PROCEEDINGS OF THE INTERNATIONAL CONFERENCE
ON FIFTH GENERATION COMPUTER SYSTEMS 1992,
edited by ICOT. © ICOT, 1992

ACYCLIC DISJUNCTIVE LOGIC PROGRAMS WITH ABDUCTIVE PROCEDURE AS PROOF PROCEDURE

Phan Minh Dung

Division of Computer Science
Asian Institute of Technology
GPO Box 2754, Bangkok 10501, Thailand.
E-mail: dung@ait.th

Abstract

We introduce and study a natural subclass of the locally stratified disjunctive logic programs, the class of acyclic disjunctive logic programs which extends the class of acyclic normal logic programs in [AB].

We show that each acyclic disjunctive program P can be transformed into an equivalent normal program N(P) where the equivalence between P and N(P) means that each perfect model of P is a stable model of N(P) and vice versa.

We show that the Eshghi and Kowalski's abductive procedure [EK,Dun] is sound with respect to the stable semantics of N(P). Thus this procedure can be used as a proof procedure for acyclic disjunctive programs.

We give sufficient conditions for the completeness and termination of the abductive procedure.

1. Introduction

Let us consider the following example

Example P: $p \vee q$

The semantics of P is defined by its two minimal models {p},{q}.

Let us translate P into N(P):

$$p \leftarrow \neg q$$
$$q \leftarrow \neg p$$

N(P) has two stable models {p},{q}. So P and N(P) are equivalent wrt stable semantics.
//

What can we gain from such translation ??

The gain is indeed significant. While no proof procedure for general disjunctive programs wrt stable semantics has been given so far in the literature, the Eshghi-Kowalski's abductive procedure given in [EK] and studied extensively in [Dun], is such a one for normal logic programs. Hence for those disjunctive programs which can be transformed into an equivalent normal programs, the Eshghi-Kowalski's abductive procedure can be used as a proof procedure for stable semantics.

Acyclic disjunctive programs constitute such a class of programs. Intuitively, an acyclic disjunctive program is a program whose atom dependency graph contains no loop. The class of acyclic disjunctive programs is a natural extension of the class of acyclic normal logic programs in [AB]. Similarly to [AB], we will show that several ways to define the semantics of logic programs, e.g. the predicate completion, perfect model semantics, stable model semantics etc., coincide in the case of acyclic disjunctive programs. The most striking characterization of acyclic disjunctive programs is that each program in this class can be transformed into an equivalent normal logic programs which themself exhibit a remarkable termination behavior as their atom dependency graph does not contain any positive loop. This result suggests immediately that the abductive procedure can be used as a proof procedure for acyclic disjunctive programs.

The paper is organized as follows: In the next paragraph, we define the acyclic disjunctive programs. Then in section 3, we show that each acyclic disjunctive program P can be transformed into an equivalent normal program N(P). In section 4, we show the soundness of the abductive procedure with respect to the stable semantics of N(P). In section 5, we give sufficient condition for the completeness of the abductive procedure.

2. Preliminary

A literal is either an atom or the negation of an atom. A disjunctive clause is a clause of the form $A_1 \vee ... \vee A_n \leftarrow L_1,...,L_m$ where $0<n$, $0 \leq m$ and A_i's are atoms and L_j's are literals. If n=1, then a disjunctive clause is called a normal clause. The head and body of a clause C are denoted by head(C) and body(C) respectively. Further, pos(C) denotes the set of atoms occurring positively in the body of C while neg(C) denotes the set of atoms under negation in the body of C. A disjunctive program is a finite set of disjunctive clauses. Similarly, a normal program is a finite

set of normal clauses. The Herbrand base of a program P is denoted by HB_P. As usual, a Herbrand interpretation is considered as a subset of HB_P. The set of all ground instances of clauses of a disjunctive program P is denoted by G_P. If L is a literal then $\neg L$ denotes the complementary of L. If S is a set of literals, then $\neg .S = \{ \neg L \mid L \in S \}$.

A disjunctive program P is locally stratified [Pr1] if it is possible to decompose the Herbrand base of P into disjoint sets, called strata $H_0, H_1, \dots, H_\alpha, \dots, H_\tau, \dots$ where $\alpha < \tau$ and τ is a countable ordinal so that for each ground clause in G_P

$$C_1 \vee .. \vee C_k \leftarrow A_1, \dots, A_n, \neg B_1, \dots, \neg B_m$$

(i) all C_i belong to the same stratum, say H_r.
(ii) all A_i belong to $\cup \{ H_j \mid j \leq r \}$
(iii) all B_i belong to $\cup \{ H_j \mid j < r \}$

The intended semantics of a locally stratified disjunctive program is captured by its perfect models [Pr1,Pr2]. A more general approach to semantics of logic programs is the stable model semantics [GL] which coincides with the perfect model semantics in the class of locally stratified programs [GL]. Since the definition of stable model semantics is simpler than that of perfect model semantics, we choose to work with the former in this paper.

Let M be a Herbrand interpretation of P. The Gelfond-Lifschitz transformation of P wrt M is the program $GL(P,M) = \{ head(C) \leftarrow pos(C) \mid C \in G_P \text{ and } neg(C) \cap M = \phi \}$. M is a stable model of P iff M is a minimal model of GL(P,M) [Pr2,GL].

We introduce now the acyclic disjunctive programs.

Definition

A disjunctive program P is acyclic if it is possible to decompose the Herbrand base of P into disjoint sets, called strata $H_0, H_1, \dots, H_i, ..$ where i is a natural number so that for each ground clause in G_P

$$C_1 \vee .. \vee C_k \leftarrow A_1, \dots, A_n, \neg B_1, \dots, \neg B_m$$

(i) all C_i belong to the same stratum, say H_r.

(ii) all A_i and B_i belong to $\cup \{ H_j \mid j < r \}$

//

Since acyclic programs are locally stratified, their intended semantics is the perfect model semantics.

3. Transforming Acyclic Disjunctive Programs into Normal Programs

Let us introduce some new notations. Let D be a disjunction of atoms. D is canonical if the atoms in D are pairwise different. For each disjunction D, the canonical form of D, denoted by can(D), is a disjunction containing only distinct atoms in D and is equivalent to D. A disjunction D' is a factor of D with most general unifier (mgu) Θ if D' is can(D) and Θ is the identity substitution or there are two or more unifiable atoms in D with mgu Θ and D' is can(DΘ). For example, the disjunction p(x,a) v p(b,y) has two factors: one is the disjunction itself and the other is p(b,a) with the mgu {b/x,a/y}.

The normal form of P, written N(P), is constructed as follows:

Let $C: A_1 \vee ... \vee A_n \leftarrow L_1, \dots, L_m$. *Define*

$$N(C) = \{ A \leftarrow A_1', .., A_k', L_1\Theta, .., L_m\Theta \mid A \vee A_1' \vee .. \vee A_k' \text{ is a factor of } A_1 \vee .. \vee A_n \text{ with mgu } \Theta \}$$

$$N(P) = \cup \{ N(C) \mid C \in P \}$$

Example P: p(x,a) v p(b,y) <-

N(P): p(x,a) <- $\neg$ p(b,y)
p(b,y) <- $\neg$ p(x,a)
p(b,a) <-

//

It has been showed [DK] that each minimal Herbrand model of a positive disjunctive programs is a model of the Clark's completion of N(P). In this chapter, we are interested in the more general question about the relationship between the stable models of P and N(P).

The following theorem shows the equivalence between P and N(P) for acyclic disjunctive programs.

Theorem 1 Let P be an acyclic disjunctive program P, and M be a Herbrand interpretation of P. Then M is a stable model of P iff M is a stable model of N(P).

Proof "=>" Let $Q = GL(G_P,M)$. Since M is a stable model of P, M is a minimal model of Q. Since M is a minimal model of Q, for each $A \in M$, there is a clause $A \vee A_1 \vee .. \vee A_n \leftarrow$ Body in Q such that for each i: $A_i \notin M$ and Body is true in M. Hence, for each $A \in M$, there is a clause $A \leftarrow$ Body' in $G_{N(P)}$ such that Body' is true in M. Thus, there exists a clause C' in $GL(G_{N(P)},M)$ such that head(C')=A and body(C') is true in M. Since P is acyclic, $GL(G_{N(P)},M)$ is acyclic, too. It follows, that M is the least Herbrand model of $GL(G_{N(P)},M)$. So M is a stable model of N(P).

"<=" Let M be a stable model of N(P). Since $GL(G_{N(P)},M) = GL(N(GL(G_P,M)),M)$, M is also a stable model of $N(GL(G_P,M))$. Thus M is a minimal model of $GL(G_P,M)$. Hence M is a stable model of P.

//

Corollary Let P be an acyclic disjunctive program, N(P) be its normal form. Then a Herbrand interpretation M is a perfect model of P iff M is a stable model of N(P).

//

The following example shows that in general, the above theorem does not hold.

Example Let P:

a <- b
b <- a
a v b

N(P):

a <- b
b <- a
a <- ¬ b
b <- ¬ a

It is clear that P is not acyclic. It is easy to see that N(P) has no stable model while the unique minimal model of P is {a,b}.

//

Since each locally stratified disjunctive program posseses at least one perfect model [Pr1,Pr2], it is obvious that there exists at least one stable model for N(P). So

Corollary If P is acyclic, then N(P) posseses at least one stable model.

//

The following theorems give important characterizations of the normal form of a acyclic disjunctive program.

Theorem 2 Let P be an acyclic disjunctive program. Then each stable model of N(P) is a Herbrand model of comp(N(P) and vice versa where comp(N(P)) denotes the Clark's predicate completion [Cla,Llo] of N(P).

//

Theorem 3 The three-valued semantics and the two-valued semantics of comp(N(P)) are equivalent in the sense that each three-valued model of comp(N(P)) can be extended into an two-valued one.

//

Let L be a ground literal. We say that L holds with respect to the stable semantics of P, written P $\models_s$ L, if L is true in each stable model of P. We say P U {L} is **stable-consistent** if there exists one stable model of P in which L is true.

Summary

Let P be an acyclic disjunctive program, and L be a ground literal.

1) P $\models_s$ L iff N(P) $\models_s$ L.

2) P U {L} is stable-consistent iff

N(P) U {L} is stable-consistent iff

comp(N(P)) U {L} is consistent.

//

The question of basic interest to us now is:

(*) "Given an acyclic disjunctive program P and a ground literal L, is P U {L} stable-consistent ?"

Eshghi and Kowalski have developed an abductive procedure [EK,Dun] which takes as input a query G and a normal program P, and delivers as output a set of ground negative literals H such that P U H U {G} is stable-consistent. From the above obtained results, it is clear that this abductive procedure can be used as a proof procedure for the question (*).

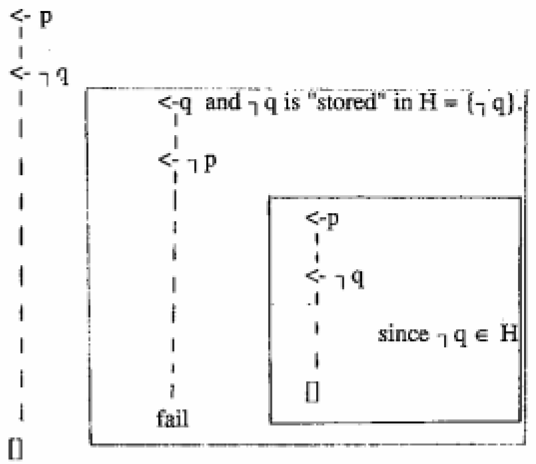
4. The Eshghi and Kowalski's Abductive Procedure

Before presenting the formal definition of the abductive procedure, let us explain the algorithm informally by an example.

Example P:

p <- ¬ q
q <- ¬ p

We want to check whether p belongs to some stable model of P, i.e. whether P U {p} is stable-consistent. It is clear that the SLDNF-resolution will not terminate for this goal due to the existence of a negative loop. To avoid getting trapped in this loop, the abductive procedure uses a loop check by "storing" all "encountered" negative literals in a set H. If a selected subgoal belongs to H, then the respected goal is simplified by deleting the selected subgoal from it.



//

Let us recall now the formal definition of the abductive procedure from [EK,Dun].

Let P be a normal logic program.

A **derivation** from (G_1,H_1) to (G_n,H_n) (wrt P) is a sequence

$$(G_1,H_1),(G_2,H_2),\ldots,(G_n,H_n)$$

such that, for each i, $1 \le i < n$, G_i has the form <–l,l' where (without loss of generality) l is selected, and l' is a (possibly empty) collection of atoms, H_i is a set of negative literals, and

ab1) If l is positive

then $G_{i+1} = C$ and $H_{i+1} = H_i$

where C is the resolvent of some clause in P with the clause G_i on the selected literal l.

ab2) If l is negative and $l \in H_i$

then $G_{i+1} =$ <–l' and $H_{i+1} = H_i$

ab3) If l is negative ($l = \neg k$) and $l \notin H_i$ and there is a consistency derivation from $(\{<\!-k\},H_i \cup \{l\})$ to (ϕ,H')

then $G_{i+1} =$ <–l' and $H_{i+1} = H'$

An **abductive refutation** is an abductive derivation to a pair ([],H).

A **consistency derivation** from (F_1,H_1) to (F_n,H_n) (wrt P) is a sequence

$$(F_1,H_1),(F_2,H_2),\ldots,(F_n,H_n)$$

such that, for each i, $0 < i \le n$, F_i has the form $\{<\!-l,l'\} \cup F_i'$, where (without loss of generality) the clause <–l,l' has been selected (to continue the search), l is selected, and

co1) If l is positive

then $F_{i+1} = C' \cup F_i'$ and $H_{i+1} = H_i$

where C' is the set of all resolvents of clauses in P with the selected clause on the selected literal, and $[] \notin C'$.

co2) If l is negative, $l \in H_i$ and l' is not empty

then $F_{i+1} = \{<\!-l'\} \cup F_i'$ and $H_{i+1} = H_i$

co3) If l is negative ($l = \neg k$), $l \notin H_i$

then if there is an abductive derivation from $(<\!-k,H_i)$ to $([],H')$

then $F_{i+1} = F_i'$ and $H_{i+1} = H'$

else if l' is not empty

then $F_{i+1} = \{<\!-l'\} \cup F_i'$ and $H_{i+1} = H_i$

A consistency derivation of the goal $(\{G\},\phi)$ is a sequentialization of the search tree of G. This sequentialization is necessary because of the need to accumulate the hypotheses found during this process.

We say that the abductive procedure is **sound** with respect to the stable semantics if whenever there exists a refutation from $(<\!-A,\phi)$ to $([],H)$ for $A \in HB$ then there exists a stable model M such that $A \in M$ and $H \cap M = \phi$.

We say that the abductive procedure is **complete** with respect to the stable semantics if for each ground literal L, if $P \cup \{L\}$ is stable-consistent then there exists a refutation for the goal $(<\!-L,\phi)$.

Note that in general, the abductive procedure is not sound with respect to the stable semantics, but it is sound with respect to the preferential semantics which is a generalization of stable semantics [Dun]. But since these two semantics coincide for programs N(P) where P is a acyclic disjunctive programs, the soundness with respect to the stable semantics follows directly from the soundness with respect to the preferential semantics.

Theorem 4 (Soundness of the Abductive Procedure)

Let P be an acyclic disjunctive program and (<-A,ϕ),..,([],H) be a refutation with respect to the program N(P). Then there exists a stable model M of P such that A ∈ M and H ∩ M = ϕ.

Proof (Sketch) Let H_0,..,H_i,... be the strata of P. Let P_i consist of those clauses A_1 v ...v A_n <- Bd in G_P such that all A_j belong to H_i. By induction, we can prove that for each i, the stable semantics and preferential semantics [Dun] of P_i coincide. It follows then that the stable and preferential semantics of P coincide. The theorem follows immediately from the fact that the abductive procedure is sound wrt preferential semantics [Dun].
//

Using Abductive Procedure For Skeptical Reasoning

The question of this chapter is:

> "Given a logic program P and a ground literal L, does L hold with respect to the stable semantics of P ?"

The following lemma shows that if the abductive procedure is complete, then it can be used to as a proof procedure for skeptical reasoning.

Lemma Let L be a ground literal and assume that the abductive procedure is sound and complete with respect to the stable semantics.

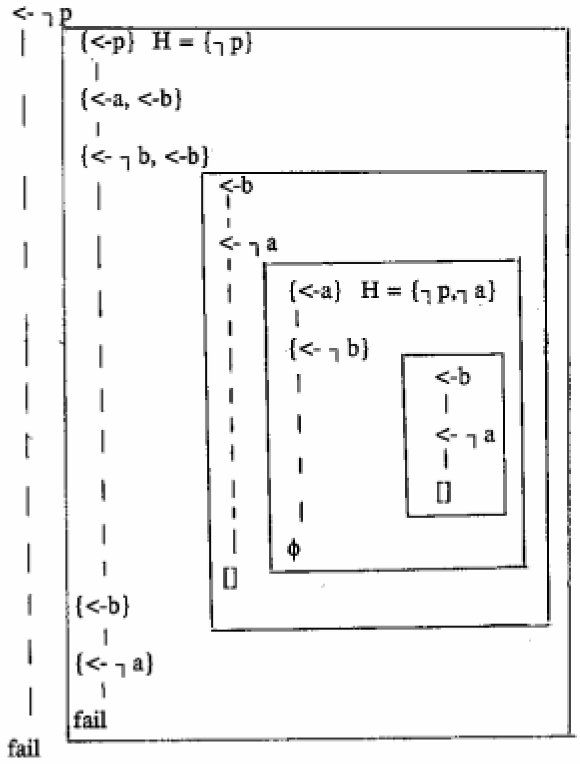
If there exists no refutation for (<- ┐L,ϕ) then P $\models_s$L.

If the abductive procedure terminates for ground goals, then it is decidable whether an arbitrary ground literal L holds with respect to the stable semantics.

//

Example p <- a
p <- b
a <- ┐b
b <- ┐a

Since the abductive procedure is complete for this program, the above lemma can be used to check whether p holds wrt stable semantics.



As there is no refutation for (<- ┐p,ϕ), P $\models_s$ p.
//

The applicability of the abductive procedure as a proof procedure for skeptical reasoning is based on its completeness. In the following paragraph, sufficient conditions for the completeness of abductive procedure are given.

5. Completeness and Termination of the Abductive Procedure

A normal program is said to be **positive acyclic**, written **p-acyclic**, if there is a level mapping |.| assigning each atom A ∈ HB_P a natural number |A| such that for each clause C in G_P, for each atom A occurring in the head of C and each atom B occurring positively in the body of C, |A| > |B|.

It is not difficult to see that if P is an acyclic disjunctive program then N(P) is always p-acyclic. Note that positive acyclicity is different to local stratifiability, i.e. there exists programs which are p-acyclic and not locally stratified and vice versa.

The atom dependency graph of P is a graph with ground atoms as its nodes such that there exists a positive (resp.

negative) edge from A to B if A occurs in the head, and B occurs positively (resp. negatively) in the body of some clause C in G_P.

An infinite path $(A_1,..,A_n,..)$ of pairwise different atoms in the atom dependency graph of P is said to be a **negative infinite loop** if the path contains infinitely many negative edges. P is said to be **free of infinite negative loop, written INL-free**, if there exists no negative infinite loop in the atom dependency graph of P.

A program P is **allowed** [Llo] if each clause in P satisfies the condition that each variable appearring in the clause appears also in a positive subgoal in the clause body.

Theorem 5 (Completeness of the Abductive Procedure)

Let P be an allowed, p-acyclic, and NIL-free normal program, and L be an arbitrary ground literal. Then the abductive procedure will terminate for the goal $(<\!-L,\phi)$, and if P U {L} is stable-consistent then there exists a refutation from $(<\!-L,\phi)$ to ([],H).
//

Let us specify now the class of disjunctive programs such that their normal form N(P) are INL-free. Two disjunctions of atoms A_1 v..v A_n and B_1 v...v B_m are said to be **related** if they have some atom in common. A sequence of disjunctions $D_1,...,D_n,...$ is said to be a **related sequence** if D_i,D_{i+1} are related for each i. A related sequence of disjunctions $D_1,...,D_n,..$ is said to be **prime** if for each i, there exists a common atom A_i in D_i and D_{i+1} such that the sequence $A_1,..,A_n,..$ contains no atom twice. A disjunctive program is said to be **free of prime related sequence (abbreviated as PRS-free)** if no prime related consequence can be built from the instances of the heads of the program clauses of P.

Corollary If P is an allowed, acyclic, PRS-free disjunctive program then the abductive procedure, applied to N(P), is sound and complete wrt perfect model semantic of P.

//

Acknowledgements

First of all, I wish to express my sincere thanks to Robert Kowalski for his generous support as well as for the many spiritful and insightful discussions. The long discussions with Paolo Mancarella, and Tony Kakas on how to find an extension of the abductive procedure which is complete with respect to the preferential semantics, are very helpful. So a lot of thanks to them.

References

[ABW] Apt K., Blair H., Walker A.
'Toward a Theory of Declarative Knowledge'
In Foundations of Deductive Databases & Logic Programming, J. Minker (ed.) 1988

[AB] Apt K., Bezem M.
'Acyclic Programs',
In Proceedings of the ICLP-90, Israel, MIT Press

[Bez] Bezem M.
'Characterizing Termination of logic programs with level mappings',
In Proceedings of the NACLP-89, USA, MIT Press

[Cav] Cavedon L.
'Continuity, consistency and completeness properties of logic programs',
In ICLP 1987, Lisbon, MIT Press

[Cla] Clark,K.L.
'Negation as Failure',
in Logic and Database, Gallaire H., Minker J. (eds), Plenum, New York,1978

[Dun] Dung P.M.
'Negations as hypotheses: an abductive foundation for logic programming'
In Proceedings of Eighth ICLP, 1991, Paris, MIT Press

[DK] Dung P.M., Kanchanasut K.,
'On the generalized predicate completion of non-Horn programs',
In Proc. of NACLP-89, USA, MIT Press

[EK] Eshghi K., Kowalski R.A.
'Abduction Compared with Negation by Failure'
In Proc. of 6th ICLP, 1989

[GL] Gelfond M.,Lifschitz V.
'The stable model semantics for logic programs'
In Proc. of the 5th Int Conf/Sym on Logic Programming, MIT Press, 1988

[HP] Henshen L., Park H.
'Compiling GCWA in indefinite deductive databases',
In Foundation in Deductive Databases and Logic Programming, J. Minker (ed.)

[Kow] Kowalski R.A.
'Logic for problem solving'
Elsvier North Holland, New York, 1979

[Llo] Lloyd J.W.
'Foundations of Logic Programming',
second edition, Springer Verlag, 1987

[Lob] Lobo J.
'Semantics for normal disjunctive logic programs'
PhD thesis, 1991

[SL] Smith B.T., Loveland D.
'A simple near Horn prolog interpreter'
Proc. of the fifth joint conference on logic programming, 1988, USA

[MR] Minker J., Rajasekar A.
'A fixpoint semantics for disjunctive logic programming',
In Journal of Logic programming, to appear

[Pr1] Przymusinski T.C. ,
'On the Declarative Semantics of Deductive Databases and Logic Programs',
In Foundations of Deductive Databases & Logic Programming, J. Minker (ed.) 1988

[Pr2] Przymusinski T.C.
'Extended stable semantics for normal and disjunctive programs',
In Proc. of seventh ICLP, Israel, 1990

[RT] Ross K.A., Topor R.W.
'Inferring negative atoms from dsijunctive databases'
Journal of Automated reasoning, Dec. 1988

[SS] Sakama C., Seki H.
'Possibel model semantics for disjunctive databases'
Preliminary report, ICOT