Ordered Programs as Abductive Systems

Davy Van Nieuwenborgh[⋆] and Dirk Vermeir[⋆⋆]

Dept. of Computer Science
Vrije Universiteit Brussel, VUB
{dvnieuwe,dvermeir}@vub.ac.be

Abstract. In ordered logic programs, i.e. partially ordered sets of clauses where smaller rules carry more preference, inconsistencies, which appear as conflicts between applicable rules, are handled by satisfying more preferred rules, at the expense of defeating lesser rules. We show that this formalism can be exploited to obtain a simple implementation of abductive systems, where abducibles are assumed false by default, but weaker rules can be used to introduce them, if necessary. Moreover, the approach can be extended, without leaving the ordered programming framework, to support abductive systems involving preference, either on the set of abducibles or on the system description. The latter case appears naturally in applications such as legal reasoning where rules carry a natural precedence. However, combining preference on abducibles with a complex theory structure brings the complexity, e.g. of the relevance problem, to Σ_3^P, and thus such systems cannot be simulated by ordered programs.

1 Introduction

Order (or preference) in logic programming represents one of the possible ways to represent non-monotonic reasoning problems. The preferred answer set semantics of [23, 21] uses a partial order defined among the rules in a program to prefer certain extended answer sets above others, where the extended answer set semantics is a natural extension of the classical one [16], dealing with inconsistencies by allowing contradictory rules to defeat each other.

Consider the program in Fig. 1, where rules in lower levels are smaller (more preferred) than rules in higher levels.

This program has two preferred answer sets, i.e. $\{a, m\}$ and $\{b, m\}$. The rule $m \leftarrow not(m)$ forces m to be in every preferred answer set of the program, without providing a justification for it. To justify m, rules such as $m \leftarrow a$ in the next higher level have to be applied, but $a \leftarrow$ or $b \leftarrow$ can only be employed if necessary to derive m, as otherwise $not(a) \leftarrow$ and $not(b) \leftarrow$ are preferred. In fact, without the rule $m \leftarrow not(m)$, the program would have the single answer set $\{not(a), not(b), not(m)\}$.

The above program can be regarded as an abductive system where the theory consists of the rules $\{m \leftarrow a, m \leftarrow b\}$, the set of abducibles is $\{a, b\}$ and $\{m\}$ contains the

⋆ Supported by the FWO

⋆⋆ This work was partially funded by the Information Society Technologies programme of the European Commission, Future and Emerging Technologies under the IST-2001-37004 WASP project

$$\begin{array}{r@{\;}l}
a \leftarrow & \\
b \leftarrow & \\
\hline
\mathit{not}(a) \leftarrow & \\
\mathit{not}(b) \leftarrow & \\
\hline
m \leftarrow & a \\
m \leftarrow & b \\
\hline
m \leftarrow & \mathit{not}(m)
\end{array}$$

Fig. 1. Ordered program simulating abduction

manifestations. This system clearly has $\{a\}$ and $\{b\}$, i.e. the corresponding program's preferred answer sets, as subset minimal explanations for $\{m\}$.

We extend the above approach by allowing the theory of an abductive system to be an ordered program. This is useful, e.g. in legal reasoning systems where laws are often ordered w.r.t. legal precedence. Abductive systems using such a theory would prefer explanations based on laws with higher precedence. It turns out that this extension also nicely maps to ordered programming using the intuition depicted in Figure 1, replacing the rules $m \leftarrow a$ and $m \leftarrow b$ with the ordered theory.

Traditionally, for abductive systems, one usually restricts to subset minimal explanations. Here, we generalize this criterion to support a preference relation between abducibles and then preferring explanations that are minimal w.r.t. the induced partial order. Having a partial order on abducibles is useful, e.g. when certain abducibles are more likely to occur or more expensive to handle. Again, this extension can be accommodated by a similar construction as the one shown in Figure 1.

Interestingly, when considering the combination of both extensions, i.e. an ordered theory and an ordered set of abducibles, the above approach breaks down, as it will treat both preference relations (on the abducibles and on the system) as a single order, and thus the program will usually yield preferred explanations that are optimal w.r.t. the (more specific) system order, ignoring the preference structure on the rules corresponding to the abducibles. It turns out that such abductive systems cannot be simulated by ordered programs because the complexity of e.g. the relevance problem for abductive reasoning with such combined systems lies in Σ_3^P, which cannot be expressed by ordered programs [23].

The remainder of this paper is organized as follows: Section 2 presents a brief overview of the preferred answer set semantics for extended ordered logic programs while Section 3 recalls classical (logic programming based) abductive systems. Section 4 generalizes these abductive systems by allowing an ordered program as the system description, thus inducing a preference relation on the possible explanations. We present an algorithm transforming such an ordered abductive system into an ordered extended logic programs. Abductive systems with a partial order relation on the abducibles are considered in Section 5, together with a transformation to abductive systems with an ordered theory. Furthermore, we show that a combination of both order relations into a multi-ordered abductive system cannot be captured by the preferred answer set semantics. Section 6 discusses the relationship with other approaches. Conclusions and directions for further research are stated in Section 7. All proofs can be found in [22].

2 Preliminaries

We use the following basic definitions and notation. A *literal* is an *atom* a or a negated atom $\neg a$. An *extended literal* is a literal or a *naf-literal* of the form $not(l)$ where l is a literal. The latter form denotes negation as failure: $not(l)$ is interpreted as "l is not true". We use $\hat{l}$ to denote the ordinary literal underlying an extended literal, i.e. $\hat{l} = a$ if $l = not(a)$ while $\hat{a} = a$ if a is an ordinary literal. Both notations are extended to sets so $\hat{X} = \{\hat{e} \mid e \in X\}$, with X a set of extended literals, while $not(Y) = \{not(l) \mid l \in Y\}$ for any set of (ordinary) literals Y.

For a set of literals X we use $\neg X$ to denote $\{\neg p \mid p \in X\}$ where $\neg(\neg a) \equiv a$. Also, X^+ denotes the positive part of X, i.e. $X^+ = \{a \in X \mid a \text{ is an atom}\}$. The *Herbrand base* of X, denoted $\mathcal{B}_X$, contains all atoms appearing in X, i.e. $\mathcal{B}_X = (X \cup \neg X)^+$. A set I of literals is *consistent* if $I \cap \neg I = \emptyset$.

For a set of extended literals X, we use X^- to denote the literals underlying elements of X that are not ordinary literals, i.e. $X^- = \{l \mid not(l) \in X\}$. It follows that X is consistent iff the set of ordinary literals $\neg(X^-) \cup (X \setminus not(X^-))$ is consistent.

An *extended rule* is a rule of the form $\alpha \leftarrow \beta$ where $\alpha \cup \beta$ is a finite set of extended literals and $|\alpha| \leq 1$. A *simple rule* is an extended one with $\alpha \cup \beta$ containing only ordinary literals. We will often confuse a singleton set with its sole element, writing rules as $a \leftarrow \beta$ or $\leftarrow \beta$.

An *extended logic program* (ELP) is a countable set P of *extended rules* of the form $\alpha \leftarrow \beta$ where $\alpha \cup \beta$ is a finite set of extended literals, and $|\alpha| \leq 1$, i.e. α is a singleton or empty. If $(\alpha \cup \beta)^- = \emptyset$, i.e. all rules are free from naf-literals, P is called a *simple program* (SLP). The *Herbrand base* $\mathcal{B}_P$ of and ELP (or SLP) P contains all atoms appearing in P. An *interpretation* I of P is any consistent subset of $\mathcal{B}_P \cup \neg\mathcal{B}_P$. An interpretation I is *total* iff $\mathcal{B}_P \subseteq I \cup \neg I$.

An extended literal l is true w.r.t. an interpretation I, denoted $I \models l$ if $l \in I$ in case l is ordinary, or $I \not\models a$ if $l = not(a)$ for some ordinary literal a. As usual, $I \models X$ for some set of (extended) literals l iff $\forall l \in X \cdot I \models l$.

A rule $r = a \leftarrow \beta$ is *satisfied* by I, denoted $I \models r$, if $I \models a$, $a \neq \emptyset$, whenever $I \models \beta$, i.e. if r is *applicable* ($I \models \beta$), then it must be *applied* ($I \models \beta \cup a$).

For a simple program P, an *answer set* is a minimal interpretation I that is *closed* under the rules of P (i.e. $\forall r \in P \cdot I \models r$).

For an ELP P and an interpretation I we use $P_I \subseteq P$ to denote the *reduct* of P w.r.t. I, i.e. $P_I = \{r \in P \mid I \models r\}$. We also define the *GL-reduct* for P w.r.t. I, denoted P^I, as the program consisting of those rules $\alpha \setminus not(\alpha^-) \leftarrow (\beta \setminus not(\beta^-))$ where $\alpha \leftarrow \beta$ is in P, $I \models not(\beta^-)$ and $I \models \alpha^-$. Note that all rules in P^I are free from negation as failure, i.e. P^I is a simple program. An interpretation I is then an *answer set* of P iff I is an answer set of the reduct P^I. An extended rule $r = \alpha \leftarrow \beta$ is *defeated* w.r.t. P and I iff there exists an applied *competing rule* $r' = \alpha' \leftarrow \beta'$ such that $\{\alpha, \alpha'\}$ is inconsistent. An *extended answer set* for P is any interpretation I such that I is an answer set of P_I and each unsatisfied rule in $P \setminus P_I$ is defeated.

An *extended ordered logic program* (EOLP) is a pair $(R, <)$ where R is an ELP and $<$ is a well-founded strict partial order on the rules in R. When R is a SLP, the pair $(R, <)$ is called an *ordered logic program* (OLP). Intuitively, $r_1 < r_2$ indicates that r_1 is more preferred than r_2. In the examples we will often represent the order implicitly using the format

$$\frac{\frac{\frac{\cdots}{R_2}}{R_1}}{R_0}$$

where each R_i, $i \geq 0$, represents a set of rules, indicating that all rules below a line are more preferred than any of the rules above the line, i.e. $\forall i \geq 0 \cdot \forall r_i \in R_i, r_{i+1} \in R_{i+1} \cdot r_i < r_{i+1}$ or $\forall i \geq 0 \cdot R_i < R_{i+1}$ for short.

Let $P = \langle R, < \rangle$ be an EOLP. For subsets R_1 and R_2 of R we define $R_1 \preceq R_2$ iff $\forall r_2 \in R_2 \backslash R_1 \cdot \exists r_1 \in R_1 \backslash R_2 \cdot r_1 < r_2$. We write $R_1 \prec R_2$ just when $R_1 \preceq R_2$ and $R_1 \neq R_2$. For M_1, M_2 extended answer sets of R, we define $M_1 \preceq M_2$ iff $R_{M_1} \preceq R_{M_2}$. As usual, $M_1 \prec M_2$ iff $M_1 \preceq M_2$ and $M_1 \neq M_2$. An *answer set* for an EOLP P is any extended answer set of R. An answer set for P is called *preferred* if it is minimal w.r.t. $\preceq$. An answer set is called *proper* if it satisfies all minimal (according to $<$) rules in R.

3 Classical Abduction

Classical abductive systems use ELP's, i.e. programs containing both classical negation and negation as failure, to describe the theory under consideration. Our definition of the classical abductive framework is based on the one from [20], which in turn is based on the belief set semantics from [9].

Definition 1. *An* ***abductive logic program*** *(ALP) is a triple $S = (T, A, M)$, where T is a ELP representing the theory, A is a set of literals representing the abducibles and M is a set of literals representing the manifestations.*

A subset $H \subseteq A$ is an ***explanation*** *for S iff there is an answer set Q for $T \cup \{x \leftarrow \mid x \in H\}$ such that $M \subseteq Q$ and $H = Q \cap A$.*

Intuitively, an explanation contains those abducibles that should be assumed true to justify the manifestations from the theory.

Example 1. Suppose that a radiator in the room is cold, while it should be hot, i.e. $M = \{cold\_radiator\}$. This may be explained by any of the abducibles in $A = \{no\_fuel,\ no\_power,\ broken\_pump,\ broken\_heater\}$. The theory T describes how these abducibles affect the central heating system.

$$\begin{array}{rr} no\_combustion \leftarrow no\_fuel & cold\_water \leftarrow no\_combustion \\ no\_combustion \leftarrow no\_power & cold\_radiator \leftarrow cold\_water \\ cold\_radiator \leftarrow broken\_pump & no\_combustion \leftarrow broken\_heater \end{array}$$

E.g., without power, fuel or a working heater, no combustion can take place which, in turn, affects the temperature of the water and the radiator. Alternatively, a broken water pump prevents warm water from passing the radiator.

Some of the explanations for $cold\_radiator$ are $H_1 = \{no\_fuel\}$, $H_2 = \{no\_power\}$, $H_3 = \{broken\_pump\}$, $H_4 = \{broken\_heater\}$ and $H_5 = \{no\_fuel,\ broken\_pump\}$.

Note that H_5 in Example 1 is likely to be redundant since $H_5 = H_1 \cup H_3$, i.e. the manifestations can be explained using a subset of the abducibles in H_5.

In general, we assume that sets of abducibles may carry a preference order. Preferred explanations then correspond to explanations that are minimal with respect to this order.

Definition 2. *Let $S = (T, A, M)$ be an ALP, with $\leq$ a partial order relation on 2^A. An explanation of S is called **$\leq$-preferred** iff it is minimal w.r.t. $\leq$.*

Often, $\leq$ is taken as the subset order although cardinality order is also used (e.g. in abductive diagnosis, if all components in a circuit are equally likely to fail, cardinality-preferred explanations are more likely). In Example 1, H_1, H_2, H_3 and H_4 are $\subseteq$-preferred explanations.

Definition 1, which follows [11, 12], differs from the definition in [6] which does not require the $H = Q \cap A$ condition. This influences the set of explanations, as illustrated in the following example.

Example 2. Consider the abductive program $S = (T, A, M)$ with T containing $air\_in\_fuel\_pump \leftarrow out\_of\_diesel$ and $car\_does\_not\_start \leftarrow out\_of\_diesel$, $A = \{out\_of\_diesel, air\_in\_fuel\_pump\}$ and $M = \{car\_does\_not\_start\}$.

The semantics of [6] yields $\{out\_of\_diesel\}$ as a $\subseteq$-preferred explanation while $\{out\_of\_diesel, air\_in\_fuel\_pump\}$ is $\subseteq$-preferred according to Definition 1. Thus, [6] returns the "root abducibles" of the problem, leaving out side effects. On the other hand, Definition 1's solution includes the side effects, which is useful in this example, as just refueling diesel will not completely fix the problem: one must also ventilate the fuel pump.

4 Abduction with Ordered Theories

Often, the rules making up the system description display a natural preference order. E.g. in legal reasoning, the clauses representing laws are ordered according to their legal precedence. Naturally, with an ordered system description, one would prefer explanations that maximally satisfy that theory; in particular more preferred rules should only be defeated as a last resort.

Definition 3. *An **abductive ordered logic program** (AOLP) is a triple $D = (P, A, M)$, where $P = \langle R, <_R \rangle$ is an EOLP representing the theory, A is a set of literals representing the abducibles and M is a set of literals containing the manifestations.*

*A subset $H \subseteq A$ is an **explanation** iff there exists an extended answer set Q for $R \cup \{h \leftarrow \mid h \in H\} \cup \{not(a) \leftarrow \mid a \in A \setminus H\}$, such that $M \subseteq Q$ and $H = Q \cap A$.*

For an explanation H, we use $R_{H,Q}$ to denote the set $R_Q \subseteq R$, i.e. the reduct of R w.r.t. the extended answer set Q.

Note that there may be several extended answer sets Q, and associated reducts R_Q, to justify an explanation H. An explanation is called *proper* iff it has an associated reduct R_Q satisfying all minimal (according to $<_R$) rules in R.

Example 3. Consider the AOLP $D = (P, A, M)$ representing the trial of shooting incidents, where P is depicted below and $A = \{unarmed, threatened, shoot, dead\}$.

$$\begin{array}{rl}
r_1 : & guilty \leftarrow shoot, dead \\
r_2 : & self\_defense \leftarrow threatened \\
\hline
r_3 : & \neg guilty \leftarrow shoot, dead, self\_defense \\
r_4 : & \neg self\_defense \leftarrow shoot, unarmed
\end{array}$$

The preferred rules r_3 and r_4 state that one cannot be found guilty if one acted out of self defense and that self defense cannot be invoked if one shot an unarmed person. The more general rules r_1 and r_2 present the default treatment for a fatal shooting and a possible justification (having been threatened by the victim) for self defense.

Assuming that the facts of the case (i.e. the manifestations) are $F = \{shoot, dead, threatened\}$, the latter claimed by the defendant, a lawyer eager to obtain a conviction will search for an optimal explanation of $M = F \cup \{guilty\}$.

D has two explanations for M, namely $H_1 = \{shoot, dead, threatened\}$, corresponding to the answer set $Q_1 = M \cup \{self\_defense\}$ and $H_2 = H_1 \cup \{unarmed\}$ corresponding to both $Q_2 = M \cup \{unarmed, \neg self\_defense\}$ and $Q'_2 = M \cup \{unarmed, self\_defense\}$. The corresponding sets of satisfied rules w.r.t. these explanations are $R_{H_1,Q_1} = P \setminus \{r_3\}$, $R_{H_2,Q_2} = P \setminus \{r_2\}$ and $R_{H_2,Q'_2} = P \setminus \{r_3, r_4\}$. Only H_2 is a proper explanation.

The preference order among explanations is based on the $\preceq$ order among the corresponding sets of satisfied rules.

Definition 4. *Let D be an AOLP with R_{H_1,Q_1} and R_{H_2,Q_2}[1] sets of rules corresponding with the explanations H_1 and H_2. Then, R_{H_1,Q_1} is preferred upon R_{H_2,Q_2}, denoted* $R_{H_1,Q_1} \sqsubset R_{H_2,Q_2}$ *iff* $\begin{cases} H_1 \subset H_2 & \text{if } R_{H_1,Q_1} = R_{H_2,Q_2} \ , \\ R_{H_1,Q_1} \prec R_{H_2,Q_2} & \text{otherwise} \ . \end{cases}$

*An explanation H is **preferred** iff it corresponds to a minimal (w.r.t. $\sqsubset$) $R_{H,Q}$.*

Note that the special clause for $R_{H_1,Q_1} = R_{H_2,Q_2}$ is necessary, e.g. when both Q_1 and Q_2 satisfy all rules in R. In such a case, the smaller (w.r.t. $\subseteq$) explanation is preferred.

Example 4. In Example 3, $R_{H_2,Q_2} = P \setminus \{r_2\}$ is the unique minimal (w.r.t. $\sqsubset$) reduct. Therefore, the lawyer should attempt to establish $unarmed$ in order to obtain a conviction.

We show that $\sqsubset$-preferred explanations of an AOLP D can be retrieved from the preferred answer sets of an EOLP $L(D)$ corresponding to D. The construction is such that, for a given AOLP $D = ((R, <_R), A, M)$, the proper preferred answer sets of $L(D)$ represent the $\sqsubset$-preferred explanations of D, i.e. for any proper preferred answer set Q of $L(D)$, $Q \cap A$ is a preferred explanation and the other way around.

The construction of $L(D)$ follows the intuition sketched in Section 1.

[1] We abuse notation by considering $R_{H,Q}$ as a tagged set, such that $R_{H',Q'}$ may not be the same as $R_{H,Q}$ although, as sets of rules, $R_{H,Q} = R_{H',Q'}$.

- The bottom component R_m of $L(D)$, whose rules will always be satisfied, consists of a set of "constraint" rules that enforce the manifestations, without providing a justification for them. As the manifestations M should be derived from an explanation combined with the theory, the constraint rules have the form $m \leftarrow \mathit{not}(m)$, $m \in M$.
- On top of the bottom component, we put the theory R ordered w.r.t. $<_R$, simulating that we want as many specific rules satisfied as possible, while still deriving the manifestations.
- On top of the highest components in $<_R$, i.e. its least preferred clauses, we put a component R_n with rules of the form $\mathit{not}(a) \leftarrow$, for each abducible $a \in A$, simulating that abducibles are normally not assumed to be true. This ensures that the semantics will prefer answer sets that maximize non-assumed abducibles, but only if this does not imply that a more preferred rule becomes defeated.
- The topmost component $R_a > R_n$ introduces the possibility to assume abducibles. For each $a \in A$, R_a contains a rule $a \leftarrow$ that provides a justification, if necessary, for a. Note that all rules in R_a have a stronger competitor in R_n.

Intuitively, if no abducibles are necessary to explain the manifestations, any proper preferred answer set will satisfy all rules in $R_m \cup R_n$, defeating all rules in R_a. If, however, the manifestations cannot be explained without assuming some abducibles, the semantics will, in order to satisfy the rules in R_m, call upon rules in R_a to introduce them.

The following definition formalizes the above construction.

Definition 5. *Let $D = (P = (R, <_R), A, M)$ be an AOLP. The EOLP version $L(D)$ of D is defined by $L(D) = \langle R_a \cup R_n \cup R \cup R_m, R_m << _r < R_n < R_a \rangle$, where $R_a = \{a \leftarrow \mid a \in A\}$, $R_n = \{\mathit{not}(a) \leftarrow \mid a \in A\}$ and $R_m = \{m \leftarrow \mathit{not}(m) \mid m \in M\}$.*

Example 5. The EOLP corresponding with the system from Example 3 is shown below.

$$
\begin{array}{rl}
\begin{array}{rr}
\mathit{dead} \leftarrow & \mathit{shoot} \leftarrow \\
\mathit{unarmed} \leftarrow & \mathit{threatened} \leftarrow \\
\hline
\mathit{not}(\mathit{dead}) \leftarrow & \mathit{not}(\mathit{shoot}) \leftarrow \\
\mathit{not}(\mathit{unarmed}) \leftarrow & \mathit{not}(\mathit{threatened}) \leftarrow
\end{array} & \\
\hline
\mathit{guilty} \leftarrow & \mathit{shoot}, \mathit{dead} \\
\mathit{self\_defense} \leftarrow & \mathit{threatened} \\
\hline
\neg \mathit{self\_defense} \leftarrow & \mathit{shoot}, \mathit{unarmed} \\
\neg \mathit{guilty} \leftarrow & \mathit{shoot}, \mathit{dead}, \mathit{self\_defense} \\
\hline
\mathit{shoot} \leftarrow & \mathit{not}(\mathit{shoot}) \\
\mathit{dead} \leftarrow & \mathit{not}(\mathit{dead}) \\
\mathit{guilty} \leftarrow & \mathit{not}(\mathit{guilty}) \\
\mathit{threatened} \leftarrow & \mathit{not}(\mathit{threatened})
\end{array}
$$

The only preferred answer set for this program is $Q = \{\mathit{guilty}, \mathit{shoot}, \mathit{dead}, \mathit{unarmed}, \neg \mathit{self\_defense}, \mathit{threatened}\}$, corresponding with the unique preferred explanation H_2.

In general we have the following correspondence.

Theorem 1. *Let $D = (P, A, M)$ be an AOLP. Then, H is a preferred explanation for D iff $H = Q \cap A$. for some proper preferred answer set Q of $L(D)$.*

In abductive logic programming [6, 20], the problems of relevance and necessity are of natural interest, where relevance means deciding if a given abducible a is contained in some preferred explanation, while necessity refers to checking whether a is contained in all preferred explanations.

For abductive ordered programs, checking whether a subset $H \subseteq A$ is an explanation for an AOLP D can obviously be done in polynomial time. Thus, checking whether H is *not* a preferred explanation is in NP, i.e. guess a subset $H' \subseteq A$ such that $H' \sqsubset H$, which can be done in polynomial time, and verify if it is an explanation. Now, finding a preferred explanation H can be done by an NP algorithm that guesses H and uses an NP oracle to verify that it is not the case that H is not a preferred explanation.

Theorem 2. *Let $D = (P, A, M)$ be an AOLP. Deciding the problem of relevance, for a given abducible $a \in A$, for D is Σ_2^P-complete, while deciding necessity for a is Π_2^P-complete.*

Showing the hardness part of Theorem 2 can be done by a reduction to the known Σ_2^P problem of deciding whether a quantified boolean formula $\phi = \exists x_1, \ldots, x_n \cdot \forall y_1, \ldots, y_m \cdot F$ is valid, where we may assume that $F = \vee_{c \in C} c$ with each c a conjunction of literals over $X \cup Y$ with $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_m\}$ $(n, m > 0)$. The construction is inspired by a similar result for abductive programming under $\subseteq$-preferredness in [6], illustrating that the preferred explanation semantics does not involve any computational overhead w.r.t. classical abductive frameworks.

5 Ordering the abducibles

Often, abducibles are themselves partially ordered according to some preference, e.g. because one abducible is more likely to occur, or cheaper to verify, than another.

An abductive logic program with ordered abducibles, induces a partial order on the explanations of the abductive system.

Definition 6. *An abductive logic program with ordered abducibles (ALPOA) is a tuple $D = (S, <)$, where $S = (T, A, M)$ is an ALP and $<$ is a strict*[2] *partial order relation on the elements in A. The explanations of D are the explanations of S.*

For explanations H_1 and H_2, $H_1 \sqsubseteq H_2$ iff $\forall a \in H_1 \backslash H_2 \cdot \exists a' \in H_2 \backslash H_1 \cdot a < a'$.

Intuitively, H_1 is preferred over H_2 if any abducible a_1 from H_1 but not in H_2 is "covered" by a "less preferred" abducible $a_2 > a_1$ in H_2 but not in H_1. It can be shown that $\sqsubseteq$ is a partial order, provided that the inverse of $<$ is well-founded, see Lemma 1 in [22].

Example 6. Extend Example 1 to an ALPOA $D = (S, <)$ with the abducibles ordered as follows $\{\mathit{no\_fuel}, \mathit{no\_power}\} < \mathit{broken\_pump} < \mathit{broken\_heater}$. It can be seen that the explanations $H_1 = \{\mathit{no\_fuel}\}$ and $H_2 = \{\mathit{no\_power}\}$ are both $\sqsubseteq$-preferred.

[2] A strict partial order $<$ on a set X is a binary relation on X that is antisymmetric, anti-reflexive and transitive.

Each $\sqsubseteq$-preferred explanation is also $\subseteq$-preferred.

Theorem 3. *Let $D = (S, <)$ be an ALPOA. Every $\sqsubseteq$-preferred explanation H of D is a $\subseteq$-preferred explanation of S.*

If the order on the abducibles is empty, $\sqsubseteq$-preference reduces to $\subseteq$-preference.

Theorem 4. *Let S be an ALP. Then, all $\subseteq$-preferred explanations of S coincide with the preferred explanations of $D = (S, \emptyset)$.*

An ALPOA can be transformed into an equivalent AOLP[3] by placing, for a given ALPOA $D = ((T, A, M), <)$, T in the most preferred component of an ordered program. On top of this component, we put a component R_n containing rules of the form $r_a = \mathit{not}(a) \leftarrow$, for each abducible $a \in A$, simulating that abducibles are normally not assumed to be true. To take into account the preference relation between abducibles, we order the rules in R_n such that the AOLP semantics, when confronted with the necessity to defeat either r_a or $r_{a'}$, will defeat r_a if $a < a'$. So, it suffices to have $r_{a'} < r_a$, i.e. the order on R_n is the reverse of the order on A.

The following definition formalizes the above construction.

Definition 7. *Let $D = ((T, A, M), <)$ be an ALPOA. The AOLP version of D, denoted $aolp(D)$, is defined by $aolp(D) = (\langle R_n \cup T, T < R_n^< \rangle, A, M)$, where $R_n = \{\mathit{not}(a) \leftarrow \mid a \in A\}$. Furthermore, $R_n^<$ stands for $\mathit{not}(a_1) \leftarrow < \mathit{not}(a_2) \leftarrow$ with $a_1, a_2 \in A$ iff $a_2 < a_1$.*

Example 7. The AOLP corresponding with the ALPOA from Example 6 is $aolp(D) = (P, A, M)$, where P is shown below.

$$\begin{array}{c}
\mathit{not}(no\_fuel) \leftarrow \quad \mathit{not}(no\_power) \leftarrow \\
\hline
\mathit{not}(broken\_pump) \leftarrow \\
\hline
\mathit{not}(broken\_heater) \leftarrow \\
\hline
\begin{array}{rlrl}
no\_combustion \leftarrow & no\_fuel & cold\_water \leftarrow & no\_combustion \\
no\_combustion \leftarrow & no\_power & cold\_radiator \leftarrow & cold\_water \\
cold\_radiator \leftarrow & broken\_pump & no\_combustion \leftarrow & broken\_heater
\end{array}
\end{array}$$

It can be seen that $aolp(D)$ has two proper preferred explanations, i.e. H_1 and H_2, the preferred explanations of D.

In general, we have the following correspondence.

Theorem 5. *Let $D = (S, <)$ be an ALPOA. Then, H is a preferred explanation for D iff H is a proper preferred explanation for $aolp(D)$.*

Since the construction of Definition 7 is polynomial, it follows that the complexity results for ALPOA's are similar to those for AOLP's (Theorem 2).

It is natural to combine ordered system descriptions with a preference order on abducibles, yielding the notion a of multi-ordered abductive logic program. A preferred explanation for such a program is obtained by selecting the best (according to the abducible ordering) explanation from among the explanations that are preferred according to the system ordering.

[3] A direct simulation by an EOLP is also possible, using a construction similar to the one proposed in Section 4.

Definition 8. *A quadruple $D = (P, A, M, <)$ is a* ***multi-ordered abductive logic program*** *(MOALP), where (P, A, M) is an AOLP as in Definition 3 and $<$ is a strict partial order relation on the elements in A.*

A subset $H \subseteq A$ is an explanation H for D iff H is a preferred explanation of (P, A, M) (Definition 3). Such an explanation H of D is preferred iff it is minimal (among the explanations) w.r.t. $\sqsubseteq$ (Definition 6).

Checking whether a subset $H \subseteq A$ is an explanation for the AOLP (P, A, M) can be done in polynomial time, while checking whether it is not a preferred explanation is in NP. Thus, checking whether H is not a preferred explanation of D can be done by an NP algorithm that guesses $H' \sqsubseteq H$ and uses an NP oracle to verify that it is not the case that H' is not a preferred explanation for (P, A, M), thus this problem is in Σ_2^P. Finding a preferred explanation H for D can then be done by guessing H and using a Σ_2^P oracle to verify that it is not the case that H is not preferred.

Theorem 6. *Let $D = (P, A, M, <)$ be a MOALP. Deciding the problem of relevance, for a given abducible $a \in A$, for D is Σ_3^P-complete, while deciding necessity for a is Π_3^P-complete.*

Hardness is obtained by showing that the complement of relevance is Π_3^P hard, this by a reduction to the known Π_3^P-hard problem of deciding whether a quantified boolean formula $\phi = \forall z_1, \ldots, z_l \cdot \exists x_1, \ldots, x_n \cdot \forall y_1, \ldots, y_m \cdot F$ is valid, where we may assume that $F = \vee_{c \in C} c$ with each c a conjunction of literals over $X \cup Y \cup Z$ with $Z = \{z_1, \ldots, z_l\}$, $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_m\}$ ($l, n, m > 0$), see also [6].

Theorem 6 implies that MOALP's go beyond the capabilities of the answer set semantics for ordered programs, which are limited to Σ_2^P [23]. However, it can be shown that, if the preferred explanations of the AOLP $E = ((R, <_R), A, M)$ all satisfy the same rules in the ordered theory, an OLP containing both the ordered theory and the ordered abducibles, using the construction of Definition 7 where T is replaced by P, will compute the correct preferred explanations of the MOALP.

6 Relationships To Other Approaches

A number of different characterizations for abduction exist, both in the context of logic and logic programming, e.g. [4, 17, 13, 7, 6]. Early formalizations of abduction used theories in first order logic, while [13] introduced an abductive framework in the context of logic programming. Later, generalized stable models [8] were introduced as an extension of the stable model semantics [10] to handle abductive reasoning. Independently, [9] formalized a similar idea, called the belief set semantics, providing an abductive reasoning formalism for systems containing disjunction, negation as failure and classical negation. In [11, 12] this semantics was used to formalize abductive extended disjunctive programs. Another formalization of abduction for logic programming was given in [6], using definitions closer to the first order logic approaches. Example 2 illustrates the difference between [6] and [11, 12].

Most approaches suggest the usefulness of a preference relation among explanations, with subset-minimality an obvious candidate, although other orderings have also been proposed, e.g. cardinality minimal explanations, where preferred explanations are minimal w.r.t. the number of abducibles in it.

[5] mentions a possible formalization of preferred explanations for a linearly prioritized set of abducibles in the context of abduction for classical logic. The more general preference relation on explanations of Definition 6 reduces to the one used in [5], for the case where the underlying partial order on abducibles is linear.

Although a variety of proposals exist for extending logic programs with some kind of preference relation [14, 15, 2, 19, 1, 3, 24, 23], we are not aware of any prior work on abduction for such programs.

Reducing abduction to model computation has been done before, e.g. [18, 20] provide a different method for transforming abductive logic programs into disjunctive logic programs, using the possible model semantics, but only for the subset-preferred case. Section 4 can be regarded as an extension to more general preference relations. Moreover, our approach does not need disjunction to obtain the simulation as it relies on a single mechanism (order) to simulate both abduction and minimality.

7 Conclusions and Direction for Further Research

We have extended abductive logic programming to systems involving preference, in either the theory or the set of abducibles. Since such reasoning can be simulated using EOLP, which is equivalent to OLP (i.e. programs without negation as failure) [21], an implementation of OLP, e.g. using the algorithms described in [23], can be envisaged to perform abduction.

References

1. Gerhard Brewka and Thomas Eiter. Preferred answer sets for extended logic programs. *Artificial Intelligence*, 109(1-2):297–356, April 1999.
2. F. Buccafurri, N. Leone, and P. Rullo. Stable models and their computation for logic programming with inheritance and true negation. *Journal of Logic Programming*, 27(1):5–43, 1996.
3. Francesco Buccafurri, Wolfgang Faber, and Nicola Leone. Disjunctive logic programs with inheritance. In Danny De Schreye, editor, *Logic Programming: The 1999 International Conference*, pages 79–93, Las Cruces, New Mexico, December 1999. MIT Press.
4. P.T. Cox and T. Pietrzykowski. General diagnosis by abductive inference. In *Proceedings of the IEEE Symposium on Logic Programming*, pages 183–189, 1987.
5. Thomas Eiter and Georg Gottlob. The complexity of logic-based abduction. *Journal of the Association for Computing Machinery*, 42(1):3–42, 1995.
6. Thomas Eiter, Georg Gottlob, and Nicola Leone. Abduction from logic programs: Semantics and complexity. *Theoretical Computer Science*, 189(1-2):129–177, 1997.
7. Thomas Eiter, Georg Gottlob, and Nicola Leone. Semantics and complexity for abduction from default logic. *Artificial Intelligence*, 90(1-2):177–222, 1997.
8. K. Eshghi and R.A. Kowalski. Abduction compared with negation by failure. In *Proceedings of the 6th International Conference on Logic Programming*, pages 234–254. MIT Press, 1989.
9. Michael Gelfond. Epistemic approach to formalization of commonsense reasoning. Technical report, University of Texas at El Paso, 1991. Technical Report TR-91-2.

10. Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. In *Logic Programming, Proceedings of the Fifth International Conference and Symposium*, pages 1070–1080, Seattle, Washington, August 1988. The MIT Press.
11. Katsumi Inoue and Chiaki Sakama. Transforming abductive logic programs to disjunctive programs. In *Proceedings of the 10th International Conference on Logic Programming*, pages 335–353. MIT Press, 1993.
12. Katsumi Inoue and Chiaki Sakama. A fi xpoint characterization of abductive logic programs. *Journal of Logic Programming*, 27(2):107.136, May 1996.
13. A. C. Kakas, R. A. Kowalski, and F. Toni. Abductive logic programming. *Journal of Logic and Computation*, 2(6):719–770, 1992.
14. E. Laenens and D. Vermeir. A fi xpoint semantics of ordered logic. *Journal of Logic and Computation*, 1(2):159–185, 1990.
15. Els Laenens and Dirk Vermeir. Assumption-free semantics for ordered logic programs: On the relationship between well-founded and stable partial models. *Journal of Logic and Computation*, 2(2):133–172, 1992.
16. Vladimir Lifschitz. Answer set programming and plan generation. *Journal of Artifi cial Intelligence*, 138(1-2):39–54, 2002.
17. D. Poole. Explanation and prediction: An architecture for default and abductive reasoning. *Computational Intelligence*, 5(1):97–110, 1989.
18. Chiaki Sakama and Katsumi Inoue. On the equivalence between disjunctive and abductive logic programs. In Pascal Van Hentenryck, editor, *Logic Programming, Proceedings of the Eleventh International Conference on Logic Programming*, pages 489–503, Santa Margherita Ligure, Italy, June 1994. MIT Press.
19. Chiaki Sakama and Katsumi Inoue. Representing priorities in logic programs. In Michael J. Maher, editor, *Proceedings of the 1996 Joint International Conference and Syposium on Logic Programming*, pages 82–96, Bonn, September 1996. MIT Press.
20. Chiaki Sakama and Katsumi Inoue. Abductive logic programming and disjunctive logic programming: their relationship and transferability. *The Journal of Logic Programming*, 44(1-3):71–96, 2000.
21. Davy Van Nieuwenborgh and Dirk Vermeir. Order and negation as failure. Submitted.
22. Davy Van Nieuwenborgh and Dirk Vermeir. Ordered programs as abductive systems. Technical report, Vrije Universiteit Brussel, Dept. of Computer Science, 2003.
23. Davy Van Nieuwenborgh and Dirk Vermeir. Preferred answer sets for ordered logic programs. In *European Workshop, JELIA 2002*, volume 2424 of *Lecture Notes in Artifi cial Intelligence*, pages 432–443, Cosenza, Italy, September 2002. Springer Verlag.
24. Kewen Wang, Lizhu Zhou, and Fangzhen Lin. Alternating fi xpoint theory for logic programs with priority. In *CL*, volume 1861 of *Lecture Notes in Computer Science*, pages 164–178, London, UK, July 2000. Springer.