

PROCESSING DEDUCTIVE DATABASES UNDER THE DISJUNCTIVE STABLE MODEL SEMANTICS

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

Abstract. Cyclic covers are shown to characterise disjunctive stable models of unstratified deductive databases, and to facilitate top-down query processing, query compilation and view updating under the disjunctive stable model semantics. Such processing is shown to be more complex than comparable processing of stratified databases.

Keywords. Disjunctive deductive databases, disjunctive stable models, cyclic covers, query processing, query compilation, view updates.

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 [Di99], WF³ [Ba91], GDWFS [Ba92] and others. A survey of these semantics is to be found in [Di95].

In contrast, relatively little work has been done on actually processing disjunctive deductive databases under these semantics. Bottom-up methods have been employed to compute perfect and stable models [Fe95] using ordered model trees, and to process positive databases under the minimal model semantics using a tableau calculus [Ni96]. The methods of [Ni96] have also been employed to compute DWFS [Bra98]. Top-down query processing is provided for positive databases by SLO-resolution [Lb92, Ra89] and the minimal model generation method of [Ya96].

In [Jo96] we introduced the concept of a cyclic tree as a means of characterising minimal model membership in positive databases, and perfect model membership in stratified databases. Specifically cyclic trees capture the relationship between a predicate lying inside such a model, and the rules which support the predicate's position in the model. For example a predicate P could be supported in a model by the rule $P \vee Q$, and the absence of Q from the model. In [Jo97] we introduced the concept of a cover, which characterise the properties of predicate lying outside of a model. For

example if our database contains the rule $R \wedge S \rightarrow Q$, and Q lies outside a model, then so must either R or S . Cyclic covers combine the properties of cyclic trees and covers, and were shown in [Jo98, Jo99a] to characterise perfect models of stratified databases, and in particular provide a top-down query processing method for such databases.

In this paper we focus on the disjunctive stable model semantics introduced in [Pr91]. We show that disjunctive stable models can be characterised by cyclic covers, and then illustrate how this allows us to perform query processing under the disjunctive stable model semantics using a hyperresolution-like operator to handle positive sub-goals, and the generation of cyclic trees to handle negative sub-goals. Such query processing is top-down, correct and terminating. In order to ensure completeness we need to add denial rules of the form $P \wedge \neg P \rightarrow \text{FALSE}$ to the database, this requirement being reduced if the database is partially stratified.

We also show that cyclic covers computed using only the intensional database can be employed to perform query compilation and view updates under the disjunctive stable model semantics.

For definite databases, disjunctive stationary models and cyclic covers have quite a close relationship, whereas in the indefinite case, this is not the case. The reasons for this discrepancy are demonstrated to lie in the differing treatments of support.

For the most part we restrict our attention to the ground (ie., propositional) level. Of course a first order deductive database consists of a finite set of first order function free rules, which represent the (finite) set of their ground instances. Our restriction to the propositional level is thus justified from a theoretical standpoint. Practical issues are discussed briefly in Section 8.

§1 DISJUNCTIVE STABLE MODELS

Throughout we assume that \mathcal{L} is a finite propositional language, $\mathcal{L} = \{P_1, P_2, \dots, P_n\}$. A *literal* is a predicate P or its negation $\neg P$. If I is a set of literals, then $I^+ = I \cap \mathcal{L}$ and $I^- = \{P \in \mathcal{L} \mid \neg P \in I\}$. I is *total* iff $I^+ \cup I^- = \mathcal{L}$. I is *consistent* iff $I^+ \cap I^- = \emptyset$. $\bar{I} = \{\neg K \mid K \in I\}$ (where of course $\neg\neg P$ is identified with P).

A deductive database T consists of a set of rules C of the form

$$A_1 \wedge A_2 \wedge \dots \wedge A_h \wedge \neg A_{h+1} \wedge \neg A_{h+2} \wedge \dots \wedge \neg A_{h+r} \rightarrow B_1 \vee B_2 \vee \dots \vee B_k$$

where each A_i, B_j is a predicate. $\text{antec}(C) = \{A_1, A_2, \dots, A_h\}$, $\mathcal{N}(C) = \{A_{h+1}, A_{h+2}, \dots, A_{h+r}\}$, and $\text{conseq}(C) = \{B_1, B_2, \dots, B_k\}$. $\text{EXT}(T)$ (the *extension* of T) denotes those rules in T whose body is empty (ie., for which $h + r = 0$), and $\text{INT}(T) = T - \text{EXT}(T)$ is the *intension* of T . A rule C is *definite* iff $|\text{conseq}(C)| = 1$, and T is *definite* iff it contains only definite rules. T is *positive* iff $\mathcal{N}(C) = \emptyset$ for each $C \in T$.

If $M \subseteq \mathcal{L}$, then M *models* C (written $M \models C$) iff $\text{antec}(C) \not\subseteq M$ or $M \cap (\text{conseq}(C) \cup \mathcal{N}(C)) \neq \emptyset$. M *models* T (written $M \models T$) iff $M \models C$ for each $C \in T$.

We will assume the existence of a distinguished predicate $\mathbb{F} \in \mathcal{L}$ (representing “false”), and moreover that \mathbb{F} appears only in the head of rules of the form $\bigwedge \mathcal{P} \rightarrow \mathbb{F}$.

We will refer to such rules as *denial rules*.

1.1 Definition [Ge88]. If $\mathcal{K} \subseteq \mathcal{L}$, then

$$T|\mathcal{K} = \{\bigwedge \text{antec}(C) \rightarrow \bigvee \text{conseq}(C) \mid \mathcal{K} \cap \mathcal{N}(C) = \emptyset, C \in T\}.$$

Note that $T|\mathcal{K}$ is positive and hence has a well-defined set of minimal models. Notice also that $\mathcal{K} \models T$ iff $\mathcal{K} \models T|\mathcal{K}$.

1.2 Definition [Pr91]. A set $M \subseteq \mathcal{L}$ is a *disjunctive stable model* of T iff $\mathbb{F} \notin M$ and M is a minimal model of $T|M$.

If Φ is a formula in \mathcal{L} , then we write $T \models_{ds} \Phi$ iff Φ holds in all disjunctive stable models of T .

Notice that if T has no disjunctive stable model, then $T \models_{ds} \Phi$ for any Φ . Disjunctive stable models coincide with stable models [Pr94] for definite databases, and with perfect models [Pr91] for stratified databases.

It is well known that the stable model semantics is not cumulative. For example ([Di95a]) if $T = \{\neg B \rightarrow A, \neg A \wedge P \rightarrow B, A \rightarrow P\}$, then $\{A, P\}$ is the only stable model of T . However, $T \cup \{P\}$ has two stable models, namely $\{A, P\}$ and $\{B, P\}$.

The reason why cumulativity fails can also be seen from this example. In the stable model of T , P needs the support of A , which in turn requires that B is false. When we add the new rule P , there is then a separate source of support for P , thus allowing B to be supported by the absence of A . The problem is thus that the introduction of the new rule affects the set of supporting relationships. Corollary 1.4 below shows that we can overcome this problem by using denial rules. It also shows that a rule of the form $\neg P \rightarrow \mathbb{F}$ *cannot* be used to justify the existence of P in some disjunctive stable model.

1.3 Theorem. Suppose that $\mathcal{P} = \{P_1, P_2, \dots, P_n, \neg Q_1, \neg Q_2, \dots, \neg Q_r\}$ is a set of literals with each $P_i, Q_j \in \mathcal{L}$. Let $\Phi = \bigwedge_i \neg P_i \wedge \bigwedge_j Q_j \rightarrow \mathbb{F}$, then M is a disjunctive stable model of $T' = T \cup \{\Phi\}$ iff M is a disjunctive stable model of T and $M \models \bigvee \mathcal{P}$.

Proof (\rightarrow). Suppose that M is a minimal model of $T'|M$ with $\mathbb{F} \notin M$. Since $M \models T' \wedge \neg \mathbb{F}$, we must have that $M \models \bigvee \mathcal{P}$.

If some $P_{i_0} \in M$, then $T'|M = T|M$, whence M is a minimal model of $T|M$. If each $P_i \notin M$, then some $Q_{j_0} \notin M$ and $T'|M = T|M \cup \{\bigwedge_j Q_j \rightarrow \mathbb{F}\}$. If $M' \subset M$ is a model of $T|M$, then since $Q_{j_0} \notin M'$ we have that $M' \models T'|M$, thus contradicting the minimality of M .

(\leftarrow). Suppose that M is a minimal model of $T|M$ with $M \models \neg \mathbb{F} \wedge \bigvee \mathcal{P}$. If some $P_{i_0} \in M$, then $T|M = T'|M$ and hence M is a minimal model of $T'|M$. If each $P_i \notin M$, then $T'|M = T|M \cup \{\bigwedge_j Q_j \rightarrow \mathbb{F}\}$ and some $Q_{j_0} \notin M$, whence $M \models T'|M$. Since M is a minimal model of $T|M$, it is immediately a minimal model of $T'|M$. ■

1.4 Corollary (Weak-cumulativity). Suppose that $\mathcal{P} = \{P_1, P_2, \dots, P_n, \neg Q_1, \neg Q_2, \dots, \neg Q_r\}$ is a set of literals with each $P_i, Q_j \in \mathcal{L}$. Let $\Phi = \bigwedge_i \neg P_i \wedge \bigwedge_j Q_j \rightarrow \mathbb{F}$.

Then $T \models_{ds} \bigvee \mathcal{P}$ iff T and $T \cup \{\Phi\}$ have the same disjunctive stable models.

§2 CYCLIC TREES

Suppose that M is a minimal model of $T|M$. If $P \in M$, then $M - \{P\} \not\models T|M$, whence we may find a rule $C \in T|M$ of the form $\bigwedge_i R_i \rightarrow P \vee \bigvee_j Q_j$, where each $R_i \in M$ and each $Q_j \notin M$. The rule C supports the presence of P in the model. We can likewise look for rules supporting the presence of each R_i .

In the disjunctive case we also need to look at support for sets of predicates. For example if our database contains the rules $A \rightarrow B$ and $B \rightarrow A$ (ie., a cycle), then the cycle may be supported by a rule of the form $R \rightarrow A \vee B$ (if R is in the model).

Cyclic trees were introduced in [Jo96] as a means of representing these supporting relationships. Specifically each predicate (node) in a cyclic tree identifies a set (cycle) of predicates along the current branch and a rule from the database that potentially supports the cycle.

In this section we define cyclic trees, and restate those properties that are needed for later sections. Motivation for the following definition appears in [Jo96, Jo98, Jo99a]. The reader should note that the properties of cyclic trees presented in Theorems 2.4/5 are specifically required in later sections, whereas a detailed understanding of their internal structure (from Definitions 2.1/2) is not.

2.1 Definition. 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 $lab(N)$. If N has a child node, then it 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 $K \in \text{antec}(C)$, RN_C has a (predicate) child node labelled with K , and these are the only child nodes of RN_C .

Then we make the following definitions.

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

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

If N has child RN_C , then define $\mathcal{O}(RN_C) = \text{conseq}(C) - \text{CYC}(N)$.

- (b) 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
 $\text{Pred}(\mathcal{T}) = \{lab(N) \mid N \text{ is a predicate node in } \mathcal{T}\}$.

2.2 Definition. Let $P \in \mathcal{L}$ and \mathcal{T} be a tree satisfying the conditions of Definition 2.1, then \mathcal{T} is said to be a *cyclic tree* for P in T iff

- (a) The root node is labelled with P ,
- (b) $Pred(\mathcal{T}) \cap (\mathcal{O}(\mathcal{T}) \cup \mathcal{N}(\mathcal{T})) = \emptyset$,
- (c) whenever RN_C is a rule node in \mathcal{T} with parent N , then
 - (i) $conseq(C) \cap CYC(N) \neq \emptyset$, and
 - (ii) $antec(C) \cap CYC(N) = \emptyset$.

In line with [Jo96, Jo99a] we will say that \mathcal{T} is *unfactored* iff \mathcal{T} has no predicate leaf node.

2.3 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. $S_3 \rightarrow Q_1 \vee Q_2 \vee Q_6$ | 5. $S_2 \vee R_5$ | 6. $S_1 \rightarrow Q_3 \vee Q_2$ |
| 7. $S_3 \vee R_7$ | 8. $R_1 \rightarrow Q_2$ | |

The only unfactored cyclic trees for Q_1 in T are depicted in Figure 2.3. $Pred(\mathcal{T}_1) = \{Q_1, Q_2, S_3, Q_3, S_2\}$, $\mathcal{O}(\mathcal{T}_1) = \{Q_5, Q_6, R_7, R_5\}$ and $\mathcal{N}(\mathcal{T}_1) = \{R_1, R_2, R_3\}$. $Pred(\mathcal{T}_2) = \{Q_1, S_3\}$, $\mathcal{O}(\mathcal{T}_2) = \{Q_2, Q_6, R_7\}$ and $\mathcal{N}(\mathcal{T}_2) = \emptyset$.

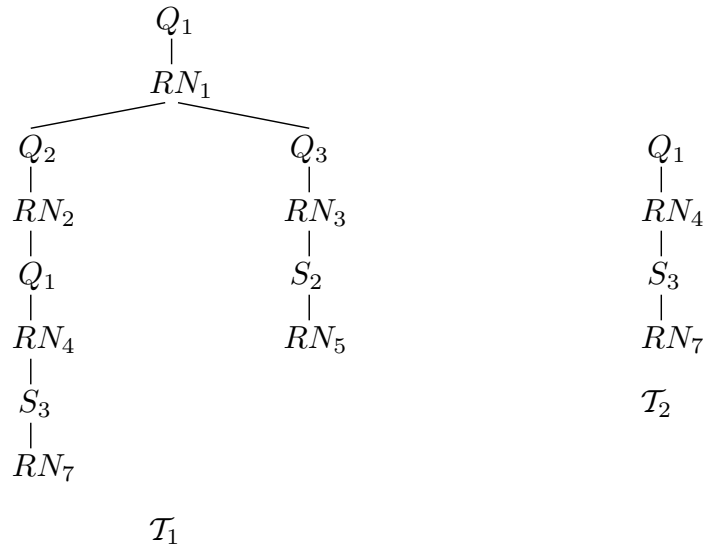


Figure 2.3.

The following two results detail the relationships between cyclic trees and models of the database needed for later sections. Both theorems either appear in, or are trivially derivable from results in [Jo96].

2.4 Theorem (a) Suppose that T is positive and M is a minimal model of T , then for each $P \in M$ there is an unfactored cyclic tree \mathcal{T} for P in T such that $Pred(\mathcal{T}) \subseteq M \subseteq \mathcal{L} - \mathcal{O}(\mathcal{T})$ (and $\mathcal{N}(\mathcal{T}) = \emptyset$).

(b) Suppose that $\mathcal{K} \subseteq \mathcal{L}$ and M is a minimal model of $T|\mathcal{K}$ with $M \subseteq \mathcal{K}$. If $P \in M$ then we may find an unfactored cyclic tree \mathcal{T} for P in T such that $Pred(\mathcal{T}) \subseteq M \subseteq \mathcal{L} - \mathcal{O}(\mathcal{T})$ and $\mathcal{N}(\mathcal{T}) \cap \mathcal{K} = \emptyset$.

2.5 Theorem. Suppose that $M \models T$ and \mathcal{T} is an unfactored cyclic tree in T . If $M \cap (\mathcal{O}(\mathcal{T}) \cup \mathcal{N}(\mathcal{T})) = \emptyset$, then $Pred(\mathcal{T}) \subseteq M$.

§3 CYCLIC COVERS AND DISJUNCTIVE STABLE MODELS

Covers were introduced in [Jo97] as a means of performing view updates in positive databases. They characterise the properties of predicates lying outside of a model. Cyclic covers combine the properties of cyclic trees and covers, and are shown in [Jo98, Jo99a] to characterise perfect models. In this section we show that they also characterise disjunctive stable models.

3.1 Definition [Jo99a]. Let \mathcal{C} be a non-complementary set of literals, then \mathcal{C} is said to be *cyclic* (in T) iff there is a sequence of unfactored cyclic trees $(\mathcal{T}_i \mid 0 \leq i \leq m)$ in T such that

- (i) $\mathcal{C}^- = \bigcup_{i=0}^m Pred(\mathcal{T}_i)$, and
 - (ii) $\mathcal{C}^+ \supseteq \bigcup_{i=0}^m (\mathcal{O}(\mathcal{T}_i) \cup \mathcal{N}(\mathcal{T}_i))$.
- (Recall from Section 1 that $\mathcal{C}^- = \{P \in \mathcal{L} \mid \neg P \in \mathcal{C}\}$, and $\mathcal{C}^+ = \mathcal{C} \cap \mathcal{L}$.)

3.2 Definition [Jo97, Jo99a]. A consistent set of literals \mathcal{C} is said to be a *cover* in T iff $\mathbb{F} \in \mathcal{C}$ and for each rule $C \in INT(T)$

$$\text{conseq}(C) \subseteq \mathcal{C} \implies \mathcal{C} \cap (\text{antec}(C) \cup \overline{\mathcal{N}(C)}) \neq \emptyset$$

(where $\overline{\mathcal{N}(C)} = \{\neg P \mid P \in \mathcal{N}(C)\}$, cf. Section 1).

If \mathcal{C} is a cover with $\mathcal{C} \supseteq \mathcal{P}$, then we say that \mathcal{C} is a *cover of* \mathcal{P} . A cover \mathcal{C} is said to be a *strong cover* in T iff there is no rule $E \in EXT(T)$ such that $E \subseteq \mathcal{C}$.

The reasons for distinguishing between covers and strong covers will become apparent later in Section 5.

Stable models are well known as being supported models ([Fa91]). In [Jo99] we introduced the concept of a supported set and showed that for definite databases, stable models are characterised by total supported strong covers. In the disjunctive case, the support (for a predicate in a model) is provided by cyclic trees, and analogously cyclic strong covers characterise disjunctive stable models. We can thus view the concept of

cyclicness as the natural generalisation of supportedness to the disjunctive case, this view being reinforced by the fact that for definite databases, supported and cyclic sets coincide.

3.3 Theorem. If $M \subseteq \mathcal{L}$, then M is a disjunctive stable model of T iff $\overline{M} \cup (\mathcal{L} - M)$ is a cyclic strong cover in T .

Proof (\rightarrow). Since $M \models T \wedge \neg \mathbb{F}$ it follows trivially that $\overline{M} \cup (\mathcal{L} - M)$ is a strong cover in T .

Now M is a minimal model of $T|M$, thus by Theorem 2.4(b) if $P \in M$ we may find an unfactored cyclic tree \mathcal{T} for P in T such that $Pred(\mathcal{T}) \subseteq M \subseteq \mathcal{L} - (\mathcal{O}(\mathcal{T}) \cup \mathcal{N}(\mathcal{T}))$. Thus $\overline{M} \cup (\mathcal{L} - M)$ is trivially cyclic.

(\leftarrow). Let $C \in T$ with $\mathcal{N}(C) \cap M = \emptyset$. If $antec(C) \subseteq M$, then we cannot have that $conseq(C) \subseteq \mathcal{L} - M$ by virtue of the fact that $\overline{M} \cup (\mathcal{L} - M)$ is a strong cover in T . This then shows that $M \models T|M$. In addition it follows immediately that $\mathbb{F} \notin M$.

Suppose that $P \in M$, then since $\overline{M} \cup (\mathcal{L} - M)$ is cyclic, we may find an unfactored cyclic tree \mathcal{T} in T such that $P \in Pred(\mathcal{T}) \subseteq M \subseteq \mathcal{L} - (\mathcal{O}(\mathcal{T}) \cup \mathcal{N}(\mathcal{T}))$. Let $T' = \{C \in T \mid C \text{ labels some rule node in } \mathcal{T}\}$, then clearly \mathcal{T} is an unfactored cyclic tree in T' . Suppose that $M' \subseteq M$ is a model of $T|M$. If $C \in T'$, then $\mathcal{N}(C) \cap M = \emptyset$, whence $M' \models C$. Clearly $M' \cap (\mathcal{O}(\mathcal{T}) \cup \mathcal{N}(\mathcal{T})) = \emptyset$, whence by Theorem 2.5, $P \in Pred(\mathcal{T}) \subseteq M'$.

Thus $M' = M$ and M is a minimal model of $T|M$. ■

3.4 Corollary. If \mathcal{P} is a set of literals, then $T \models_{ds} \bigvee \mathcal{P}$ iff \mathcal{P} has no total cyclic strong cover in T .

Thus disjunctive stable models are characterised by total cyclic strong covers. We can force all covers to be total by insisting that T implicitly contains $\{P \wedge \neg P \rightarrow \mathbb{F} \mid P \in \mathcal{L}\}$. It follows from Theorem 1.3 that the addition of such denial rules does not alter the disjunctive stable models of T .

Note that in certain cases cyclic (strong) covers of \mathcal{P} might fail to exist (in which case $T \models_{ds} \bigvee \mathcal{P}$ follows trivially). This could occur because of the requirement that covers are consistent (eg., if $\mathcal{P} = \{P, Q\}$ and $\neg P \rightarrow Q \in T$), because of the requirement that \mathbb{F} belongs to any cover, or because disjunctive stable models of the database fail to exist.

The above theorem also allows us to prove some interesting and useful properties of cyclic sets.

3.5 Theorem. The following are equivalent.

- (a) \mathcal{C} is cyclic,
- (b) for each $P \in \mathcal{C}^-$ there is an unfactored cyclic tree \mathcal{T} for P in T such that $\overline{Pred(\mathcal{T})} \cup \mathcal{O}(\mathcal{T}) \cup \mathcal{N}(\mathcal{T}) \subseteq \mathcal{C}$.

Proof (a) \rightarrow (b). Let $(\mathcal{T}_i \mid 0 \leq i \leq m)$ be a sequence of unfactored cyclic trees in T such that $\bigcup_{i=0}^m \text{Pred}(\mathcal{T}_i) = \mathcal{C}^-$ and $\bigcup_{i=0}^m (\mathcal{O}(\mathcal{T}_i) \cup \mathcal{N}(\mathcal{T}_i)) \subseteq \mathcal{C}^+$.

Let $T' = \{C \in T \mid \exists i \leq m (C \text{ labels some rule node in } \mathcal{T}_i)\}$, then clearly \mathcal{C} is cyclic in T' . Note also that if C labels some rule node in \mathcal{T}_i , then by Definitions 2.1/2, $\text{conseq}(C) \cap \mathcal{C}^- \supseteq \text{conseq}(C) \cap \text{Pred}(\mathcal{T}_i) \neq \emptyset$, and $\text{conseq}(C) \subseteq \text{Pred}(\mathcal{T}_i) \cup \mathcal{O}(\mathcal{T}_i) \subseteq \mathcal{C}^- \cup \mathcal{C}^+$.

But then $\overline{\mathcal{C}^-} \cup (\mathcal{L} - \mathcal{C}^-)$ is a cyclic strong cover in T' . By Theorem 3.3, \mathcal{C}^- is a disjunctive stable model of T' , thus if $P \in \mathcal{C}^-$ we may (by Theorem 2.4(b)) find an unfactored cyclic tree \mathcal{T} for P in T' such that $\text{Pred}(\mathcal{T}) \subseteq \mathcal{C}^- \subseteq \mathcal{L} - (\mathcal{O}(\mathcal{T}) \cup \mathcal{N}(\mathcal{T}))$. Clearly \mathcal{T} is a cyclic tree in T .

To see that $\mathcal{O}(\mathcal{T}) \cup \mathcal{N}(\mathcal{T}) \subseteq \mathcal{C}^+$, let RN_D be a rule node in \mathcal{T} . D labels a rule node in some \mathcal{T}_j , whence $\mathcal{N}(D) \subseteq \mathcal{N}(\mathcal{T}_j) \subseteq \mathcal{C}^+$. By the above remarks, $\text{conseq}(D) \subseteq \mathcal{C}^- \cup \mathcal{C}^+$, whence if $R \in \text{conseq}(D) \cap \mathcal{O}(\mathcal{T})$, then $R \in \mathcal{C}^+$ (since $\mathcal{C}^- \cap \mathcal{O}(\mathcal{T}) = \emptyset$).

(b) \rightarrow (a). This is trivial. ■

The proof of the above theorem has the following corollary.

3.6 Corollary. If \mathcal{C} is cyclic, then \mathcal{C}^- is a disjunctive stable model of $\{C \in T \mid \mathcal{C}^- \models C\}$.

§4 TOP-DOWN QUERY PROCESSING

In this section we informally present a top-down query processing method for the disjunctive stable model semantics, this method being an adaptation of a method first presented for stratified disjunctive databases under the perfect model semantics [Jo98].

The method, illustrated in Example 4.3, is based upon computing cyclic covers using database rules via a hyperresolution-like operation similar to that employed in SLO-resolution [Ra89] and by expanding negative “sub-goals” using cyclic trees. In the latter case, there is a danger that the expansion will produce a high branching factor, and for this reason we attempt to prune the construction of cyclic trees using the following two results.

4.1 Theorem. A consistent set of literals \mathcal{C} is cyclic iff we may find a sequence of cyclic trees $(\mathcal{T}_i \mid i \leq m)$ in T such that

- (a) $\mathcal{C}^- = \bigcup_{i=0}^m \text{Pred}(\mathcal{T}_i)$,
- (b) $\mathcal{C}^+ \supseteq \bigcup_{i=0}^m (\mathcal{O}(\mathcal{T}_i) \cup \mathcal{N}(\mathcal{T}_i))$, and
- (c) if N is a predicate leaf node in \mathcal{T}_i , then $\text{lab}(N) \in \bigcup_{j < i} \text{Pred}(\mathcal{T}_j)$.

Proof. The implication from left to right follows trivially from Definition 3.1.

Suppose that $(\mathcal{T}_i \mid i \leq m)$ satisfies conditions (a)-(c). Let $T' = \{C \in T \mid C \text{ labels a rule node in some } \mathcal{T}_i\}$. Notice that if $M = \mathcal{C}^-$, then as in the proof of Theorem 3.5, \mathcal{C}^- intersects $\text{conseq}(C)$ for every rule $C \in T'$, whence trivially $M \models T'|M$.

We first show that M is a minimal model of $T'|M$.

Suppose that $M' \subset M$ with $M' \models T'|M$. Pick $i \leq m$ such that $\bigcup_{j < i} \text{Pred}(\mathcal{T}_j) \subseteq M'$ and $\text{Pred}(\mathcal{T}_i) \not\subseteq M'$. Pick predicate nodes $N_0 > N_1$ in \mathcal{T}_i such that

- (i) $\{\text{lab}(N) \mid N_0 \geq N \geq N_1\} \cap M' = \emptyset$,
- (ii) if $N > N_0$, then $\text{lab}(N) \in M'$, and
- (iii) if RN_C is the child node of N_1 and N'' is a child node of RN_C , then $\text{lab}(N'') \in M'$.

Notice that by condition (c), N_1 cannot be a leaf node in \mathcal{T}_i , whence RN_C certainly exists. Now $\mathcal{N}(\mathcal{C}) \subseteq \mathcal{N}(\mathcal{T}_i) \subseteq \mathcal{C}^+ \subseteq \mathcal{L} - M$ and $\text{antec}(\mathcal{C}) \subseteq M'$. In addition, $\text{CYC}(N_1) \subseteq \{\text{lab}(N) \mid N_0 \geq N \geq N_1\} \subseteq \mathcal{L} - M'$, whence $\text{conseq}(\mathcal{C}) \subseteq \text{CYC}(N_1) \cup (\text{conseq}(\mathcal{C}) - \text{CYC}(N_1)) \subseteq \text{CYC}(N_1) \cup \mathcal{O}(\mathcal{T}_i) \subseteq \mathcal{L} - M'$. This then contradicts the fact that $M' \models T'|M$.

Thus by Theorem 2.4(b), if $P \in M$, we may find an unfactored cyclic tree \mathcal{T} for P in T' such that $\text{Pred}(\mathcal{T}) \subseteq M \subseteq \mathcal{L} - (\mathcal{O}(\mathcal{T}) \cup \mathcal{N}(\mathcal{T}))$. Clearly \mathcal{T} is cyclic in T , and as in the proof of Theorem 3.5 we have that $\mathcal{O}(\mathcal{T}) \cup \mathcal{N}(\mathcal{T}) \subseteq \mathcal{C}^+$, thus showing that \mathcal{C} is cyclic in T . ■

4.2 Corollary. Suppose that \mathcal{P} is cyclic with associated cyclic trees $(\mathcal{T}_i \mid i \leq m)$ satisfying the conditions of Theorem 4.1. Let \mathcal{C} be cyclic with $\mathcal{P} \subseteq \mathcal{C}$. Then $P \in \mathcal{C}^-$ iff there is a cyclic tree \mathcal{T} for P in T such that

- (a) $\overline{\text{Pred}(\mathcal{T})} \cup \mathcal{O}(\mathcal{T}) \cup \mathcal{N}(\mathcal{T}) \subseteq \mathcal{C}$,
- (b) $\mathcal{P} \cup \overline{\text{Pred}(\mathcal{T})} \cup \mathcal{O}(\mathcal{T}) \cup \mathcal{N}(\mathcal{T})$ is consistent, and
- (c) if N is a predicate leaf node in \mathcal{T} , then $\text{lab}(N) \in \bigcup_{i \leq m} \text{Pred}(\mathcal{T}_i)$.

Moreover if \mathcal{T} satisfies conditions (b) and (c), then $\mathcal{P} \cup \overline{\text{Pred}(\mathcal{T})} \cup \mathcal{O}(\mathcal{T}) \cup \mathcal{N}(\mathcal{T})$ is cyclic.

4.3 Example. Let T consist of the following rules.

- | | | |
|---|---|--|
| 1. $E_0 \wedge \neg B_0 \rightarrow B_3 \vee A_1$ | 2. $B_5 \wedge \neg E_3 \rightarrow B_0 \vee B_1$ | 3. $E_1 \wedge \neg E_5 \rightarrow B_0$ |
| 4. $\neg B_4 \wedge B_5 \rightarrow A_2$ | 5. $E_2 \vee E_3$ | 6. $E_0 \vee E_2$ |
| 7. E_1 | 8. $E_5 \vee E_2$ | 9. $\neg E_2 \rightarrow B_5 \vee B_3$ |
| 10. $\neg A_1 \wedge \neg A_2 \rightarrow P \vee A_0$ | 11. $\neg A_2 \rightarrow B_0$ | |

Suppose that we wish to generate cyclic strong covers of $\mathcal{P} = \{P, A_0, A_1\}$. Applying rule 10, any cover of \mathcal{P} must contain either $\neg A_1$ or $\neg A_2$. The set $\{A_1, \neg A_1\}$ is inconsistent and therefore cannot be contained in a cover. $RN_{A_1 \vee \neg A_1}$ denotes the application of the rule $A_1 \vee \neg A_1$ in verifying the inconsistency. We denote this by the partial tree \mathcal{T}_1 (Figure 4.3(i)). RN_{10} denotes the application of rule 10.

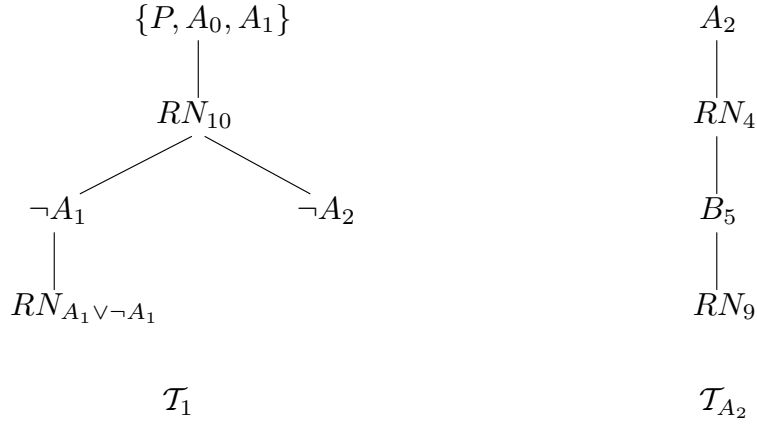


Figure 4.3(i).

Now any set of predicates is trivially cyclic, thus by Corollary 4.2, $\mathcal{P} \cup \{\neg A_2\}$ is contained in some cyclic cover \mathcal{C} iff there is an unfactored cyclic tree \mathcal{T}_{A_2} for A_2 such that $\mathcal{P} \cup \overline{\text{Pred}(\mathcal{T}_{A_2})} \cup \mathcal{O}(\mathcal{T}_{A_2}) \cup \mathcal{N}(\mathcal{T}_{A_2}) \subseteq \mathcal{C}$.

The only such cyclic tree \mathcal{T}_{A_2} is illustrated in Figure 4.3(i). We thus append $\overline{\text{Pred}(\mathcal{T}_{A_2})} \cup \mathcal{O}(\mathcal{T}_{A_2}) \cup \mathcal{N}(\mathcal{T}_{A_2}) = \{\neg A_2, \neg B_5, B_4, B_3, E_2\}$ below $\neg A_2$ (Figure 4.3(ii)). The “rule node” $RN_{\rightarrow \neg A_2}$ simply denotes the generation of cyclic trees to handle the negative “sub-goal” $\neg A_2$. Note that (by Corollary 4.2) the effect of doing so is to restore the cyclic property, ie., $\{P, A_0, A_1, \neg A_2, \neg B_5, B_4, B_3, E_2\}$ is cyclic.

Applying rule 1 (cf., Figure 4.3(ii)), and noting that the set $\{E_0, E_2\}$ cannot be extended to a strong cover (by virtue of rule 6) yields the tree \mathcal{T}_2 depicted in Figure 4.3(ii). The right hand branch again calls for the generation of cyclic trees for B_0 . In this case we can prune the construction of such trees using Corollary 4.2, ie., if we encounter a predicate node whose label occurs in $\text{Pred}(\mathcal{T}_{A_2}) = \{A_2, B_5\}$. We can also discard trees \mathcal{T} for which $\overline{\text{Pred}(\mathcal{T})} \cup \mathcal{O}(\mathcal{T}) \cup \mathcal{N}(\mathcal{T})$ is inconsistent with the literals already lying on the branch to $\neg B_0$.

The trees that require consideration are depicted as \mathcal{T}_{B_0} and \mathcal{T}'_{B_0} in Figure 4.3(ii).

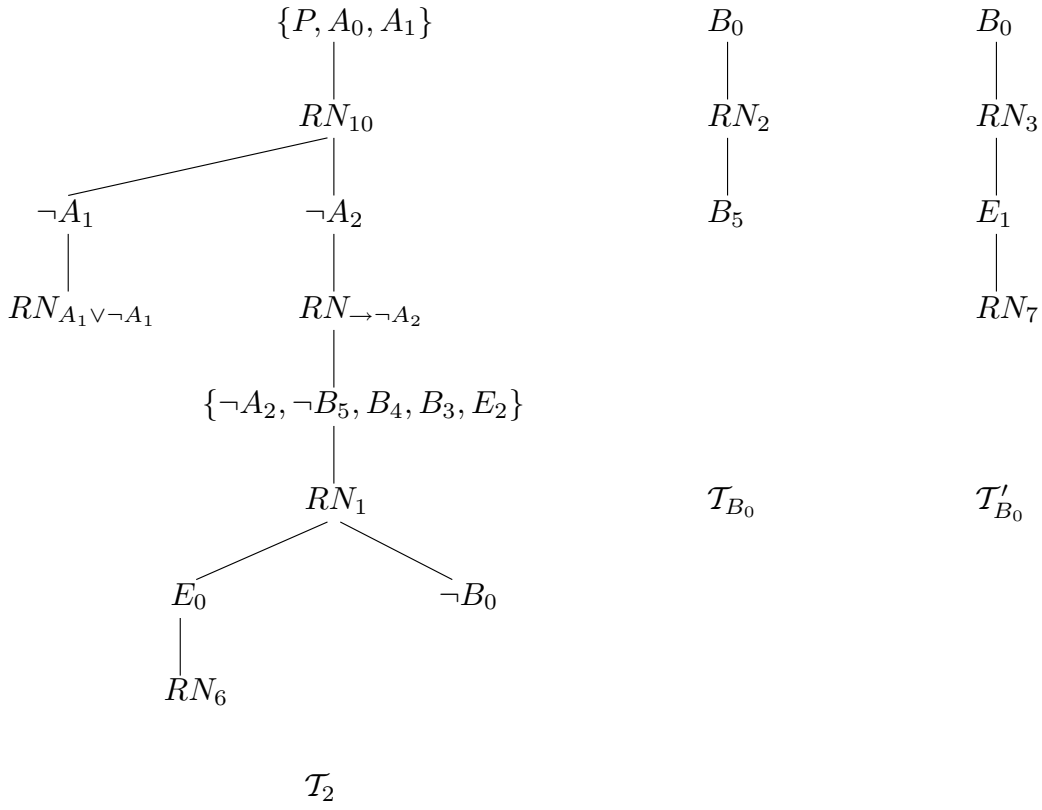


Figure 4.3(ii).

Appending the corresponding sets of literals below $\neg B_0$, and solving the generated branches using rules 5 and 8 yields the final tree depicted in Figure 4.3(iii). This then shows that \mathcal{P} has no cyclic strong cover in T , and hence (by Corollary 3.4) shows that $T \models_{ds} \bigvee \mathcal{P}$. Note that in this case we have not had to resort to the use of the denial rules $P \wedge \neg P \rightarrow \mathbb{F}$. In other cases such denial rules would indeed be required.

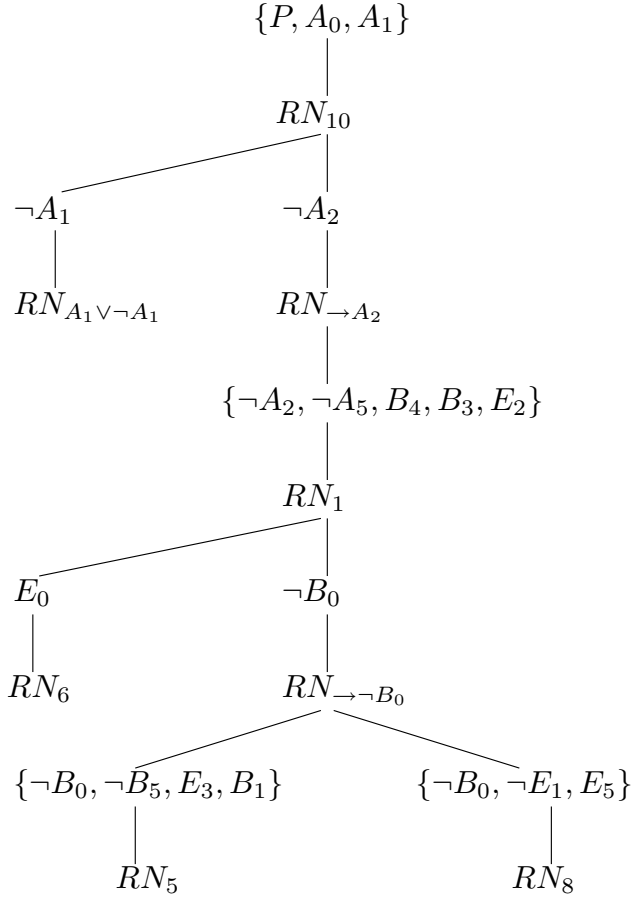


Figure 4.3(iii).

4.4 Query processing.

Note the features of the above construction. Suppose that \mathcal{B} is a branch through the tree and that the set \mathcal{C} of literals lying on the branch is consistent. If \mathcal{B} does not terminate in a negative literal, then \mathcal{C} cyclic. We attempt to expand \mathcal{B} using a rule $C \in T$ such that $\text{conseq}(C) \subseteq \{\mathbf{F}\} \cup \mathcal{C}$ and $\mathcal{C} \cap (\text{antec}(C) \cup \overline{\mathcal{N}(C)}) = \emptyset$. If no such rule exists, then we may conclude that $\mathcal{C} \cup \{\mathbf{F}\}$ is a cyclic strong cover.

If \mathcal{B} terminates in a negative literal, then we construct cyclic trees (satisfying the conditions of Corollary 4.2) in order to restore the cyclic property of the branch. In this case if no such cyclic tree exists, then we may conclude that \mathcal{C} cannot be extended to a cyclic cover.

If, by the above method, we are able to show that \mathcal{P} has no cyclic strong cover \mathcal{C} , then as above we may conclude that $T \models_{ds} \bigvee \mathcal{P}$. The method is thus sound.

The converse (ie., completeness) can be guaranteed by Corollary 3.4 and the implicit addition of the denial rules $P \wedge \neg P \rightarrow \mathbb{F}$. As mentioned earlier, the addition of these denial rules does not alter the disjunctive stable models of T , and each branch generated in the tree is then an attempt to construct a disjunctive stable model of the database that does not satisfy the goal $\bigvee \mathcal{P}$. As we saw in the previous example, we will not in general need to be generate such models in their entirety, this being an important efficiency consideration.

The above method is clearly terminating, since duplicate predicate nodes are never generated along any branch. The method is also top-down. (Top-down methods for constructing cyclic trees are presented in [Jo95, Jo96].)

Note that the use of Corollary 4.2 represents a pruning of the query processing method. Further pruning is also possible. For example would could employ factorisation [Jo98, Section 4.2], or the pruning method described in [Jo98, Section 4.3]. We note also that the above construction insists upon restoring the cyclic property as soon as a negative literal is generated. There is no reason why this could not be delayed until further use of the database rules had been attempted. This might be especially useful if the branching factor produced by generating the cyclic trees were high.

§5 QUERY COMPILATION AND VIEW UPDATES

Throughout this section we assume that \mathcal{L} is the disjoint union, $\mathcal{L} = \text{EXT}(\mathcal{L}) \cup \text{INT}(\mathcal{L})$ with $\mathbb{F} \in \text{INT}(\mathcal{L})$ and that for each rule C , if $\text{antec}(C) \cup \mathcal{N}(C) \neq \emptyset$ (ie., $C \in \text{INT}(T)$) then $\text{conseq}(C) \subseteq \text{INT}(\mathcal{L})$, and if $\text{antec}(C) \cup \mathcal{N}(C) = \emptyset$ (ie., $C \in \text{EXT}(T)$) then $\text{conseq}(C) \subseteq \text{EXT}(\mathcal{L})$. Such assumptions are standard in deductive database research.

Notice that if \mathcal{T} is a cyclic tree in T , $P \in \text{EXT}(\mathcal{L}) \cap \text{Pred}(\mathcal{T})$ and $\text{lab}(N) = P$, then either N is a leaf node, or the child node of N is of the form RN_C , where $C \in \text{EXT}(T)$, $P \in C$, $C - \{P\} \subseteq \mathcal{O}(T)$, and RN_C is a leaf.

5.1 Int-total cyclic covers.

5.1.1 Proposition. Suppose that \mathcal{C} is a cyclic strong cover, then we may extend \mathcal{C} to a cyclic strong cover $\mathcal{D} \supseteq \mathcal{C}$ such that $\mathcal{D}^+ \cup \mathcal{D}^- \supseteq \text{EXT}(\mathcal{L})$.

Proof. If $E \in \text{EXT}(T)$, then $E \not\subseteq \mathcal{C}^+$, whence we may find a minimal model M of $\text{EXT}(T)$ with $M \subseteq \text{EXT}(\mathcal{L}) - \mathcal{C}^+$. If $P \in \mathcal{C}^- \cap \text{EXT}(\mathcal{L})$, then by the above remarks we may find a rule $E \in \text{EXT}(T)$ such that $P \in E$ and $E - \{P\} \subseteq \mathcal{C}^+$, whence $P \in M$. Thus $\mathcal{D} = \mathcal{C} \cup \overline{M} \cup (\text{EXT}(\mathcal{L}) - M)$ is consistent. \mathcal{D} is a strong cover by virtue of the fact that \mathcal{C} is a strong cover, and M models $\text{EXT}(T)$.

If $P \in \text{INT}(\mathcal{L}) \cap \mathcal{D}^- \subseteq \mathcal{C}^-$, then we may find an unfactored cyclic tree \mathcal{T} for P in T such that $\overline{\text{Pred}(\mathcal{T})} \cup \mathcal{O}(\mathcal{T}) \cup \mathcal{N}(\mathcal{T}) \subseteq \mathcal{C} \subseteq \mathcal{D}$.

If $P \in \text{EXT}(\mathcal{L}) \cap \mathcal{D}^- \subseteq M$, then by the minimality of M we may find a rule $E' \in \text{EXT}(T)$ such that $P \in E'$ and $E' - \{P\} \subseteq \text{EXT}(\mathcal{L}) - M \subseteq \mathcal{D}$. Let \mathcal{T}' be the unfactored cyclic tree consisting of a single predicate node (labelled with P) and a single rule node $RN_{E'}$, whence $\overline{\text{Pred}(\mathcal{T}')} \cup \mathcal{O}(\mathcal{T}') \cup \mathcal{N}(\mathcal{T}') = \{\neg P\} \cup (E' - \{P\}) \subseteq \mathcal{D}$. This then shows that \mathcal{D} is cyclic. ■

5.1.2 Definition. A set of literals I is *int-total* iff $I^+ \cup I^- \supseteq \text{INT}(\mathcal{L})$.

5.1.3 Corollary (a) $T \models_{ds} \bigvee \mathcal{P}$ iff whenever \mathcal{C} is an int-total cyclic cover of \mathcal{P} , then $\text{EXT}(T) \models \bigvee \text{EXT}(\mathcal{L}) \cap \mathcal{C}^+$.

(b) If \mathcal{C} is an int-total cyclic cover in T , then

$$T \models_{ds} \bigvee \mathcal{C} \iff \text{EXT}(T) \models \bigvee \text{EXT}(\mathcal{L}) \cap \mathcal{C}^+$$

The preceding corollary shows that we can characterise processing under the disjunctive stable model semantics using int-total (as opposed to total) cyclic strong covers.

5.2 Query compilation.

The general structure of a deductive database is expected to be a large and transient extension, and a smaller and relatively static intension. This suggests that for recurrent queries we might employ query compilation, in which we pre-process the intensional database once, and following which, processing of the compiled query requires a manipulation of the extensional database only. Changes to the intensional database would of course require re-compilation, but the more frequent changes to the extensional database would not. Compilation under the GCWA and the perfect model semantics is discussed in [He88] and [Jo99a] respectively.

Suppose that our query is of the form $?\bigvee \mathcal{P}$, where \mathcal{P} is a set of literals.

5.2.1 Definition. A consistent set of literals \mathcal{C} is *weakly cyclic* iff there is a sequence of cyclic trees $(\mathcal{T}_i \mid i \leq m)$ in $\text{INT}(T)$ such that

- (i) $\mathcal{C}^- = \bigcup_{i \leq m} \text{Pred}(\mathcal{T}_i)$,
- (ii) $\mathcal{C}^+ \supseteq \bigcup_{i \leq m} (\mathcal{O}(\mathcal{T}_i) \cup \mathcal{N}(\mathcal{T}_i))$, and
- (iii) if N is a predicate leaf node in some \mathcal{T}_i , then $\text{lab}(N) \in \text{EXT}(\mathcal{L})$.

Note that the definition of weakly cyclic is independent of $\text{EXT}(T)$. Note also that since each of the cyclic trees \mathcal{T}_i is a tree in $\text{INT}(T)$, if N is a predicate node in \mathcal{T}_i with $\text{lab}(N) \in \text{EXT}(\mathcal{L})$, then (by the remark preceding Section 5.1), N is a leaf.

Clearly in order to extend a weakly cyclic set to a fully cyclic set, we need to find for each $P \in \mathcal{C}^- \cap \text{EXT}(\mathcal{L})$, a rule $E_P \in \text{EXT}(T)$ such that $P \in E_P$ and $(E_P - \{P\}) \cap \mathcal{C}^- = \emptyset$, and then append RN_{E_P} as a child of each predicate node labelled with P . Such an

extension would then allow us to extend each of the cyclic trees \mathcal{T}_i to an unfactored tree.

We show below that computing weakly cyclic covers in $\text{INT}(T)$ allows us to compile the query $?\bigvee \mathcal{P}$.

5.2.2 Theorem. If \mathcal{P} is a set of literals then $T \models_{ds} \bigvee \mathcal{P}$ iff whenever \mathcal{C} is an int-total weakly cyclic cover of \mathcal{P} (in $\text{INT}(T)$), then $\bigvee \{K \in \mathcal{C} \mid K = P \text{ or } K = \neg P, \text{ where } P \in \text{EXT}(\mathcal{L})\}$ is true in every minimal model of $\text{EXT}(T)$.

Proof (\rightarrow). Suppose that \mathcal{C} is an int-total weakly cyclic cover of \mathcal{P} , and that M is a minimal model of $\text{EXT}(T)$ such that $M \not\models \bigvee \{K \in \mathcal{C} \mid K = P \text{ or } K = \neg P, \text{ where } P \in \text{EXT}(\mathcal{L})\}$. If $P \in \text{EXT}(\mathcal{L}) \cap \mathcal{C}^-$, then $P \in M$, whence by the minimality of M we may find a rule $E_P \in \text{EXT}(T)$ such that $P \in E_P$ and $(E_P - \{P\}) \cap M = \emptyset$. But then by the above remarks, $\mathcal{D} = \mathcal{C} \cup \{E_P - \{P\} \mid P \in \text{EXT}(\mathcal{L}) \cap \mathcal{C}^-\}$ is cyclic. \mathcal{D} is trivially a cover of \mathcal{P} , since \mathcal{C} is a cover, and $\mathcal{D}^+ - \mathcal{C}^+ \subseteq \text{EXT}(\mathcal{L})$.

If $E \in \text{EXT}(T)$ with $E \subseteq \mathcal{D}^+$, then $E \subseteq \text{EXT}(\mathcal{L}) \cap \mathcal{D}^+ \subseteq \text{EXT}(\mathcal{L}) - M$, thus contradicting the fact that $M \models \text{EXT}(T)$. Thus \mathcal{D} is a strong cover, hence contradicting Corollary 5.1.3(a).

(\leftarrow). If \mathcal{C} is a total cyclic strong cover of \mathcal{P} , then \mathcal{C} is also int-total and weakly cyclic. Since \mathcal{C} is a total strong cover, $\text{EXT}(\mathcal{L}) \cap \mathcal{C}^- \models \text{EXT}(T)$, and for each $P \in \text{EXT}(\mathcal{L}) \cap \mathcal{C}^-$ we may (by the remarks preceding Section 5.1) find a rule $E \in \text{EXT}(T)$ such that $P \in E$ and $(E - \{P\}) \cap \mathcal{C}^- = \emptyset$, thus showing that $\text{EXT}(\mathcal{L}) \cap \mathcal{C}^-$ is a minimal model of $\text{EXT}(T)$. \mathcal{C} thus contradicts the conditions of the right hand side. ■

5.2.3 Query compilation and processing.

Query compilation thus consists of computing int-total weakly cyclic covers, this requiring a manipulation of $\text{INT}(T)$ only. It is easy to see that such covers can be computed by a variant of the method suggested in Example 4.3. Query processing then requires us to check that such covers satisfy the condition given in Theorem 5.2.2, this requiring a manipulation of the extensional database only.

5.3 View updates

View updating is concerned with implicitly updating derived predicates (ie., predicates in $\text{INT}(\mathcal{L})$) by means of an explicit updating of $\text{EXT}(T)$.

5.3.1 Definition [Gr93]. Given a set $\mathcal{Q} \subseteq \mathcal{L}$, the problem of inserting $\bigvee \mathcal{Q}$ into T is that of finding a database T' such that

- (i) $T' \models \bigvee \mathcal{Q}$,
- (ii) $\text{INT}(T') = \text{INT}(T)$, $\text{EXT}(T') \supseteq \text{EXT}(T)$, and
- (iii) if T^* satisfies conditions (i) and (ii) above, then $\text{EXT}(T^*) \models \text{EXT}(T')$.

Condition (iii) captures the notion that T' should be the weakest strengthening of T which satisfies $\bigvee \mathcal{Q}$.

5.3.2 Definition. Let $cl(T, \mathcal{Q})$ denote the minimal superset of T such that if \mathcal{C} is an int-total cyclic cover of \mathcal{Q} in $cl(T, \mathcal{Q})$, then $\bigvee \text{EXT}(\mathcal{L}) \cap \mathcal{C}^+ \in cl(T, \mathcal{Q})$.

We can in fact form $cl(T, \mathcal{Q})$ by computing int-total weakly cyclic covers in $\text{INT}(T)$ once, and then completing these covers in the manner suggested in the remark preceding Theorem 5.2.2. This promotes greater efficiency in performing the closure operation.

It is easy to see that $cl(T, \mathcal{Q})$ is unique and (by Corollary 5.1.3(a)) satisfies conditions (i) and (ii) of Definition 5.3.1. Theorem 5.3.4 below also shows that $cl(T, \mathcal{Q})$ satisfies condition (iii) of Definition 5.3.1.

5.3.3 Lemma [Jo99a]. Suppose that $\text{INT}(T^*) = \text{INT}(T^+)$ and $\text{EXT}(T^*) \models \text{EXT}(T^+)$. If \mathcal{C} is a cyclic cover in T^+ , then either $\text{EXT}(T^*) \models \bigvee \text{EXT}(\mathcal{L}) \cap \mathcal{C}^+$, or \mathcal{C} is a cyclic cover in T^* .

5.3.4 Theorem. If $T^* = T \cup \{\bigvee E_j \mid j \leq m\} \models_{ds} \bigvee \mathcal{Q}$ with each $E_j \subseteq \text{EXT}(\mathcal{L})$, then $\text{EXT}(T^*) \models \text{EXT}(cl(T, \mathcal{Q}))$.

Proof. Let $T^+ = \text{INT}(T) \cup \{E \in \text{EXT}(cl(T, \mathcal{Q})) \mid \text{EXT}(T^*) \models E\}$. Suppose that \mathcal{C} is an int-total cyclic cover of \mathcal{Q} in T^+ , then $\bigvee \text{EXT}(\mathcal{L}) \cap \mathcal{C}^+ \in cl(T, \mathcal{Q})$, and by the preceding lemma, either $\text{EXT}(T^*) \models \bigvee \text{EXT}(\mathcal{L}) \cap \mathcal{C}^+$ or \mathcal{C} is an int-total cyclic cover in T^* (in which case $\text{EXT}(T^*) \models \bigvee \text{EXT}(\mathcal{L}) \cap \mathcal{C}^+$ by Corollary 5.1.3(a)).

Thus $\bigvee \text{EXT}(\mathcal{L}) \cap \mathcal{C}^+ \in T^+$, whence by the minimality of $cl(T, \mathcal{Q})$, we have that $T^+ = cl(T, \mathcal{Q})$, thus proving the result. ■

§6 PARTIAL STRATIFICATION

We can view the partitioning of \mathcal{L} into $\text{EXT}(\mathcal{L})$ and $\text{INT}(\mathcal{L})$ as a very weak form of stratification. We saw in the last section that this yields improvements in the efficiency of our query processing, in that we are then only required to compute int-total (as opposed to total) cyclic strong covers.

In this section we take this idea one step further, assuming that part of the database is indeed stratified. Specifically, we assume that \mathcal{L} is the disjoint union, $\mathcal{L} = \mathcal{L}_s \cup \mathcal{L}_u$, and that a level function $\ell : \mathcal{L}_s \rightarrow \{0, 1, 2, \dots\}$ exists such that if C is a rule in T with $P \in \text{conseq}(C) \cap \mathcal{L}_s$, then

- (i) $\text{antec}(C) \cup \mathcal{N}(C) \cup \text{conseq}(C) \subseteq \mathcal{L}_s$,
- (ii) if $R \in \text{conseq}(C)$, then $\ell(R) = \ell(P)$,
- (iii) if $R \in \text{antec}(C)$, then $\ell(R) \leq \ell(P)$, and
- (iv) if $R \in \mathcal{N}(C)$, then $\ell(R) < \ell(P)$.

Let $\text{STRAT}(T) = \{C \in T \mid \text{conseq}(C) \cap \mathcal{L}_s \neq \emptyset\}$. A set of literals I is \mathcal{L}_u -total iff $I^+ \cup I^- \supseteq \mathcal{L}_u$.

6.1 Definition. If $M \subseteq \mathcal{L}_s$, then $T \parallel M$ is formed from $T - \text{STRAT}(T)$ by

- (i) discarding any rule whose body (ie., $\text{antec}(C) \cup \overline{\mathcal{N}(C)}$) contains some literal in $\overline{M} \cup (\mathcal{L}_s - M)$, and then
- (ii) discarding from the body of all remaining rules any literals in $\mathcal{L}_s \cup \overline{\mathcal{L}_s}$.

Definition 6.1 is an extension of the Gelfond-Lifschitz transformation (Definition 1.1) in that both positive and negative literals in the body of a rule are evaluated in $\overline{M} \cup (\mathcal{L}_s - M)$.

6.2 Lemma. If \mathcal{C} is a cyclic strong cover in T , then $\mathcal{C} \cap (\mathcal{L}_s \cup \overline{\mathcal{L}_s})$ is cyclic strong cover in $\text{STRAT}(T)$.

6.3 Theorem. M is a disjunctive stable model of T iff

- (i) $M \cap \mathcal{L}_s$ is a disjunctive stable model of $\text{STRAT}(T)$, and
- (ii) $M \cap \mathcal{L}_u$ is a disjunctive stable model of $T \parallel (M \cap \mathcal{L}_s)$.

Proof. Let $M_s = M \cap \mathcal{L}_s$, $M_u = M \cap \mathcal{L}_u$ and $T_u = T \parallel M_s$.

(\rightarrow). Suppose that M is a disjunctive stable model of T , then by Theorem 3.3, $\overline{M} \cup (\mathcal{L} - M)$ is a cyclic strong cover in T , whence $\overline{M_s} \cup (\mathcal{L}_s - M_s)$ is a cyclic strong cover in $\text{STRAT}(T)$, thus showing that M_s is a disjunctive stable model of $\text{STRAT}(T)$.

Since $M \models T$, it is trivially the case that $M_u \models T_u$, whence $M_u \models T_u \parallel M_u$. Suppose that $M' \subset M_u$ is a model of $T_u \parallel M_u$. We show that $M_s \cup M' \models T \parallel M$, thus contradicting the minimality of M . Suppose that $C \in T$ with $\mathcal{N}(C) \cap M = \emptyset$ and $\text{antec}(C) \subseteq M_s \cup M'$. If $C \in \text{STRAT}(T)$, then since $M_s \models \text{STRAT}(T)$, we must have that $\text{conseq}(C) \cap M_s \neq \emptyset$. Suppose that $C \notin \text{STRAT}(T)$, then $C \parallel M_s \in T_u$, $\mathcal{N}(C \parallel M_s) \cap M_u = \emptyset$ and $\text{antec}(C \parallel M_s) \subseteq M'$, whence $\text{conseq}(C) \cap M' \neq \emptyset$.

(\leftarrow). Since $M_s \models \text{STRAT}(T)$ and $M_u \models T \parallel M_s$, it is trivially the case that $M \models T$, whence $M \models T \parallel M$. Suppose that $M' \subset M$ with $M' \models T \parallel M \supseteq \text{STRAT}(T) \parallel M_s$. Since M_s is a disjunctive stable model of $\text{STRAT}(T)$, we must have by minimality that $M' \cap \mathcal{L}_s = M_s$.

But then $M' \cap \mathcal{L}_u \models T_u \parallel M_u$, for suppose that $C \in T - \text{STRAT}(T)$ with $C \parallel M_s \in T_u$, $\mathcal{N}(C \parallel M_s) \cap M_u = \emptyset$ and $\text{antec}(C \parallel M_s) \subseteq M'$. Then $\text{antec}(C) \subseteq M'$ and $\mathcal{N}(C) \cap M = \emptyset$, whence $M' \cap \text{conseq}(C) \neq \emptyset$.

This then contradicts the minimality of M_u . ■

6.4 Theorem. If \mathcal{C} is a cyclic strong cover in T , then \mathcal{C} may be extended to a cyclic strong cover $\mathcal{D} \supseteq \mathcal{C}$ such that $\mathcal{D}^+ \cup \mathcal{D}^- \supseteq \mathcal{L}_s$.

Proof. By Lemma 6.2, $\mathcal{C} \cap (\mathcal{L}_s \cup \overline{\mathcal{L}_s})$ is a cyclic strong cover in $\text{STRAT}(T)$, whence

by [Jo99a, Corollary 2.2.7], we may find a perfect (and hence disjunctive stable) model $M \subseteq \mathcal{L}_s$ of $\text{STRAT}(T)$ such that $M \not\models \bigvee \mathcal{C} \cap (\mathcal{L}_s \cup \overline{\mathcal{L}_s})$.

As in the proof of Proposition 5.1.1, $\mathcal{D} = \mathcal{C} \cup \overline{M} \cup (\mathcal{L}_s - M)$ is the required cyclic strong cover. ■

6.5 Corollary. If \mathcal{P} is a set of literals, then $T \models_{ds} \bigvee \mathcal{P}$ iff \mathcal{P} has no \mathcal{L}_u -total cyclic strong cover in T .

§7 DISJUNCTIVE STATIONARY MODELS

7.1 Definition [Pr91]. A consistent set of literals I is a *disjunctive stationary model* of T iff

- (i) $\neg \mathbb{F} \in I$,
- (ii) I^+ is a minimal model of $T|(\mathcal{L} - I^-)$ and
- (iii) $\mathcal{L} - I^-$ is a minimal model of $T|I^+$.

The definition of disjunctive stationary model given in [Pr91] is shown to be equivalent to the above in [Pr91, Remark 3.1]. Disjunctive stationary models are also known as disjunctive partial stable models. For definite databases, disjunctive stationary/stable models coincide with stationary/stable models [Pr94], and in which case the well-founded model is the smallest (with respect to inclusion) stationary model. In the stratified case, disjunctive stationary models coincide with the perfect models.

In [Jo99] we showed that in the definite case, query processing under the disjunctive stationary model semantics can be characterised by supported covers. Specifically if I is a stationary model of T with $\neg \mathbb{F} \in I$, then \bar{I} is a supported strong cover, and if \mathcal{C} is a supported strong cover, then $\bar{\mathcal{C}}$ may be extended to a stationary model of T . As mentioned earlier, we regard cyclicity as the natural generalisation of supportedness to the disjunctive case. However the analogous characterisation does not hold, as is indicated by the following example.

7.2 Example. Let $T = \{A \vee B \vee C, \neg B \rightarrow C, \neg C \rightarrow B, \neg B \wedge \neg C \rightarrow A\}$, then $I = \{A, \neg \mathbb{F}\}$ is a disjunctive stationary model of T since $I^+ = \{A\}$ is a minimal model of $T|(\mathcal{L} - I^-) = \{A \vee B \vee C\}$, and $\mathcal{L} - I^- = \{A, B, C\}$ is a minimal model of $T|I^+ = \{A \vee B \vee C, A, B, C\}$. However, $\bar{I} = \{\neg A, \mathbb{F}\}$ is not cyclic, nor can it be extended to a cyclic cover in T .

The reason for this discrepancy is that disjunctive stationary models and cyclic covers employ different approaches to support. If I is disjunctive stationary with $P \in I^+$, then P belongs to a minimal model of $T|(\mathcal{L} - I^-)$, whence by Theorem 2.4(b), we may find an unfactored cyclic tree \mathcal{T} for P in T such that $\text{Pred}(\mathcal{T}) \subseteq I^+ \subseteq \mathcal{L} - \mathcal{O}(\mathcal{T})$ and $\mathcal{N}(\mathcal{T}) \subseteq I^-$. Notice that there is no requirement that $\mathcal{O}(\mathcal{T}) \subseteq I^-$ as would be the case if \bar{I} were cyclic.

We have been unable to establish whether for every cyclic strong cover \mathcal{C} , we may find a disjunctive stationary model containing $\bar{\mathcal{C}}$.

§8 CONCLUSIONS AND SUGGESTIONS FOR FUTURE RESEARCH

We have shown that disjunctive stable models can be characterised in terms of cyclic covers, and in particular that such covers provide a powerful technique for characterising the properties of query processing, compilation and view updating. It is perhaps worthy of note that our top-down method (apparently) requires the addition of denial rules (of the form $P \wedge \neg P \rightarrow \mathbb{F}$) in order to achieve completeness in the unstratified case, thus suggesting that in general query processing will be less efficient in this case.

This addition of denial rules is in turn related to the non-existence of (disjunctive) stable models, and suggests the following questions. If we partition \mathcal{L} into $\text{EXT}(\mathcal{L})$ and $\text{INT}(\mathcal{L})$ as in Section 5, then under what conditions can every minimal model of $\text{EXT}(T)$ be extended to a stable model of T ? (Theorem 6.3 provides a partial answer to this question.) Under what conditions can disjunctive stationary models be extended to disjunctive stable models?

Cyclic trees can also be employed to perform top-down computation of DWFS [Di99], although the relationship between the two is somewhat more complex than that seen in the current paper. These details will be presented in a sequel. This raises the question as to whether the methods of the current paper could be employed, suitably amended, to other semantics for the unstratified case.

A first order deductive database represents the set of its ground instances. Since such databases are assumed to be function free, this set of ground instances is finite, and hence our restriction to the ground level is justified from a theoretical standpoint. The issue of computing cyclic covers for first order databases is discussed at length in [Jo95, Jo98]. In fact this is not significantly more complex than at the propositional level, since in particular cyclic trees are always ground, even for first order databases. One issue that is significantly more complex at the first order level is termination of top-down methods. In the stratified case the methods of [Jo98, Section 6] may be applied. These techniques could also be applied to the unstratified case provided we were prepared to assume that part of the database were stratified (as in Section 6).

References

- [Ba91] C. Baral, J. Lobo and J. Minker, WF^3 : 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.

- [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).
- [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.
- [Di95a] J. Dix, A classification theory of semantics of normal logic programs: II. Weak properties, *Fundamenta Informaticae*, vol. 12 (1995), 257-288.
- [Di99] J. Dix, A general framework for semantics of disjunctive logic programs based upon partial evaluation, *J. Logic Programming*, to appear.
- [Fa91] F. Fages, A new fixpoint semantics for general logic programs compared with the well-founded and the stable model semantics, *New Generation Computing*, vol. 9 (1991), 425-443.
- [Fe95] J. Fernández, J. Minker and A. Yahya, Computing perfect and stable models using ordered model trees, *Computational Intelligence*, vol. 11 (1995), 89-112.
- [Ge88] M. Gelfond and V. Lifschitz, The stable model semantics for logic programming, in: R. Kowalski and K. Bowen (eds.), Proc. 5th International 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.
- [Gr93] J. Grant, J. Horty, J. Lobo and J. Minker, View updates in stratified disjunctive databases, *J. Automated Reasoning*, vol. 11 (1993), 249-267.
- [He88] L. J. Henschen and H. Park, Compiling the GCWA in indefinite databases, in J. Minker, ed., *Foundations of Deductive Databases* (Morgan Kaufmann, Washington, 1988), 395-438.
- [Jo95] C. A. Johnson, Query processing in indefinite stratified databases, Computer Science technical report TR95-14, Keele University (1995).
- [Jo96] C. A. Johnson, On computing minimal and perfect model membership, *Data and Knowledge Engineering*, vol. 18 (1996), 225-276.
- [Jo97] C. A. Johnson, Deduction trees and the view update problem in indefinite deductive databases, *J. Automated Reasoning*, vol. 19 (1997), 31-85.
- [Jo98] C. A. Johnson, Top-down query processing in indefinite stratified databases, *Data and Knowledge Engineering*, vol. 26 (1998), 1-36. (Extracted from [Jo95].)
- [Jo99] C. A. Johnson, Processing indefinite deductive databases under the possible model semantics, Computer Science technical report, Keele University (1999).
- [Jo99a] C. A. Johnson, On cyclic covers and perfect models, *Data and Knowledge Engineering*, to appear.
- [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, Fachberichte Informatik, Universität Koblenz-Landau, research report 5/96.
- [Pr88] T. C. Przymusiński, On the declarative semantics of deductive databases and logic programs, in: J. Minker (ed.), Foundations of Deductive Databases and Logic Programming, (Morgan Kaufman, Washington, 1988), 193-216.
- [Pr91] T. C. Przymusiński, Stable semantics for disjunctive programs, New Generation Computing, vol. 9 (1991), 401-424.
- [Pr94] T. C. Przymusiński, Well-founded and stationary models of logic programs, Annals of Mathematics and Artificial Intelligence, vol. 12 (1994), 141-187.
- [Ra89] A. Rajasekar, Semantics for Disjunctive Logic Programs, PhD thesis, University of Maryland (1989).
- [Ya96] A. Yahya, A goal-driven approach to efficient query processing in disjunctive databases, Institut für Informatik, Universität München, research report PMS-FB-1996-12.