of distinct predicates in \mathcal{L} such that $(\forall i > 0)(\exists j < i)(P_j \succeq P_i)$.

4.7 Theorem. Suppose that T is well-founded. If M is a minimal model of $T|_gN$ and $P \in M$, then we may find a quasi cyclic tree \mathcal{T} for P in T such that $Pred(\mathcal{T}) \subseteq M \subseteq \mathcal{L} - \mathcal{O}(T)$ and $\mathcal{N}(\mathcal{T}) \cap N = \emptyset$.

Proof. We construct \mathcal{T} iteratively. Initially \mathcal{T} consists of just the root node P . Suppose that n is a predicate leaf node in \mathcal{T} , then since $\text{CYC}(n) \subseteq M$, we may find a rule $C \in T$ such that $\mathcal{N}(C) \cap N = \emptyset$ and $M - \text{CYC}(n) \not\models \text{pos}(C)$, whence we may extend n with rn_C (and the corresponding child nodes from $\text{antec}(C)$).

Suppose that the above construction does not terminate, then an infinite branch $\mathcal{B} = n_0 > rn_{C_0} > n_1 > rn_{C_1} > n_2 > rn_{C_2} > \dots$ is generated. Let $\mathcal{C} = \{\text{lab}(n_r) : r = 0, 1, 2, \dots\}$. Define a sequence $(P_i : i = 0, 1, 2, \dots)$ of distinct predicates in \mathcal{L} as follows. Suppose that we are given $(P_j : j < i)$ such that $\{P_j : j < i\} = \{\text{lab}(n_r) : r < r_i\}$ and

$lab(n_{r_i}) \notin \{P_j : j < i\}$. Then set $P_i = lab(n_{r_i})$. Notice that $(\forall i > 0)(\exists j < i)(P_j \succeq P_i)$.

We are thus required to show that the sequence $(P_i : i = 0, 1, 2, \dots)$ can be extended indefinitely. If not then \mathcal{C} is finite, whence pick $u < v$ such that $\mathcal{C} = \{lab(n_r) : r \leq u\}$ and $\{lab(n_r) : u < r\} = \{lab(n_i) : u < i \leq v\}$. Now $lab(n_v) \in \{lab(n_r) : r \leq u\}$, thus $lab(n_{v+1}) \notin CYC(n_v) \supseteq \{lab(n_i) : u \leq i \leq v\} \supseteq \{lab(n_r) : u < r\}$, a contradiction. ■

Note that the above theorem clearly fails without well-foundedness. For example if $T = \{P_i \vee P_{i+1} : i = 0, 1, 2, \dots\} \cup \{P_{i+1} \rightarrow P_i : i = 0, 1, 2, \dots\}$, then T has a single minimal model, namely $M = \{P_i : i = 0, 1, 2, \dots\}$, but there is no quasi cyclic tree \mathcal{T} in T for which $\mathcal{O}(\mathcal{T}) = \emptyset$.

4.8 Theorem. If T is well-founded then T is cyclic-finite.

Proof. Let $\mathcal{T}(P)$ be the set of “partial” quasi cyclic trees for P in T , ie., trees satisfying the conditions of Theorem 4.2 with the exception that predicate leaf nodes are permitted. \mathcal{T} is an immediate extension of \mathcal{T}' if it is formed from \mathcal{T}' by appending a rule node (and the corresponding child nodes there-of) to a branch in \mathcal{T}' which currently terminates in a predicate node. By condition (a)(i) of Definition 4.5, each tree in $\mathcal{T}(P)$ has only a finite number of immediate extensions, thus if the theorem fails, we may construct a sequence $(\mathcal{T}_i : i = 0, 1, 2, \dots)$ such that each \mathcal{T}_{i+1} is an immediate extension of \mathcal{T}_i . But then we may find an infinite branch passing through $\bigcup\{\mathcal{T}_i : i = 0, 1, 2, \dots\}$, which (as in the proof of Theorem 4.7) contradicts the well-foundedness of T . ■

It should be noted that the converse of Theorem 4.8 does not hold. However it is clear that databases in which well-foundedness fails are prime candidates for the failure of cyclic-finiteness. Moreover, without well-foundedness it is clearly impossible to construct quasi cyclic trees finitely. Condition (a)(i) (of Definition 4.5) ensures that there are only a finite number of ways to (immediately) extend a given partial tree, and conditions (a)(i+ii) ensure that such a sequence of extensions cannot extend indefinitely.

We thus view well-foundedness as a natural structural property of deductive databases which fairly closely matches our requirements for constructing quasi cyclic trees.

In order to perform top-down testing of membership in $DWFS^-$, we will need the following characterisation.

4.9 Theorem. Let T be well-founded. Suppose that D is of the form given in Definition 1.2, and D is closed. Then the following are equivalent.

- (a) $(\forall N \subseteq \mathcal{L} - \overline{D^-})(N \models D^+ \implies T|_g N \models_{min} \neg Q)$,
- (b) for each quasi cyclic tree \mathcal{T} for Q in T , either $D^+ \models \bigvee \mathcal{N}(\mathcal{T})$ or $T|_g(\mathcal{L} - \mathcal{N}(\mathcal{T}) \cup \overline{D^-}) \models \bigvee \mathcal{O}(\mathcal{T})$.

Proof (a) \rightarrow (b). Let \mathcal{T} be a quasi cyclic tree for Q in T with $D^+ \not\models \bigvee \mathcal{N}(\mathcal{T})$. By the closure of D we have $\mathcal{L} - \mathcal{N}(\mathcal{T}) \cup \overline{D^-} \models D^+$, whence by hypothesis, $T|_g(\mathcal{L} - \mathcal{N}(\mathcal{T}) \cup$

$\overline{D^-} \models_{min} \neg Q$.

If $T|_g(\mathcal{L} - \mathcal{N}(\mathcal{T}) \cup \overline{D^-}) \not\models \bigvee \mathcal{O}(\mathcal{T})$, then pick a minimal model M of $T|_g(\mathcal{L} - \mathcal{N}(\mathcal{T}) \cup \overline{D^-})$ such that $M \cap \mathcal{O}(\mathcal{T}) = \emptyset$. By Theorem 4.3, $Q \in Pred(\mathcal{T}) \subseteq M$, thus yielding a contradiction.

(b) \rightarrow (a). Suppose $N \subseteq \mathcal{L} - \overline{D^-}$ with $N \models D^+$ and $T|_g N \not\models_{min} \neg Q$.

By Theorem 4.7 we may find a quasi cyclic tree \mathcal{T} for Q in T such that $T|_g N \not\models \bigvee \mathcal{O}(\mathcal{T})$ and $\mathcal{N}(\mathcal{T}) \cap N = \emptyset$. Since $N \models D^+$ we cannot have that $D^+ \models \bigvee \mathcal{N}(\mathcal{T})$. Finally $T|_g(\mathcal{L} - \mathcal{N}(\mathcal{T}) \cup \overline{D^-}) \subseteq T|_g N$, whence $T|_g(\mathcal{L} - \mathcal{N}(\mathcal{T}) \cup \overline{D^-}) \not\models \bigvee \mathcal{O}(\mathcal{T})$. ■

4.10 Corollary. Let T be well-founded, then the following are equivalent.

(a) $\neg Q \in D_{\alpha+1}^-$,

(b) for each quasi cyclic tree \mathcal{T} for Q in T , either $\bigvee \mathcal{N}(\mathcal{T}) \in D_{\alpha+1}^+$ or $T|_g(\mathcal{L} - \mathcal{N}(\mathcal{T}) \cup \overline{D_\alpha^-}) \models \bigvee \mathcal{O}(\mathcal{T})$.

Proof. $D = D_\alpha^- \cup D_{\alpha+1}^+$ is closed since $D_\alpha^- \subseteq D_{\alpha+1}^-$, and $D_{\alpha+1}$ is closed. ■

§5. Top-down testing of DWFS membership

In this section we combine deduction trees with quasi cyclic trees to develop a top-down method of testing DWFS membership. Theorem 2.4 and Corollary 4.10 tells us that we will need to prove statements of the form $T|_g(\mathcal{L} - \mathcal{N} \cup \overline{D_\alpha^-}) \models \bigvee \mathcal{P}$ (where $\mathcal{N} = \emptyset$ in the application of Theorem 2.4). We will denote this by a goal node of the form $(\bigvee \mathcal{P}, \mathcal{N})$. By Lemma 3.1(b) such nodes are attacked by finding an appropriate rule $C \in T$, and we will denote such applications by rule nodes rn_C . We will also need to prove goals of the form $\neg Q \in DWFS$, and by Corollary 4.10 such goals are attacked by generated quasi cyclic trees for Q in T . We will use a “rule node” $rn_{\rightarrow \neg Q}$ to denote the generation of such trees.

5.1 Definition. A *goal node* can either have the form (i) $(\bigvee \mathcal{P}, \mathcal{N})$, where \mathcal{P} and \mathcal{N} are finite sets of predicates, or (ii) $\neg Q$, where $Q \in \mathcal{L}$.

A *rule node* can either have the form (i) rn_C , where $C \in T$, or (ii) $rn_{\rightarrow \neg Q}$, where $Q \in \mathcal{L}$.

A *goal tree* for the goal g^* is a finite tree of goal nodes and rule nodes satisfying the following conditions.

- (a) The root node is g^* ,
- (b) Each goal node of the form $(\bigvee \mathcal{P}, \emptyset)$ or $\neg Q$ appears at most once along any branch,
- (c) If $g = (\bigvee \mathcal{P}, \mathcal{N})$ then g has a single child node of the form rn_C , where
 - $conseq(C) \subseteq \mathcal{P}$,
 - $antec(C) \cap \mathcal{P} = \emptyset$,
 - rn_C has child (goal) nodes of the form

- $(? \bigvee (\mathcal{P} \cup \{A\}), \mathcal{N})$ for each $A \in \text{antec}(C)$, and
 - $\neg Q$ for each $Q \in \mathcal{N}(C) - \mathcal{N}$.
- (d) If $g = \neg Q$, then g has a single child node of the form $rn_{\rightarrow \neg Q}$. $rn_{\rightarrow \neg Q}$ has, for each quasi cyclic tree \mathcal{T} for Q in T , a single child node of the form $(? \bigvee \mathcal{N}(\mathcal{T}), \emptyset)$ or $(? \bigvee \mathcal{O}(\mathcal{T}), \mathcal{N}(\mathcal{T}))$.
- (e) Each leaf node is a rule node.

Condition (e) again follows from conditions (c) and (d) and is included simply for the sake of clarity.

5.2 Theorem. Suppose that T is well-founded.

- (a) $\bigvee \mathcal{P} \in \text{DWFS}$ iff $(? \bigvee \mathcal{P}, \emptyset)$ has a goal tree.
- (b) $\neg Q \in \text{DWFS}$ iff $\neg Q$ has a goal tree.

Proof. If $\bigvee \mathcal{P}' (\neg Q')$ belongs to $D_{\gamma+n+1} - D_{\gamma+n}$ (where γ is 0 or a limit ordinal, and n is finite), then define $\ell(\bigvee \mathcal{P}') = \gamma + 2n$ ($\ell(\neg Q') = \gamma + 2n + 1$).

We proceed by induction on ℓ to develop a tree \mathcal{T} such that if the goal node $(? \bigvee \mathcal{P}', \emptyset) (\neg Q')$ appears in \mathcal{T} then $\bigvee \mathcal{P}' \in \text{DWFS}$ ($\neg Q' \in \text{DWFS}$), and such that the ℓ -values of such goal nodes are monotonic decreasing along all branches.

Suppose that $\bigvee \mathcal{P} \in D_{\alpha+1}^+ - D_{\alpha}^+$, then $T|_g(\mathcal{L} - \overline{D_{\alpha}^-}) \models \bigvee \mathcal{P}$, whence by the results of Section 3 we may find a partial goal tree \mathcal{T} for $(? \bigvee \mathcal{P}, \emptyset)$ satisfying our conditions above, and in which each leaf goal node in \mathcal{T} is of the form $\neg Q'$ with $\ell(\neg Q') < \ell(\bigvee \mathcal{P})$. The result then follows by inductive hypothesis.

Suppose that $\neg Q \in D_{\alpha+1}^- - D_{\alpha}^-$, then by Corollary 4.10, for each quasi cyclic tree \mathcal{T} for Q in T , we have that either $\bigvee \mathcal{N}(\mathcal{T}) \in D_{\alpha+1}^+$ or $T|_g(\mathcal{L} - \mathcal{N}(\mathcal{T}) \cup \overline{D_{\alpha}^-}) \models \bigvee \mathcal{O}(\mathcal{T})$. In the former case we can append $(? \bigvee \mathcal{N}(\mathcal{T}), \emptyset)$ as a child of $rn_{\rightarrow \neg Q}$, and then use the inductive hypothesis. Thus suppose that $\bigvee \mathcal{N}(\mathcal{T}) \notin D_{\alpha+1}^+$ (whence $T|_g(\mathcal{L} - \mathcal{N}(\mathcal{T}) \cup \overline{D_{\alpha}^-}) \models \bigvee \mathcal{O}(\mathcal{T})$).

If $\mathcal{N}(\mathcal{T}) = \emptyset$, then $T|_g(\mathcal{L} - \overline{D_{\alpha}^-}) \models \bigvee \mathcal{O}(\mathcal{T})$, whence by Theorem 2.4, $\bigvee \mathcal{O}(\mathcal{T}) \in D_{\alpha+1}^+$, thus we may append $(? \bigvee \mathcal{O}(\mathcal{T}), \emptyset)$ as a child of $rn_{\rightarrow \neg Q}$, and then apply our inductive hypothesis.

If $\mathcal{N}(\mathcal{T}) \neq \emptyset$, we may use the results of Section 3 to construct a partial goal tree for $(? \bigvee \mathcal{O}(\mathcal{T}), \mathcal{N}(\mathcal{T}))$ in which each leaf node is of the form $\neg Q'$, where $\neg Q' \in D_{\alpha}^-$, from whence we can again apply our inductive hypothesis.

The converse result follows trivially from Theorems 2.4/3.6 and Corollary 4.10. ■

5.3 Example. Let T consist of the following rules.

- | | | |
|--|---|--|
| 1. $E_0 \wedge \neg E_2 \wedge \neg B \rightarrow B_3$ | 2. $B_2 \wedge \neg E_3 \rightarrow B \vee B_1$ | 3. $E_1 \wedge \neg E_4 \rightarrow B$ |
| 4. $\neg B_1 \wedge B_2 \rightarrow A$ | 5. $E_2 \vee E_3$ | 6. $\neg B_1 \rightarrow E_0$ |
| 7. $\neg E_2 \rightarrow E_1$ | 8. $E_4 \vee E_2$ | 9. $\neg E_2 \rightarrow B_2 \vee B_3$ |
| 10. $E_3 \wedge \neg A \rightarrow P \vee Q$ | 11. $\neg A \rightarrow B_3$ | 12. $\neg E_2 \wedge \neg E_3 \rightarrow B_3$ |

Suppose that we wish to check whether $\bigvee\{P, Q, E_2\}$ is in DWFS. Since rule 10 is the only rule whose consequent is a subset of $\{P, Q, E_2\}$, by Corollaries 3.2/3 we have that $\bigvee\{P, Q, E_2\} \in D_\alpha^+$ iff $\bigvee\{P, Q, E_2, E_3\} \in D_\alpha^+$ and $\neg A \in D_{\alpha-1}^-$. $\bigvee\{P, Q, E_2, E_3\}$ is trivially in D_1^+ by virtue of rule 5. This is depicted by the tree \mathcal{T}_1 in Figure 5.3(i). Note that for brevity, goals of the form $(?\bigvee\{P, Q, E_2, E_3\}, \mathcal{N})$ will be represented by the branch as a whole. rn_{10} and rn_5 denote the applications of rules 10 and 5.

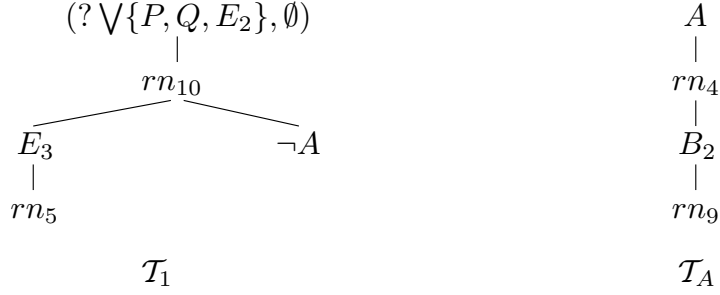


Figure 5.3(i).

In order to check that $\neg A \in D_{\alpha-1}^-$, we need to generate quasi cyclic trees for A in \mathcal{T} . The only such tree is depicted in Figure 5.3(i) as \mathcal{T}_A , with $\mathcal{O}(\mathcal{T}_A) = \{B_3\}$ and $\mathcal{N}(\mathcal{T}_A) = \{B_1, E_2\}$.

We are thus required to check that either $\bigvee\{B_1, E_2\} \in D_{\alpha-1}^+$, or that $T|_g(\mathcal{L} - \{B_1, E_2\} \cup D_{\alpha-2}^-) \models B_3$. Notice that the former statement is trivially false, since no rule has its consequent contained within $\{B_1, E_2\}$. Thus we append the goal node $(?B_3, \{B_1, E_2\})$ as the child node of $rn_{\neg A}$ (Figure 5.3(ii)).

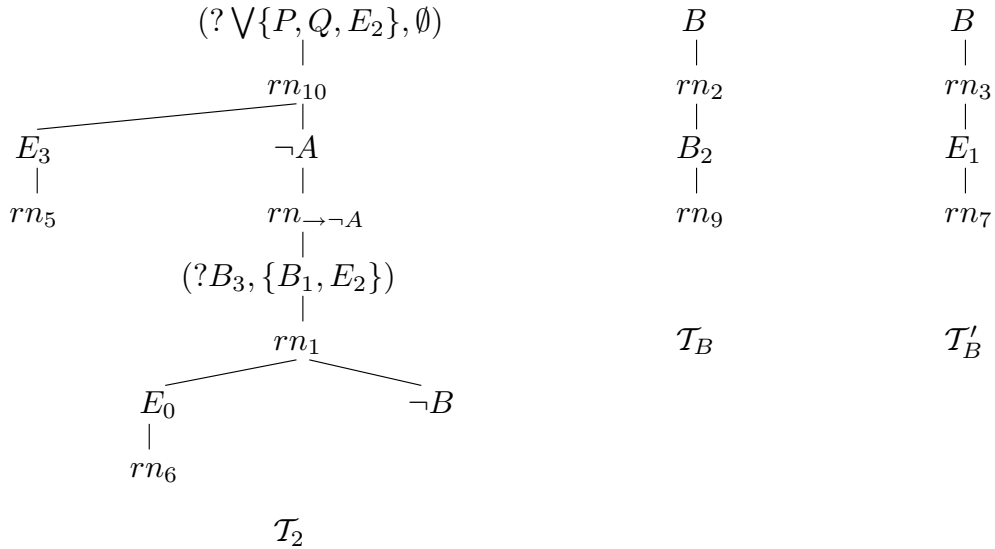


Figure 5.3(ii).

The only rules in T whose consequent is $\{B_3\}$ are rules 1, 11 and 12. Rule 11 however would introduce a duplicate goal (ie., $\neg A$). Rule 12 would introduce the goal $\neg E_3$, but E_3 has a single quasi cyclic tree \mathcal{T}_{E_3} with $\mathcal{N}(\mathcal{T}_{E_3}) = \emptyset$, and $\mathcal{O}(\mathcal{T}_{E_3}) = \{E_2\}$. Since no rule has $\{E_2\}$ as its consequent, we cannot have $\neg E_3 \in \text{DWFS}$.

We thus apply rule 1, and then employ rule 6 to solve the left-hand child. Notice that in both cases we can ignore predicates in $\mathcal{N}(C)$ which appear in $\{B_1, E_2\}$ (cf., condition (c) of Definition 5.1). The goal $\neg B$ calls for the generation of quasi cyclic trees for B , these being depicted as \mathcal{T}_B and \mathcal{T}'_B in Figure 5.3(ii). $\mathcal{N}(\mathcal{T}_B) = \{E_3, E_2\}$, which is trivially in D_1^+ by virtue of rule 5. $\mathcal{N}(\mathcal{T}'_B) = \{E_4, E_2\}$, which again is in D_1^+ by virtue of rule 8. The final goal tree is depicted in Figure 5.3(iii), and shows that $\bigvee\{P, Q, E_2\} \in \text{DWFS}$.

We have seen that $\{E_3 \vee E_2, E_2 \vee E_4, P \vee Q \vee E_2 \vee E_3\} \subseteq D_1^+$, whence $\neg B \in D_1^-$, $\neg A \in D_2^-$, and hence $P \vee Q \vee E_2 \in D_3^+$.

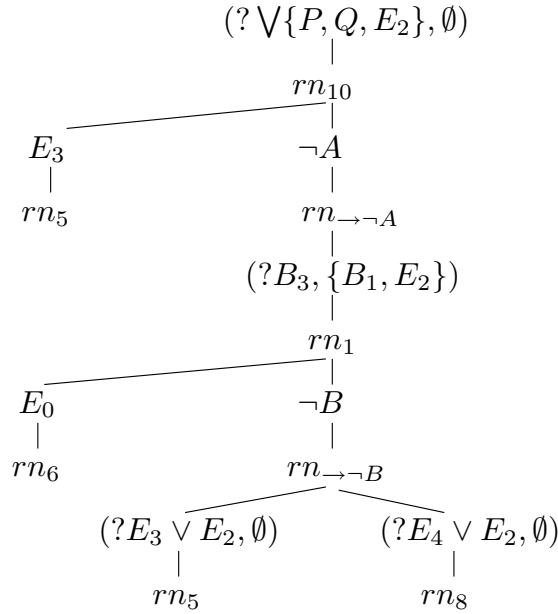


Figure 5.3(iii).

5.4 Top-down vs. bottom-up computation.

The above method raises the question as to the relative advantages of top-down and bottom-up approaches. Bottom-up methods are often criticised for generating many formulae that are unrelated to the intended goal (i.e. it is difficult to direct the search). A bottom-up approach to testing DWFS membership using the D_α construction (Definition 2.1) is inherently weak in this respect, since ascertaining that $\phi \in D_{\alpha+1}$ using this method requires the computation of part of $D_{\alpha+1} - D_\alpha$, which in turn requires the *complete* construction of $D_0, D_1, \dots, D_\alpha$. This limitation can be overcome using the more flexible bottom-up computation described in Sections 6 and 7 below.

A further limitation of the bottom-up D_α construction is that it might require a transfinite iteration if the Herbrand base is infinite. (The corresponding problem for top-down methods would be non-termination.) It is unclear whether a bottom-up construction of DWFS is possible without transfinite iteration.

Finally we note that many bottom-up constructions are relatively easy to visualise (for example when constructing the minimal model of a positive database), whence directing the search seems at least plausible. This is not the case however when constructing DWFS since *disjuncts* of predicates are being generated, and new negative atoms are identified by a highly non-trivial operation (cf. Theorem 2.5) which makes it very difficult to foresee which negative atoms can be generated at the next stage.

§6. A flexible bottom-up computation of DWFS

The D_α construction of the DWFS suggests that we build DWFS by constructing D_α , then $D_{\alpha+1}$, etc. This is somewhat inflexible, particularly since the sets D_α could be large (or even infinite) and their construction has to be complete before starting the construction of $D_{\alpha+1}$. In the finite case, generating $D_{\alpha+1}^+$ from D_α is a relatively simple closure operation, but constructing $D_{\alpha+1}^-$ from $D_\alpha \cup D_{\alpha+1}^+$ requires the execution of a complex test for *every* predicate which does not appear in D_α^- .

The construction detailed below in Theorem 6.1 allows DWFS to be generated in a more flexible fashion as a simple closure operation, with new formulae inserted one at a time in any order.

6.1 Theorem. Let $\emptyset = W_0 \subseteq W_1 \subseteq \dots \subseteq W_\kappa$ be a sequence satisfying conditions (a) - (c) below.

- (a) Each W_α is of the form given in Definition 1.2, and is closed. If γ is a limit ordinal, then $W_\gamma = \bigcup_{\alpha < \gamma} W_\alpha$.
- (b) $W_{\alpha+1}$ is the closure (in the sense of Definition 1.3) of $W_\alpha \cup \{\phi_\alpha\}$, where ϕ_α satisfies either condition (i) or (ii) below.
 - (i) $\phi_\alpha = \bigvee(\mathcal{P} - \overline{W_\alpha^-})$, where $W_\alpha^+ \not\models \bigvee \mathcal{P}$, and there exists $C \in T$ with
 - $\text{conseq}(C) \subseteq \mathcal{P}$ and $\text{antec}(C) \cap \mathcal{P} = \emptyset$,
 - $W_\alpha^+ \models A \vee \bigvee \mathcal{P}$ for each $A \in \text{antec}(C)$, and
 - $\mathcal{N}(C) \subseteq \overline{W_\alpha^-}$.
 - (ii) $\phi_\alpha = \neg Q$, where $\neg Q \notin W_\alpha^-$ and
 - $(\forall N \subseteq \mathcal{L} - \overline{W_\alpha^-})(N \models W_\alpha^+ \implies T|_g N \models_{\min} \neg Q)$.
- (c) No formula ϕ_κ can be found to satisfy condition (b) for W_κ .

Then $\text{DWFS} = \{\bigvee \mathcal{P} : W_\kappa^+ \models \bigvee \mathcal{P}\} \cup W_\kappa^-$.

Proof. Let $\mathcal{W} = \{\bigvee \mathcal{P} : W_\kappa^+ \models \bigvee \mathcal{P}\} \cup W_\kappa^-$. We first show that $\mathcal{W} \subseteq \text{DWFS}$. To do this it suffices to show that each $W_\rho \subseteq \text{DWFS}$. Suppose that $W_\alpha \subseteq \text{DWFS}$.

Case (i) ϕ_α is of the form $\bigvee(\mathcal{P} - \overline{W_\alpha^-})$. By the conditions on C and our inductive hypothesis, we have that $A \vee \bigvee \mathcal{P} \in \text{DWFS}$ for each $A \in \text{antec}(C)$, and $\mathcal{N}(C) \subseteq \overline{\text{DWFS}^-}$. But then by Corollary 3.3, we have that $\bigvee \mathcal{P} \in \text{DWFS}$, whence by the closure of DWFS, we have $\phi_\alpha \in \text{DWFS}$.

Case (ii) ϕ_α is of the form $\overline{\neg Q}$. Pick η such that $W_\alpha \subseteq D_\eta$. If $N \subseteq \mathcal{L} - \overline{D_\eta^-}$ with $N \models D_{\eta+1}^+$, then $N \subseteq \mathcal{L} - \overline{W_\alpha^-}$ with $N \models W_\alpha^+$, whence $T|_g N \models_{\min} \neg Q$. Thus by Theorem 2.5, $\neg Q \in D_{\eta+1}^-$. Again by the closure of DWFS we have $W_{\alpha+1} \subseteq \text{DWFS}$.

We now show that each $D_\rho \subseteq \mathcal{W}$. Suppose that $D_\alpha \subseteq \mathcal{W}$.

(i) If $D_{\alpha+1}^+ \not\subseteq \mathcal{W}$, then pick $\bigvee \mathcal{P} \in D_{\alpha+1}^+ - \mathcal{W}$. Let \mathcal{T} be a deduction tree for \mathcal{P} in $T|_g(\mathcal{L} - \overline{D_\alpha^-})$, and pick a goal node $?\bigvee \mathcal{Q}$ in \mathcal{T} such that $\bigvee \mathcal{Q} \notin \mathcal{W}$, but $\bigvee \mathcal{Q}' \in \mathcal{W}$ for each goal node $?\bigvee \mathcal{Q}'$ below $?\bigvee \mathcal{Q}$ in \mathcal{T} .

Let $rn_{C'}$ be the child node of $?\bigvee \mathcal{Q}$ in \mathcal{T} , where $C' \in T|_g(\mathcal{L} - \overline{D_\alpha^-})$. Let C be the corresponding rule in T , then $\text{conseq}(C) \subseteq \mathcal{Q}$, $\text{antec}(C) \cap \mathcal{Q} = \emptyset$, $\mathcal{N}(C) \subseteq \overline{W_\kappa^-}$, and by our choice of $?\bigvee \mathcal{Q}$ we have that $A \vee \bigvee \mathcal{Q} \in \mathcal{W}$ for each $A \in \text{antec}(C)$. But then by condition (c), we have that $\bigvee \mathcal{Q} \in \mathcal{W}$.

(ii) Suppose that $D_{\alpha+1}^+ \subseteq \mathcal{W}$ and $\neg Q \in D_{\alpha+1}^-$. Let $N \subseteq \mathcal{L} - \overline{W_\kappa^-}$ with $N \models W_\kappa^+$, then $N \subseteq \mathcal{L} - \overline{D_\alpha^-}$ and $N \models D_{\alpha+1}^+$, whence $T|_g N \models_{\min} \neg Q$. Thus again by condition (c), $\neg Q \in W_\kappa^- \subseteq \mathcal{W}$. ■

The following trivial proposition will be needed in Section 7.

6.2 Proposition. Let $(W_\alpha : \alpha = 0, 1, 2, \dots)$ be any sequence satisfying the conditions of Theorem 6.1.

(a) If $M \subseteq \mathcal{L} - \overline{W_\alpha^-}$ with $M \models T|_g(\mathcal{L} - \overline{W_\alpha^-})$, then $M \models W_\alpha^+$.

(b) If $\neg Q \in W_\alpha^-$, $N \subseteq \mathcal{L} - \overline{W_\alpha^-}$ with $N \models W_\alpha^+$ then $T|_g N \models_{\min} \neg Q$.

The following example uses the same database as was given in Example 5.3. For well-founded databases, condition (b)(ii) of Theorem 6.1 is (by Theorem 4.9) equivalent to “for each quasi cyclic tree \mathcal{T} for Q in T , either $W_\alpha^+ \models \bigvee \mathcal{N}(\mathcal{T})$ or $T|_g(\mathcal{L} - \mathcal{N}(\mathcal{T}) \cup \overline{W_\alpha^-}) \models \bigvee \mathcal{O}(\mathcal{T})$ ”. Such a characterisation however introduces an element of top-down processing.

6.3 Example. Let T consist of the following rules.

- | | | |
|--|---|--|
| 1. $E_0 \wedge \neg E_2 \wedge \neg B \rightarrow B_3$ | 2. $B_2 \wedge \neg E_3 \rightarrow B \vee B_1$ | 3. $E_1 \wedge \neg E_4 \rightarrow B$ |
| 4. $\neg B_1 \wedge B_2 \rightarrow A$ | 5. $E_2 \vee E_3$ | 6. $\neg B_1 \rightarrow E_0$ |
| 7. $\neg E_2 \rightarrow E_1$ | 8. $E_4 \vee E_2$ | 9. $\neg E_2 \rightarrow B_2 \vee B_3$ |
| 10. $E_3 \wedge \neg A \rightarrow P \vee Q$ | 11. $\neg A \rightarrow B_3$ | 12. $\neg E_2 \wedge \neg E_3 \rightarrow B_3$ |

Suppose that we wish to generate DWFS. Initially set $W_2 = \{E_2 \vee E_3, E_4 \vee E_2\}$. As in Example 5.3, B has two quasi cyclic trees \mathcal{T}_B and \mathcal{T}'_B such that $\mathcal{N}(\mathcal{T}_B) = \{E_3, E_2\}$ and $\mathcal{N}(\mathcal{T}'_B) = \{E_4, E_2\}$, whence we may set $W_3 = W_2 \cup \{\neg B\}$.

Again, as in Example 5.3, A has a single quasi cyclic tree \mathcal{T}_A with $\mathcal{O}(\mathcal{T}_A) = \{B_3\}$ and $\mathcal{N}(\mathcal{T}_A) = \{B_1, E_2\}$. As in Example 5.3 we may use rules 1 and 6 to show that $T|_g(\mathcal{L} - \{B_1, E_2, B\}) \models B_3$, whence we may set $W_4 = W_3 \cup \{\neg A\}$. Applying rules 10 and 11, we may then set $W_6 = W_4 \cup \{P \vee Q \vee E_2, B_3\}$.

B_1 has a single quasi cyclic tree in T , depicted as \mathcal{T}_1 in Figure 6.3, with $\mathcal{N}(\mathcal{T}_1) = \{E_3, E_2\}$, whence we may set $W_7 = W_6 \cup \{\neg B_1\}$ (by rule 5) and $W_8 = W_7 \cup \{E_0\}$ (by rule 6).



Figure 6.3.

B_2 has a single quasi cyclic tree in T , depicted as \mathcal{T}_2 in Figure 6.3, with $\mathcal{O}(\mathcal{T}_2) = \{B_3\}$. By virtue of rule 11, we have that $T|_g(\mathcal{L} - \{A\}) \models B_3$, whence we may set $W_9 = W_8 \cup \{\neg B_2\}$.

Notice that $\{E_2, E_3, E_4\} \cap \text{DWFS} = \emptyset$, since no rule has $\{E_2\}$, $\{E_3\}$ or $\{E_4\}$ as its consequent. The closure of DWFS then allows us to conclude that $\{\neg E_2, \neg E_3, \neg E_4\} \cap \text{DWFS} = \emptyset$. Similarly we may conclude that $E_1 \notin \text{DWFS}$ (since this would require $\neg E_2 \in \text{DWFS}$), and $\neg E_1 \notin \text{DWFS}$ (since this would require $E_2 \in \text{DWFS}$).

No rule has its consequent contained in $\{Q, E_2\}$, whence $Q \vee E_2 \notin \text{DWFS}$ and in particular $\neg P \notin \text{DWFS}$. Similarly $\neg Q \notin \text{DWFS}$.

Finally we can easily check that no further elements of DWFS can be generated from W_9 using the operation detailed in Theorem 6.1(b)(i), whence we may conclude that $\text{DWFS} = \{\bigvee \mathcal{P} : W_9^+ \models \bigvee \mathcal{P}\} \cup W_9^-$.

§7. Reduction and bottom-up computation

Definition 2.1 constructs DWFS as a set of formulae D_α that are “known to be true”, and then uses these formulae to reduce the database using the $/$ operator before going on to construct $D_{\alpha+1} = \text{Dis}(T/D_\alpha)$. The reduction of T clearly helps in facilitating a more efficient computation of the Dis function. It would be useful if we could employ similar reduction in the generation of sequences $(W_\alpha : \alpha \leq \kappa)$ of the form given in Theorem 6.1. In this section we show that we can in fact employ a reduction operator that is somewhat stronger than $/$.

7.1 Definition. Suppose that $W = W^+ \cup W^-$ is of the form given in Definition 1.2, and is closed. Given T , let $T \parallel W$ be formed from T by performing the following steps against each $C \in T$.

1. remove C if $W^+ \models \bigvee \mathcal{N}(C)$,
2. replace $\mathcal{N}(C)$ by $\mathcal{N}(C) - \overline{W^-}$,
3. replace $\text{conseq}(C)$ by $\text{conseq}(C) - \overline{W^-}$,
4. remove C if $\text{antec}(C) \cap \overline{W^-} \neq \emptyset$, and
5. remove C if $W^+ \models \bigvee \text{conseq}(C)$.

Note that \parallel represents an extension of $/$ in that T/W is formed from T by applying steps 1 and 2 only. Again, if W is closed, then the order in which the operations 1 - 5 are carried out is immaterial. Note that step 3 may result in denial rules (whose consequent is empty).

Step 5 is in effect removing rules from T by subsumption. The subsuming rules will be re-inserted in Theorem 7.3 below.

In the following two theorems, let $(W_\beta : \beta \leq \alpha)$ be a sequence satisfying conditions (a) and (b) of Theorem 6.1. Note that $T \parallel W_{\beta+1} = (T \parallel W_\beta) \parallel (W_{\beta+1} - W_\beta)$, whence the following two theorems show that $W_{\alpha+1}$ can be computed from $T \parallel W_\alpha$.

7.2 Theorem. $\phi_\alpha = \bigvee \mathcal{Q}$ satisfies condition (b)(i) of Theorem 6.1 with respect to W_α iff $\mathcal{Q} \subseteq \mathcal{L} - \overline{W_\alpha^-}$, $W_\alpha^+ \not\models \bigvee \mathcal{Q}$ and there is a rule $C \in T \parallel W_\alpha$ such that $\text{conseq}(C) \subseteq \mathcal{Q}$, $\text{antec}(C) \cap \mathcal{Q} = \emptyset$, $W_\alpha^+ \models A \vee \bigvee \mathcal{Q}$ for each $A \in \text{antec}(C)$, and $\mathcal{N}(C) = \emptyset$.

Proof (\rightarrow). Suppose that $W_\alpha^+ \not\models \bigvee \mathcal{P}$ and $C^* \in T$ satisfies condition (b)(i) of Theorem 6.1 with respect to \mathcal{P} and W_α . Let $\mathcal{Q} = \mathcal{P} - \overline{W_\alpha^-}$.

Firstly note that $\text{antec}(C^*) \cap \overline{W_\alpha^-} = \emptyset$, for otherwise we have $W_\alpha^+ \models \bigvee \mathcal{P}$ by the closure of W_α . Since $W_\alpha^+ \not\models \bigvee \mathcal{P}$, we have that $W_\alpha^+ \not\models \bigvee \text{conseq}(C^*)$. Also by the closure of W_α we have that $W_\alpha^+ \not\models \bigvee \mathcal{N}(C^*)$.

Thus the reduced rule $C = \bigwedge \text{antec}(C^*) \rightarrow \bigvee (\text{conseq}(C^*) - \overline{W_\alpha^-})$ is in $T \parallel W_\alpha$, and satisfies the conditions of the right-hand side.

(\leftarrow). Suppose that $C \in T \parallel W_\alpha$ is as given. Let C^* be the corresponding rule in T , then C^* and $\mathcal{P} = \mathcal{Q} \cup (\text{conseq}(C^*) \cap \overline{W_\alpha^-})$ satisfy condition (b)(i) of Theorem 6.1. ■

7.3 Theorem. Suppose that $N \subseteq \mathcal{L} - \overline{W_\alpha^-}$ with $N \models W_\alpha^+$. Then $T|_g N$ and $W_\alpha^+ \cup (T \parallel W_\alpha)|_g N$ share the same minimal models.

Proof. By Lemma 1.6(c), $T|_g N = (T/W_\alpha)|_g N$. Now T/W_α is formed from T by applying steps 1 and 2 from Definition 7.1. In addition, $(T \parallel W_\alpha)|_g N$ is formed from $(T/W_\alpha)|_g N$ by applying steps 3, 4 and 5 from Definition 7.1.

Let T^* be the result of applying step 3 to $(T/W_\alpha)|_g N$, ie., $T^* = \{\bigwedge \text{antec}(C) \rightarrow \bigvee (\text{conseq}(C) - \overline{W_\alpha^-}) : C \in (T/W_\alpha)|_g N\}$, then

- (a) $T^* \models (T/W_\alpha)|_g N$ (since each rule in $(T/W_\alpha)|_g N$ is subsumed by a rule in T^*),
and
(b) $T^* \supseteq (T \parallel W_\alpha)|_g N$.

In order to show that $T|_g N$ and $W_\alpha^+ \cup (T \parallel W_\alpha)|_g N$ share the same minimal models, it suffices to show that the minimal models of one are models of the other.

If M is a minimal model of $T|_g N = (T/W_\alpha)|_g N$, then by Proposition 6.2(b), $M \cap \overline{W_\alpha^-} = \emptyset$, whence by the definition of T^* we must have that $M \models T^*$. Thus $M \models (T \parallel W_\alpha)|_g N$. Now $N \subseteq \mathcal{L} - \overline{W_\alpha^-}$, whence $T|_g N \supseteq T|_g(\mathcal{L} - \overline{W_\alpha^-})$, and hence by Proposition 6.2(a) we have that $M \models W_\alpha^+$. Thus $M \models W_\alpha^+ \cup (T \parallel W_\alpha)|_g N$.

Conversely, if M is a minimal model of $W_\alpha^+ \cup (T \parallel W_\alpha)|_g N$, then since no predicate in $\overline{W_\alpha^-}$ appears in $(T \parallel W_\alpha)|_g N$ and W_α is closed, we must have that $M \cap \overline{W_\alpha^-} = \emptyset$. If $C \in T^* - (T \parallel W_\alpha)|_g N$, then either $\text{antec}(C) \cap \overline{W_\alpha^-} \neq \emptyset$, or C is subsumed by a rule in W_α^+ . Thus $M \models T^*$ and again, $M \models (T/W_\alpha)|_g N$. ■

The following example gives an insight into the kind of saving that might result from the application of \parallel .

7.4 Example. Let T contain the following rules.

- | | | |
|--|---|--|
| 1. $E_0 \wedge \neg E_2 \wedge \neg B \rightarrow B_3$ | 2. $B_2 \wedge \neg E_3 \rightarrow B \vee B_1$ | 3. $E_1 \wedge \neg E_4 \rightarrow B$ |
| 4. $\neg B_1 \wedge B_2 \rightarrow A$ | 5. $E_2 \vee E_3$ | 6. $\neg B_1 \rightarrow E_0$ |
| 7. $\neg E_2 \rightarrow E_1$ | 8. $E_4 \vee E_2$ | 9. $\neg E_2 \rightarrow B_2 \vee B_3$ |
| 10. $E_3 \wedge \neg A \rightarrow P \vee Q$ | 11. $\neg A \rightarrow B_3$ | 12. $\neg E_2 \wedge \neg E_3 \rightarrow B_3$ |

In Example 6.3 we found $W_8 = \{E_2 \vee E_3, E_4 \vee E_2, \neg B, \neg A, P \vee Q \vee E_2, B_3, \neg B_1, E_0\}$. Let's consider how the reduction operator helps in completing $\text{DWFS} = W_9$.

$T \parallel W_8 = \{B_2 \wedge \neg E_3 \rightarrow, E_1 \wedge \neg E_4 \rightarrow, B_2 \rightarrow, \neg E_2 \rightarrow E_1, E_3 \rightarrow P \vee Q\}$. It is thus immediate that B_2 can belong to no (minimal) model of a set of the form $W_8^+ \cup (T \parallel W_8)|_g N$, whence we may set $W_9 = W_8 \cup \{\neg B_2\}$.

$T \parallel W_9 = \{E_1 \wedge \neg E_4 \rightarrow, \neg E_2 \rightarrow E_1, E_3 \rightarrow P \vee Q\}$. From this we can see that none of the rules in $T \parallel W_9$ allows us to generate any new disjuncts using Theorem 7.2. It is also trivial see W_9 cannot be extended using the operator detailed in Theorem 6.1(b)(ii), for if $N = \{E_3, E_4, P, B_3, E_0\}$, then $N \models W_9^+$ and $T^* = W_9^+ \cup (T \parallel W_9)|_g N = \{E_2 \vee E_3, E_4 \vee E_2, P \vee Q \vee E_2, B_3, E_0, E_1, E_3 \rightarrow P \vee Q\}$, whence it is clear that for each $p \in \{E_1, E_2, E_3, E_4, P, Q\}$, T^* has a minimal model containing p .

§8. The first order level

Our results have been presented for the propositional level, which immediately raises the question as to how they can be extended to first order deductive databases. In [Jo93, Jo98a] we adapted Bibel's connection method [Bi87] to present a top-down

construction of deduction trees which allows correct and complete processing of queries of the form $\bigvee \mathcal{P}$ within positive first order databases/logic programs.

The main difficulty with top-down methods at the first order level is termination, the main problem being the possible presence of rules of the form $P(x, y) \rightarrow Q(x)$ in which y can be viewed as existentially quantified. In [Jo98a] we employed powerful search space pruning techniques, semi-definite predicates, and a constraint [Jo98a, Section 6.1.2] on such existentially quantified variables to guarantee termination in the function free case.

Cyclic trees are always ground, even for first order databases (cf., [Jo96, Theorem 6.4.7]). However, because their construction is inherently top-down, termination is again problematic. In [Jo96, Jo98] we showed that termination is guaranteed at the first order (function free) level provided that we impose further constraints on existentially quantified variables.

In [Jo00] we extend the results of the present paper to the first order function free level. Partial compilation of cyclic trees is employed and this allows a terminating construction of such trees using the same variable constraints as are needed for the terminating construction of deduction trees.

The above remarks concerning cyclic trees are also applicable to a bottom-up approach in which we employ cyclic trees to process condition (b)(ii) of Theorem 6.1. Condition (b)(i) of Theorem 6.1 provides no special problems at the first order level, assuming (as is normal) that for each rule $C \in T$, any variable appearing in $\text{conseq}(C)$ also appears in $\text{antec}(C) \cup \mathcal{N}(C)$.

Finally we note that the definition of DWFS adopted in this paper is of a different nature to the original definition presented in [Bra94, Bra98a]. There, the DWFS was presented as the weakest semantics satisfying certain desirable properties/transformations, most importantly the generalised principle of partial evaluation. This approach has also been extended to the first order level by Dix and Stolzenburg [Di97], where they define a calculus of transformations for first order (constraint) programs. The weakest semantics that is invariant under this calculus is referred to as the constraint disjunctive well-founded semantics, and is then shown to be a generalisation of the DWFS. ■

§9. Conclusions and criticisms

We have presented a characterisation of the DWFS in terms of Gelfond-Lifschitz transformations of the initial database, and used this characterisation to present a top-down method of testing DWFS membership and a bottom-up method of computing DWFS. Our bottom-up method allows a flexible computation of DWFS, and moreover admits a powerful reduction operator, thus improving the efficiency of the computation. All of our methods operate in polynomial space for finite propositional databases.

Our methods differ from those developed in the DISLOP project [Ar97], in that the latter employs the (the more rigid) bottom-up D_α construction (Definition 2.1) and the bottom-up methods of [Ni96] for minimal model reasoning.

References

- [Ar97] C. Aravindan, J. Dix and I. Niemelä, DISLOP: A research project on disjunctive logic programming, *AI communications*, vol. 10 (1997), 151-165.
- [Ba91] C. Baral, J. Lobo and J. Minker, WF³: A semantics for negation in normal disjunctive logic programs, in Z. Ras and M. Zemankova (eds.), *Proceedings of the 6th International Symposium on Methodologies for Intelligent Systems*, Springer Lecture Notes in Computer Science, vol. 542, pp 459-468.
- [Ba92] C. Baral, J. Lobo and J. Minker, Generalised disjunctive well-founded semantics for logic programs, *Annals of Mathematics and Artificial Intelligence*, vol. 5 (1992), 89-132.
- [Bi87] W. Bibel, *Automated Theorem Proving* (Vieweg, Wiesbaden, 1987).
- [Bra94] S Brass and J. Dix, A disjunctive semantics based upon unfolding and bottom-up evaluation, in : B Wolfinger, (ed.), *Innovationen bei Rechen- und Kommunikationssystemen (IFIP- Congress, Workshop FG2: Disjunctive Logic Programming and Disjunctive Databases)*, (Springer, 1994), 83-91.
- [Bra95] S. Brass and J. Dix, Disjunctive semantics based upon partial and bottom-up evaluation, in : L. Sterling (ed.), *Proc. 12th Int'l Conf. on Logic Programming*, Tokyo (MIT Press).
- [Bra98] S. Brass, J. Dix, I. Niemelä and T. C. Przymusiński, A comparison of the static and disjunctive well-founded semantics, in: A.G. Cohn, L.K. Schubert and S.C. Shapiro (eds.), *Proceedings of the 6th International Conference on Principles of Knowledge Representation and Reasoning* (Morgan Kaufmann, 1998).
- [Bra98a] S. Brass and J. Dix, Characterisations of the disjunctive well-founded semantics: confluent calculi and iterated GCWA, *J. Automated Reasoning*, vol. 20 (1998), 143-165.
- [Bra99] S. Brass and J. Dix, Semantics of disjunctive logic programs based upon partial evaluation, *J. Logic Programming*, vol. 40 (1999), 1-46.
- [Di95] J. Dix, Semantics of logic programs: Their intuitions and formal properties, in A. Fuhrman and H. Rott (eds.), *Logic, Action and Information, Essays on Logic in Philosophy and Artificial Intelligence* (DeGruyter, 1995), 241-327.
- [Di97] J. Dix and F. Stolzenburg, A framework to incorporate non-monotonic reasoning into constraint logic programming, *Journal of Logic Programming*, vol. 37 (1997), 47-76.
- [Ge88] M. Gelfond and V. Lifschitz, The stable model semantics for logic programming, in: R. Kowalski and K. Bowen (eds.), *Proc. 5th Int'l Conference on Logic Programming*, Seattle (1988), 1070-1080.
- [Gr86] J. Grant and J. Minker, Answering queries in indefinite databases and the null value problem, *Advances in Computing Research*, vol. 3 (1986), 247-267.
- [Jo93] C. A. Johnson, Top-down deduction in indefinite deductive databases, in F. Bry

- (ed.), Proceedings of the 1993 Journées Bases de Données Avancées, Toulouse, (INRIA, France), 119-138.
- [Jo96] C. A. Johnson, On computing minimal and perfect model membership, *Data and Knowledge Engineering*, vol. 18 (1996), 225-276.
- [Jo98] C. A. Johnson, Extended deduction trees and query processing, Computer Science technical report TR98-07, Keele University (1997).
- [Jo98a] C. A. Johnson, Top-down query processing in indefinite stratified databases, *Data and Knowledge Engineering*, vol. 26 (1998), 1-36. (Extracted from [Jo98].)
- [Jo99] C. A. Johnson, On cyclic covers and perfect models, *Data and Knowledge Engineering*, vol. 31 (1999), 25-65.
- [Jo00] C. A. Johnson, Top-down query processing in first order deductive databases under the DWFS, 12th International Symposium on Methodologies for Intelligent Systems (Charlotte, USA, October 2000), accepted.
- [Lb92] J. Lobo, J. Minker, and A. Rajasekar, *Foundations of Disjunctive Logic Programming*, (MIT Press, Cambridge, Massachusetts, 1992).
- [Ni96] I. Niemelä, A tableau calculus for minimal model reasoning, in: *Proceedings of the 5th Workshop on Theorem Proving with Analytic Tableaux and Related Methods*, Terrasini, Italy (Springer, 1996), 278-294.
- [Pr88] T. Przymusiński, On the declarative semantics of deductive databases and logic programs, in: J. Minker (ed.), *Foundations of deductive Databases and Logic Programming* (Morgan Kaufmann, Washington, 1998), 193-216.
- [Pr91] T. Przymusiński, Stable semantics for disjunctive programs, *New Generation Computing*, vol. 9 (1991), 401-424.