Updates in Disjunctive Deductive Databases: A Minimal Model Based Approach

Adnan H. Yahya

Electrical Engineering Department, Birzeit University, Palestine
Email: yahya@ee.birzeit.edu

Abstract. The issue of updates in Disjunctive Deductive Databases (DDDBs) under the minimal model semantics is addressed. We consider ground clause addition and deletion in a DDDB. The approach of this paper is based on manipulating the clauses of the theory to produce the required change to the minimal model structure necessary to achieve the clause addition/deletion update. First we deal with ground positive clause updates in ground DDDBs. Later we consider positive, then general, clause addition/deletion in the class of range restricted DDDBs. When we give more than one algorithm for a case we comment on the comparative merits and limitations of each. We use the freedom offered by the multiple possibilities for achieving an update to select the one with the least change to the minimal model structure of the theory. We argue that such minimality is desirable if one interprets the minimal model structure as representing the possible states of the modeled world and therefore an update must affect them minimally.

Keywords: Disjunctive Deductive Databases (DDDBs), Database Updates, Minimal Model Semantics, Minimal Model Generation.

1 Introduction

As a routine task in database maintenance, the update problem was extensively covered in the literature [1, 7, 4, 25, 27, 14, 15, 13, 26, 28]. Different semantics were suggested and several methods for accomplishing updates were advanced. Since more than one way may be available to accomplish an update, minimality under certain criteria can be used to decide in favor of a particular choice.

Minimal models play an important role in understanding disjunctive deductive databases (DDDBs). Given that the minimal model structure is used to define the semantics of a DDDB, it is natural to specify updates in terms of changes to the minimal model structure. Update minimality in this case may be interpreted as the least change to the minimal model structure of the theory necessary to accomplish the required update. This need not be minimal in terms of change to the clausal structure. It is

known that minor changes to the clausal structure can reflect drastically on the minimal model structure of a theory and vice versa. Therefore, it is of interest to study the change to the clausal representation sufficient so that certain changes to the minimal model structure of the database can be achieved.

In this paper we address the issue of executing clause addition/deletion updates using an approach based on manipulating the minimal model structure of the theory. We consider general clause updates and do not restrict ourselves to fact insertions and deletions. We present several methods to deal with the update problem and show the merits and limitations of each. Both the update itself and update minimality are defined in terms of the effect on the minimal model structure of the theory. However, updates are accomplished by changing the theory itself and not the individual minimal models directly. This is, in our view, the better choice since the program encodes more than the minimal model structure of the theory. That is, the theory and its minimal model representation are generally not exchangeable [19, 29]

The rest of the paper is organized as follows. In the next section we give some definitions and background material. In section 3 we address the issue of updating ground DDDBs through isolating the component of the theory that needs to be modified to achieve the required change to the minimal model structure. In section 4 we treat the class of range restricted DDDBs: a larger and more natural class of DDDBs and offer two algorithms to accomplish each type of update. In section 5 we extend our results to accommodate the case of nonpositive clause updates. We comment on the possible interpretations of such updates and the way these interpretations can influence the update algorithms. We also expand on the update minimality criteria used in the paper. In the last section we compare our approach with others reported in the literature, make some comments and point to possible directions for further research.

2 Background and Notation

We adopt the usual notation relating to DDDBs as, e.g., in [21]. Here we limit ourselves to the basic material needed for presenting the results of this paper.

Definition 1. (DDDB) A *disjunctive deductive database (DDDB)*, DB, is a set of clauses in implication form: $C = A_1 \vee \cdots \vee A_m \leftarrow B_1 \wedge \ldots \wedge B_n$, where $m, n \geq 0$ and the A_i and B_j are atoms in a First Order Language (FOL) $\mathcal{L}$. C is positive if $n = 0$ (that is, the body is $\top$, *true*, empty) and negative or denial if $m = 0$ (that is, the head is $\bot$, *false*, empty).

By $Head(C)$ we denote the disjunction of atoms $A_1 \vee \cdots \vee A_m$ and by $Body(C)$ we denote the conjunction of atoms $B_1 \wedge \ldots \wedge B_n$. So $C = Head(C) \leftarrow Body(C)$.

The Herbrand base of DB, HB_{DB}, is the set of all ground atoms that can be formed using the predicate symbols and constants in $\mathcal{L}$. We identify an interpretation by the set of ground atoms it assigns the truth value *true*. A *Herbrand interpretation* is any subset of HB_{DB}. A Herbrand model of DB, M, is a Herbrand interpretation such that $M \models DB$ (all clauses of DB are *true* in M). M is *minimal* if no proper subset of M is a model of DB. The set of all minimal models of DB is denoted by $\mathcal{MM}(DB)$. Frequently, a DDDB is divided into three components: the extensional database (EDB) or the, possibly disjunctive, facts; the intensional database (IDB) consisting of the derivation rules; and the integrity constraints (IC) which can be denials or general clauses [21].

Definition 2. (Range-Restriction) A clause C is range-restricted if every variable occurring in the head of C also appears in the body of C. A database is range-restricted if and only if all its clauses are range-restricted.

Definition 3. If C is a clause and DB is a DDDB then by $\mathcal{MM}(DB)_C$ and $\mathcal{MM}(DB)_{\neg C}$ we denote the set of minimal models of DB is which C is *true* and *false*, respectively. Clearly, $\mathcal{MM}(DB) = \mathcal{MM}(DB)_C \cup \mathcal{MM}(DB)_{\neg C}$.

This definition is extended to the case of a set of clauses in a straightforward manner.

By $Atoms(C)$ we denote the set of atoms in a clause C.

By $min(S)$ we denote the set of minimal elements (relative to set inclusion) of the set S.

If A is an atom and DB is a DDDB then $DB \vee A = \{C \vee A | C \in DB\}$.

A model is treated as a conjunction of atoms and its negation is the disjunction of all negative literals corresponding to the atoms in that model. If $M = P_1 \wedge ... \wedge P_n$ then $\neg M = M \rightarrow \bot = \neg P_1 \vee ... \vee \neg P_n$

When no confusion arises we may refer to both positive clauses and models as sets of their constituent atoms.

Definition 4. A positive clause, C, is minimally derivable from DB iff it is derivable from DB but no proper subclause of C is derivable from DB. Clearly, a positive clause C is derivable from DB iff a (not necessarily proper) subclause of C is minimally derivable from DB.

Lemma 5. *[33] A positive clause C is minimally derivable from a DDDB, DB, iff $DB \vdash C$ and $\forall A \in Atoms(C)\ \exists M \in \mathcal{MM}(DB) | M \cap Atoms(C) = \{A\}$.*

Under the minimal model semantics the natural way to define database completions is in terms of the (Extended) Generalized Closed World Assumption, (E)GCWA [22, 33]. It reconciles the concepts of derivability from the completed database and being *true* in all minimal models.

We consider two classes of updates: adding a clause to a DDDB and deleting a clause from a DDDB. Under the minimal model semantics the addition can be viewed as making the clause *true* in all minimal models and the deletion as making it *false* in at least one minimal model of the database.

Definition 6. (Clause Addition) Let DB be a DDDB and C be a positive clause such that $DB \nvdash C$. We say that DB' is the result of adding C to DB if $DB' \vdash C$. In terms of minimal models[1]: $\exists N \in \mathcal{MM}(DB)\ s.t. N \not\models C$ and $\forall M \in \mathcal{MM}(DB')\ M \models C$

Definition 7. (Clause Deletion) Let DB be a DDDB and C be a positive clause such that $DB \vdash C$. We say that DB' is the result of deleting C from DB iff $DB' \nvdash C$. In terms of minimal models[1]: $\forall M \in \mathcal{MM}(DB)\ M \models C$ and $\exists N \in \mathcal{MM}(DB')\ s.t. N \not\models C$.

There may exist a number of DB's accomplishing an update. We are interested in minimal change in some sense. In particular, we are concerned with databases accomplishing the update with the least change to the minimal model structure of DB[2]. Our approach is based on manipulating the minimal model structure of the theory to achieve the required update.

Lemma 8. *Let DB be a DDDB and let $\mathcal{M}$ be a set of models for DB.*

1. *If C is a positive clause and $\mathcal{C} = \{M \rightarrow C | M \in \mathcal{M}\}$ and $DB'' = DB \cup \mathcal{C}$ then DB'' has the set of minimal models $\mathcal{MM}(DB'') = min(\mathcal{MM}(DB)_{\mathcal{C}} \cup \{(M \cup \{A\})^* | M \in \mathcal{M}$ and $A \in Atom(C)\})$, where $(M \cup \{A\})^*$ is the extension of $(M \cup \{A\})$ to a model of DB''.*
2. *If $DB' = DB \cup \mathcal{C}$, where $\mathcal{C} = \{M \rightarrow \bot | M \in \mathcal{M}\}$ then DB' has the set of minimal models $\mathcal{MM}(DB') = \mathcal{MM}(DB) \setminus \mathcal{M}$.*

[1] The statement of this definition in terms of minimal models is applicable to nonpositive clauses as well.

[2] More on the update minimality issue to be found in paragraph 5.3.

Proof. 1. Let N be a minimal model of DB''. $N \models DB$. If $N \in \mathcal{MM}(DB)$ and since $N \models \mathcal{C}$ then $N \in \mathcal{MM}(DB)_{\mathcal{C}}$. If not, N must be a minimal model for $DB \cup \mathcal{C}$ for otherwise N', a subset of N ($M \subseteq N' \subset N$, for some $M \in \mathcal{M}$), is a minimal model for DB'' contradicting our assumption.

If $N \in \mathcal{MM}(DB)_{\mathcal{C}}$ then $N \models DB$ and $N \models \mathcal{C}$. $N \in \mathcal{MM}(DB'')$ since $\forall N' \subset N, N' \not\models DB''$. Or else N is of the form $(M \cup \{A\})^*$. If $M \models \mathcal{C}$ then a subset of N is will be in $\mathcal{MM}(DB)_{\mathcal{C}}$ and N will not survive the minimality test. As a superset of M, $N \models DB$. If $M \not\models \mathcal{C}$ then $N \cap C \neq \emptyset$. $N \models \mathcal{C}$ and therefore $N \models DB''$. N survives the minimality test only if it is minimal for DB''.

2. See [5].

Lemma 9. *Let M be an interpretation for a DDDB, DB. If $DB' = DB \vee M = \{DB \vee A | A \in M\}$ then DB' has the set of minimal models $\mathcal{MM}(DB') = min(\mathcal{MM}(DB) \cup \{M\})$.*

Proof. Immediate by noting that a model of DB' is either a model of DB or is (a superset of) M.

In the following two sections we address the issue of adding/deleting a ground positive clause C to/from a disjunctive deductive database DB. We consider several possibilities for accomplishing such an update in different classes of theories all of which lead to altering the minimal model structure of the original theory to obtain the updated database. As shown in [14, 15] it is possible to convert several other update types to this form. We will discuss possible extensions of our results to the case of nonpositive clauses in later sections.

3 Updates in Ground DDDBs

If the database, DB, is ground then it is possible to use a positive clause C, $Atoms(C) \subseteq HB_{DB}$, to split DB into components with particular properties of minimal models regarding C.

Definition 10. Let DB be a ground DDDB and let $C = A_1 \vee A_2 \vee ... \vee A_m$, where $A_i \in HB_{DB}$ be a positive clause not necessarily derivable from DB. Let $A = A_1$. Rewrite DB as $\{F_i = A \vee F_i'\} \cup \{E_j = \neg A \vee E_j'\} \cup \{C_k\}$, where F_i are the clauses containing A, E_j the clauses containing $\neg A$ and C_k the clauses with no occurrences of A. Let $DB_A = \{A\} \cup \{E_i'\} \cup \{C_k\}$ and $DB_{\neg A} = \{F_j'\} \cup \{C_k\}$ Clearly, $\mathcal{MM}(DB) = \mathcal{MM}(DB_A \vee DB_{\neg A})$ [32]. Additionally, we let $DB^{-A} = \{F_i' \vee E_j'\} \cup \{C_k\}$. Both $DB_{\neg A}$ and DB^{-A} have no occurrences of A.

All the minimal models for DB_A (if any) contain A while none of the minimal models of $DB_{\neg A}$ has A since A doesn't occur in $DB_{\neg A}$. Thus we get $DB_1 = DB_{A_1}$ and $DB_{-1} = DB_{\neg A_1}$.

In the next iteration we let $DB = DB_{\neg A}$, $A = A_2$ and repeat the expansion for the new DB and A and continue in this fashion to produce $DB_2, DB_3, ..., DB_m$ and $DB_{-2}, DB_{-3}, ..., DB_{-m}$ corresponding to the remaining atoms of C.

$DB_{-(i+1)}$ and DB_{i+1} are derived from expanding DB_{-i}, on A_{i+1}. That is, $DB_{-(i+1)} = (DB_{-i})_{\neg A_{i+1}}$ and $DB_{i+1} = (DB_{-i})_{A_{i+1}}$ for $m \perp 1 \leq i \leq 0$, where $DB_{-0} = DB_0 = DB$ (the original database itself).

DB_i has A_i as a unit clause and has no occurrences of A_j for $j < i$.

The remaining component of the database, DB_{-m}, has no occurrences of any atom of C. Both DB_i and DB_{-i} are defined for $m \leq i \leq 0$. However, for compactness of representation, by DB_{m+1} we denote the set DB_{-m}:

$DB_1 = (DB_{-0})_{A_1} = DB_{A_1}$ and $DB_{-1} = (DB_{-0})_{\neg A_1} = DB_{\neg A_1}$.
$DB_2 = (DB_{-1})_{A_2} = (DB_{\neg A_1})_{A_2}$ and $DB_{-2} = (DB_{-1})_{\neg A_2} = (DB_{\neg A_1})_{\neg A_2}$.
$DB_3 = (DB_{-2})_{A_3} = ((DB_{\neg A_1})_{\neg A_2})_{A_3}$ and $DB_{-3} = (DB_{-2})_{\neg A_3} = ((DB_{\neg A_1})_{\neg A_2})_{\neg A_3}$.
:
:
$DB_m = (DB_{-(m-1)})_{A_m} = ((...((DB_{\neg A_1})_{\neg A_2})...)_{\neg A_{m-1}})_{A_m}$.
$DB_{m+1} = DB_{-m} = (DB_{-(m-1)})_{\neg A_m} = ((...((DB_{\neg A_1})_{\neg A_2})...)_{\neg A_{m-1}})_{\neg A_m}$.

The following lemma is an extension of a result proved in [32]:

Lemma 11. *[32] Let DB be a DDDB, C be a positive clause and let DB_i, for $m+1 \leq i \leq 0$ be as given in Definition 10. Then:*

$$DB = \vee_{i=1}^{i=m+1} DB_i.$$

Lemma 12. *[32, 5] Let DB be a DDDB, C be a positive clause and let DB_i, for $m+1 \leq i \leq 0$ be as given in Definition 10. Then: A minimal model of DB_{m+1} cannot be subsumed by a minimal model of DB_i, for $m \leq i \leq 1$. $\forall M \in \mathcal{MM}(DB_{m+1}), \forall N \in \mathcal{MM}(DB_i)$ for $m \leq i \leq 1, N \not\subset M$. Subsumption in the other direction is possible.*

3.1 Clause Addition

Let DB be a DDDB and let C be a positive clause of length m. Assume that $DB \not\vdash C$ and we want to modify DB to DB' such that $DB' \vdash C$.

It is possible to accomplish this task by adding the clause C to DB and have $DB' = DB \cup \{C\}$. However, one may demand that the change to the minimal model structure of DB be minimal. That is, we would like DB' to have exactly the same set of minimal models as DB except that an atom (or more) from C is added to those models of DB with no atom of C. This leaves open the question of which atom(s) of C to add to each model. We can accomplish this task: by adding a single atom, some atoms or all atoms of C to every minimal model of DB not having an atom of C. To do that we need to determine the component of DB that needs to be modified in order for DB' to derive C. That is, we would like to isolate the part of DB that generates minimal models of DB not containing atoms of C. Earlier it was shown that the only component that makes the clause C nonderivable from DB is $DB_{m+1} = DB\_{m}$. We can restrict our modification to this component of DB.

Theorem 13. (Clause Addition) *Let DB be a DDDB and C be a positive clause such that $DB \not\vdash C$ and let DB_i, for $m+1 \geq i \geq 1$ be as defined above. Then: C derivable from $DB' = \vee_{i=1}^{i=m} DB_i \vee DB'_{m+1}$ ($DB' \vdash C$) if DB_{m+1} is changed to DB'_{m+1} so that $DB'_{m+1} \vdash C$.*

Proof. By Lemma 11, $DB = \vee_{i=1}^{i=m+1} DB_i$. Since our modification is limited to DB_{m+1} to generate DB'_{m+1} then $DB' = \vee_{i=1}^{i=m} DB_i \vee DB'_{m+1}$.

The only component that underwent modification is DB_{m+1}. The minimal models of DB' are among those of any of $\vee_i DB_i$ for $m \geq i \geq 1$ and those for DB'_{m+1}. The models of DB'_{m+1} depend on the way in which DB_{m+1} changed so that all its models have at least one element of C.

The modification of DB_{m+1} to get DB'_{m+1} can be accomplished in any of the following ways:

1. To add a single atom of C, say A, as a unit clause to DB_{m+1}. This is equivalent to adding A to every minimal model of DB falsifying C.
2. To add all the atoms of C, $A_1, A_2, ..., A_m$, as unit clauses to DB_{m+1}. This is equivalent to adding the set $\{A_1, A_2, ..., A_m\}$ to every minimal model of DB not having an atom of C.
3. An arrangement that is intermediate between the discussed options (adding a subset of the atoms of C to DB_{m+1}.
4. To add C itself as a clause to DB_{m+1}. This is equivalent to generating m new models from each model not having an atom of C by adding one atom of C.
5. If $E_1 \vee E_2 \in DB_{m+1}$ and $C = A_1 \vee A_2$ then we can let $DB'_{m+1} = DB_{m+1} \cup \{A_1 \vee A_2, E_1 \vee E_2, A_1 \vee E_2, A_2 \vee E_1\}$. This is equivalent to adding an atom of C to each minimal model of DB_{m+1} and different

models may get different atoms. The approach can be easily extended to nonbinary clauses.

The minimal models of DB_{m+1} are the only models of DB that didn't contain atoms of C. The (retained, surviving minimality test) minimal models of DB'_{m+1} are also minimal models for DB' and therefore C is derivable from DB' and the degree of change to the minimal model structure of DB is determined by the degree of change to the minimal models of DB_{m+1} when transformed into DB'_{m+1}.

Note that in all cases clauses with no occurrences of atoms from the clause under consideration, $\{C_k\}$, will be retained in the updated database. Therefore, we can limit our consideration to clauses with occurrences of atoms from C. These are the clauses that may need to undergo change [26]. This can reduce the average cost of performing updates, though not in the worst case.

To select among the available choices for adding C one may demand that C be minimally derivable in the sense that no proper subclause of C is derivable from the updated database DB'.

Consider the following example[3]:

Example 1. $DB = \{P(a) \vee P(b), P(a) \vee P(c) \vee P(e), P(b) \vee P(c) \vee P(d), P(c) \vee P(d) \vee P(e)\}$. $\mathcal{MM}(DB) = \{\{P(a), P(c)\}, \{P(a), P(d)\}, \{P(b), P(c)\}, \{P(b), P(e)\}\}$. Assume we want to add the clause $P(c) \vee P(d)$.

$DB_1 = DB_{P(c)} = \{P(a) \vee P(b), P(c)\}$, $DB_{\neg P(c)} = \{P(a) \vee P(b), P(a) \vee P(e), P(b) \vee P(d), P(d) \vee P(e)\}$, $DB_2 = DB_{\neg P(c)\, P(d)} = \{P(a) \vee P(b), P(a) \vee P(e), P(d)\}$,

$DB_3 = DB_{\neg P(c) \neg P(d)} = DB_{m+1} = \{P(b), P(e)\}$.

Selecting $P(c)$ for addition we get $DB'_3 = \{P(b), P(e), P(c)\}$ and

$DB' = \vee_{i=1}^{i=2} DB_i \vee DB'_3 = \{P(a) \vee P(b), P(c) \vee P(d), P(a) \vee P(c) \vee P(e)\}$

which has the set of minimal models

$\mathcal{MM}(DB') = \{\{P(a), P(c)\}, \{P(a), P(d)\}, \{P(b), P(c)\}, \{P(b), P(d), P(e)\}\}$

and in which the clause $P(c) \vee P(d)$ is derivable.

We could have selected $DB'_3 = \{P(b), P(e), P(d)\}$ to get

$DB' = \vee_{i=1}^{i=2} DB_i \vee DB'_3 = \{P(a) \vee P(b), P(c) \vee P(d), P(a) \vee P(c) \vee P(e)\}$

which has the set of minimal models

$\mathcal{MM}(DB') = \{\{P(a), P(c)\}, \{P(a), P(d)\}, \{P(b), P(c)\}, \{P(b), P(d), P(e)\}\}$

and in which the clause $P(c) \vee P(d)$ is also derivable.

[3] The entire minimal model structure is usually not needed for the update but is given in the examples of this paper to illustrate the degree of change that structure undergoes as a result of the update.

The next example shows that the result of updating DB_{m+1} to get DB'_{m+1} that derives C can produce a database DB' that is different from the direct addition of C to DB. It also shows that different choices for modifying DB_{m+1} to get DB'_{m+1} can give different updated theories.

Example 2. $DB = \{P(c) \vee P(b) \vee P(e), P(c) \rightarrow P(a)\}$.

$\mathcal{MM}(DB) = \{\{P(a), P(c)\}, \{P(b)\}, \{P(e)\}\}$.

Assume we want to add the clause $P(c) \vee P(d)$.

$DB_1 = DB_{P(c)} = \{P(a), P(c)\}$, $DB_{\neg P(c)} = \{P(b) \vee P(e)\}$, $DB_2 = \{P(b) \vee P(e)\}$.

$DB'_2 = \{P(b) \vee P(e), P(b) \vee P(d), P(c) \vee P(e), P(c) \vee P(d)\}$ and

$DB' = DB_1 \vee DB'_2 = \{P(a) \vee P(b) \vee P(e), P(a) \vee P(b) \vee P(d), P(c) \vee P(e), P(c) \vee P(d)\}$ which has the set of minimal models

$\mathcal{MM}(DB') = \{\{P(a), P(c)\}, \{P(b), P(c)\}, \{P(d), P(e)\}\}$ and in which the clause $P(c) \vee P(d)$ is derivable.

We could have selected $DB'_2 = \{P(b) \vee P(e), P(b) \vee P(c), P(d) \vee P(e), P(c) \vee P(d)\}$ and $DB' = DB_1 \vee DB'_2 = \{P(a) \vee P(b) \vee P(e), P(a) \vee P(d) \vee P(e), P(b) \vee P(c), P(c) \vee P(d)\}$ which has the set of minimal models $\mathcal{MM}(DB') = \{\{P(a), P(c)\}, \{P(b), P(d)\}, \{P(c), P(e)\}\}$ and in which the clause $P(c) \vee P(d)$ is derivable.

Note that if $DB'' = DB \cup \{P(c) \vee P(d)\} =$
$\{P(c) \vee P(b) \vee P(e), P(c) \rightarrow P(a), P(c) \vee P(d)\}$ then we have $\mathcal{MM}(DB'') =$
$\{\{P(a), P(c)\}, \{P(b), P(d)\}, \{P(d), P(e)\}$.

So the result of our updates can be different from direct addition of the required clause.

We usually change the models of DB_{m+1} by expanding them to have atoms of the added clause C. This growth cannot remove any of the good old models of DB. As a result the change to the model structure tends to be minimal in the sense that the clause under consideration is derivable and we retain as many minimal models of DB as possible. However, it is possible that some models of DB_i, $1 \leq i \leq m$ become minimal because they are no more suppressed by the minimal models of DB'_{m+1}.

3.2 Clause Deletion

Now given a positive ground clause $C = A_1 \vee A_2 \vee ... \vee A_m$ derivable from DB, $(DB \vdash C)$, we want to modify DB to DB' so that $DB' \nvdash C$. First we have the following lemma:

Lemma 14. *Let DB be a DDDB, A be an atom of the Herbrand base of DB and $DB^{-A} = \{F'_i \vee E'_j\} \cup \{C_k\}$, where F_i are the clauses containing*

A, E_j the clauses containing $\neg A$ and C_k the clauses with no occurrences of A and F' and E' are the result of removing the occurrences of A from F and E, respectively (as in Definition 10). Then M is a minimal model of DB iff $M \setminus \{A\}$ is a minimal model of DB^{-A}.

Proof. $\rightarrow$ Let M be a minimal model of DB. It is a model of DB^{-A} since it satisfies the clauses of the set $\{C_k\}$ of DB^{-A} as they belong to DB as well. If $A \in M$ then M satisfies E'_j as the only way for M to satisfy the E clauses of DB. If $A \notin M$ then M satisfies F'_i as the only way for M to satisfy the F clauses of DB. In either case, $F'_i \vee E'_j$ is satisfied by M. Since DB^{-A} has no occurrences of A then $M \setminus \{A\}$ is still a model for DB^{-A}. Assume that $M' = M \setminus \{A\}$ is not a minimal model for DB^{-A}. Let $M'' \subset M'$ be a minimal model of DB^{-A}. We show that $M'' \cup \{A\} \subset M$ is a model of DB. M'' satisfies the set $\{C_k\} \subseteq DB$. M'' must satisfy all of the $\{F'_i\}$ or all of the $\{E'_j\}$. This is so since otherwise there must exist two clauses F'_l and E'_s not satisfied by M'' and consequently, the clause $F'_l \vee E'_s \in DB^{-A}$ will be *false* in M'' contradicting that M'' is a model for DB^{-A}.

Now, assume M'' satisfies the set $\{F'_i\}$. M'' contains no A and therefore it also satisfies the set $\{E_j\}$. M'' is a model for DB and so is its superset $M'' \cup \{A\}$. Or else assume M'' satisfies the set $\{E'_i\}$. $M'' \cup \{A\}$ is a model of DB since the A atom satisfies all the F clauses and the E clauses are satisfied by the satisfaction of their primed components (E').
That is $M'' \cup \{A\} \subset M$ is a model of DB contradicting the assertion that $M \in \mathcal{MM}(DB)$.

$\leftarrow$ Let M' be a minimal model of DB^{-A}. Clearly $A \notin M'$. Two cases are possible:

M' is a minimal model for DB by satisfying the $\{C_k\}$ clauses and the F' clauses and the E clauses by the absence of A from M'. M corresponding to M' is equal to M' with no occurrences of A.
Or M' is not a model for DB but a model for E' clauses. $M = M' \cup \{A\}$ is a minimal model for DB. In both cases M corresponding to M' is a minimal model for DB.

Theorem 15. (Clause Deletion) *Let DB be a DDDB and $C = A_1 \vee ... \vee A_m$ be a positive clause such that $DB \vdash C$ (minimally). Let A be an atom occurring in C. Let M be a minimal model of DB containing A and none of the other atoms of C ($M \in \mathcal{MM}(DB)$ and $A \in M$ and $\forall A_i \neq A, A_i \notin M$)[4]. Let $M' = M \setminus \{A\}$ and let DB^{-A} and $DB_{\neg A}$ be as defined earlier. Let $AUG_A^M(DB) = \{E'_j\} \cup \{C_k\} \cup \{A \vee B | B \in M'\}$ and let*

[4] Such a model exists by Lemma 5.

$DB' = DB_{\neg A} \vee AUG_A^M(DB)$. Then $DB' \not\vdash C$ and DB' has as its set of minimal models the minimal elements of the set $\mathcal{MM}(DB) \cup \{M'\}$. $\mathcal{MM}(DB') = min(\mathcal{MM}(DB) \cup \{M'\})$.

Proof. $DB' = DB_{\neg A} \vee AUG_A^M(DB) = DB^{-A} \cup (\{F'\} \vee \{A \vee B | B \in M'\})$. In view of Lemma 14, minimal models of DB not having A are also minimal models of DB^{-A} and satisfy $(\{F'\} \vee \{A \vee B | B \in M'\})$ by making $\{F'\}$ *true*. The remaining models of DB are of the form $\{\{A\} \cup M | M \in \mathcal{MM}(DB^{-A})\}$. Clearly all models of DB^{-A} with A added satisfy DB' due to the presence of A in every clause of $(\{F'\} \vee \{A \vee B | B \in M'\})$. On the other hand M' is a model of DB^{-A} by Lemma 14 and satisfies the remaining clauses of DB' by construction and has no A neither any of the other atoms of C. It is minimal because it is a minimal model of DB^{-A}.

This approach is biased towards a particular atom A and requires that M contain an occurrence of A and none of the other atoms in the clause to be deleted[5].

Example 3.
- $DB = \{P(a) \vee P(b), P(a) \rightarrow P(c) \vee P(d), P(b) \rightarrow P(c) \vee P(d)\}$. $\mathcal{MM}(DB) = \{\{P(a), P(c)\}, \{P(a), P(d)\}, \{P(b), P(c)\}, \{P(b), P(d)\}\}$. We want to delete $C = P(c) \vee P(d)$. Take $P(c)$ as the candidate atom. We can take $\{P(a), P(c)\}$ as the candidate model for removal (update).
  $\{F_i'\} = \{P(a) \rightarrow P(d), P(b) \rightarrow P(d)\}, \{E_j'\} = \emptyset$, $\{C_k\} = \{P(a) \vee P(b)\}$.
  $DB_{\neg P(c)} = \{P(a) \vee P(b), P(a) \rightarrow P(d), P(b) \rightarrow P(d)\}$.
  $AUG_{P(c)}^{\{P(a),P(c)\}}(DB) = \{P(a) \vee P(c)\} \cup \{C_k\}$.
  $DB' = \{P(a) \vee P(b), P(b) \rightarrow P(c) \vee P(d) \vee P(a)\}$ with the minimal models $\{\{P(a)\}, \{P(b), P(c)\}, \{P(b), P(d)\}\}$. Or alternatively, if we take $\{P(b), P(c)\}$ as the candidate model:
  $DB' = \{P(a) \vee P(b), P(a) \rightarrow P(c) \vee P(d) \vee P(b)\}$ with the minimal models $\{\{P(b)\}, \{P(a), P(c)\}, \{P(a), P(d)\}\}$.
- Now let's consider the equivalent (in terms of minimal models) database
  $DB = \{P(a) \vee P(b), P(c) \vee P(d)\}$.
  $\mathcal{MM}(DB) = \{\{P(a), P(c)\}, \{P(a), P(d)\}, \{P(b), P(c)\}, \{P(b), P(d)\}\}$.
  We want to delete $C = P(c) \vee P(d)$. We also take $P(c)$ as the candidate atom. We take $\{P(a), P(c)\}$ as the candidate model.
  $\{F_i'\} = \{P(d)\}, \{E_j'\} = \emptyset$, $\{C_k\} = \{P(a) \vee P(b)\}$.
  $DB_{\neg P(c)} = \{P(a) \vee P(b), P(d)\}$.

[5] One can modify the procedure to make it less sensitive to the atom and the model at hand.

$AUG_{P(c)}^{\{P(a),P(c)\}}(DB) = \{P(a) \vee P(c)\} \cup \{C_k\}$.
$DB' = \{P(a) \vee P(b), P(c) \vee P(d) \vee P(a)\}$ with the minimal models $\{\{P(a)\}, \{P(b), P(c)\}, \{P(b), P(d)\}\}$. Or alternatively, if we take $\{P(b), P(c)\}$ as the candidate model:
$DB' = \{P(a) \vee P(b), P(c) \vee P(d) \vee P(b)\}$ with the minimal models $\{\{P(b)\}, \{P(a), P(c)\}, \{P(a), P(d)\}\}$.

In each case, both databases, and any others resulting from the different choices of the candidate atom, can serve as a solution to the problem at hand. The change to the model structure is minimal in the sense that the clause under consideration is not derivable anymore, and we retained as many minimal models as possible from the original minimal model structure. Note that the change in the structure of the candidate for removal model rendered other models nonminimal and forced their removal.

So, the candidate for removal model shrinks by removing atoms of C from it. This may render other models nonminimal and force their removal. The degree of update minimality, or model retention, will generally depend on the choice of the candidate model. One may insist on selecting the best such choice (one that keeps the most minimal models of DB) but that may be computationally expensive. Removing the least number of atoms from that model may serve as a good heuristic to achieve update minimality.

4 Updates in Range Restricted DDDBs

The approach to update discussed so far is applicable to ground disjunctive databases. Groundness was needed to isolate the components that have to be modified to accomplish the update. One may argue that it can be applied to nonground theories if we choose to ground them. The size of the resulting theory may be prohibitive. In this section we present algorithms for update in range restricted theories: a larger class than ground theories but still a natural one that is encountered frequently [5].

4.1 Update Through Model Suppression/Augmentation

Given a positive clause C and a DDDB, DB such that $DB \nvdash C$ and in view of Lemma 8 it is possible to add negative clauses to DB so as to suppress the minimal models that falsify C. On the other hand, if $DB \vdash C$ and in view of Lemma 9 it is possible expand the model structure of DB so as to add a minimal model that falsifies C. We state the following results the proofs of which are immediate.

Theorem 16. (Clause Addition via Model Suppression) *Let DB be a DDDB and C be a positive ground clause such that $DB \nvdash C$. Let $DB' = DB \cup \{M \rightarrow \bot | M \in \mathcal{MM}(DB)_{\neg C}\}$. $\mathcal{MM}(DB') = \mathcal{MM}(DB)_C$*[6]*. $DB' \vdash C$*[7].

Proof. Immediate from Lemma 8

As for clause deletion, this approach is not applicable since all minimal models of DB satisfy C and removing them all will result in an inconsistent theory. This asymmetry is the result of having clause addition defined in terms of *all* minimal models and clause deletion in terms of *at least one* minimal model.

Theorem 17. (Clause Deletion via Model Expansion) *Let DB be a DDDB and C be a positive ground clause such that $DB \vdash C$. Let $DB' = DB \vee M'$, where $M' = M \setminus Atoms(C)$ and $M \in \mathcal{MM}(DB)$. $DB \nvdash C$ and $\mathcal{MM}(DB') = min(\mathcal{MM}(DB) \cup \{M'\})$.*

Proof. Immediate from Lemma 9.

Example 4. Let DB be the set of clauses:

$DB = \{$
$C_1 = \top \rightarrow P(a)$, $C_2 = \top \rightarrow Q(b)$
$C_3 = P(x) \rightarrow Q(x) \vee R(x)$, $C_4 = P(x) \wedge R(x) \rightarrow S(x)$,
$C_5 = Q(x) \rightarrow P(x) \vee R(x)$, $C_6 = S(a) \wedge R(b) \rightarrow \bot\}$.

$\mathcal{MM}(DB) = \{\{P(a), Q(b), P(b), Q(a)\}, \{P(a), Q(b), R(b), Q(a)\}, \{P(a), Q(b), P(b), R(a), S(a)\}\}$.

Clearly, $DB \nvdash C_7 = R(a) \vee S(b)$ and $DB \vdash C_8 = R(b) \vee P(b)$.

– Addition:
Add $C_7 = R(a) \vee S(b)$.
$\mathcal{MM}(DB)_{\neg C} = \{\{P(a), Q(b), P(b), Q(a)\}, \{P(a), Q(b), R(b), Q(a)\}\}$.
$DB' = DB \cup \{ P(a) \wedge Q(b) \wedge R(b) \wedge Q(a) \rightarrow \bot$,
$P(a) \wedge Q(b) \wedge R(b) \wedge Q(a) \rightarrow \bot\}$ or equivalently (for the given theory):
$DB' = DB \cup \{ Q(a) \wedge R(b) \rightarrow \bot$, $Q(a) \wedge P(b) \rightarrow \bot\}$.
$\mathcal{MM}(DB') = \{\{P(a), Q(b), P(b), R(a), S(a)\}\}$.

[6] A similar approach can be used to make C *false* in every minimal model of DB'': $DB'' = DB \cup \{M \rightarrow \bot | M \in \mathcal{MM}(DB)_C\}$. $\mathcal{MM}(DB'') = \mathcal{MM}(DB)_{\neg C}$. This may be of interest when deleting a clause is interpreted as being *false* in all minimal models of the theory. However, we don't adopt this interpretation here.

[7] Note that if C is *true* in no minimal model of DB then DB' is inconsistent.

– Deletion:
Delete $C_8 = R(b) \vee P(b)$.
$M = \{P(a), Q(b), P(b), Q(a)\}$.
$M' = M \setminus \{P(b), R(b)\} = \{P(a), Q(b), Q(a)\}$.
$M' = \{P(a) \wedge Q(b) \wedge Q(a)\}$.

$DB' = \{$
$C_1 = \top \rightarrow P(a)$, $C_2 = \top \rightarrow Q(b)$
$C_3' = P(x) \rightarrow Q(x) \vee R(x) \vee M'$, $C_4' = P(x) \wedge R(x) \rightarrow S(x) \vee M'$,
$C_5' = Q(x) \rightarrow P(x) \vee R(x) \vee M'$, $C_6 = S(a) \wedge R(b) \rightarrow \bot\}$.

$\mathcal{MM}(DB') = \{\{P(a), Q(b), Q(a)\}, \{P(a), Q(b), P(b), R(a), S(a)\}\}$.
The second element of $\mathcal{MM}(DB)$ was abandoned in $\mathcal{MM}(DB')$ for becoming nonminimal.
Note that rather than augmenting each clause with every atom of M' we augmented clauses by the conjunction of atoms in M' and simplified the formulas with heads intersecting with M'.

Note also that removing minimal models for clause addition retains all the good models of the original theory ($\mathcal{MM}(DB)_C$). However, adding the shrunk model to accomplish clause deletion may remove more than one minimal model of the original theory for becoming nonminimal.

4.2 Update Through Clause Expansion/Addition

Theorem 18. (Clause Addition) *Let DB be a DDDB, C be a positive clause such that $DB \nvdash C$. Define $DB' = DB \cup \{M \rightarrow C | M \in \mathcal{MM}(DB)_{\neg C}\}$. DB' accomplishes the addition update of C. Additionally, $\mathcal{MM}(DB)_C \subseteq \mathcal{MM}(DB')$.*

Proof. Let $N \in \mathcal{MM}(DB')$. Either N is in $\mathcal{MM}(DB)$ or $\exists M \subset N$ such that $M \in \mathcal{MM}(DB)$ and $(M \rightarrow C) \in DB'$; $(N \setminus M) \cap Atoms(C) \neq \phi$. C is *true* in N. The rest is a direct application of Lemma 8.

There may exist several ways to add $M \rightarrow C$ and the discussion of the point for Theorem 13 is applicable here as well.

Theorem 19. (Clause Deletion) *Let DB be a DDDB, C be a positive clause such that $DB \vdash C$ and let $M \in \mathcal{MM}(DB)$ and $N = M \setminus Atoms(C)$. Let $\mathcal{C} = \{E \in DB$ such that $N \not\models E\}$. Define $DB' = (DB \setminus \mathcal{C}) \cup \{E' | Head(E) \cap Head(E') = Head(E)$ and $Body(E) = Body(E')$ and $E \in \mathcal{C}$ and $E' \cap N \neq \phi\}$. DB' accomplishes the deletion update of C and $\mathcal{MM}(DB') = min(\mathcal{MM}(DB) \cup \{N\})$.*

Proof. N has no atoms of C by construction. N satisfies every clause in DB'. Assume that is not the case. There must exist a clause $D \in DB'$ such that $N \not\models D$. If $D \in DB$ then it would have been falsified by N and $Head(D)$ must consist entirely of atoms not in N. As a clause in DB that is falsified by N, D must have been changed to include an element of N by construction to produce D' and $D \notin DB'$. $N \models D'$. A contradiction.

In view of Lemma 5 we may select the model that contains a single atom, say A, of C. In this case only clauses with A in the head will be falsified by N. However, the deletion process may still be nondeterministic due to the existence of more than one such model. The choices can be exploited to achieve better update minimality, may be at the expense of more expensive computations.

Note that rather than adding N to every clause of DB we restrict our modification to elements of $\mathcal{C}$.

Example 5. Let DB be the set of clauses of Example 4, i.e.:

$DB = \{$
$C_1 = \top \rightarrow P(a)$, $C_2 = \top \rightarrow Q(b)$,
$C_3 = P(x) \rightarrow Q(x) \vee R(x)$, $C_4 = P(x) \wedge R(x) \rightarrow S(x)$,
$C_5 = Q(x) \rightarrow P(x) \vee R(x)$, $C_6 = S(a) \wedge R(b) \rightarrow \bot\}$.

$\mathcal{MM}(DB) = \{\{P(a), Q(b), P(b), Q(a)\}, \{P(a), Q(b), R(b), Q(a)\}, \{P(a), Q(b), P(b), R(a), S(a)\}\}$.

- Addition: Add $C_7 = R(a) \vee S(b)$.
  $\mathcal{MM}(DB)_{\neg C} = \{\{P(a), Q(b), P(b), Q(a)\}, \{P(a), Q(b), R(b), Q(a)\}\}$.
  $DB' = DB \cup \{ P(a) \wedge Q(b) \wedge P(b) \wedge Q(a) \rightarrow R(a) \vee S(b), P(a) \wedge Q(b) \wedge R(b) \wedge Q(a) \rightarrow R(a) \vee S(b)\}$.
  $\mathcal{MM}(DB') = \{\{P(a), Q(b), P(b), R(a), S(a)\}, \{P(a), Q(b), R(b), Q(a), S(b)\}, \{P(a), Q(b), P(b), Q(a), S(b)\}\}$.
  Note that when dealing with a minimal model M it is frequently possible to restrict our consideration (in the bodies of added clauses) to those atoms of M that are not common to all other minimal models of the theory and still achieve the required results, possibly with more efficiency[8].
- Deletion: Delete $C_8 = R(b) \vee P(b)$.
  $M = \{P(a), Q(b), P(b), Q(a)\}$. $N = \{P(a), Q(b), Q(a)\}$.
  $\mathcal{C} = \{Q(b) \rightarrow P(b) \vee R(b)\}$.

[8] This was also the case when adding denials in Example 4.

$$
\begin{array}{l}
DB' = \{ \\
\quad C_1 = \top \rightarrow P(a), \qquad\qquad\qquad C_2 = \top \rightarrow Q(b) \\
\quad C_3 = P(x) \rightarrow Q(x) \vee R(x) \vee P(a), C_4 = P(x) \wedge R(x) \rightarrow S(x) \\
\quad C_5 = Q(a) \rightarrow P(a) \vee R(a), \qquad C_6 = S(a) \wedge R(b) \rightarrow \bot, \\
\quad C_9' = Q(b) \rightarrow P(b) \vee R(b) \vee N\}.
\end{array}
$$

$$
\mathcal{MM}(DB') = \{\{P(a), Q(b), Q(a)\}, \{P(a), Q(b), P(b), R(a), S(a)\}\}.
$$

5 Nonpositive Clause Updates

In this section we address the issue of adding/deleting a nonpositive clause to a DDDB. As is the case for positive clauses, a clause in implication form $C = Head(C) \leftarrow Body(C)$ is *true* in DB if and only if all minimal models of DB satisfy C. That is, C is not *true* in DB if and only if there exists a minimal model of DB falsifying C: $\exists M \in \mathcal{MM}(DB)$ such that $M \models Body(C)$ and $M \not\models Head(C)$. The previous results can be viewed as special cases of the general clause updates [29].

5.1 Clause Addition Update

Let DB be a DDDB and C be a clause such that $\mathcal{MM}(DB)_{\neg C} = \{M \in \mathcal{MM}(DB)\ M \models Body(C)$ and $M \not\models Head(C)\}$ is not empty. Let $DB' = DB \cup \{M \rightarrow Head(C) | M \in \mathcal{MM}(DB)_{\neg C}\}$. DB' accomplishes the addition of C to DB.

The result can be viewed as an application of Theorem 18. Clearly there may exist many ways to have $M \rightarrow Head(C)$ as was discussed following Theorem 13. The update can also be accomplished by suppressing the models of the theory not satisfying C (those in $\mathcal{MM}(DB)_{\neg C}$) in the spirit of Theorem 16.

5.2 Clause Deletion Update

To delete C from DB we have to create a minimal model of the updated theory that doesn't satisfy C. To minimize the change we always select a model that satisfies $Body(C)$ and make sure that the modification of that model retains the satisfiability of $Body(C)$ but falsifies $Head(C)$. Once that model is selected and since $Head(C)$ is a positive clause then we can use the results of earlier sections to accomplish the deletion of $Head(C)$ (Theorem 19) by making it *false* in the modified model.

It may be the case that such a model doesn't exist. That is, all minimal models of DB satisfy C by falsifying its body. Then we have to create it.

We need to create a model that satisfies the body of C but not its head. For update minimality reasons we may select this model to be as close as possible to a minimal model of DB

Example 6. Let DB be the set of clauses of Example 4, i.e.:

$DB = \{$
$C_1 = \top \rightarrow P(a)$, $C_2 = \top \rightarrow Q(b)$,
$C_3 = P(x) \rightarrow Q(x) \vee R(x)$, $C_4 = P(x) \wedge R(x) \rightarrow S(x)$,
$C_5 = Q(x) \rightarrow P(x) \vee R(x)$, $C_6 = S(a) \wedge R(b) \rightarrow \bot\}$.

$\mathcal{MM}(DB) = \{\{P(a), Q(b), P(b), Q(a)\}, \{P(a), Q(b), R(b), Q(a)\}, \{P(a), Q(b), P(b), R(a), S(a)\}\}$.

- Addition: Add $C_{11} = P(a) \rightarrow Q(a)$.
  $\mathcal{MM}(DB)_{\neg C_{11}} = \{\{P(a), Q(b), P(b), R(a), S(a)\}\}$.
  $DB' = DB \cup \{ P(a) \wedge Q(b) \wedge P(b) \wedge R(a) \wedge S(a) \rightarrow Q(a)\}$.
  $\mathcal{MM}(DB') = \{\{P(a), Q(b), P(b), Q(a)\}, \{P(a), Q(b), R(b), Q(a)\}\}$.
- Deletion: Delete $C_{12} = Q(b) \rightarrow P(b) \vee R(b)$.
  Select $M = \{P(a), Q(b), P(b), Q(a)\}$. $M \cap Body(C_{12}) = Body(C_{12})$,
  $M \cap Head(C_{12}) \neq \emptyset$.
  $N = M \setminus Head(C_{12}) = M \setminus \{P(b), R(b)\} = \{P(a), Q(b), Q(a)\}$.
  $\mathcal{C} = \{Q(b) \rightarrow P(b) \vee R(b)\}$.

  $DB' = \{$
  $C_1 = \top \rightarrow P(a)$, $C_2 = \top \rightarrow Q(b)$
  $C_3 = P(x) \rightarrow Q(x) \vee R(x) \vee P(a)$, $C_4 = P(x) \wedge R(x) \rightarrow S(x)$
  $C_5 = Q(a) \rightarrow P(a) \vee R(a)$, $C_6 = S(a) \wedge R(b) \rightarrow \bot$,
  $C'_9 = Q(b) \rightarrow P(b) \vee R(b) \vee N\}$.

  $\mathcal{MM}(DB') = \{\{P(a), Q(b), Q(a)\}, \{P(a), Q(b), P(b), R(a), S(a)\}\}$.
  Note that the clause $S(b) \rightarrow R(b)$ is true in all minimal models but no minimal model has $S(b)$.

5.3 On the Issue of Update Minimality

The concept of update minimality was described as the least change to the minimal model structure but was not formally defined. We think of it as consisting of two components:

First we would like to preserve as many minimal models as as possible of those of the old: certainly all the models that satisfy the update criteria (minimal models in which the added clause is *true* and minimal models

in which the deleted clause is *false*). We would like also to have as few as possible minimal models change. In a sense we would like to have $(\mathcal{MM}(DB') \cap \mathcal{MM}(DB))$ with maximal cardinality and $(\mathcal{MM}(DB') \setminus \mathcal{MM}(DB))$ with minimal cardinality.

This criterion alone is usually not sufficient to guarantee the uniqueness of the resulting updated theory since it deals with the number of changed models and doesn't touch on the degree of change models undergo. For that we use the second criterion and borrow from interpretation update concepts: we require that the modified models undergo the least change necessary. That is, if M is a model in $\mathcal{MM}(DB)$ modified to M' in $\mathcal{MM}(DB')$ then we require $(M \setminus M') \cup (M' \setminus M)$ to have minimum cardinality[19].

Generally, our updates tend to meet the first minimality criterion since we try to limit changes to the models that do not satisfy the update requirements (in deriving the update clause). For clause addition updates we always retain the models that do not need to undergo changes since the changed models either grow or get deleted and both operations have no effect on the "good old models". However, for deletion updates the modified models may shrink, a fact that may render some of the old minimal models nonminimal. Different ways of modifying the bad models can give different results under the first minimality criterion. The selection of the one with maximum "good model" retention, will usually be at the expense of more expensive computations[9].

The second criterion calls for adding/deleting the least number of atoms to/from the updated model. That is not difficult to accomplish although it may require more computation to select the model that will require the least change as opposed to selecting an arbitrary model. We may use the approach of [5, 32] to focus our search for such a model. However, this criterion may conflict with the first one in the sense that selecting the model needing the least change may suppress some of the good old models that we strive to retain. The different algorithms presented in the paper tend to fair reasonably good by both minimality criteria, while we generally select the modification that is not very expensive to perform[10].

6 Conclusion and Remarks

We presented an approach to the update problem in DDDBs based on modifying the theory to achieve the required change to the minimal model

[9] See Theorem 15 and the note following Theorem 19.

[10] If more than one update option is available one may use additional criteria such as minimal syntactic change to the theory to make the final choice.

structure. We considered adding/deleting ground positive clauses to/from ground DDDBs then range-restricted DDDBs and finally extended the results to general clause updates.

Update minimality was defined in terms of the effect of the update on the minimal model structure of the theory. This sounds natural if one interprets minimal models as the possible states in which the theory can be (had our information about the world been complete). In such a case it is natural to try to minimize the effect of an update on both the number of models as well as its effect on the content of individual models. These are the criteria we tried to use to select among the possible updates. Our motivation for relaxing these requirements was to achieve more efficient update algorithms.

The issue of update minimality received much attention in the literature [2, 16, 8, 10, 11, 13, 14]. Other approaches to define update minimality were advanced. Examples are to define minimality in terms of the weakness of the update [13] or in terms of the number of clauses modified to accomplish the update [14]. Some authors choose to impose no minimality criteria on updates [6]. Clearly, least change to the minimal model structure discussed here need not be minimal by other measures of update minimality.

Our approach depends heavily on computing one or more minimal models of the theory with particular properties with respect to the clause to be added/deleted. An algorithm to perform this computation was given in [5] and proven to be minimal model sound and complete for the class of range-restricted theories: it returns all and only minimal models of its input theory. It uses a clause C to focus its search for minimal models with specific characteristics regarding C. The algorithm depends heavily on utilizing denial rules to guide the model expansion process [32]. The detailed discussion, an implementation and comments on efficiency results can also be found in [5]. Another minimal model generating procedure for ground DDDBs is given in [23].

The goal of an update is specified in terms of the minimal model structure (and consequently in terms of the contents of individual models). However, we don't achieve that by direct manipulation of individual models but by modifying the theory specifying the database. This is in line with the approach of [19, 24] and avoids the shortcomings of direct model updates [17], discussed in [19, 29]. In this sense our approach is closer to program update [19] or rule update [24] or formula update [28] than it is to direct model (interpretation) update [17], and is applied to the case of DDDBs.

Our approach differs from those discussed in [2, 25, 7] in that we

consider a more general class of theories and updates. We treat DDDBs and consider adding/deleting general clauses to such theories. In [12, 9] DDDBs are considered and the approach can be extended to the case of range-restricted theories. However, they treat only positive clause updates and our approach tends to expand much less of the model structure through exploiting the ordering on model generation induced by the update clause [32, 5]. We don't assume a particular representation of the model structure [12, 16] nor do we require it all to be present for an update. The models are generated "ad-hoc" and in the order most appropriate for the update. Combined with the algorithms given in [32, 5] for minimal model generation our approach can result in substantial savings in constructing the needed portion of the minimal model structure as opposed to generating a complete set of models then minimizing among them. Additionally, we can have the update clause guide the system to generate the most relevant models [29]. In a sense we can integrate the updating process into the model generation process.

We offered several methods for accomplishing updates as opposed to the one offered in [12] which can be viewed as a special case of our methods. Our approach, being more general, can still be combined with algorithms reported in the literature for performing more complex tasks such as controlling database dynamics [27].

In [13] the view update problem in stratified disjunctive databases was addressed and extended to normal disjunctive databases in [9]. This reduces to positive clause addition/deletion updates when DDDBs are considered. Our results can be employed in that approach as well. The work handles nonpositive additions as well but we offer more choices and we advise employing algorithms that enable directed search for the required models. The approach of [14] deals with normal databases and it is of interest to extend our results to normal disjunctive databases.

[16] treats the issue of positive clause updates, both addition and deletion, in a DDDB through the construction of so called *deduction trees*, a representation for top-down processing of the theory to generate the clauses needed to accomplish the required update. The relationship between deduction trees expansion and minimal models was established in [31]. A similar approach is used in [2] for view deletion updates in definite deductive databases.

We don't restrict ourselves to modifying the extensional part of the theory, as is the case in [2, 6, 14, 12, 13, 28], to accomplish an update. Rather, we do that by adding arbitrary clauses to the DDDB including modifying its IDB clauses. Such a modification is also employed in [26, 20, 3]. We believe that it is possible to modify our approach so that it

keeps track of the clauses added for the sole purpose of accomplishing an update as opposed to the *original* clauses of the theory. This may help in making the updates reversible in the sense that a sequence of adding and deleting the same clause may take us back to the original theory which is not usually the case for our algorithms as presented here or for most other algorithms available.

Certain requirements regarding update minimality may produce databases that are outside the class of theories to which the update is applied. An example is that the updated theory can become indefinite to account for the various ways of accomplishing an update to a definite theory [8, 13]. Certain approaches were advanced to avoid this [9]. This drawback doesn't apply to our case since the theories we treat can be indefinite to start with[11].

The updates as presented here were interpreted as additions/deletions to the theory (EDB, IDB). One can also talk about adding/deleting integrity constraints[12]. While the interpretation of the integrity constraint addition is obvious: retain all the minimal models that satisfy the constraint and remove the others, the deletion is not. To delete a constraint it may be sufficient to just stop enforcing it and not necessarily making it false in some minimal model. So a deleted integrity constraints may be true in all minimal models still.

Our restriction to the case of range-restricted theories is dictated by the class of theories that can be handled by the minimal model generation procedure rather than a limitation of the adopted approach.

Efficiency improvement methods can be employed to improve the performance of the algorithms presented here. Examples are the incremental generation of models and the inclusion of only the relevant portions of the constraints added to the theory during the model generation process [5, 32].

Topics for future work include extending the approach reported here to larger classes of theories such as theories with negation in rule bodies and for theories with more than one type of negation as well as treating nonground clause updates efficiently. Our updates are not reversible. Adding then deleting the same clause generally doesn't result in the original theory. It is of interest to study the conditions needed to ensure update reversibility. Another possibility is to investigate the feasibility of accomplishing minimal updates under our criteria augmented by others

[11] Updates can be one of the sources of indefiniteness in databases.

[12] Under the *weak* interpretation of integrity constraints: a set of constraints is satisfied if there is a model of the theory satisfying all of the constraints or if the theory together with the constraints is consistent [18].

such as minimizing the syntactic change and restricting modification to certain components, say the EDB, of the theory, and to compare them with the methods discussed here.

Acknowledgments:

The author thanks Dietmar Seipel for his helpful comments on earlier drafts of this paper and the two anonymous referees of [30] for their valuable reports.

References

1. S. Abiteboul; Updates, a new frontier; *Proceedings of the Second International Conference on Database Theory*; Springer-Verlag LNCS series Vol. number 326. PP. 1-18, 1988.
2. C. Aravindan and P. Baumgartner; A rational and efficient algorithm for view deletion in databases; *Proceedings of the sixth International Logic Programming Symposium ILPS97*; 1997.
3. K. Benkerimi and J. C. Shepherdson. Partial deduction of updatable definite logic programs. *Journal of Logic Programming*, 18:1–26, 1994.
4. F. Bry. Intensional Updates: Abduction via deduction. In D. Warren and P. Szeredi, editors, *Proceedings of the International Conference on Logic Programming*, PP. 561–575, MIT Press, 1990.
5. F. Bry and A. Yahya. Minimal model generation with positive unit hyper-resolution tableaux. *Proceedings of the Fifth Workshop on Theorem Proving with Analytic Tableaux and Related Methods, (P. Miglioli, U. Moscato, D. Mundici, and M. Ornaghi, editors.)*, PP. 143–159, Palermo, Italy, May 1996. Springer-Verlag LNAI series. Vol. 1071, Full version: http://www.pms.informatik.uni-muenchen.de/publikationen/.
6. H. Decker. An Extension of SLD by abduction and integrity maintenance for view updating in deductive databases. *Proceedings of JICSLP96 (M. Maher, editor)*, PP. 157–161, 1996.
7. R. Fagin, G. Kuper, J. Ullman, and M. Vardi. Updating logical databases. In *Advances in Computing Research*, PP. 1–18, 1986.
8. R. Fagin, J. Ullman and M. Vardi; On the semantics of updates in databases; *Proceedings of the second ACM symposium on Principles of database systems* PP. 352–365, 1983.
9. J.A. Fernandez, J.Grant, and J. Minker. Model theoretic approach to view updates in deductive databases. Technical Report CS-TR-3335, Department of Computer Science and UMIACS, University of Maryland, College Park, July 1994.
10. P. Gärdenfors; Belief revision: an introduction. *Belief Revision (P. Gaŕdenfors, editor)*, Cambridge University Press, PP. 1–20, July 1995.
11. P. Gärdenfors and H. Rott; Belief Revision. *Handbook of Logic in Artificial Intelligence and Logic Programming Vol. 4 (Gabbay, Hogger and Robinson eds.)*, Oxford University Press, PP. 35–132, 1995.

12. J. Grant, J. Gryz, and J. Minker. Updating disjunctive databases via model trees. Technical Report CS-TR-3407 UMIACS-TR-95-11, Department of Computer Science and UMIACS, University of Maryland, College Park, July 1995.
13. J.Grant, J.Horty, J. Lobo, and J. Minker. View updates in stratified deductive databases. *Journal of Automated Reasoning*, 11:249–267, 1993.
14. A. Guessoum and Lloyd J. Updating knowledge bases. *New Generation Computing Journal*, 8:71–89, 1990.
15. A. Guessoum and Lloyd J. Updating knowledge bases ii. *New Generation Computing Journal*, 10:73–100, 1991.
16. C. Johnson; Deduction trees and the view update problem in indefinite deductive databases; *University of Keele-Department of Computer Science Technical Report Number tr94.08*; University of Keele, April, 1994.
17. H. Katsuno and A. Mendelzon. On the difference between updating a knowledge base and revising it. *Proccedings of the international conference of the Principles of Knowledge Representation and Reasoning (J. Allen, R. Fikes and E. Sandewall, eds.)*, PP. 230–237, 1991.
18. R. Kowalski and F. Sadri. A theorem proving approach to database integrity. In J. Minker, editor, *Foundations of Deductive Databases and Logic Programming*. Morgan Kaufmann, 1988.
19. J. A. Leite and L. M. Pereira; Generalizing updates: from models to programs; *Proceedings of the Fourth Workshop on Logic Programming and Knowledge Representation LPKR'97*, 1997.
20. N. Leone, L. Palopoli and M. Romeo; Updating logic programs; *Proceedings of ISMIS93*, PP. 235–244, 1997.
21. J. Lobo, J. Minker, and A. Rajasekar. *Foundations of Disjunctive Logic Programming*. MIT Press, 1992.
22. J. Minker. On indefinite databases and the closed world assumption. In *Lecture Notes in Computer Science 138*, PP. 292–308. Springer-Verlag, 1982.
23. I. Niemelä. A tableau calculus for minimal model reasoning. In P. Miglioli, U. Moscato, D. Mundici, and M. Ornaghi, editors, *Proceedings of the Fifth Workshop on Theorem Proving with Analytic Tableaux and Related Methods*, pages 278–294, Palermo, Italy, May 1996. Springer-Verlag LNAI series. Vol. 1071.
24. T. Przymusinski, and H. Turner; Update by means of inference rules; *Journal of Logic Programming*; Vol. 30 No. 2. PP. 125–143, 1997.
25. R. Reiter. On specifying database updates. *J. Logic Programming*, 25(6):53–91, June 1995.
26. F. Rossi and S. Naqvi. Contributions to the view update problem. *Proceedings of the International Conference on Logic Programming, Lisbon*, PP. 398–415, 1989.
27. C. Wichert and , B. Freitag; Capturing database dynamics by diferred updates; *proceedings of the Fourteenth ILCP (Naish, Ed.)*, PP. 226–240, 1997.
28. M. Winslett; Reasoning about action using a possible models approach; *Proceedings of the AAAI-88*; PP. 89–93, 1988.
29. A. Yahya. Generalized query answering in disjunctive databases: Procedural and nonmonotonic aspects. *Proceedings of the Fourth International Conference on Logic Programming and Nonmonotonic Reasoning LPNMR'97, Springer-Verlag LNAI Series*, Volume Number 1265, 324-340, 1997.
30. A. Yahya. Updates in disjunctive deductive databases: A minimal model based approach, (An Extended Abstract); *Proceedings of the Fourth Workshop on Logic Programming and Knowledge Representation LPKR'97*, 1997.

31. A. Yahya; A Goal-driven approach to efficient query answering in disjunctive databases. *Technical report number PMS–FB-1996-12*; Institut für Informatik, University of Munich, Department of computer science, 1996.
32. A. Yahya, J.A. Fernandez, and J. Minker. Ordered model trees: A normal form for disjunctive deductive databases. *J. Automated Reasoning*, 13(1):117–144, 1994.
33. A. Yahya and L.J. Henschen. Deduction in non-Horn databases. *J. Automated Reasoning*, 1(2):141–160, 1985.

This article was processed using the LaTeX macro package with LLNCS style