Partial Deduction of Disjunctive Logic Programs: A Declarative Approach*

Chiaki Sakama[1] and Hirohisa Seki[2]

[1] ASTEM Research Institute of Kyoto
17 Chudoji Minami-machi, Shimogyo, Kyoto 600 Japan
sakama@astem.or.jp
[2] Department of Artificial Intelligence and Computer Science
Nagoya Institute of Technology, Gokiso, Showa-ku, Nagoya 466 Japan
seki@ics.nitech.ac.jp

Abstract. This paper presents a partial deduction method for disjunctive logic programs. We first show that standard partial deduction in logic programming is not applicable as it is in the context of disjunctive logic programs. Then we introduce a new partial deduction technique for disjunctive logic programs, and show that it preserves the minimal model semantics of positive disjunctive programs, and the stable model semantics of normal disjunctive programs. Goal-oriented partial deduction is also presented for query optimization.

1 Introduction

Partial deduction or *partial evaluation* is known as one of the optimization techniques in logic programming. Given a logic program, partial deduction derives a more specific program through performing deduction on a part of the program, while preserving the meaning of the original program. Such a specialized program is usually more efficient than the original program when executed.

Partial deduction in logic programming was firstly introduced by Komorowski [Kom81] and has been developed by several researchers from various viewpoints (for an introduction and bibliographies, see [Kom92] and [SZ88], for example). From semantic points of view, Lloyd and Shepherdson [LS91] formalized partial evaluation for normal logic programs and showed its correctness with respect to Clark's program completion semantics. On the other hand, Tamaki and Sato [TS84] showed that partial deduction preserves the least Herbrand model semantics of definite logic programs in the context of unfold/fold transformation. The result is extended by Seki to the perfect model semantics for stratified logic

* In *Proceedings of the 4th International Workshop on Logic Program Synthesis and Transformation (LOPSTR'94), Lecture Notes in Computer Science* 883, Springer-Verlag, pp. 170-182, 1994.

programs [Seki91], and the well-founded semantics for normal logic programs [Seki93].

Recent studies of logic programming extended its framework to include indefinite information in a program. *Disjunctive logic programs* are such extensions of logic programming, which possibly include disjunctive clauses in programs. A disjunctive logic program enables us to reason with indefinite information in a program, and its growing importance in logic programming and artificial intelligence is recognized these days. Disjunctive logic programs increase expressiveness of logic programming on the one hand, but their computation is generally expensive on the other hand. Then optimizations of disjunctive logic programs are important issues for practical usage, however, there have been few studies on the subject.

In this paper, we develop partial deduction techniques for disjunctive logic programs. We first show that standard partial deduction is not useful in the presence of disjunctive information in a program, and introduce new partial deduction for disjunctive logic programs. We prove that the proposed partial deduction method preserves the minimal model semantics of positive disjunctive programs, and the disjunctive stable model semantics of normal disjunctive programs.

The rest of this paper is organized as follows. In Section 2, we introduce notations of disjunctive logic programs. In Section 3, we present new partial deduction for positive disjunctive programs and show its correctness with respect to the minimal model semantics. Section 4 extends the result to normal disjunctive programs containing negation as failure, and shows that proposed partial deduction also works well for the disjunctive stable model semantics. Section 5 discusses some connections between normal partial deduction and the proposed one. In Section 6, partial deduction techniques are applied to goal-oriented partial deduction for query optimization. Section 7 summarizes the paper and addresses future work.

2 Disjunctive Logic Programs

A *normal disjunctive program* is a finite set of clauses of the form:

$$A_1 \vee \ldots \vee A_l \leftarrow B_1 \wedge \ldots \wedge B_m \wedge not\, B_{m+1} \wedge \ldots \wedge not\, B_n \quad (l \geq 0,\, n \geq m \geq 0) \quad (1)$$

where A_i's and B_j's are atoms and *not* denotes the *negation-as-failure* operator. The left-hand side of the clause (1) is called the *head*, while the right-hand side of the clause is called the *body*. The clause (1) is called *disjunctive* (resp. *normal*) if $l > 1$ (resp. $l = 1$). Else if $l = 0$ and $n \neq 0$, it is called an *integrity constraint*. A normal disjunctive program containing no *not* is called a *positive*

disjunctive program, while a program containing no disjunctive clause is called a *normal logic program*. A normal logic program containing no *not* is called a *Horn logic program*, and a Horn logic program without integrity constraints is called a *definite logic program*.

In this paper, when we write $A \vee \Sigma \leftarrow \Gamma$, Σ denotes a disjunction (possibly *false*) in the head, and Γ denotes a conjunction (possibly *true*) in the body.[3]

In logic programming, a program is semantically identified with its *ground program*, which is the set of all ground clauses from the program. Then we consider ground programs throughout this paper. We also assume without loss of generality that a disjunction in the head of a ground clause is already *factored*, that is, each atom in the disjunctive head of a clause is different.

An *interpretation* of a program is a subset of the Herbrand base of the program. For a positive disjunctive program P, an interpretation I is called a *minimal model* of P if there is no smaller interpretation J satisfying the program. A program is *consistent* if it has a minimal model, otherwise a program is *inconsistent*. The *minimal model semantics* [Min82] of a positive disjunctive program P is defined as the set of all minimal models of P (denoted by $\mathcal{MM}_P$).

For a normal disjunctive program P, an interpretation I is called a *stable model* of P if I coincides with a minimal model of the positive disjunctive program P^I obtained from P as follows:

$$P^I = \{A_1 \vee \ldots \vee A_l \leftarrow B_1 \wedge \ldots \wedge B_m \mid \text{ there is a ground clause of the form (1) from } P \text{ such that } \{B_{m+1}, \ldots, B_n\} \cap I = \emptyset \}.$$

A normal disjunctive program has no, one, or multiple stable models in general. A program which has no stable model is called *incoherent*.

The *disjunctive stable model semantics* [Prz91] of a normal disjunctive program P is defined as the set of all stable models of P (denoted by $\mathcal{ST}_P$). The disjunctive stable model semantics coincides with Gelfond and Lifschitz's stable model semantics [GL88] in normal logic programs.

3 Partial Deduction of Positive Disjunctive Programs

Partial deduction in logic programming is usually defined as *unfolding* of clauses in a program.[4] For a Horn logic program P, partial deduction is formally presented as follows.

[3] When we write a clause as $A \vee \Sigma \leftarrow \Gamma$, it does not necessarily mean that A should be the leftmost atom in the head of the clause. That is, any two clauses are identified modulo the permutation of disjuncts/conjuncts in their heads/bodies.

[4] Partial deduction is also called partial evaluation. However, we prefer to use the term partial deduction since partial evaluation often includes non-deductive procedures.

Given a Horn clause C from P:

$$C:\ H \leftarrow A \wedge \Gamma\,,$$

suppose that $C_1, \ldots, C_k$ are all of the clauses in P such that each of which has the atom A in its head:

$$C_i:\ A \leftarrow \Gamma_i \quad (1 \leq i \leq k)\,.$$

Then *normal partial deduction* of P (with respect to C on A) is defined as the program $\pi^N_{\{C;A\}}(P)$ (called a *residual program*) such that

$$\pi^N_{\{C;A\}}(P) = (P \setminus \{C\}) \cup \{C'_1, \ldots, C'_k\}$$

where each C'_i is defined as

$$C'_i:\ H \leftarrow \Gamma \wedge \Gamma_i\,.$$

When we simply say normal partial deduction of P (written $\pi^N(P)$), it means normal partial deduction of P with respect to any clause on any atom.

Example 3.1 Let P be the program:

$$P = \{\ a \leftarrow b, \quad b \leftarrow c, \quad b \leftarrow a, \quad c \leftarrow\ \}.$$

Then normal partial deduction of P with respect to $a \leftarrow b$ on b becomes

$$\pi^N_{\{a \leftarrow b;b\}}(P) = \{\ a \leftarrow c, \quad a \leftarrow a, \quad b \leftarrow c, \quad b \leftarrow a, \quad c \leftarrow\ \}. \quad \square$$

In the context of unfold/fold transformation of logic programs, Tamaki and Sato [TS84] showed that normal partial deduction preserves the least Herbrand model semantics of definite logic programs.

Lemma 3.1 ([TS84]) Let P be a definite logic program and M_P be its least Herbrand model. Then, for any residual program $\pi^N(P)$ of P, $M_P = M_{\pi^N(P)}$. $\square$

The result also holds for Horn logic programs, that is, programs containing integrity constraints.

Theorem 3.2 Let P be a Horn logic program and $\pi^N(P)$ be any residual program of P. Then $M_P = M_{\pi^N(P)}$.

Proof. By identifying each integrity constraint $\leftarrow G$ with $false \leftarrow G$, M_P contains $false$ iff $M_{\pi^N(P)}$ contains $false$. In this case, both programs are inconsistent. Then the result follows from Lemma 3.1.

Thus, in what follows we do not take special care for the treatment of integrity constraints, that is, they are identified with normal clauses during partial deduction as presented above.

Now we consider partial deduction in disjunctive logic programs. If we consider to extend normal partial deduction to a program possibly containing disjunctive clauses, however, normal partial deduction does not preserve the minimal models of the program in general.

Example 3.2 Let P be the program:

$$P = \{\ a \vee b \leftarrow,\quad a \leftarrow d,\quad c \leftarrow a\ \}$$

where $\mathcal{MM}_P = \{\{a, c\}, \{b\}\}$. On the other hand,

$$\pi^N_{\{c \leftarrow a; a\}}(P) = \{\ a \vee b \leftarrow,\quad a \leftarrow d,\quad c \leftarrow d\ \}$$

where $\mathcal{MM}_{\pi^N_{\{c \leftarrow a; a\}}(P)} = \{\{a\}, \{b\}\}$. $\square$

The problem is that normal partial deduction of logic programs is defined as unfolding between normal clauses. In the above example, however, there is the disjunctive clause $a \vee b \leftarrow$ containing the atom a in its head, so unfolding between $c \leftarrow a$ and $a \vee b \leftarrow$ would be needed.

Then our first task is to extend the normal partial deduction method to the one which supplies unfolding for disjunctive clauses.

Definition 3.1 Let P be a positive disjunctive program and C be a clause in P of the form:

$$C:\ \Sigma \leftarrow A \wedge \Gamma\,. \tag{2}$$

Suppose that $C_1, \ldots, C_k$ are all of the clauses in P such that each of which includes the atom A in its head:

$$C_i:\ A \vee \Sigma_i \leftarrow \Gamma_i \quad (1 \leq i \leq k)\,. \tag{3}$$

Then *disjunctive partial deduction* of P (with respect to C on A) is defined as the program $\pi^D_{\{C;A\}}(P)$ (called a *residual program*) such that

$$\pi^D_{\{C;A\}}(P) = (P \setminus \{C\}) \cup \{C'_1, \ldots, C'_k\}$$

where each C'_i is defined as

$$C'_i:\ \Sigma \vee \Sigma_i \leftarrow \Gamma \wedge \Gamma_i\,, \tag{4}$$

in which $\Sigma \vee \Sigma_i$ is factored. $\square$

Disjunctive partial deduction is a natural extension of normal partial deduction. In fact, the clause (4) is a resolvent of the clauses (2) and (3). In Horn logic programs, disjunctive partial deduction coincides with normal partial deduction.

Now we show that disjunctive partial deduction preserves the minimal model semantics of positive disjunctive programs. We first present a preliminary lemma.

Lemma 3.3 Let P be a positive disjunctive program and M be its minimal model. Then an atom A is in M iff there is a clause $C: A \vee \Sigma \leftarrow \Gamma$ in P such that $M \setminus \{A\} \models \Gamma$ and $M \setminus \{A\} \not\models \Sigma$.

Proof. ($\Rightarrow$) Suppose that for some atom A in M, there is no clause C in P such that $M \setminus \{A\} \models \Gamma$ and $M \setminus \{A\} \not\models \Sigma$. Then, for each clause C, $M \setminus \{A\} \not\models \Gamma$ or $M \setminus \{A\} \models \Sigma$, and hence it holds that $M \setminus \{A\} \models \Gamma$ implies $M \setminus \{A\} \models \Sigma$. In this case, since $M \setminus \{A\}$ satisfies each clause C, it becomes a model of P, which contradicts the assumption that M is a minimal model. Hence the result follows.

($\Leftarrow$) Assume that A is not in M. Then $M \setminus \{A\} = M$, and for a clause C in P, $M \models \Gamma$ and $M \not\models \Sigma$ imply $A \in M$, contradiction.

Theorem 3.4 Let P be a positive disjunctive program and $\pi^D(P)$ be any residual program of P. Then $\mathcal{MM}_P = \mathcal{MM}_{\pi^D(P)}$.

Proof. ($\subseteq$) Let M be a minimal model of P. Since the clause (4) is a resolvent of the clauses (2) and (3) in P, M also satisfies each clause (4) in $\pi^D(P)$. Then M is a model of $\pi^D(P)$. Assume that there is a minimal model N of $\pi^D(P)$ such that $N \subset M$. Since N is not a model of P, N does not satisfy the clause (2). Then $N \models \Gamma$, $N \models A$, and $N \not\models \Sigma$. As a minimal model N of $\pi^D(P)$ implies A, it follows from Lemma 3.3 that there is a clause C of the form (3) or (4) in $\pi^D(P)$ such that C contains A in its head. (i) Suppose first that C is of the form (3). Then $N \models A$ implies $N \setminus \{A\} \models \Gamma_i$ and $N \setminus \{A\} \not\models \Sigma_i$ (by Lemma 3.3). Here $N \setminus \{A\} \models \Gamma_i$ implies $N \models \Gamma_i$. Besides, the disjunctive head $A \vee \Sigma_i$ is assumed to be already factored, then Σ_i does not include A. Thus $N \setminus \{A\} \not\models \Sigma_i$ also implies $N \not\models \Sigma_i$. In this case, however, N does not satisfy the clause (4). This contradicts the assumption that N is a model of $\pi^D(P)$. (ii) Next suppose that C is of the form (4) such that $\Sigma = A \vee \Sigma'$. Then $N \models A$ implies $N \models \Sigma$, which contradicts the fact $N \not\models \Sigma$. Hence, M is also a minimal model of $\pi^D(P)$.

($\supseteq$) Let M be a minimal model of $\pi^D(P)$. If M is not a model of P, M does not satisfy the clause (2). In this case, $M \not\models \Sigma$, $M \models A$, and $M \models \Gamma$. Since a minimal model M of $\pi^D(P)$ implies A, it follows from Lemma 3.3 that there is a clause C of the form (3) or (4) in $\pi^D(P)$ such that C contains A in its head. When C is of the form (3), $M \models A$ implies $M \models \Gamma_i$ and $M \not\models \Sigma_i$ (by Lemma 3.3 and the discussion presented above). Thus M does not satisfy the corresponding clause (4), which contradicts the assumption that M is a model of $\pi^D(P)$. Else

when C is of the form (4) such that $\Sigma = A \vee \Sigma'$, $M \models A$ implies $M \models \Sigma$, which contradicts the fact $M \not\models \Sigma$. Hence M is a model of P. Next assume that there is a minimal model N of P such that $N \subset M$. By $(\subseteq)$, N is also a minimal model of $\pi^D(P)$, but this is impossible since M is a minimal model of $\pi^D(P)$.

Corollary 3.5 Let P be a positive disjunctive program. Then P is inconsistent iff $\pi^D(P)$ is inconsistent. □

Example 3.3 (cont. from Example 3.2) Given the program P, its disjunctive partial deduction $\pi^D_{\{c \leftarrow a;a\}}(P)$ becomes

$$\pi^D_{\{c \leftarrow a;a\}}(P) = \{\ a \vee b \leftarrow,\quad a \leftarrow d,\quad c \leftarrow d,\quad b \vee c \leftarrow\ \},$$

and $\mathcal{MM}_{\pi^D_{\{c \leftarrow a;a\}}(P)} = \{\{a, c\}, \{b\}\}$, which is exactly the same as $\mathcal{MM}_P$. □

4 Partial Deduction of Normal Disjunctive Programs

In this section, we extend disjunctive partial deduction to normal disjunctive programs.

The definition of disjunctive partial deduction of normal disjunctive programs is the same as Definition 3.1, except that in this case each clause possibly contains negation as failure.

Example 4.1 Let P be the normal disjunctive program:

$$P = \{\ a \vee b \leftarrow not\, c,\quad a \leftarrow d,\quad c \leftarrow a\ \}.$$

Then disjunctive partial deduction of P with respect to $c \leftarrow a$ on a becomes

$$\pi^D_{\{c \leftarrow a;a\}}(P) = \{\ a \vee b \leftarrow not\, c,\quad a \leftarrow d,\quad c \leftarrow d,\quad b \vee c \leftarrow not\, c\ \}.\quad \square$$

As shown in the above example, disjunctive partial deduction is not affected by the presence of negation as failure in a program. Thus we can directly apply previously defined disjunctive partial deduction to normal disjunctive programs and the following result holds.

Theorem 4.1 Let P be a normal disjunctive program. Then $\mathcal{ST}_P = \mathcal{ST}_{\pi^D(P)}$.

Proof. Let M be a stable model of P. Then M is a minimal model of P^M. Since P^M is a positive disjunctive program, by Theorem 3.4, M is also a minimal model of $\pi^D(P^M)$. Now let us consider the clauses:

$$\Sigma \leftarrow A \wedge \Gamma \wedge not\, \Gamma' \quad (*)$$

and

$$A \vee \Sigma_i \leftarrow \Gamma_i \wedge not\, \Gamma'_i \quad (1 \leq i \leq k) \quad (\dagger)$$

in P, where $not\ \Gamma'$ is the conjunction of negation-as-failure formulas in the body.

(i) If $M \not\models \Gamma'$ and $M \not\models \Gamma'_i$ for some i $(1 \leq i \leq k)$, the clauses:

$$\Sigma \leftarrow A \wedge \Gamma \quad (*')$$

and

$$A \vee \Sigma_i \leftarrow \Gamma_i \quad (\dagger')$$

are in P^M. From these clauses, disjunctive partial deduction generates the clauses:

$$\Sigma \vee \Sigma_i \leftarrow \Gamma \wedge \Gamma_i \quad (\ddagger')$$

in $\pi^D(P^M)$. On the other hand, from $(*)$ and $(\dagger)$ in P, there are the clauses:

$$\Sigma \vee \Sigma_i \leftarrow \Gamma \wedge \Gamma_i \wedge not\ \Gamma' \wedge not\ \Gamma'_i \quad (\ddagger)$$

in $\pi^D(P)$, which become $(\ddagger')$ in $\pi^D(P)^M$.

(ii) Else if $M \models \Gamma'$ or $M \models \Gamma'_i$ for any i $(1 \leq i \leq k)$, the clauses $(*)$ or $(\dagger)$ is respectively eliminated in P^M. Then the clauses $(\ddagger')$ are not included in $\pi^D(P^M)$. In this case, each clause $(\ddagger)$ in $\pi^D(P)$ is also eliminated in $\pi^D(P)^M$.

Thus, there is a one-to-one correspondence between the clauses in $\pi^D(P^M)$ and the clauses in $\pi^D(P)^M$, hence $\pi^D(P^M) = \pi^D(P)^M$. Therefore M is also a minimal model of $\pi^D(P)^M$, and a stable model of $\pi^D(P)$.

The converse is also shown in the same manner.

Corollary 4.2 Let P be a normal disjunctive program. Then P is incoherent iff $\pi^D(P)$ is incoherent. □

The above theorem also implies that in normal logic programs, normal partial deduction preserves Gelfond and Lifschitz's stable model semantics.

Corollary 4.3 Let P be a normal logic program. Then $\mathcal{ST}_P = \mathcal{ST}_{\pi^N(P)}$. □

The above result is also presented in [Seki90].

5 Connections between Normal and Disjunctive Partial Deduction

In this section, we consider connections between normal and disjunctive partial deduction. We first give a sufficient condition such that normal partial deduction preserves the meaning of disjunctive logic programs.

Theorem 5.1 Let P be a normal disjunctive program and C be a clause of the form $\Sigma \leftarrow A \wedge \Gamma$ from P. If A does not appear in the head of any disjunctive clause in P, then $\mathcal{ST}_P = \mathcal{ST}_{\pi^N_{\{C;A\}}(P)}$. That is, normal partial deduction of P with respect to C on A preserves the disjunctive stable model semantics.

Proof. In this case, disjunctive partial deduction coincides with normal one, hence the result follows from Theorem 4.1.

Next we present a method to compute disjunctive partial deduction in terms of normal partial deduction.

Definition 5.1 Let P be a normal disjunctive program. The *nlp-transformation* transforms P into the normal logic program $\eta(P)$ which is obtained from P by replacing each disjunctive clause:

$$C:\ A_1 \vee \ldots \vee A_l \leftarrow \Gamma \tag{5}$$

with l normal clauses:

$$C_i^-:\ A_i \leftarrow \Gamma \wedge A_1^- \wedge \ldots \wedge A_{i-1}^- \wedge A_{i+1}^- \wedge \ldots \wedge A_l^- \quad (1 \leq i \leq l)\,. \tag{6}$$

where each A_j^- is a new atom introduced for each A_j.
In particular, $C = C_i^-$ if $l \leq 1$. $\square$

Now we show that disjunctive partial deduction of a normal disjunctive program P with respect to a clause C is obtained through normal partial deduction of $\eta(P)$ with respect to each C_i^-. In the following, the function η^{-1} is the reverse transformation which shifts each atom A_j^- appearing in the body of each clause in a program to the atom A_i in the head of the clause. Also Σ^- means $A_1^- \wedge \ldots \wedge A_l^-$ where $\Sigma = A_1 \vee \ldots \vee A_l$.

Theorem 5.2 Let P be a normal disjunctive program. Then $\pi^D_{\{C;A\}}(P) = \eta^{-1}(\pi^N_{\{C_i^-;A\}}(\eta(P)))$ where $\pi^N_{\{C_i^-;A\}}(\eta(P))$ means normal partial deduction of $\eta(P)$ with respect to each normal clause C_i^- on A.

Proof. Corresponding to the clauses (2) and (3) in P, there are the clauses:

$$A' \leftarrow A \wedge \Gamma \wedge \Sigma'^- \quad (\text{where } \ \Sigma = \Sigma' \vee A'\ ) \quad (*)$$

and

$$A \leftarrow \Gamma_i \wedge \Sigma_i^- \quad (1 \leq i \leq k) \quad (\dagger)$$

in $\eta(P)$, respectively. Then the clauses:

$$A' \leftarrow \Gamma \wedge \Gamma_i \wedge \Sigma'^- \wedge \Sigma_i^- \quad (\ddagger)$$

are obtained from $(*)$ and $(\dagger)$ by normal partial deduction in $\eta(P)$. In this case, by the reverse transformation η^{-1}, each clause of the form $(\ddagger)$ becomes a disjunctive clause of the form (4). Hence, $\pi^D_{\{C;A\}}(P) = \eta^{-1}(\pi^N_{\{C_i^-;A\}}(\eta(P)))$.

Example 5.1 Let P be the program:

$$P = \{\ a \vee b \leftarrow,\quad a \leftarrow b,\quad b \leftarrow a\ \}.$$

Then,

$$\pi^D_{\{a \leftarrow b; b\}}(P) = \{\ a \vee b \leftarrow,\quad a \leftarrow,\quad a \leftarrow a,\quad b \leftarrow a\ \}.$$

On the other hand, the *nlp*-transformation of P becomes

$$\eta(P) = \{\ a \leftarrow b^-,\quad b \leftarrow a^-,\quad a \leftarrow b,\quad b \leftarrow a\ \},$$

and

$$\pi^N_{\{a \leftarrow b; b\}}(\eta(P)) = \{\ a \leftarrow b^-,\quad b \leftarrow a^-,\quad a \leftarrow a^-,\quad a \leftarrow a,\quad b \leftarrow a\ \}.$$

Thus,

$$\eta^{-1}(\pi^N_{\{a \leftarrow b; b\}}(\eta(P))) = \{\ a \vee b \leftarrow,\quad a \leftarrow,\quad a \leftarrow a,\quad b \leftarrow a\ \}.$$

Therefore, $\pi^D_{\{a \leftarrow b; b\}}(P) = \eta^{-1}(\pi^N_{\{a \leftarrow b; b\}}(\eta(P)))$. $\square$

The above theorem presents that disjunctive partial deduction $\pi^D_{\{C;A\}}(P)$ is obtained by the transformation sequence: $P \rightarrow \eta(P) \rightarrow \pi^N_{\{C_i^-;A\}}(\eta(P)) \rightarrow \eta^{-1}(\pi^N_{\{C_i^-;A\}}(\eta(P)))$. That is, together with the *nlp*-transformation, normal partial deduction can also be used for normal disjunctive programs.

6 Goal-Oriented Partial Deduction

In this section, we present goal-oriented partial deduction in disjunctive logic programs. Goal-oriented partial deduction specializes a program with respect to a given goal, which is useful to optimize programs for query-answering. Lloyd and Shepherdson [LS91] discuss a framework of goal-oriented partial evaluation for normal logic programs and provide conditions to assure its correctness with respect to Clark's completion semantics and SLDNF proof procedures. In our framework, goal-oriented partial deduction is presented as follows.

Let us consider a *query* of the form:

$$Q:\ Q(\mathbf{x}) \leftarrow B_1 \wedge \ldots \wedge B_m \wedge not\, B_{m+1} \wedge \ldots \wedge not\, B_n \tag{7}$$

where $Q(\mathbf{x})$ is a new atom not appearing elsewhere in a program and $\mathbf{x}$ represents variables appearing in the body of the clause.

Then, given a normal disjunctive program P, partial deduction of P with respect to Q is defined as $\pi^D_{\{Q;B_i\}}(P_Q)$ where B_i is any atom occurring positively in the body of Q and P_Q is the program $P \cup \{Q\}$. When a query contains variables, we consider partial deduction with respect to its ground instances.

An *answer* to a query is defined as a ground substitution σ for variables in $Q(\mathbf{x})$. When Q contains no variable, σ is the empty substitution.

A query Q is *true* in P under the disjunctive stable model semantics if for every stable model I of P_Q there is an answer σ such that $Q(\mathbf{x})\sigma$ is included in I. Else if for some stable model I of P_Q there is an answer σ such that $Q(\mathbf{x})\sigma$ is included in I, the query is *possibly true*. Otherwise, if there is no such answer, the query is *false*. By Theorem 4.1, the following results hold.

Theorem 6.1 Let P be a normal disjunctive program and Q be a query. Then, under the disjunctive stable model semantics,

(i) Q is true in P iff Q is true in $\pi^D_{\{Q;B_i\}}(P_Q)$.
(ii) Q is possibly true in P iff Q is possibly true in $\pi^D_{\{Q;B_i\}}(P_Q)$.
(iii) Q is false in P iff Q is false in $\pi^D_{\{Q;B_i\}}(P_Q)$. □

Example 6.1 Let P be the program:

$$\{\ p(a) \vee p(b) \leftarrow \},$$

in which the query $Q:\ q(x) \leftarrow p(x)$ is true. Then,

$$\pi^D_{\{Q;p(x)\}}(P_Q) = \{\ q(a)\vee p(b) \leftarrow,\quad q(b)\vee p(a) \leftarrow,\quad q(a)\vee q(b) \leftarrow,\quad p(a)\vee p(b) \leftarrow \}$$

and Q is also true in $\pi^D_{\{Q;p(x)\}}(P_Q)$ under the disjunctive stable model semantics. □

Note that in the above example, we assume that the ground queries $q(a) \leftarrow p(a)$ and $q(b) \leftarrow p(b)$ are unfolded consecutively in the program. That is, $\pi^D_{\{Q;p(x)\}}(P_Q)$ means $\pi^D_{\{Q;p(b)\}}(\pi^D_{\{Q;p(a)\}}(P \cup \{\ q(a) \leftarrow p(a),\ \ q(b) \leftarrow p(b)\ \}))$. In this case, the order of unfolding does not affect the result of partial deduction since each partial deduction preserves the stable models of the program P_Q.

7 Summary

This paper presented a method of partial deduction for disjunctive logic programs. We first showed that normal partial deduction is not applicable to disjunctive logic programs in its present form. Then we introduced disjunctive partial deduction for disjunctive logic programs, which is a natural extension of normal partial deduction for normal logic programs. Disjunctive partial deduction was shown to preserve the minimal model semantics of positive disjunctive programs, and the disjunctive stable model semantics of normal disjunctive programs. We also showed a method of translating disjunctive partial deduction into normal partial deduction, and presented an application to goal-oriented partial deduction for query optimization.

The partial deduction technique presented in this paper is also directly applicable to disjunctive logic programs possibly containing classical negation [GL91]. Moreover, since positive disjunctive programs are identified with first-order theories, disjunctive partial deduction has potential application to first-order theorem provers. Recently, Brass and Dix [BD94] independently developed a partial deduction technique for disjunctive logic programs which is equivalent to ours. They discuss several abstract properties of disjunctive logic programs and conclude partial deduction as one of the fundamental properties that logic programming semantics should satisfy.

In this paper, we have mainly concerned with declarative aspects of partial deduction and considered propositional programs as a first step. Then our next step is to apply the partial deduction method to programs containing variables and investigate the procedural aspect of disjunctive partial deduction.

References

[BD94] Brass, S. and Dix, J., A Disjunctive Semantics Based on Unfolding and Bottom-up Evaluation, *Proc. 13th World Computer Congress'94, IFIP, GI-Workshop W2, Disjunctive Logic Programming and Disjunctive Databases*, 1994.

[GL88] Gelfond, M. and Lifschitz, V., The Stable Model Semantics for Logic Programming, *Proc. Joint Int. Conf. and Symp. on Logic Programming*, 1070-1080, 1988.

[GL91] Gelfond, M. and Lifschitz, V., Classical Negation in Logic Programs and Disjunctive Databases, *New Generation Computing* 9, 365-385, 1991.

[Kom81] Komorowski, J., A Specification of an Abstract Prolog Machine and its Application to Partial Evaluation, Technical Report LSST 69, Linköping Univ., 1981.

[Kom92] Komorowski, J., An Introduction to Partial Deduction, *Proc. 3rd Int. Workshop on Meta-programming in Logic*, Lecture Notes in Computer Science 649, Springer-Verlag, 49-69, 1992.

[LS91] Lloyd, J. W. and Shepherdson, J. C., Partial Evaluation in Logic Programming, *J. Logic Programming* 11, 217-242, 1991.

[Min82] Minker, J., On Indefinite Data Bases and the Closed World Assumption, *Proc. 6th Int. Conf. on Automated Deduction*, Lecture Notes in Computer Science 138, Springer-Verlag, 292-308, 1982.

[Prz91] Przymusinski, T. C., Stable Semantics for Disjunctive Programs, *New Generation Computing* 9, 401-424, 1991.

[Seki90] Seki, H., A Comparative Study of the Well-Founded and the Stable Model Semantics: Transformation's Viewpoint, *Proc. Workshop on Logic Programming and Nonmonotonic Logic*, Association for Logic Programming and Mathematics Sciences Institute, Cornell University, 115-123, 1990.

[Seki91] Seki, H., Unfold/Fold Transformation of Stratified Programs, *Theoretical Computer Science* 86, 107-139, 1991.

[Seki93] Seki, H., Unfold/Fold Transformation of General Logic Programs for the Well-Founded Semantics, *J. Logic Programming* 16, 5-23, 1993.

[SZ88] Sestoft, P. and Zamulin, A. V., Annotated Bibliography on Partial Evaluation and Mixed Computation, *New Generation Computing* 6, 309-354, 1988.

[TS84] Tamaki, H. and Sato, T., Unfold/Fold transformation of Logic Programs, *Proc. 2nd Int. Conf. on Logic Programming*, 127-138, 1984.