Fundamenta Informaticae 1(1996)1-6
IOS Press

Reducing Disjunctive to Non-Disjunctive Semantics by Shift-Operations*

Jürgen Dix
Department of Computer Science
University of Koblenz-Landau
Rheinau 1, 56075 Koblenz, Germany
dix@mailhost.informatik.uni-koblenz.de

Georg Gottlob
Institut für Informationssysteme
Technical University of Vienna
Paniglgasse 14, 1040 Wien, Austria
gottlob@vexpert.dbai.tuwien.ac.at

Wiktor Marek
Department of Computer Scienc
University of Kentucky
Lexington, KY 40506, USA
marek@cs.engr.uky.edu

Abstract. It is wellknown that Minker's semantics GCWA for positive disjunctive programs P is Π_2^P-complete , i.e. to decide if a literal is true in all minimal models of P. This is in contrast to the same entailment problem for semantics of non-disjunctive programs such as STABLE and SUPPORTED (both are co-NP-complete) as well as M_P^{supp} and WFS (that are even polynomial).

Recently, the idea of reducing disjunctive to non-disjunctive programs by using so called *shift-operations* was introduced independently by *Bonatti*, *Dix/Gottlob/Marek*, and *Schaerf*. In fact, Schaerf associated to each semantics SEM for normal programs a corresponding semantics Weak-SEM for disjunctive programs and asked for the properties of these weak semantics, in particular for the complexity of their entailment relations. While Schaerf concentrated on Weak-STABLE and Weak-SUPPORTED, we investigate the weak versions of Apt/Blair/Walker's stratified semantics M_P^{supp} and of vanGelder/Ross/Schlipf's wellfounded semantics WFS.

We show that credulous entailment for both semantics is NP-complete (consequently, sceptical entailment is co-NP–complete). Thus, unlike GCWA, the complexity of these semantics belongs to the *first* level of the polynomial hierarchy. Note that, unlike Weak-WFS, the semantics Weak-M_P^{supp} is not always defined: testing consistency of Weak-M_P^{supp} is also NP–complete.

We also show that Weak-WFS and Weak-M_P^{supp} are cumulative and rational and that, in addition, Weak-WFS satisfies some of the *well-behaved principles* introduced by Dix.

Keywords: Disjunctive Logic Programming, Semantics, Complexity, Abstract Properties.

*acknowledgements

1. Introduction

One is often tempted to consider as desired models of a theory T only *intended models*. But what is an *intended model*? Clearly, such model depends on the possible applications that the programmer has in mind while writing a theory. Various intentions lead to different results. For instance, the analysis of the frame problem leads to the acceptance of minimal models as the class of desired models and, subsequently to the notion of *circumscription* ([McC80]). The analysis of closed systems of *beliefs* leads to the acceptance of *supported* models of programs ([Cla78, MT93]).

In this paper we are looking at logical theories (described by means of a disjunctive program, possibly with negation in the body) as expressing a possible *causal* relationship between various atoms of the underlying language. Moreover we want to express the interpretation of negation as *negation by failure to prove*. The idea is that when we observe a *state of affairs*, we write a program describing it, and we want to find the possible ways of causal interplay of atoms.

We consider a general reduction method to associate to any disjunctive program P a set of normal programs. Given a semantics SEM for non-disjunctive programs, we assign to a disjunctive program P all SEM-models of the normal programs $P_1^{shift}, \ldots, P_n^{shift}$. These are obtained from P by a series of *shift-operations* which move atoms from the head to the body (and negate them). The procedure is unidirectional – we cannot move a literal from the body to the head. Therefore we can keep an additional control over the way the causal models are produced – if we want an atom not to depend on other atoms we can move it to the body (provided it appears in a head consisting of a proper disjunction). The resulting semantics is called Weak-SEM. This approach has also been followed be Schaerf in [Sch93, Sch95] and by Bonatti in an even more general context ([Bon93]).

Thus our idea is that some theories carry in their syntactic form one or more computational procedures that can be associated with that theory. In this we are taking a position similar to that of [ABW88, Prz88, vGRS91]. While Schaerf considered Weak-STABLE and Weak-Supported, we investigate in this paper the two semantics

- Weak-M_P^{supp}, and
- Weak-WFS.

Results about properties of Weak-M_P^{supp} have been already given in [DGM94]. These results are now extended to Weak-WFS and more properties of these semantics are investigated.

Weak-M_P^{supp} essentially gives a *stratified* interpretation to causality. That is, when a program is stratified, it imposes on atoms of the underlying language an ordering; some atoms are decided earlier than other atoms. Such ordering, together with the program itself leads to the unique model which can be viewed as a description of the causal relationship. The way it went in the earlier strata (together with the program) determines the way the matters stand in the next stratum. A theory (that is a disjunctive program) may or may not admit a transformation to a stratified logic program. Weak-M_P^{supp} is only defined for theories which admit such representation. Like in the case of stratified normal programs, theories logically equivalent may be different from the point of view of causality. Weak-M_P^{supp} has as intended models the set of all perfect models of all stratified shifts of P.

Weak-WFS is based on the wellfounded semantics WFS and associates to any program P the whole set of all shifts of P, no matter whether they are stratified or not. Unlike the M_P^{supp} semantics, WFS is always defined and therefore no restriction is needed. Weak-WFS declares as intended models the set of all wellfounded models of all shifts of P

The paper is organized as follows. Section 2. contains some terminology used throughout the paper, the definition of the perfect model for stratified non-disjunctive programs and its complexity. In Section 3. we introduce the shifting-operations and define the class of *causal* programs. This is the class where Weak-M_P^{supp} is consistent. Its NP-completeness is shown and a smaller class, *simple* disjunctive programs, is shown to be polynomial. In

Section 4. we formally define Weak-WFS, Weak-M_P^{supp} and investigate their relationship with GCWA. We also determine the complexity of their induced entailment relations and show some interesting abstract properties of them. Section 5. compares our work to the approach of Schaerf, Bonatti and Ben-Eliyahu/Dechter. We end in Section 6. with some conclusions.

2. Preliminaries

A *disjunctive rule* is a formula $a_1 \vee \ldots \vee a_n \leftarrow b_1, \ldots, b_m, \neg c_1, \ldots, \neg c_l$, where $n \geq 1,\ m, l \geq 0$ and a_i, b_i, c_i are arbitrary propositional atoms. As usual, the comma represents conjunction. We call such a rule *positive* if $l = 0$, *normal* if $n = 1$. One can think of a rule C as a pair of sets $\langle head(C), body(C)\rangle$, where $head(C) = \{a_1, \ldots, a_n\}$ and $body(C) = \{b_1, \ldots, b_m, \neg c_1, \ldots, \neg c_l\}$.

A *disjunctive logic program* is a set of disjunctive rules: it straightforwardly inherits the typology of rules. A normal logic program is often also called *general logic program*. The *Herbrand base* induced by a program P is denoted B_P.

Here we only deal with *finite* disjunctive logic programs. Since all the clauses of a disjunctive program have non-empty heads, a disjunctive logic program is always consistent (viewed as a first-order theory).

We say that a normal program P is *stratified* if there exists a *rank* function f, $f : At \rightarrow \mathbb{N}$ such that for every rule

$$C = p \leftarrow q_1, \ldots, q_n, \neg s_1, \ldots, \neg s_m \quad (p, q_1, \ldots, q_n, s_1, \ldots, s_m \text{ are atoms})$$

from P: $rk(p) \geq rk(q_i),\ i \leq n$, and $rk(p) > rk(s_j),\ j \leq m$.

Let P be a stratified normal logic program. We can assign to P a model, called the *perfect* model ([ABW88] as presented in [MT93]) as follows: first, split the program P into the union of programs P_i according to the ranks of heads. Let $P = \bigcup_{n \in \mathbb{N}} P_n$ be this decomposition. Define M_0 to be the least model of P_0 (notice that according to stratification condition, P_0, if non-empty, must be a Horn program). Next, assuming that $M_i, i < j$, are already computed proceed as follows: for every clause C in P_j perform the following *reduction*. If some atom p in $\bigcup_{i<j} M_i$ appears negatively in C then eliminate C. If all the atoms appearing negatively in the body of C do not belong to $\bigcup_{i<j} M_i$ then eliminate all these negated atoms. The resulting reduced program Q_j is a Horn program, and M_j is defined as the smallest model of $Q_j \cup \{a \leftarrow\ : a \in \bigcup_{i<j} M_i\}$.

The following fact is proved by Apt, Blair and Walker ([ABW88]):

Theorem 2.1. (Perfect Model for stratified Normal Programs)
If P is a stratified normal logic program, then its perfect model is a minimal model of P. Moreover the perfect model of P does not depend on stratification: every stratification of P generates the same perfect model.

We also note the following well-known fact, which follows immediately from the quadratic complexity of the wellfounded semantics ([Sch92a]):

Lemma 2.1. (Quadratic Complexity of the Perfect Model)
Let P be a stratified normal logic program. Then the perfect model of P can be computed in quadratic time.

3. Reducing Disjunctive to Normal Programs

Our intention is to define a semantics for disjunctive programs (or on a certain subclass of them) with good computational behaviour. Since the perfect model M_P^{supp} of a stratified normal program or the wellfounded model of an arbitrary normal program can be computed in quadratic time (see Lemma 2.1. in Section 2.) it is promising to try to reduce a disjunctive program to a set of (stratified) non-disjunctive programs.

We introduce the shifting operations in Subsection 3.1. and define the class of *causal* programs which is a proper subclass of the class of all disjunctive programs. Section 3.2. solves the complexity problem of testing causality: this has been stated as an open problem in [Sch93].

3.1. The Shift-Operation

The important notion to reduce disjunctive into non-disjunctive programs is a *shift*:

Definition 3.1. (Shift, Complete Shift σ)
A *shift* in a disjunctive logic program consists in moving a literal of a rule containing more than one literals in the head from the head to the body and negating it: A shift of "a_1" in $a_1 \vee \ldots \vee a_n \leftarrow b_1, \ldots, b_m, \neg c_1, \ldots, \neg c_l$ results in the clause

$$a_2 \vee \ldots \vee a_n \leftarrow b_1, \ldots, b_m, \neg a_1, \neg c_1, \ldots, \neg c_l.$$

We call any sequence of shifts σ that transform P into a normal program a *complete shift*.

Clearly, a shift does not change the classical models of a rule (viewed as a first-order formula). If C is a disjunctive rule and C' the result of a shift of some atom from the head to the body, then every model of C is a model of C' and conversely. But from a negation-as-failure viewpoint there is obviously a difference between "$a \vee b$" and "$a \leftarrow \neg b$".

Now the idea of reducing disjunctive to non-disjunctive programs is to appply all possible shifts until a non-disjunctive programs is reached, i.e. to consider complete shifts.

This means that we can associate to any disjunctive program P a set of non-disjunctive programs

$$\{P^\sigma : \ \sigma \text{ is a complete shift } \}.$$

Of particular importance are those complete shifts that result in stratified programs:

Definition 3.2. (Causal Program)
A disjunctive logic program is called *causal* if it can be transformed by a sequence of shift operations to a stratified normal logic program.

There is a simple syntactic condition to ensure causality. Let us define

$$b(P) := \bigcup_{C \in P} \{a : \ a \text{ an atom s.t. } a \text{ or } \neg a \in \mathit{body}(C)\}$$

Then

Lemma 3.1. Let P be a disjunctive logic program such that for every head $\mathit{head}(C)$ of a clause $C \in P$, $\mathit{head}(C) \setminus b(P) \neq \emptyset$. Then P is causal.

Proof:
Shift all the elements of $b(P)$ to the bodies. Notice that such elements may appear in the heads of clauses from P as well. Our assumption guarantees that after such sequence of shifts every clause will have at least one atom in the head. Next, order all the elements of $At \setminus b(P)$ in order $\prec$ of type at most ω. As before, leave at the head of the modified clause only the atom highest in the ordering $\prec$, shifting all the remaining atoms to the body. Now, assign to every atom the following rank: the elements of $b(P)$ are assigned the rank 0. Similarly, the atoms which do not appear in the heads of clauses of P are assigned 0 as well. For the remaining atoms p (these are precisely the atoms appearing in the heads of clauses after the initial shift) are assigned the rank $n+1$ where n is the position of p in the ordering $\prec$. We claim that the resulting program P' is stratified. Indeed, let

$C = p \leftarrow q_1, \ldots, q_n, \neg s_1, \ldots, \neg s_m$ be a clause of P'. First of all, the atoms q_i, $i \leq n$ must belong to $b(P)$, and therefore they have rank 0. Hence their rank is smaller than that of p. Concerning the remaining (negated) literals: they either come from $b(P)$ and then they have the rank 0 (whereas the rank of p is not zero) or they appear in the ordering $\prec$ before p and so they also have smaller rank. Thus P' is stratified.
We notice that the program constructed in the proof of Lemma 3.1. is in fact *hierarchical* (see [Llo87]), not only stratified. A special instance of the previous Lemma is that all programs whose bodies are empty are causal. But not even every positive disjunctive program is causal:

Example 3.1. (A Non-Causal Positive Disjunctive Program)
Let P_{nc} be the following disjunctive logic program:

$$\begin{aligned} P_{nc} : p \vee q &\leftarrow r \\ p \vee r &\leftarrow q \\ r \vee q &\leftarrow p \end{aligned}$$

Then P_{nc} is not causal. Indeed, by symmetry we can shift p in the first clause. Then $rk(p) < rk(q)$, and $rk(r) \leq rk(q)$. This forces us to select r for the shift in the third clause. This implies that $rk(p) = rk(r)$ and $rk(q) < rk(p)$. Now, in the second clause, neither p nor r can be shifted to the body, for one of them has to have smaller rank than the other. On the other hand, every proper subprogram of P_{nc} is causal.

3.2. Testing Causality

In [Sch93], Schaerf asked (in our terminology) to determine the complexity of testing causality and of determining tractable subclasses of programs. Theorem 3.1. and Lemma 3.2. are solutions to these problems.

Theorem 3.1. (NP-Completeness of Deciding Causality)
Testing whether a positive disjunctive logic program is causal is NP-complete.

Proof:
The problem is obviously in NP: if the given disjunctive logic program is stratifiable we may nondeterministically guess a correct sequence of shifts and check in polynomial time by well-known methods that the resulting normal logic program is stratifiable. NP-hardness is shown by a polynomial transformation from EXACT HITTING SET, a well-known NP-complete problem.

An instance I of EXACT HITTING SET consists of a finite set S and a family of subsets $S_1, \ldots, S_n$ of S. The question is whether there exists a set $H \subseteq S$ such that $\forall i : 1 \leq i \leq n \ \ |H \cap S_i| = 1$. If such a set exists it is called an *exact hitting set* of $S_1, \ldots, S_n$.

To each instance $\{S, S_1, \ldots, S_n\}$ of EXACT HITTING SET we define a disjunctive logic program $DLP(I)$ as follows

- The atoms of $DLP(I)$ are: $S \cup \{q\}$, where q is a new predicate symbol.
- $DLP(I)$ contains for each S_i a rule R_i of the form: $\bigvee_{x \in S_i} x \leftarrow q$.
- In addition, $DLP(I)$ contains an extra rule R of the form: $q \leftarrow \bigwedge_{x \in S} x$.

We claim that

$$DLP(I) \text{ is causal } \textit{ if and only if } \text{ the } S_i \text{ have an exact hitting set.}$$

Let us first show the if-direction. Assume the $S_1, \ldots, S_n$ have an exact hitting set H. Transform the disjunctive logic program $DLP(I)$ to a normal logic program P' by shifting each atom of each rule $R_1, \ldots, R_n$ not occurring in H to the right (i.e., to the rule body). First observe that P' is effectively a normal logic program, since each rule contains only one atom in its head (because H is an exact hitting set). Now observe that P' is stratified. Indeed, none of the negative literals that occur in the rule bodies occurs also in the rule head; this allows a stratification of two strata: the top stratum consists of $H \cup \{q\}$ and the bottom stratum consists of all other predicate symbols.

Let us now show the only-if direction. Assume $DLP(I)$ is causal. Then $DLP(I)$ can be transformed by shift operations to a stratified normal logic program P'. Obviously the special rule R remains unaffected by the shifts and is therefore also present in P'. Thus P' is of the form $\{R'_1, \ldots, R'_n, R\}$ where R'_i is the transform of R_i for $1 \leq i \leq n$. Let H be the set of all head-atoms of the rules $R'_1, \ldots, R'_n$. Obviously, H is a hitting set of the family $S_1, \ldots, S_n$, since H intersects each S_i. We claim that H is an *exact* hitting set of $S_1, \ldots, S_n$. Assume it is not. Then for some S_i it holds that $|S_i \cap H| \geq 2$, hence, there are at least two different atoms p and s in $H \cap S_i$. This means that during the *shift* from $DLP(I)$ to P', at least one of these atoms, say p, is shifted from the head to the rule body of R_i. Hence p occurs negatively in the body of R'_i. By definition of H, however, there must exist a rule R'_j whose head is p. Now it is easy to see that the existence of the three rules R'_i, R'_j, and R in P' constitutes a contradiction to our assumption that P' is stratified. Indeed, from R'_i we know that for some atom t (namely the head of R'_i), we have $t > p$; from R we further know that $q \geq t$, hence it follows $q > p$. But from the existence of R'_j we deduce that $p \geq q$, a contradiction. Therefore, H must be an exact hitting set.

Observe that the constructed program does not contain any negated literal (if written in implicational form of course). This shows that NP-completeness of causality-testing holds for the restricted class of *positive* disjunctive logic programs. Actually, adding negated literals to the rule bodies makes things easier because some choices are prohibited.

If a disjunctive logic program has only negated literals in the rule bodies, then causality can be tested in polynomial time. Let us therefore call *simple disjunctive logic programs* those disjunctive logic programs whose rule bodies contain only negated literals.

Lemma 3.2. (Polynomial Complexity of Simple Programs)
There is a polynomial time algorithm for testing causality of simple disjunctive logic programs.

Proof:
One first shows the following two claims:

1. If a simple disjunctive logic program P is causal, then there must exist an atom p in some rule head of P such that $\neg p$ does not occur in any rule body (exploiting the finiteness ...). Call such an atom a *top-atom*.
2. If a simple disjunctive logic program P contains a top-atom p then it is causal if and only if the program $P' \subset P$ consisting of all rules of P in which p does not occur is causal.

These claims imply that a polynomial algorithm for testing the causality of a simple disjunctive logic program P is easily derived by choosing top-atoms of smaller and smaller programs. If the algorithm ends-up with the empty program then the input-program is causal; if the algorithm gets stuck because at some level there is no top-atom, then the input-program is not causal.

We mention that in recent work of the first author ([BD95b, BD95a]) it has been shown that under any semantics satisfying two simple properties (*Partial Evaluation* and *Elimination of Tautologies*) a program may be transformed in an equivalent simple program. This transformation itself is, unfortunately, exponential.

4. Weak-WFS, Weak-M_P^{supp} and their Properties

In this section we first define the notions of *causal* and *weak wellfounded* model and consider their induced entailment relations *truth in all causal (resp. weak wellfounded) models* by comparing them with GCWA (Section 4.1.). We then determine their complexity (Section 4.2.) and consider in Section 4.3. abstract properties introduced by Dix ([Dix95a, Dix95b]) into logic programming.

4.1. Definition of Weak-WFS, Weak-M_P^{supp}

We have associated to every disjunctive program the set of its complete shifts. This gives us the following

Definition 4.1. (Weak WFS)
The weak-wellfounded semantics of a disjunctive program P is defined as the set of all wellfounded models of all complete shifts of P (we call these models *weakly wellfounded*:

$$\text{Weak-WFS}(P) = \{\text{WFS}(P^\sigma) : \sigma \text{ is a complete shift of } P\}.$$

A literal l is sceptically entailed via Weak-WFS from P, if it is true in all weakly wellfounded models of P.

We again note that this definition has already be given in [Sch93].

Of course, one can argue that there is no general agreement about the "right" semantics on the class of all normal programs and that, therefore, WFS is only one among several candidates. Schaerf for example considered in [Sch93, Sch95] the *supported* and the *stable* models as competing approaches and discussed their induced weak versions for general disjunctive programs.

We choose here, since there is general agreement about the right semantics for the class of *stratified normal* programs, the *causal* models as natural candidates:

Definition 4.2. (Weak-M_P^{supp})
The weak-M_P^{supp} semantics of a disjunctive program P is defined as the set of all wellfounded models of all complete shifts of P that result in stratified normal programs (we call those models *causal*):

$$\text{Weak-M}_P^{supp}(P) = \{\text{WFS}(P^\sigma) : \sigma \text{is a complete shift of } P \text{ and } P^\sigma \text{ is stratified}\}.$$

Literal l is sceptically entailed by Weak-M_P^{supp} from P, if it is true in all causal models of P.

Hence, we are looking at the possible stratifications of a disjunctive logic program and in this fashion a disjunctive logic program may possess none, unique, or several causal models. The following observations are obvious:

1. Every Horn program P possesses the unique causal model. This model coincides with the least model M_P of P.
2. Every stratified normal program P possesses its perfect model M_P^{supp} as the unique causal model. The causal semantics therefore extends the stratified semantics.
3. The disjunctive logic program $\{p \vee q\}$ possesses two causal models: $\{p\}$ and $\{q\}$.

Let us compare our semantics with the GCWA (introduced by Minker in [Min82]). For a positive disjunctive program P, the GCWA entails all literals true in all *minimal* models of P. Note that for *atoms* it makes no difference between considering (classical) entailment or minimal entailment. This is no longer the case for our causal entailment.

Example 4.1. (Minimal vs. Causal Entailment)
Let P_{c-ent} be the following disjunctive logic program:

$$\begin{aligned} P_{c-ent}: p &\leftarrow q \\ q &\leftarrow p \\ p \vee q \vee r & \end{aligned}$$

Clearly, P_{c-ent} possesses two minimal models, one contains both p and q but not r, the other contains only r (but neither p nor q). When we look at the stratified programs obtained from P_{c-ent} by shifts, then we see that there is only one such program, in which both p and q are shifted to the right. This is because both p and q are in the same stratum, so the shifts of the third clause must move them both. Therefore only the second of two minimal models is a causal model of P_{c-ent}, and so P_{c-ent} causally entails r. On the other hand, P_{c-ent} does not minimally entail any atom: GCWA$(P)=\emptyset$.

In the last example the causal semantics is stronger than GCWA (more literals are derivable). On the other hand there are non-causal positive programs (see Example 3.1.): for such programs the causal semantics is not defined but the GCWA is.

This consistency problem obviously does not occur for Weak-WFS. But Weak-WFS has other shortcomings. We consider the program P_1 consisting of "$a \vee b$" and P_2 obtained from P_1 by adding "$a \vee b \vee c$". The weak-wellfounded semantics of P_1 consists of the two minimal models $\{a\}$ and $\{b\}$ and therefore coincides with GCWA. The weak-wellfounded semantics of P_2 consists not only of the two two-valued models $\{a\}$, $\{b\}$ but it also contains the three-valued model $\langle\emptyset; \{c\}\rangle$ where $\neg c$ is true and both a and b are undefined. It is therefore weaker than GCWA. In fact, we cannot even derive $a \vee b$! We have the following easy observations:

Lemma 4.1. (Relationship of Weak-WFS, Weak-M_P^{supp} and GCWA)

- **a.** If P is positive causal: sceptical causal-entailment is stronger than sceptical GCWA.
- **b.** If P is positive: sceptical GCWA is stronger than sceptical weak-WFS-entailment.
- **c.** If P is causal: sceptical causal-ent. is stronger than sceptical Weak-WFS-ent.

Let us define another interesting class of programs:

Definition 4.3. (Strongly Causal Programs)
A disjunctive program is called *strongly causal* if every complete shift results in a stratified normal program.

It is immediate that Weak-WFS and Weak-M_P^{supp} coincide for all strongly causal programs, because WFS extends the stratified semantics M_P^{supp} and Weak-M_P^{supp} is consistent on this class.

The main reason for the difference of GCWA and the causal or the weak-wellfounded semantics is their complexity: while GCWA is Π_2^P-complete ([CS90, EG93]), both causal and weak-wellfounded entailment are located one level below in the polynomial hierarchy as we will show in the next section.

4.2. Complexity of the Entailment Problem

In the previous section we introduced causal and weakly-wellfounded entailment and noticed that these notions, even for atoms, are different from the usual entailment defined by GCWA. We also determined in Section 3.2. the complexity of the existence of a causal model. In this section we use these results to analyze the complexity of causal entailment for positive programs. A different proof will show us that this complexity is identical to the complexity of causal entailment for the class of *strongly causal* programs. Therefore (since weakly-wellfounded and causal models coincide for strongly causal programs) we get the same complexity for Weak-WFS and Weak-M_P^{supp}.

Let us define $P \mathrel{|\!\sim}^{sc} l$ ("l follows *sceptically* from P") to denote that l is true in *every* causal model of theory P. Similarly $P \mathrel{|\!\sim}^{cr} l$ ("l follows *credulously* from P") denotes that l is true in *some* causal model of P.

Theorem 4.1. (Complexity of Causal Entailment for Positive Programs)

- **a.** Determining whether "$P \mathrel{|\!\sim}^{sc} l$" is a co-$NP$-complete problem even for positive programs P and l being atoms.
- **b.** Determining whether $P \mathrel{|\!\sim}^{cr} l$ is an NP-complete problem even for positive programs P and l being atoms.

Proof:
First, we need to prove that the problem complementary to our problem is in the class NP and that our problem is co-NP-hard.

1. To test that $P \mathrel{|\!\sim}^{sc} l$ can be done in NP time is done as follows: first we guess a stratification for P. Next, using Lemma 2.1. we compute the corresponding causal model. Finally we check that the constructed causal model of P does not satisfy l.
2. *"$P \mathrel{|\!\sim}^{sc} a$ for some atom a not occurring in P"* is equivalent to the fact that P is not causal. Hence there is a trivial polynomial reduction from the problem complementary to causality testing to testing of $P \mathrel{|\!\sim}^{sc} l$. Thus, by Theorem 3.1. our problem is co-NP-hard.

Second, we need to prove that our problem belongs to the class NP and that it is NP-hard.

1. To establish that our problem is in the class NP we proceed as above. We guess a stratification of P, compute the corresponding causal model of P and then check that that model satisfies l. This is, of course, done in polynomial time.
2. Now, it is clear that P possesses a causal model if and only if "$P \mathrel{|\!\sim}^{cr} a$ for some atom occurring in P". Thus we get a trivial reduction of the stratifiability problem to the $\mathrel{|\!\sim}^{cr}$ entailment problem. This, by Proposition 3.1. implies that our problem is NP-hard.

The next theorem proves the same result for the class of *strongly causal* programs. Note that the proof is completely different from the previous one, because it is based on a reduction to 3-SAT.

Theorem 4.2. (Complexity of Causal Entailment for Strongly Causal Programs)

1. Determining whether $P \mathrel{|\!\sim}^{sc} l$ is a co-NP-complete problem even for strongly causal programs P and l being atoms.
2. Determining whether $P \mathrel{|\!\sim}^{cr} l$ is an NP-complete problem even for strongly causal programs P and l being atoms.

Proof:
In view of the proof of Theorem 4.1. we will only show that the problem "$P \mathrel{|\!\sim}^{sc} a$" is co-$NP$-complete. In fact, we show that there is a polynomial transformation from the complement of 3-SAT to this problem.

An instance I of 3-SAT consists of a clause set $C_1, \ldots, C_m$ over a set of variables $\{p_1, \ldots, p_n\}$ such that each C_i contains at most 3 literals. The question is whether there is an assignment of the variables such that the whole clause set is satisfiable.

For each instance I of 3-SAT we construct a strongly causal disjunctive logic program $DLP(I)$ as follows:

- The $2n+1$ atoms of $DLP(I)$ are $p_1, \ldots, p_n$, $not\_p_1, \ldots, not\_p_n$, a_{new}.
- $DLP(I)$ contains for each $1 \leq i \leq n$ the rule: $p_i \vee not\_p_i \leftarrow$.

- In addition $DLP(I)$ contains the m rules

$$a_{new} \leftarrow f(l_1), f(l_2), f(l_3)$$

where $C = \{l_1, l_2, l_3\}$ is one of the m clauses in I. Here we denote by f the function defined on $\{p_1, \ldots, p_n, \neg p_1, \ldots, \neg p_n\}$ by

$$f(x) := \begin{cases} p_1 & \text{if } x = \neg p_1, \\ \vdots & \vdots \\ p_n & \text{if } x = \neg p_n, \\ not\_p_1 & \text{if } x = p_1, \\ \vdots & \vdots \\ not\_p_n & \text{if } x = p_n, \end{cases}$$

We claim that

$$I \text{ is unsatisfiable } \textit{ if and only if } \text{ Weak-M}_P^{supp}(DLP(I)) \models a_{new}.$$

First we show that $DLP(I)$ is strongly causal. It suffices to consider only the rules $p_i \vee not\_p_i \leftarrow$ because all other rules are already stratified. But any shift on these rules determines a stratification (with 2 strata): if p_i is shifted then not_p_i is in the higher stratum and p_i will be false and not_p_i be true in the corresponding model. If not_p_i is shifted then p_i is in the higher stratum and not_p_i will be false and p_i be true in the corresponding model. Thus any complete shift results in a stratified normal program with exactly 2 strata. The bottom stratum contains all p_i that are false and all not_p_j such that p_j is true. The top stratum contains all other atoms.

Therefore any complete shift of $DLP(I)$ corresponds to a variable assignment to the set $\{p_1, \ldots, p_n\}$ and vice versa. The atom a_{new} is true in $\mathrm{M}_{P\sigma}^{supp}$ if and only if one of the clauses $a_{new} \leftarrow f(l_1), f(l_2), f(l_3)$ has been applied, i.e. if the complete shift σ induced a variable assignment which falsified the corresponding clause $\{l_1, l_2, l_3\}$ of I.

Corollary 4..1 (Complexity of Weak-WFS)
Credulous entailment for Weak-WFS is NP-complete, even for strongly causal programs and deriving atoms. Consequently, sceptical entailment for Weak-WFS is co-NP-complete for the same class of programs.

4.3. Abstract Properties

In [Dix95a, Dix95b] the first author adapted various abstract conditions known in the context of general nonmonotonic reasoning to logic programming semantics. It was argued that the properties of *Cumulativity* and *Rationality*

Cumulativity: If $P \mid\!\sim^{sc} a$ then: $P \mid\!\sim^{sc} l$ if and only if $P \cup \{a\} \mid\!\sim^{sc} l$.

Rationality: If <u>not</u> $P \mid\!\sim^{sc} \neg a$ then: $P \mid\!\sim^{sc} l$ implies $P \cup \{a\} \mid\!\sim^{sc} l$.

(originally introduced by Gabbay and Makinson for general nonmonotonic theories) are connected with the complexity of a semantics. This was supported by two famous examples: the wellfounded semantics WFS and the WGCWA. Both are cumulative and rational [Dix91, Dix92b] and have a lower complexity than their non-rational “competitors” STABLE and GCWA:

- While WFS is polynomial (this was already cited in Section 2.), STABLE is at the first level of the polynomial hierarchy ([MT91]),

- While WGCWA is at the first level, GCWA is at the second level of the polynomial hierarchy ([CS90]).

In addition, Fernandez defined in [Fer93] a semantics WICWA for general disjunctive programs which he claims to be cumulative and rational. He showed that WICWA is of lower complexity than PERFECT (which extends GCWA and is not rational). The same holds if we compare our causal and weak-wellfounded semantics with GCWA: we have already shown that they have lower complexity. Indeed, they are cumulative but not rational:

Theorem 4.3. (Cumulativity for Weak-WFS and Weak-M_P^{supp})
Sceptical entailment $\mid\!\sim^{sc}$ of both Weak-WFS and Weak-M_P^{supp} is cumulative. Neither of the two semantics is rational.

Proof:
The proof is very similar for both semantics. We show it first for Weak-WFS. Note that for any complete shift σ

$$P^\sigma \cup \{a\} = (P \cup \{a\})^\sigma$$

and therefore, by cumulativity of WFS (see [Dix95a]), if $\mathrm{WFS}(P^\sigma) \models a$ then

$$\mathrm{WFS}((P \cup \{a\})^\sigma) = \mathrm{WFS}(P^\sigma \cup \{a\}) = \mathrm{WFS}(P^\sigma).$$

Now suppose a is true in all weakly wellfounded models of P, i.e. in all $\mathrm{WFS}(P^\sigma)$ where σ is a complete shift. By the last identity we have that $\mathrm{WFS}(P^\sigma) = \mathrm{WFS}((P \cup \{a\})^\sigma)$ so that also the set of all literals true in the intersection of all $\mathrm{WFS}(P^\sigma)$ coincides with the set of all literals true in the intersection of all $\mathrm{WFS}((P \cup \{a\})^\sigma)$. This is exactly the cumulativity of Weak-WFS.

Obviously, the same proof works also for Weak-M_P^{supp}, because the addition of an atom a has no effect on the stratification.

The counterexample against rationality is the following

$$\begin{array}{lll} P_{rat}: e \vee f & \leftarrow & \\ \quad a & \leftarrow & \neg f \\ \quad x & \leftarrow & a, \neg e \end{array}$$

Note that only one complete shift σ_f (the one shifting f) results in a program whose wellfounded model contains a. In this model, $\neg x$ is also contained. The wellfounded model of the other complete shift σ_e contains $\neg a$ and also $\neg x$. Therefore $\neg x$ is weakly wellfounded derivable from P_{rat} but $\neg a$ is not. Adding a however to the program $P_{rat}^{\sigma_e}$ results in a wellfounded model that derives x: $\mathrm{WFS}(P_{rat}^{\sigma_e} \cup \{a\}) \models x$. This counterexample also applies to Weak-M_P^{supp} because all programs are stratified.

This shows that the original claim from the first author, namely that cumulativity and rationality of a semantics might have a strong relation on the complexity is not true. Our semantics have a good complexity even without being rational. Another counterexample is the semantics WFS_C of Schlipf ([Sch92b]), which is equivalent to WFS^+ ([Dix95a]): WFS is of the same (high) complexity as STABLE (for normal programs) but it is cumulative and rational.

We end this section with some comments about consistency. The inconsistency of the stable semantics is not the only shortcoming. In fact, often when asking a query to a program under a semantics it would be nice when the answer only depends on that part of the program, where the query *depends on*, i.e. the subset of rules determined by the call-graph below the query. This property has been called *Relevance* and was introduced in [Dix92a]. In [BD95a] this was extended to disjunctive programs and shown that it implies the following property, called *Independence*

$$SEM(P) \models \phi \iff SEM(P \cup P') \models \phi,$$

provided that the predicates occurring in P and P' are disjoint, and ϕ contains only predicates from P. *Independence* in turn implies *Consistency*. Another property satisfied by most semantics is the *Elimination of Tautologies*: the semantics of a program does not change, if rules that contain an atom a at the same in their head and in their body can be eliminated without changing the semantics of the program.

It is therefore natural to ask how our semantics behave in view of these properties.

Lemma 4.2. (More Abstract Properties)

- **a.** *Elimination of Tautologies* holds for Weak-M_P^{supp} on the class of causal programs and for Weak-WFS on the class of strongly causal programs.
- **b.** *Relevance* holds for Weak-WFS in general as well as for Weak-M_P^{supp} for causal programs.

The reason that *Elimination of Tautologies* does not hold in general for Weak-M_P^{supp} is that this transformation may transform a non-causal to a causal program. Concerning Weak-WFS note that it derives from the program consisting of "$a \vee p \leftarrow p$" and "$p \leftarrow \neg p$" neither p nor a nor their negations. But if the first clause is eliminated, $\neg a$ wis derivable.

Note also that *Relevance* does not hold for STABLE even on the class of programs where STABLE is consistent.

5. Relation to other approaches

We already mentioned Schaerf who also considered recently ([Sch93, Sch95]) the technique of shifting a disjunctive program into a normal one. The semantics that is closest to our Weak-M_P^{supp} is his Weak-SUPPORTED semantics, where he considers all supported models of all complete shifts, whereas we are only considering those complete shifts that result in stratified programs (and we only take the unique perfect model).

Therefore even for stratified programs his semantics Weak-SUPPORTED is different from ours. The complexity of his semantics, however, is identical to ours.

Another approach is due to R. Ben-Eliyahu and R. Dechter [BED92]. They tried to find classes of programs where the complexity of the stable semantics is low. They defined the class of *head-cycle-free* programs and proved that the entailment relation ("truth in all stable models") is co-NP-complete for this class. A program is *head-cycle-free*, if any two literals that occur in the same head do not depend positively on each other. Here "A depends positively on B" is the transitive closure of "A depends immediately positively on B" (which means that there is a rule C containing A in its head and B positively in its body). Note that in this definition negative dependencies are ignored.

As an example "$A \vee B \leftarrow A, B$" is not *head-cycle-free* while "$A \leftarrow \neg A$" is. Therfore the classes of *head-cycle-free* programs and of *causal* programs are incomparable. But for positive programs we have:

Lemma 5.1. The class of positive causal programs strictly includes the class of positive head-cycle free clauses.

Therefore our semantics can be seen as an extension of the class of *positive head-cycle free clauses* retaining the attractive low complexity. But obviously both semantics are different even on the smallest class of head-cycle free programs. This is because causal semantics relies on *perfect* models while Ben-Eliyahu and Dechter's semantics is based on *stable* models.

We also note that causal entailment is stronger than stable entailment (this was pointed out by Piero Bonatti):

Lemma 5.2. Every causal model of P is also a disjunctive stable model of P.

The reason is that every causal model M of P is also a minimal model of P^M, the Gelfond-Lifschitz transform of P with respect to M. By definition, the minimal models M of P^M (a *positive* program) are exactly the stable models of P.

6. Conclusions

We found that some propositional theories S (we termed them *causal*) carry with them one or more computational procedures which determine the order of construction of atoms in some model of S. Once such a procedure is known, we can construct this model in polynomial time. We showed that the problem of existence of such order is itself NP-complete, thereby solving a problem recently raised by Schaerf.

We defined *causal* models and investigated the induced entailment relation as well as Weak-WFS. Causal entailment is different from all other semantics proposed in the literature and is of low complexity. In addition these semantics behave in a nice way: sceptical entailment is *cumulative* and, on certain classes of programs, it also satisfies *Relevance* and *Elimination of Tautologies*.

We gave simple syntactic conditions on the program that ensure the existence of causal models. In general, the problem of existence of a causal model is as complex as the satisfiability problem. We find a class of programs for which we can test the existence of a causal model in polynomial time.

We also compared our approach to work of Ben-Eliyahu/Dechter and Schaerf.

Acknowledgements

We thank Piero Bonatti, Marco Schaerf and some anonymous referees for useful comments on the subject of the paper.

References

[ABW88] K. R. Apt, H. A. Blair, and A. Walker. Towards a theory of declarative knowledge. In Jack Minker, editor, *Foundations of Deductive Databases*, chapter 2, pages 89–148. Morgan Kaufmann, 1988.

[BD95a] Stefan Brass and Jürgen Dix. Characterizations of the Stable Semantics by Partial Evaluation. In A. Nerode, W. Marek, and M. Truszczyński, editors, *Logic Programming and Non-Monotonic Reasoning, Proceedings of the Third International Conference*, LNCS 928, pages 85–98, Berlin, June 1995. Springer. *Extended and revised version will appear soon in Journal of Logic Programming under the same title.*

[BD95b] Stefan Brass and Jürgen Dix. Disjunctive Semantics based upon Partial and Bottom-Up Evaluation. In Leon Sterling, editor, *Proceedings of the 12th Int. Conf. on Logic Programming, Tokyo*, pages 199–213. MIT Press, June 1995. *Extended and revised version will appear soon in Journal of Logic Programming under the title "Semantics of Disjunctive Logic Programs Based on Partial Evaluation".*

[BED92] Rachel Ben-Eliyahu and Rina Dechter. Propositional Semantics for Disjunctive Logic Programs. In K. R. Apt, editor, *LOGIC PROGRAMMING: Proceedings of the 1992 Joint International Conference and Symposium*, Cambridge, Mass., November 1992. MIT Press.

[Bon93] Piero Bonatti. Shift-based semantics: general results and applications. Technical Report CD-TR-93-59, Technical University of Vienna, Inst. für Informationssysteme, 1993.

[Cla78] K. L. Clark. Negation as Failure. In H. Gallaire and J. Minker, editors, *Logic and Data-Bases*, pages 293–322. Plenum, New York, 1978.

[CS90] Jan Chomicki and V.S. Subrahmanian. Generalized Closed World Assumption is Π_2^0-Complete. *Information Processing Letters*, 34:289–291, 1990.

[DGM94] Jürgen Dix, Georg Gottlob, and Viktor Marek. Causal Models for Disjunctive Logic Programs. In Pascal Van Hentenryck, editor, *Proceedings of the 11th Int. Conf. on Logic Programming, S. Margherita Ligure*, pages 290–302. MIT, June 1994.

[Dix91] Jürgen Dix. Classifying Semantics of Logic Programs. In Anil Nerode, Wiktor Marek, and V. S. Subrahmanian, editors, *Logic Programming and Non-Monotonic Reasoning, Proceedings of the first International Workshop*, pages 166–180, Cambridge, Mass., July 1991. Washington D.C, MIT Press.

[Dix92a] Jürgen Dix. A Framework for Representing and Characterizing Semantics of Logic Programs. In B. Nebel, C. Rich, and W. Swartout, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference (KR '92)*, pages 591–602. San Mateo, CA, Morgan Kaufmann, 1992.

[Dix92b] Jürgen Dix. Classifying Semantics of Disjunctive Logic Programs. In K. R. Apt, editor, *LOGIC PROGRAMMING: Proceedings of the 1992 Joint International Conference and Symposium*, pages 798–812, Cambridge, Mass., November 1992. MIT Press.

[Dix95a] Jürgen Dix. A Classification-Theory of Semantics of Normal Logic Programs: I. Strong Properties. *Fundamenta Informaticae*, XXII(3):227–255, 1995.

[Dix95b] Jürgen Dix. A Classification-Theory of Semantics of Normal Logic Programs: II. Weak Properties. *Fundamenta Informaticae*, XXII(3):257–288, 1995.

[EG93] Thomas Eiter and Georg Gottlob. Propositional Circumscription and Extended Closed World Reasoning are Π_2^P-complete. *Theoretical Computer Science*, 144(2):231–245, Addendum: vol. 118, p. 315, 1993, 1993.

[Fer93] Jose Alberto Fernández. Weak Models for Disjunctive Logic Programs. In *Proceedings of Workshop on Logic Programming with Incomplete Information, Vancouver Oct. 1993, following ILPS' 93*, pages 190–205, 1993.

[Llo87] John W. Lloyd. *Foundations of Logic Programming*. Springer, Berlin, 2nd edition, 1987.

[McC80] John McCarthy. Circumscription: A Form of Nonmonotonic Reasoning. *Artificial Intelligence*, 13:27–39, 1980.

[Min82] Jack Minker. On indefinite databases and the closed world assumption. In *Proceedings of the 6th Conference on Automated Deduction, New York*, pages 292–308, Berlin, 1982. Springer.

[MT91] Wiktor Marek and Mirek Truszczyński. Computing Intersection of Autoepistemic Expansions. In *Logic Programming and Non-Monotonic Reasoning, Proceedings of the first International Workshop*, Cambridge, Mass., 1991. MIT Press.

[MT93] Wiktor Marek and Mirek Truszczyński. *Nonmonotonic Logics; Context-Dependent Reasoning*. Springer, Berlin, 1st edition, 1993.

[Prz88] Teodor Przymusinski. On the declarative semantics of deductive databases and logic programs. In Jack Minker, editor, *Foundations of Deductive Databases*, chapter 5, pages 193–216. Morgan Kaufmann, 1988.

[Sch92a] John S. Schlipf. A Survey of Complexity and Undecidability Results in Logic Programming. In H. Blair, W. Marek, A. Nerode, and J. Remmel, editors, *Proceedings of the Workshop on Complexity and Recursion-theoretic Methods in Logic Programming, following the JICSLP'92*. informal, 1992.

[Sch92b] John S. Schlipf. Formalizing a Logic for Logic Programming. *Annals of Mathematics and Artificial Intelligence*, 5:279–302, 1992.

[Sch93] Marco Schaerf. Negation and minimality in non-Horn databases. In Catriel Beeri, editor, *Proceedings of the Twelfth Conference on Principle Of Database Systems (PODS-93)*, pages 147–157. ACM Press, 1993.

[Sch95] Marco Schaerf. Negation and minimality in disjunctive databases. *Journal of Logic Programming*, 23(1):63–83, 1995.

[vGRS91] Allen van Gelder, Kenneth A. Ross, and John S. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38:620–650, 1991.