Head-Elementary-Set-Free Logic Programs

Martin Gebser[1], Joohyung Lee[2], and Yuliya Lierler[3]

[1] Institut für Informatik, Universität Potsdam, Germany
[2] School of Computing and Informatics, Arizona State University, USA
[3] Department of Computer Sciences, University of Texas at Austin, USA
gebser@cs.uni-potsdam.de,joolee@asu.edu,yuliya@cs.utexas.edu

Abstract. The recently proposed notion of an elementary set yielded a refinement of the theorem on loop formulas, telling us that the stable models of a disjunctive logic program can be characterized by the loop formulas of its elementary sets. Based on the notion of an elementary set, we propose the notion of head-elementary-set-free (HEF) programs, a more general class of disjunctive programs than head-cycle-free (HCF) programs proposed by Ben-Eliyahu and Dechter, that can still be turned into nondisjunctive programs in polynomial time and space by "shifting" the head atoms into the body. We show several properties of HEF programs that generalize earlier results on HCF programs. Given an HEF program, we provide an algorithm for finding an elementary set whose loop formula is not satisfied, which has a potential for improving stable model computation by answer set solvers.

1 Introduction

Disjunctive logic programs under the stable model semantics are more expressive than nondisjunctive programs. The problem of deciding whether a disjunctive program has a stable model is Σ_2^P-complete [1], while the same problem for a nondisjunctive program is NP-complete.

However, Ben-Eliyahu and Dechter [2] showed that a class of disjunctive programs called "head-cycle-free (HCF)" programs can be turned into nondisjunctive programs in polynomial time and space, by "shifting" the head atoms into the body—a simple operation defined in [3]. This tells us that an HCF program is an "easy" disjunctive program, which is merely a syntactic shortcut of a nondisjunctive program. Thus, HCF programs play an important role in efficient computation of stable models for disjunctive programs. Indeed, the HCF property is exploited by answer set solvers DLV[1] [4] and CMODELS[2] [5].

In this paper, we propose the notion of head-elementary-set-free (HEF) programs, a more general class of disjunctive programs than HCF programs, that can still be turned into nondisjunctive programs in polynomial time and space by shifting. This is motivated by the recent study on elementary sets [6], which yielded a refinement of the theorem on loop formulas by Lin and Zhao [7]. All elementary sets are loops, but not all loops are elementary sets; still stable models can be characterized by elementary

[1] http://www.dbai.tuwien.ac.at/proj/dlv/
[2] http://www.cs.utexas.edu/users/tag/cmodels/

C. Baral, G. Brewka, and J. Schlipf (Eds.): LPNMR 2007, LNAI 4483, pp. 149–161, 2007.

sets' loop formulas. Our definition of an HEF program is similar to the definition of an HCF program except that the former refers to elementary sets instead of loops. We observe that some other properties of nondisjunctive programs and HCF programs can be extended to HEF programs, including the main results by Lin and Zhao [8] characterizing the stable models of a nondisjunctive program by "inherent tightness," and the operational characterization of stable models of HCF programs by Leone *et al.* [9].

The properties of HEF programs studied here may be useful for improving the computation of disjunctive answer set solvers, such as DLV and CMODELS. As a first step, we provide an algorithm for finding an elementary set whose loop formula is not satisfied for a given HEF program, which is simpler and more efficient than the algorithm described in [10].

The outline of the paper is as follows. In Section 2, we review the definition of an elementary set introduced in [6] and show some of its properties. In Section 3, we introduce the notion of HEF programs and show that shifting preserves their stable models. In Section 4, we demonstrate that the notion of inherent tightness can be generalized to HEF programs, but not to general disjunctive programs. This section also includes simplifications of earlier notions. In Section 5, we show that the operational characterization of stable models by Leone *et al.* [9] can be extended to HEF programs as well. We also define "bounding" loops that allow for enhancing the model checking approach for disjunctive programs introduced in [9,11]. In Section 6, we present an algorithm for computing an elementary set for a given HEF program.

2 Review of Elementary Sets for Disjunctive Programs

We begin with a review of elementary sets, introduced in [6], which are a reformulation and generalization of elementary loops [12].

A *disjunctive program* is a finite set of *(disjunctive) rules* of the form

$$a_1; \ldots; a_k \leftarrow a_{k+1}, \ldots, a_l, \mathit{not}\ a_{l+1}, \ldots, \mathit{not}\ a_m, \mathit{not\ not}\ a_{m+1}, \ldots, \mathit{not\ not}\ a_n \quad (1)$$

where $n \geq m \geq l \geq k \geq 0$ and $a_1, \ldots, a_n$ are propositional atoms. We will identify a rule of the form (1) with the propositional formula

$$(a_{k+1} \wedge \cdots \wedge a_l \wedge \neg a_{l+1} \wedge \cdots \wedge \neg a_m \wedge \neg\neg a_{m+1} \wedge \cdots \wedge \neg\neg a_n) \rightarrow (a_1 \vee \cdots \vee a_k)\ .$$

We will also write (1) as

$$A \leftarrow B, F \quad (2)$$

where A is $a_1; \ldots; a_k$, B is $a_{k+1}, \ldots, a_l$, and F is

$$\mathit{not}\ a_{l+1}, \ldots, \mathit{not}\ a_m, \mathit{not\ not}\ a_{m+1}, \ldots, \mathit{not\ not}\ a_n\ ,$$

and we identify A and B with their corresponding sets of atoms.

Let Π be a disjunctive program. A nonempty set X of atoms occurring in Π is called a *loop* of Π if, for all nonempty proper subsets Y of X, there is a rule (2) in Π such that $A \cap Y \neq \emptyset$ and $B \cap (X \setminus Y) \neq \emptyset$. As shown in [6], this definition of a loop is equivalent to the definition based on a positive dependency graph given in [13].

We say that a subset Y of X is *outbound* in X for Π if there is a rule (2) in Π such that $A \cap Y \neq \emptyset$, $B \cap (X \setminus Y) \neq \emptyset$, $A \cap (X \setminus Y) = \emptyset$, and $B \cap Y = \emptyset$. A nonempty set X of atoms that occur in Π is *elementary* for Π if all nonempty proper subsets of X are outbound in X for Π. It is clear that every elementary set is also a loop, but the converse does not hold. The definition of an elementary set above remains equivalent even if we restrict Y to be loops or even elementary sets.

Proposition 1. *For any disjunctive program Π and any nonempty set X of atoms that occur in Π, X is elementary for Π iff all proper subsets of X that are elementary for Π are outbound in X for Π.*

For any set Y of atoms, the *external support formula* of Y, denoted by $ES_\Pi(Y)$, is the disjunction of conjunctions $B \wedge F \wedge \bigwedge_{a \in A \setminus Y} \neg a$ for all rules (2) of Π such that $A \cap Y \neq \emptyset$ and $B \cap Y = \emptyset$.

The following proposition describes the relationship between the external support formula of an arbitrary set of atoms and the external support formulas of its subsets.

Proposition 2. *Let Π be a disjunctive program, and let X, Y, Z be sets of atoms such that $X \supseteq Y \supseteq Z$. If Z is not outbound in Y for Π and $X \models ES_\Pi(Z)$, then $X \models ES_\Pi(Y)$.*

This proposition is similar to Lemma 5 in [14], which states that $ES_\Pi(Z) \models ES_\Pi(Y)$ holds if there is no rule (2) in Π such that $A \cap Z \neq \emptyset$ and $B \cap (Y \setminus Z) \neq \emptyset$. Proposition 2 is more general in the sense that it refers to the stronger condition of "outboundness."

For any set Y of atoms, by $LF_\Pi(Y)$ we denote the following formula:

$$\bigwedge_{a \in Y} a \rightarrow ES_\Pi(Y) \, . \tag{3}$$

Formula (3) is called the *(conjunctive) loop formula* of Y for Π. Note that we still call (3) a loop formula even when Y is not a loop.

From Proposition 2, we derive the following relationship among loop formulas.

Proposition 3. *For any disjunctive program Π and any nonempty set X of atoms that occur in Π, there is a subset Y of X such that Y is elementary for Π and $LF_\Pi(Y) \models LF_\Pi(X)$.*

Proposition 3 allows us to restrict the attention to loop formulas of elementary sets only, rather than those of arbitrary sets or even loops. This yields the following theorem.

Theorem 1. *[6] For any disjunctive program Π and any model X of Π whose atoms occur in Π, the following conditions are equivalent:*

(a) X is stable for Π;[3]
(b) X satisfies $LF_\Pi(Y)$ for all nonempty sets Y of atoms occurring in Π;
(c) X satisfies $LF_\Pi(Y)$ for all loops Y of Π;
(d) X satisfies $LF_\Pi(Y)$ for all elementary sets Y of Π.

[3] For a model of Π, we will say that it is "stable for Π" if it is a stable model of Π.

3 Head-Elementary-Set-Free Logic Programs

Ben-Eliyahu and Dechter [2] defined a class of disjunctive programs called "head-cycle-free" programs that can be mapped in polynomial time and space to nondisjunctive programs, preserving the stable models. A disjunctive program Π is called *Head-Cycle-Free* (HCF) if, for every rule (2) in Π, there is no loop Y of Π such that $|A \cap Y| > 1$.

By referring to elementary sets in place of loops in the definition, we can define a class of programs that is more general than HCF programs. We will call a program Π *Head-Elementary-set-Free* (HEF) if, for every rule (2) in Π, there is no elementary set Y of Π such that $|A \cap Y| > 1$. From the fact that every elementary set is a loop, it is clear that every HCF program is an HEF program as well. However, not all HEF programs are HCF. For example, consider the following program Π_1:

$$\begin{array}{l} p \leftarrow r \\ q \leftarrow r \\ r \leftarrow p, q \\ p\,;q \leftarrow . \end{array} \tag{4}$$

The program has 6 loops, $\{p\}$, $\{q\}$, $\{r\}$, $\{p, r\}$, $\{q, r\}$, $\{p, q, r\}$. Since the head of the last rule contains two atoms from loop $\{p, q, r\}$, the program is not HCF. However, it is HEF since $\{p, q, r\}$ is not elementary for Π_1 (its subsets $\{p, r\}$ and $\{q, r\}$ are not outbound in $\{p, q, r\}$ for Π_1).

Let us write rule (2) in the following form:

$$a_1; \ldots; a_k \leftarrow B, F\ . \tag{5}$$

Gelfond *et al.* [3] defined a mapping of a disjunctive program Π into a nondisjunctive program Π_{sh}, the "shifted" variant of Π, by replacing each rule (5) with k new rules:

$$a_i \leftarrow B,\ F,\ not\ a_1, \ldots,\ not\ a_{i-1},\ not\ a_{i+1}, \ldots,\ not\ a_k\ . \tag{6}$$

They showed that every stable model of Π_{sh} is also a stable model of Π, but not vice versa. Ben-Eliyahu and Dechter [2] showed that the other direction holds as well if Π is HCF. Here we extend the result to HEF programs.

Theorem 2. *If a program Π is HEF, then Π and Π_{sh} have the same stable models.*

For instance, one can check that both Π_1 and $(\Pi_1)_{sh}$ have $\{p\}$ and $\{q\}$ as their only stable models. Theorem 2 shows that HEF programs are not more expressive than nondisjunctive programs, so that one can regard the use of disjunctive rules in such programs as a syntactic shortcut. Another consequence is that the problem of deciding whether a model is stable for an HEF program is tractable, as in the case of nondisjunctive and HCF programs. (In the general disjunctive case, it is coNP-complete [4].)

Comparing the elementary sets of Π and the elementary sets of Π_{sh} gives the following result.

Proposition 4. *For any disjunctive program Π, if X is an elementary set of Π, then X is an elementary set of Π_{sh}.*

The converse of Proposition 4 does not hold, even if Π is HEF. For instance, consider the following HEF program Π_2:

$$p\ ;\ q \leftarrow r$$
$$r \leftarrow p$$
$$r \leftarrow q\ .$$

Set $\{p, q, r\}$ is not elementary for Π_2 since, for instance, $\{p\}$ is not outbound in $\{p, q, r\}$. On the other hand, $\{p, q, r\}$ is elementary for $(\Pi_2)_{sh}$:

$$\begin{array}{l} p \leftarrow r, \mathit{not}\ q \\ q \leftarrow r, \mathit{not}\ p \\ r \leftarrow p \\ r \leftarrow q\ . \end{array} \tag{7}$$

However, there is a certain subset of Π_{sh} whose elementary sets are also elementary sets of Π. For a set X of atoms, by Π_X we denote the set of all rules in Π whose bodies are satisfied by X.

Proposition 5. *Let Π be a disjunctive program, X a set of atoms that occur in Π, and Y a subset of X. If Y is elementary for $(\Pi_{sh})_X$, then Y is elementary for Π as well.*

For $X = \{p, q, r\}$ and $(\Pi_2)_{sh}$, we have that $[(\Pi_2)_{sh}]_X$ consists of the last two rules of (7) only. Only singletons $\{p\}$, $\{q\}$, and $\{r\}$ are elementary for $[(\Pi_2)_{sh}]_X$, and they are elementary for Π_2 as well.

4 HEF Programs and Inherent Tightness

When we add more rules to a program, a stable model of the original program remains to be a stable model of the extended program as long as it satisfies the new rules.

Proposition 6. *For any disjunctive program Π and any model X of Π, X is stable for Π iff there is a subset Π' of Π such that X is stable for Π'.*

In view of Theorem 1, Proposition 6 tells us that, provided that X is a model of Π, it is sufficient to find a subset Π' of Π such that X is stable for Π', in order to verify that X is stable for Π. Of course, one can trivially take Π itself as the subset Π', but there are nontrivial subsets that deserve attention. If Π is nondisjunctive in Proposition 6, it is known that the subset Π' can be further restricted to a "tight" program [15,16]—the result known as "inherently tight", or "weakly tight" programs [8,17]. We will reformulate these results and show that they can be extended to HEF programs.

As in [13], we call a set of atoms occurring in Π *trivial* if it consists of a single atom a that has no rule (2) in Π such that $a \in A \cap B$. Recall that by Π_X we denote the set of all rules in Π whose bodies are satisfied by X.

Definition 1. *[16,13] A disjunctive program Π is called tight if every loop of Π is trivial. Program Π is called tight on a set X of atoms if every loop of Π_X is trivial.*

As defined in [18], a set X of atoms is *supported* by a nondisjunctive program Π if, for every atom $a \in X$, there is a rule (2) in Π_X such that $A = \{a\}$. We reformulate Lin and Zhao's notion of inherent tightness [8] as follows.

Definition 2. *A nondisjunctive program Π is called* inherently tight *on a set X of atoms if there is a subset Π' of Π such that Π' is tight and X is supported by Π'.*

Theorem 1 from [8] can be reformulated as follows.

Proposition 7. *For any nondisjunctive program Π and any model X of Π, X is stable for Π iff Π is inherently tight on X.*

One may wonder whether Proposition 7 can be extended to disjunctive programs as well, since the definition of a tight program (Definition 1) applies to disjunctive programs as well, and the notion of support was already extended to disjunctive programs [19,20,13]: a set X of atoms is *supported* by a disjunctive program Π if, for every atom $a \in X$, there is a rule (2) in Π_X such that $A \cap X = \{a\}$. We extend Definition 2 to disjunctive programs with these extended notions.

Unfortunately, for disjunctive programs, this straightforward extension of inherent tightness is not sufficient to characterize the stability of a model. In other words, only one direction of Proposition 7 holds for disjunctive programs.

Proposition 8. *For any disjunctive program Π and any model X of Π, if Π is inherently tight on X, then X is stable for Π.*

The following program Π_3 illustrates that the converse does not hold:

$$\begin{aligned} p\,;q &\leftarrow \\ p &\leftarrow q \\ q &\leftarrow p\,. \end{aligned}$$

Set $\{p, q\}$ is the only stable model of Π_3, but there is no subset Π' of Π_3 such that Π' is tight and $\{p, q\}$ is supported by Π'.

However, one may expect that Proposition 7 can be extended to HEF programs since, as we noted in Section 3, HEF programs are merely a syntactic shortcut of nondisjunctive programs. Indeed, the following proposition holds.

Proposition 9. *For any HEF program Π and any model X of Π, X is stable for Π iff Π is inherently tight on X.*

Since every HCF program is HEF, the proposition also holds for HCF programs.

We observed that by turning to the notion of an elementary set in place of a loop, we can get generalizations of results known for loops, such as Theorem 2 and Proposition 9. This brings our attention to the following question. Can the notion of a tight program, which is based on loops, be generalized by referring to elementary sets instead? To answer this, let us modify Definition 1 as follows.

Definition 3. *A disjunctive program Π is called* e-tight *if every elementary set of Π is trivial. Program Π is called* e-tight on *a set X of atoms if every elementary set of Π_X is trivial.*

Since every elementary set is a loop, it is clear that a tight program is e-tight as well. But is the class of e-tight programs strictly more general than the class of tight programs? The reason why this is an interesting question to consider is because, if so, it would

lead to a generalization of Fages' theorem [15], which would provide a more general class of programs for which the stable model semantics and the completion semantics coincide. However, it turns out that e-tight programs are not truly more general than tight programs.

Proposition 10. *(a) A disjunctive program is e-tight iff it is tight.*
(b) A disjunctive program is e-tight on a set X of atoms iff it is tight on X.

This result also indicates that the notion of an inherently tight program does not become more general by referring to elementary sets. That is, replacing "Π' is tight" in the statement of Definition 2 by "Π' is e-tight" does not affect the definition.

In the remainder of this section, we compare our reformulation of inherent tightness above with the original definition by Lin and Zhao.

Definition 4. *[8] A nondisjunctive program Π is called* inherently tight *on a set X of atoms if there is a subset Π' of Π such that Π' is tight on X and X is a stable model of Π'.*

There are two differences between our reformulation (Definition 2) and Definition 4. The former does not rely on the relative notion of tightness ("tight on a set of atoms") and uses a weaker condition of supportedness. Nevertheless it is not difficult to check that the two definitions are equivalent.

Proposition 7 above is a simplification of Theorem 1 from [8].

Proposition 11. *[8, Theorem 1] For any nondisjunctive program Π and any set X of atoms, X is a stable model of Π iff X is a model of the completion of Π and Π is inherently tight on X.*

Our reformulation of inherently tight programs is closely related to what Fages' called "well-supported" models [15]. We do not reproduce Fages' definition here due to lack of space, but it is not difficult to check that, for a nondisjunctive program Π and a set X of atoms, X is well-supported by Π iff Π is inherently tight on X. Proposition 7 is similar to Theorem 3.1 from [15], which showed that well-supported models coincide with stable models.

The notion of an inherently tight program is also closely related to the notion of a weakly tight program presented in [17].

5 Checking the Stability of Models for HEF Programs

The problem of deciding whether a given model is stable is coNP-complete for a disjunctive program, while it is tractable for HCF programs [9]. Leone *et al.* [9] presented an operational framework for checking the stability of a model in polynomial time for HCF programs. Given a disjunctive program Π and sets X, Y of atoms, they defined a sequence $R^0_{\Pi,X}(Y), R^1_{\Pi,X}(Y), \ldots$ that converges to a limit $R^\omega_{\Pi,X}(Y)$ as follows:

- $R^0_{\Pi,X}(Y) = Y$ and
- $R^{i+1}_{\Pi,X}(Y)$ is obtained from $R^i_{\Pi,X}(Y)$ by removing every atom a for which there is a rule (2) in Π_X such that $A \cap X = \{a\}$ and $B \cap R^i_{\Pi,X}(Y) = \emptyset$.[4]

[4] Recall that Π_X consists of all rules (2) in Π such that $X \models B, F$.

A set Y of atoms is called *unfounded* by Π w.r.t. X if $X \not\models ES_\Pi(Y)$. Set X is *unfounded-free* for Π if it contains no nonempty subset that is unfounded by Π w.r.t. X. As shown in Corollary 2 from [21] and Theorem 4.6 from [9], unfounded-free models coincide with stable models.

Proposition 6.5 from [9] shows that X is unfounded-free for Π if $R^\omega_{\Pi,X}(X) = \emptyset$. The converse also holds if Π is restricted to be a HCF program, as shown in Theorem 6.9 from the same paper. That theorem can be extended to HEF programs.[5]

Proposition 12. *For any HEF program Π and any set X of atoms, X is unfounded-free for Π iff $R^\omega_{\Pi,X}(X) = \emptyset$.*

As an example, consider again program Π_1 ((4) in Section 3), which is HEF but not HCF. Theorem 6.9 from [9] does not apply since it is limited to HCF programs. However, for set $X_1 = \{p, q, r\}$, it holds that $R^\omega_{\Pi_1,X_1}(X_1) = X_1$, and in accordance with Proposition 12, X_1 is not a stable model of Π_1. For set $X_2 = \{p\}$, the limit $R^\omega_{\Pi_1,X_2}(X_2) = \emptyset$, and X_2 is a stable model of Π_1.

The following proposition shows how the HEF property and $R^\omega_{\Pi,X}$ can be used to decide whether a set Y of atoms contains a nonempty unfounded set for Π w.r.t. X. By $\Pi_{X,Y}$ we denote the set of all rules (2) in Π_X such that $X \cap (A \setminus Y) = \emptyset$.

Proposition 13. *For any disjunctive program Π, any set X of atoms, and any subset Y of X such that $\Pi_{X,Y}$ is HEF, $R^\omega_{\Pi,X}(Y) \neq \emptyset$ iff Y contains a nonempty unfounded subset for Π w.r.t. X.*

If we replace "$R^\omega_{\Pi,X}(Y) \neq \emptyset$" by "$R^\omega_{\Pi,X}(Y) = Y$ and Y is nonempty" in Proposition 13, only the left-to-right direction still holds. In the next section, we present an algorithm based on this for finding a non-trivial unfounded set for a HEF (sub)program.

As defined in [6], we say that a set Y of atoms occurring in a disjunctive program Π is *elementarily unfounded* by Π w.r.t. a set X of atoms if

- Y is an elementary set of $\Pi_{X,Y}$ that is unfounded by Π w.r.t. X, or
- Y is a singleton that is unfounded by Π w.r.t. X.

For a model X of Π, Theorem 1(e′) from [6] states that X is stable for Π iff no subset of X is elementarily unfounded by Π w.r.t. X. Thus stability checking can be cast into a problem of ensuring the absence of elementarily unfounded sets. Since every elementarily unfounded set is a loop, every elementarily unfounded set is clearly contained in a maximal loop, which allows us to split the search for elementarily unfounded sets by maximal loops. Below we describe a notion called "bounding loops," which give tighter bounds than maximal loops. We remark that the idea of using maximal loops for partitioning the program and splitting stability checking by subprograms was already presented by Leone *et al.* [9] and Koch *et al.* [11]. Their results can be enhanced by referring to bounding loops.

For a disjunctive program Π and a set X of atoms, let S be the set of all sets Y of atoms such that Y is a loop of $\Pi_{X,Y}$ and $R^\omega_{\Pi,X}(Y) = Y$. We call a maximal element

[5] We here consider slightly more general rules than those considered in [9], since the body of a rule may contain double negation (*not not*).

of S a *bounding* loop for Π w.r.t. X. The following two propositions describe properties of bounding loops, that are similar to maximal loops used for modular stability checking.

Proposition 14. *For any disjunctive program Π and any set X of atoms, bounding loops for Π w.r.t. X are disjoint.*

Proposition 15. *For any disjunctive program Π and any set X of atoms, every non-singleton elementarily unfounded set for Π w.r.t. X belongs to a bounding loop for Π w.r.t. X.*

Clearly, every bounding loop is contained in a maximal loop. However, as shown in the example below, bounding loops provide tighter bounds than maximal loops for locating elementarily unfounded sets. Propositions 14 and 15 tell us that the process of checking the absence of elementarily unfounded sets can be split by bounding loops.

Proposition 16. *For any disjunctive program Π and any model X of Π, X is stable for Π iff X is supported by Π and X contains no bounding loop Y for Π w.r.t. X such that Y has a nonempty unfounded subset for Π w.r.t X.*

We note that computing all bounding loops for Π w.r.t. X that are contained in X can be done in polynomial time using the following method:

1. Let $Y := X$.
2. Let $Z := R^{\omega}_{\Pi,X}(Y)$. (Note that $Z = R^{\omega}_{\Pi,X}(Z)$ holds.)
3. If $Z \neq \emptyset$, then consider the following cases:
   (a) If Z is a loop of $\Pi_{X,Z}$, then mark Z as a bounding loop for Π w.r.t. X.
   (b) Otherwise, proceed with step 2 for every maximal loop Y of $\Pi_{X,Z}$ that is contained in Z.

For example, consider program Π_4,

$$\begin{array}{lll} p \leftarrow r & s\,;t \leftarrow & p\,;q \leftarrow s \\ q \leftarrow r & s \leftarrow t & t\,;u \leftarrow q \\ r \leftarrow p, q & t \leftarrow s, u & u\,;v \leftarrow , \end{array}$$

and its model $X = \{p, q, r, s, t, u\}$. It holds that $(\Pi_4)_{X,X} = \Pi_4$, and X is a maximal loop of Π_4. Note that $R^{\omega}_{\Pi_4,X}(X) = \{p, q, r, s, t\} \neq X$, so that X is not a bounding loop for Π_4 w.r.t. X. Set $Z = \{p, q, r, s, t\}$ is not a loop of $(\Pi_4)_{X,Z}$; the maximal loops of $(\Pi_4)_{X,Z}$ contained in Z are $Y_1 = \{p, q, r\}$ and $Y_2 = \{s, t\}$. Indeed, Y_1 and Y_2 are the two bounding loops for Π_4 w.r.t. X.

From Proposition 13 and the definition of a bounding loop, we derive the following.

Corollary 1. *Let Π be a disjunctive program, X a set of atoms, and Y a bounding loop for Π w.r.t. X that is contained in X. If $\Pi_{X,Y}$ is HEF, then there is a nonempty subset of Y that is unfounded by Π w.r.t. X.*

Recall program Π_4, its model X, and bounding loop Y_1. Note that $(\Pi_4)_{X,Y_1}$ is HEF. By Corollary 1, the fact that $(\Pi_4)_{X,Y_1}$ is HEF implies that X is not stable for Π_4. In fact, Y_1 contains $\{p, r\}$ and $\{q, r\}$, which are both elementarily unfounded by Π_4 w.r.t. X.

6 Computing Elementarily Unfounded Sets

It is inevitable that exponentially many loop formulas have to be considered in the worst case [22]. Hence, SAT-based answer set solvers do not try to find all loop formulas at once; loop formulas are added incrementally until a stable model is found (if there is any). As shown in [6], it is sufficient to consider only loop formulas of elementarily unfounded sets in this process. Thus, it is important to design an efficient algorithm for finding elementarily unfounded sets.

For a general disjunctive program, it has been shown that deciding whether a given set of atoms is elementary is coNP-complete [6]. While we do not expect a tractable algorithm for computing elementarily unfounded sets of general disjunctive programs, it is possible for HEF programs. Below we present a tractable algorithm for HEF programs, which is simpler and more efficient than the one described in [10].[6]

For any disjunctive program Π and any set Y of atoms, we define $(Y, EC_\Pi(Y))$ as a directed graph where:

$$\begin{aligned} EC^0_\Pi(Y) &= \emptyset \\ EC^{i+1}_\Pi(Y) &= \{\ (a,b) \mid \text{there is a rule (2) in } \Pi \text{ such that } A \cap Y = \{a\} \text{ and} \\ &\qquad\qquad \text{all atoms } b \text{ in } B \cap Y \text{ belong to the same} \\ &\qquad\qquad \text{strongly connected component of } (Y, EC^i_\Pi(Y))\ \} \\ EC_\Pi(Y) &= \textstyle\bigcup_{i \geq 0} EC^i_\Pi(Y)\ . \end{aligned}$$

This graph is equivalent to the "elementary subgraph" defined in [6], and it is closer to the algorithm for computing an elementarily unfounded set described below.

We first note that Theorem 2 in [6] can be extended to HEF programs.

Proposition 17. *For any HEF program Π and any nonempty set Y of atoms that occur in Π, Y is elementary for Π iff $(Y, EC_\Pi(Y))$ is a strongly connected graph.*

Given a disjunctive program Π, a set X of atoms occurring in Π, and a nonempty subset Y of X such that $\Pi_{X,Y}$ is HEF and $R^\omega_{\Pi,X}(Y) = Y$, Figure 1 shows an algorithm for computing an elementarily unfounded set by Π w.r.t. X that is contained in Y.[7]

Due to Step I, E-SET never considers any rule (2) of $\Pi_{X,Y}$ such that $|A \cap Y| > 1$. This is similar to the definition of $EC^{i+1}_\Pi(Y)$ above, where only rules (2) satisfying $A \cap Y = \{a\}$ contribute to any edge. In a bottom-up manner, Step 1(a) of E-SET adds edges to $EC_{\Pi_{X,Y}}(Y)$ for rules (2) such that $|B \cap Y| = 1$. This ensures that all rules contributing to edges depend on a single SCC of $(Y, EC_{\Pi_{X,Y}}(Y))$. In rules (2) of $\Pi_{X,Y}$ such that B contains multiple atoms from a recently computed SCC, Step 1(b) replaces all atoms of the SCC by a single representative. If this leads to $|B \cap Y| = 1$, rule (2) contributes an edge in the next iteration of Step 1(a). The described process is iterated until no further edges can be added. If a single SCC is obtained, i.e., if $(Y, EC_{\Pi_{X,Y}}(Y))$ is strongly connected, then Y is elementarily unfounded by Π w.r.t. X. Otherwise, in Step 2, we remove atoms from Y that belong to some SCC C that is not reached ($Y \setminus C$ still contains an elementarily unfounded set for Π w.r.t. X). In the next iteration

[6] That algorithm was designed for nondisjunctive programs, but also applies to HEF programs.

[7] "SCC" is used as a shorthand for "Strongly Connected Component."

E-SET($\Pi_{X,Y}, Y$)

I. $\Pi_{X,Y} := \Pi_{X,Y} \setminus \{(A \leftarrow B, F) \in \Pi_{X,Y} \mid |A \cap Y| > 1\}$

II. $EC_{\Pi_{X,Y}}(Y) := \emptyset$

III. While $(Y, EC_{\Pi_{X,Y}}(Y))$ is not strongly connected Do

1. While there is a rule $(A \leftarrow B, F)$ in $\Pi_{X,Y}$ such that $|A \cap Y| = 1$ and $|B \cap Y| = 1$ Do
   (a) For each rule $(A \leftarrow B, F)$ in $\Pi_{X,Y}$ such that $|A \cap Y| = 1$ and $|B \cap Y| = 1$ Do
      i. $EC_{\Pi_{X,Y}}(Y) := EC_{\Pi_{X,Y}}(Y) \cup \{(a, b) \mid A \cap Y = \{a\}, B \cap Y = \{b\}\}$
      ii. $\Pi_{X,Y} := \Pi_{X,Y} \setminus \{(A \leftarrow B, F)\}$ /* the rule needs not be considered further */
   (b) For each (non-trivial) SCC $(C, EC_{\Pi_{X,Y}}(Y) \cap (C \times C))$ of $(Y, EC_{\Pi_{X,Y}}(Y))$ Do
      i. Select an atom $b \in C$
      ii. $\Pi_{X,Y} := (\Pi_{X,Y} \setminus \{(A \leftarrow B, F) \in \Pi_{X,Y} \mid |B \cap C| > 1\}) \cup$
      $\{(A \leftarrow b, B \setminus C, F) \mid (A \leftarrow B, F) \in \Pi_{X,Y}, |B \cap C| > 1\}$
2. If $(Y, EC_{\Pi_{X,Y}}(Y))$ is not strongly connected Then
   (a) Select some SCC $(C, EC_{\Pi_{X,Y}}(Y) \cap (C \times C))$ of $(Y, EC_{\Pi_{X,Y}}(Y))$ that is not reached in $(Y, EC_{\Pi_{X,Y}}(Y))$
   (b) $Y := Y \setminus C$ /* some $Z \subseteq Y \setminus C$ is elementarily unfounded by Π w.r.t. X */
   (c) $EC_{\Pi_{X,Y}}(Y) := EC_{\Pi_{X,Y}}(Y) \setminus \{(a, b) \in EC_{\Pi_{X,Y}}(Y) \mid a \in C\}$

IV. Return Y

Fig. 1. E-SET: An algorithm to compute an elementarily unfounded set

of Step 1, this might allow to add more edges to $EC_{\Pi_{X,Y}}(Y)$ for rules (2) of $\Pi_{X,Y}$ such that $B \cap C \neq \emptyset$. The process is repeated until $(Y, EC_{\Pi_{X,Y}}(Y))$ becomes a strongly connected graph. Note that the computed set Y can be a proper subset of the Y in the invocation of E-SET($\Pi_{X,Y}, Y$).

When we apply E-SET to Π_1 ((4) in Section 3) and $Y = \{p, q, r\}$, it adds edges (p, r) and (q, r) to $EC_{\Pi_1}(Y)$. As the resulting graph is not strongly connected, either q or p is removed from Y. After this, adding edge (r, p) or (r, q), respectively, to $EC_{\Pi_1}(Y)$ leads to a strongly connected graph. The result of E-SET is thus either $\{p, r\}$ or $\{q, r\}$, which are the two elementarily unfounded sets for Π_1 w.r.t. $\{p, q, r\}$.

The following proposition states the correctness of the E-SET algorithm.

Proposition 18. *Let Π be a disjunctive program, X a set of atoms that occur in Π, and Y a nonempty subset of X. If $\Pi_{X,Y}$ is HEF and $R^{\omega}_{\Pi,X}(Y) = Y$, then E-SET($\Pi_{X,Y}$,$Y$) returns an elementarily unfounded set for Π w.r.t. X.*

It is reasonable to take a bounding loop Y for Π w.r.t. X such that $\Pi_{X,Y}$ is HEF as input for E-SET since every elementarily unfounded set is a subset of some bounding loop. For the correctness of E-SET, it is however sufficient that $\Pi_{X,Y}$ is HEF and $R^{\omega}_{\Pi,X}(Y) = Y$.

Finally, we comment on the complexity of E-SET. Note that E-SET successively merges atoms from an input set Y into SCCs until finally obtaining a single SCC.

Whenever a new SCC C is produced, all its atoms are replaced by a single element of C in rules (2) such that $|B \cap C| > 1$. This can be regarded as counting down body elements until only one atom from Y is left, in which case a rule "fires." This behavior is similar to the *Dowling-Gallier* algorithm [23], also used to compute the minimal model of a set of Horn clauses. Since the computation of SCCs and the Dowling-Gallier algorithm have linear complexity, the same is concluded for E-SET. In contrast, the elementary set computation algorithm in [10] has complexity $O(n \times log\ n)$.

7 Conclusion

The main contribution of this paper is identifying the class of HEF programs, a more general class of disjunctive programs than HCF programs, that can be turned into nondisjunctive programs in polynomial time and space by shifting head atoms into the body. We showed that several properties of nondisjunctive programs and HCF programs can be extended to HEF programs in a straightforward way. Since HCF programs have played an important role in the computation of stable models for disjunctive programs, we expect that HEF programs can be useful as well. As a first step, we have provided an algorithm for finding an elementarily unfounded set for a HEF program, which has a potential for improving the stable model computation for disjunctive programs.

As a future work, we plan to implement algorithm E-SET, presented in this paper, in CMODELS for an empirical evaluation. It is an open question whether identifying HEF programs is tractable, while it is known that identifying HCF programs can be done in linear time.

Acknowledgments

We are grateful to Selim Erdoğan, Tomi Janhunen, Dan Lessin, Vladimir Lifschitz, Torsten Schaub, Jicheng Zhao, and the anonymous referees for their useful comments. Joohyung Lee was partially supported by DTO AQUAINT.

References

1. Eiter, T., Gottlob, G.: Complexity results for disjunctive logic programming and application to nonmonotonic logics. In: Proceedings of International Logic Programming Symposium. (1993) 266–278
2. Ben-Eliyahu, R., Dechter, R.: Propositional semantics for disjunctive logic programs. Annals of Mathematics and Artificial Intelligence **12** (1994) 53–87
3. Gelfond, M., Lifschitz, V., Przymusińska, H., Truszczyński, M.: Disjunctive defaults. In: Proceedings of International Conference on Principles of Knowledge Representation and Reasoning. (1991) 230–237
4. Leone, N., Pfeifer, G., Faber, W., Eiter, T., Gottlob, G., Perri, S., Scarcello, F.: The DLV system for knowledge representation and reasoning. ACM Transactions on Computational Logic **7** (2006) 499–562
5. Lierler, Y.: Cmodels: SAT-based disjunctive answer set solver. In: Proceedings of International Conference on Logic Programming and Nonmonotonic Reasoning. (2005) 447–452

6. Gebser, M., Lee, J., Lierler, Y.: Elementary sets for logic programs. In: Proceedings of National Conference on Artificial Intelligence. (2006)
7. Lin, F., Zhao, Y.: ASSAT: Computing answer sets of a logic program by SAT solvers. Artificial Intelligence **157** (2004) 115–137
8. Lin, F., Zhao, J.: On tight logic programs and yet another translation from normal logic programs to propositional logic. In: Proceedings of International Joint Conference on Artificial Intelligence. (2003) 853–858
9. Leone, N., Rullo, P., Scarcello, F.: Disjunctive stable models: Unfounded sets, fixpoint semantics, and computation. Information and Computation **135** (1997) 69–112
10. Anger, C., Gebser, M., Schaub, T.: Approaching the core of unfounded sets. In: Proceedings of International Workshop on Nonmonotonic Reasoning. (2006) 58–66
11. Koch, C., Leone, N., Pfeifer, G.: Enhancing disjunctive logic programming systems by SAT checkers. Artificial Intelligence **151** (2003) 177–212
12. Gebser, M., Schaub, T.: Loops: Relevant or redundant? In: Proceedings of International Conference on Logic Programming and Nonmonotonic Reasoning. (2005) 53–65
13. Lee, J.: A model-theoretic counterpart of loop formulas. In: Proceedings of International Joint Conference on Artificial Intelligence. (2005) 503–508
14. Ferraris, P., Lee, J., Lifschitz, V.: A generalization of the Lin-Zhao theorem. Annals of Mathematics and Artificial Intelligence **47** (2006) 79–101
15. Fages, F.: Consistency of Clark's completion and existence of stable models. Journal of Methods of Logic in Computer Science **1** (1994) 51–60
16. Erdem, E., Lifschitz, V.: Tight logic programs. Theory and Practice of Logic Programming **3** (2003) 499–518
17. You, J.H., Yuan, L.Y., Zhang, M.: On the equivalence between answer sets and models of completion for nested logic programs. In: Proceedings of International Joint Conference on Artificial Intelligence. (2003) 859–866
18. Apt, K., Blair, H., Walker, A.: Towards a theory of declarative knowledge. In: Foundations of Deductive Databases and Logic Programming. Morgan Kaufmann (1988) 89–148
19. Baral, C., Gelfond, M.: Logic programming and knowledge representation. Journal of Logic Programming **19/20** (1994) 73–148
20. Inoue, K., Sakama, C.: Negation as failure in the head. Journal of Logic Programming **35** (1998) 39–78
21. Saccá, D., Zaniolo, C.: Stable models and non-determinism in logic programs with negation. In: Proceedings of ACM Symposium on Principles of Database Systems. (1990) 205–217
22. Lifschitz, V., Razborov, A.: Why are there so many loop formulas? ACM Transactions on Computational Logic **7** (2006) 261–268
23. Dowling, W., Gallier, J.: Linear-time algorithms for testing the satisfiability of propositional Horn formulae. Journal of Logic Programming **1** (1984) 267–284