Proofs of the theorems in *Scalable Non-observational Predicate Learning in ASP*

Proof of Theorem 1

Theorem 1. *Let T be a learning task and e be a WCDPI. For any hypothesis $H \subseteq S_M$, $B \cup H$ accepts e iff there is at least one possibility $p \in poss^*(T, e)$ s.t. H accepts p.*

Proof. Assume $\exists p' \in poss^*(T, e)$ s.t. H accepts p'_{pi}.

$\Leftrightarrow \exists p' \in poss^*(T, e)$ s.t. there is an answer set of $H \cup p'_{as}$ that extends p'_{pi}.

$\Leftrightarrow \exists p' \in poss^*(T, e)$ s.t. $\exists \Delta \subseteq Ab$ s.t. Δ extends p'_{pi} and the unique answer set of $p'_{as} \cup H$ is $p'_{as} \cup \Delta$.

$\Leftrightarrow \exists p' \in poss^*(T, e)$ s.t. the unique answer set of $p'_{as} \cup H \cup top(B \cup e_{ctx})$ extends e_{pi} – by point 3 of Definition 5, the answer set of $p'_{as} \cup \Delta \cup top(B \cup e_{ctx})$ extends e_{pi}.

$\Leftrightarrow$ There is an answer set A of $bottom(B \cup e_{ctx})$ s.t. the unique answer set of $A \cup H \cup top(B \cup e_{ctx})$ extends e_{pi}.

$\Leftrightarrow B \cup H \cup e_{ctx}$ has an answer set that extends e_{pi} (by the splitting set theorem (Lifschitz and Turner 1994)). □

Proof of Theorem 2

We first recall some of the details of the FastLAS algorithm that are necessary for the proof (for full details please see (Law et al. 2020a)).

Definition 5. *(Law et al. 2020a) Let $e \in E^+$ and a be a ground atom. A rule R is in the* characteristic ruleset *of T w.r.t. a and e (written $\mathcal{C}(T, a, e)$) if and only if: (i) $R \in S_M$; (ii) there is at least one ground instance R^g of R s.t. $a = head(R^g)$ and $body(R^g)$ is satisfied by the unique answer set of $B \cup e_{ctx}$; and (iii) there is no rule R' that satisfies (i) and (ii) s.t. R is a strict subrule of R'.*

Definition 6. *(Law et al. 2020a) The* characterisation *of an example $e \in E^+$ (written $\mathcal{C}(T, e)$) is the pair $\langle e_I, e_V \rangle$, where $e_I = \bigcup\limits_{a \in e^{inc}} \mathcal{C}(T, a, e)$ and $e_V = \bigcup\limits_{a \in e^{exc}} \mathcal{C}(T, a, e)$.*

A rule R is said to be a *subrule* (Law et al. 2020a) of a rule R' if and only if $head(R) = head(R')$ and $body(R) \subseteq body(R')$ (we call R a *strict-subrule* if $R \neq R'$).

Definition 7. *(Law et al. 2020a) For any rule $R \in S_M$, let c_R be the set of all rules R' in $\mathcal{C}(T)$ s.t. R is a subrule of R'. The* generalised characteristic hypothesis space *of T, written $\mathcal{G}(T)$, is the set containing every rule R for which $c_R \neq \emptyset$ and there is no rule $R' \in S_M$ s.t. R is a strict subrule of R' and $c_R = c_{R'}$.*

By the end of FastLAS's second phase (generalisation), it has computed $\mathcal{G}(T)$. Note that for each rule $R \in S_M$, $\mathcal{G}(T)$ is guaranteed to contain at least one rule R' s.t. R is a subrule of R' and $c_R = c_{R'}$. Next FastLAS applies the *opt* algorithm to each rule in $\mathcal{G}(T)$. The *opt* algorithm takes as input a program P describing a scoring function,[1] a rule R from $\mathcal{G}(T)$ and the task T. It makes use of an ASP program $\mathcal{M}_{opt}$ defined in (Law et al. 2020a). Any answer set A of $\mathcal{M}_{opt}(P, R, T)$ can be mapped back into an ASP rule. $\mathcal{M}^{-1}_{rule}(A)$ denotes the rule extracted from A. $\mathcal{O}(T, \mathcal{S}_{len}) = \bigcup\limits_{R \in \mathcal{G}(T)} opt(P_{len}, R, T)$.

Algorithm 1 $opt(P, R, T)$

> **procedure** OPT(P, R, T)
> $CS = \emptyset$; $RS = \emptyset$;
>   **while** $AS(\mathcal{M}_{opt}(P, R, T) \cup CS) \neq \emptyset$ **do**
>   Fix A to be an optimal answer set of the program
> $R_{new} = \mathcal{M}^{-1}_{rule}(A)$;
> $RS = RS \cup \{R_{new}\}$;
> $CS = CS \cup \left\{ \texttt{:-} \bigwedge\limits_{\texttt{v(id}_\texttt{i}\texttt{)} \in \texttt{A}} \texttt{v(id}_\texttt{i}\texttt{)}. \right\}$;
>   **end while**
>   **return** RS;
> **end procedure**

The following lemma is a direct consequence of Lemma 2 from (Law et al. 2020b).

Lemma 1. *Let R be a rule in $\mathcal{G}(T)$. For each subrule R' of R:*

- *If there is an example $e \in E^+$ with an infinite penalty s.t. R' is a subrule of at least one rule in e_V then there is no answer set A of $\mathcal{M}_{opt}(P_{len}, R, T)$ s.t. $\mathcal{M}^{-1}_{rule}(A) = R'$.*
- *If not, there is exactly one answer set A of $\mathcal{M}_{opt}(P_{len}, R, T)$ s.t. $\mathcal{M}^{-1}_{rule}(A) = R'$. Furthermore, A satisfies the following properties:*

[1] In FastNonOPL, this program is always fixed as P_{len} from (Law et al. 2020a), which represents the standard $\mathcal{S}_{len}$ scoring function

1. *For each $e \in E^+$, $\mathtt{v}(\mathtt{e_{id}}) \in A$ iff R' is a subrule of at least one rule in e_V.*
2. *The score of A (at its only priority level) is equal to $|R'|$.*

The following Lemma is necessary to prove Theorem 2. The proof is similar to the proof of Theorem 4 of (Law et al. 2020a), given in (Law et al. 2020b). For any rule R, let v_R be the set of all $e \in E^+$ st R is a subrule of a rule in e_V.

Lemma 2. *Let T be the ILP_{LAS}^{OPL} task $\langle B, M, E^+\rangle$ and let $\mathcal{O}(T, \mathcal{S}_{len})$ be the OPT-sufficient subset constructed by FastLAS when it is executed on T. For each $H \subseteq S_M$, there is at least one $H' \subseteq \mathcal{O}(T, \mathcal{S}_{len})$ such that $\mathcal{S}_{len}(H', T) \leq \mathcal{S}_{len}(H, T)$ and $\forall e \in E^+$ s.t. H covers e, H' also covers e.*

Proof. Let T be the ILP_{LAS}^{OPL} task $\langle B, M, E^+\rangle$ and let $\mathcal{O}(T, \mathcal{S}_{len})$ be the OPT-sufficient subset constructed by FastLAS when it is executed on T. Let $H \subseteq S_M$ s.t. $\mathcal{S}_{len}(H, T)$ is finite. We must show that there is at least one $H' \subseteq \mathcal{O}(T, \mathcal{S}_{len})$ such that $\mathcal{S}_{len}(H', T) \leq \mathcal{S}_{len}(H, T)$ and $\forall e \in E^+$ s.t. H covers e, H' also covers e.

Assume for contradiction that there is no such H'. Then there must be at least one rule $h \in H$ for which there is no rule in $R \in O$ s.t. $|h| \leq |R|$, $c_h \subseteq c_R$ and $v_R \subseteq v_h$.

There must be a rule R' in $\mathcal{G}(T)$ such that $c_{R'} = c_h$ and h is a subrule of R'. $opt(P_{len}, R', T)$ must not contain any rule R'' such that $v_{R''} \subseteq v_h$. But if this were the case then in the final iteration of the while loop, by Lemma 1, there would be an answer set A of $\mathcal{M}_{opt}(P_{len}, R', T)$ s.t. $\mathcal{M}_{rule}^{-1}(A) = h$ and A did not violate any of the constraints in CS. This is a contradiction, as it would mean that the while loop would not terminate after the final iteration. Hence, O is OPT-sufficient. □

We can now prove Theorem 2.

Theorem 2. *Let $T = \langle B, M, E^+\rangle$, and T_{poss} be the FastLAS task $\langle \emptyset, M, \{\langle p_{id}, 1, p_{pi}, p_{as}\rangle \mid e \in E^+, p \in poss^*(T, e)\}\rangle$. The subset of the hypothesis space (written $\mathcal{O}(T_{poss}, \mathcal{S}_{len})$) constructed by FastLAS for T_{poss} is OPT-sufficient for T.*

Proof. Let H be a hypothesis $H \subseteq S_M$ s.t. $\mathcal{S}_{len}(H, T)$ is finite. We must show that there is an $H' \subseteq \mathcal{O}(T_{poss}, \mathcal{S}_{len})$ s.t. $\mathcal{S}_{len}(H', T) \leq \mathcal{S}_{len}(H, T)$. By Lemma 2, there must be an $H' \in \mathcal{O}(T_{poss}, \mathcal{S}_{len})$ s.t. $\mathcal{S}_{len}(H', T_{poss}) \leq \mathcal{S}_{len}(H, T_{poss})$ and H' covers every example in T_{poss} that is covered by H. Therefore, by Theorem 1, H' must also cover every example in T that H does. Hence, $\mathcal{S}_{len}(H', T) \leq \mathcal{S}_{len}(H, T)$. Hence, $\mathcal{O}(T_{poss}, \mathcal{S}_{len})$ is OPT-sufficient. □

Proof of Theorem 3

Theorem 3. *For any learning task T and any WCDPI e, abduce_possibilities(T, e) is guaranteed to terminate, and returns a set S such that $poss^*(T, e) \subseteq S \subseteq poss(T, e)$.*

Proof. As each of the for loops is iterating over finite sets, in order to show termination, it suffices to show that there can only be a finite number of iterations of the main while loop. In an arbitrary iteration of the while loop, let n be the smallest number of elements in a partial possibility in *pp*, and n' be this number in the next iteration. As each fix Δ_{fix}^+ must contain at least one element (or Δ could not have been an exception), n' must clearly be larger than n; hence, as n can be at most $|Ab|$, there can only be a finite number of iterations.

Next, we show that $S \subseteq poss(T, e)$. Let p be an arbitrary element of S. We must show that p satisfies the three conditions in the definition of a possibility. As $p_{as} \in AS(bottom(B \cup e_{ctx}))$ (due to line 1 of the algorithm) and p_{pi}^{inc}, $p_{pi}^{exc} \subseteq Ab$ (because fixes are always a subset of *Ab*, by definition), we only need to show that $\forall \Delta \subseteq Ab$ s.t. $\Delta \lhd p_{pi}$, $\exists I \in AS(p_{as} \cup \Delta \cup top(B \cup e_{ctx}))$ s.t. $I \lhd e_{pi}$. Due to the way p was constructed, p_{pi}^{exc} must be a negative fix to the set of minimal exceptions to $\langle\langle p_{pi}^{inc}, \emptyset\rangle, p_{as}\rangle$. Assume for contradiction that $\exists \Delta \subseteq Ab$ s.t. $\Delta \lhd p_{pi}$, $\nexists I \in AS(p_{as} \cup \Delta \cup top(B \cup e_{ctx}))$ s.t. $I \lhd e_{pi}$. This would mean that Δ would be an exception to p that does not intersect with any of the minimal exceptions to p (which is a clear contradiction). Hence, $p \in poss(T, e)$.

It remains to show that $poss^*(T, e) \subseteq S$. Let p be an arbitrary possibility. It suffices to show that in an arbitrary iteration, if there is at least one *partial_p* $\in$ *pp* s.t. $partial\_p_{pi}^{inc} \subseteq p_{pi}^{inc}$, $partial\_p_{pi}^{exc} \subseteq p_{pi}^{exc}$ and $partial\_p_{as} \subseteq p_{as}$ at the start of the iteration, then there must be such a *partial_p* in either *pp'* or *poss* at the end of the iteration (as *pp* is empty when the algorithm terminates and because this condition must hold at the start of the algorithm, there must be one such *partial_p* in S for each possibility p, proving that $poss^*(T, e) \subseteq S$). Fix an arbitrary *partial_p* $\in$ *pp* at the start of the iteration. Case (1): there is a positive fix Δ_{fix}^+ to a minimal exception of *partial_p* s.t. $\Delta_{fix}^+ \subseteq p_{pi}^{inc}$; in this case, *pp'* contains $\langle\langle partial\_p_{pi}^{inc} \cup \Delta_{fix}^+, partial\_p_{pi}^{exc}\rangle, partial\_p_{as}\rangle$, and so satisfies the condition. Case (2): there is no positive fix to a minimal exception of *partial_p* that is a subset of p_{pi}^{inc}. As p is a possibility, this must mean that all minimal exceptions intersect with p_{pi}^{exc}, meaning that at least one minimal negative fix is a subset of p_{pi}^{exc}; the resulting new CDPI that is added to *poss* satisfies the condition.

□

Searching for exceptions

Definition 8. *Let e be a WCDPI and p be a partial possibility. The program $\mathcal{M}_{except}(e, p)$ consists of the following:*

- *For each rule $R \in ground(top(B \cup e_{ctx}))$:*
  - $\mathtt{rule(R_{id})}$, *where $\mathtt{R_{id}}$ is a unique identifier for R;*
  - $\mathtt{in\_rule(R_{id}, head, h)}$, $\mathtt{h} = head(R)$*;*
  - $\mathtt{in\_rule(R_{id}, b\_pos, b)}$ *for each $b \in body^+(R)$;*
  - $\mathtt{in\_rule(R_{id}, b\_neg, b)}$ *for each $b \in body^-(R)$;*
- *For each $\mathtt{a} \in e_{pi}^{inc}$, $\mathtt{t(a)}$;*
- *For each $\mathtt{a} \in e_{pi}^{exc}$, $\mathtt{f(a)}$;*
- *For each $\mathtt{a} \in Ab$, $\mathtt{abducible(a)}$;*
- *For each $\mathtt{a} \in p_{pi}^{inc}$, $\mathtt{delta(a)}$;*
- *For each $\mathtt{a} \in p_{pi}^{as}$, $\mathtt{bot(a)}$.*

Before proving the correctness of our approach, it is useful to introduce the following lemma.

```
1:  0 { delta(A) } 1 :- abducible(A).
2:  t(A) ; f(A) :- in_r(R, _, A).
3:  prec(A1, A2) ; prec(A2, A1) :- t(A1), t(A2).
4:  prec(A1, A3) :- prec(A1, A2), prec(A2, A3).
5:  t(A) :- delta(A).
6:  t(A) :- bot(A).
7:  t(A) :- top(R), in_r(R, head, A); t(A) : in_r(R, pos, A); f(A) : in_r(R, neg, A).
8:  f(A) :- in_r(R, _, A), not delta(A), not bot(A), u(R) : in_r(R, head, A), top(R).
9:  u(R) :- in_r(R, b_pos, A), f(A).
10: u(R) :- in_r(R, b_neg, A), t(A).
11: u(R) :- prec(H, A), in_r(R, head, H), in_r(R, b_pos, A).
12: t(A) :- saturate, in_r(R, _, A).
13: f(A) :- saturate, in_r(R, _, A).
14: prec(A1, A2) :- t(A1), t(A2), saturate.
15: saturate :- prec(A1, A2), prec(A2, A1), A1 != A2.
16: saturate :- t(A), f(A).
17: :- not saturate.
```

Figure 1: The general encoding Π. Note that this encoding uses conditional literals of the form $\mathtt{b} : \mathtt{c_1}, \ldots, \mathtt{c_m}$. In Π, the conditions $\mathtt{c_1}, \ldots, \mathtt{c_m}$ use predicates which are defined as facts (in the program in Definition 8). This means that the conditional literal $\mathtt{b} : \mathtt{c_1}, \ldots, \mathtt{c_m}$ is essentially shorthand for the conjunction of all $\mathtt{b^g}$'s such that $\mathtt{b^g} : \mathtt{c_1^g}, \ldots, \mathtt{c_m^g}$ is a ground instance of the conditional literal and $\mathtt{c_1^g}, \ldots, \mathtt{c_m^g}$ all occur as facts in the program. Note that outside of conditional literals, ";" and "," can be used interchangeably in the body of a rule (to denote conjunction); but a conditional literal must end with ";" (or ".").

Lemma 3. *X be a set of atoms and P be an ASP program such that all negation as failure in P is over predicates that only appear in X and not P. The answer sets of $X \cup P$ are the minimal models of $(X \cup P)^X$.*

Proof. Assume $A \in AS(X \cup P)$
$\Leftrightarrow A$ is a minimal model of $(X \cup P)^A$.
$\Leftrightarrow A$ is a minimal model of $(X \cup P)^A$ and $X \subseteq A$.
$\Leftrightarrow A$ is a minimal model of $(X \cup P)^X$ ($(X \cup P)^A$ is equal to $(X \cup P)^X$, as the only negation as failure in P refers to atoms in X and not $A \backslash X$). □

We now prove the correctness of our encoding, showing that it can be used to generate the exceptions to a partial possibility. Note that in Algorithm 1, exclusions are never added to a partial possibility until they are added (as complete possibilities) to the set *poss*, meaning that we can assume that all partial possibilities have an empty set of exclusions.

Theorem 4. *For any WCDPI e and any partial possibility p the set of all exceptions to p is equal to the set $\{\{\delta \mid \mathtt{delta}(\delta) \in A\} \mid A \in AS(\Pi \cup \mathcal{M}_{except}(e, p))\}$.*

Proof. We show this in two parts. First, we show that for any exception there is such an answer set, and then we show that any answer set represents a exception. We use the notations $\Pi[i]$ and $\Pi[i \ldots j]$ to denote the i^{th} line and the i to j^{th} lines of Π.

- Let Δ be an exception of p.
  There is clearly an answer set of $\Pi[1] \cup \mathcal{M}_{except}(e, p)$ s.t. $\Delta = \{\delta \mid \mathtt{delta}(\delta) \in A\}$. Hence, it remains to show that $\Pi[2..17] \cup \mathcal{M}_{except}(e, p) \cup \{\mathtt{delta}(\delta). \mid \delta \in \Delta\}$ is satisfiable. This is the case iff $\Pi[2..16] \cup \mathcal{M}_{except}(e, p) \cup \{\mathtt{delta}(\delta). \mid \delta \in \Delta\}$ has at least one answer set that contains $\mathtt{saturate}$. For ease of notation, let $A' = \mathcal{M}_{except}(e, p) \cup \{\mathtt{delta}(\delta). \mid \delta \in \Delta\}$.
  The only instances of negation as failure in $\Pi[2 \ldots 16]$ are (in $\Pi[8]$) over the predicates in A'. Therefore, the answer sets of $\Pi[2 \ldots 16] \cup A'$ are equal to the minimal models of $(\Pi[2 \ldots 16] \cup A')^{A'}$ (by Lemma 3). Hence, $\Pi[2 \ldots 16] \cup A'$ has at least one answer set. It therefore suffices to show that any minimal model of $\Pi[2 \ldots 16] \cup A'$ must contain $\mathtt{saturate}$.
  Assume for contradiction that there is a minimal model A^* of $A' \cup \Pi[2 \ldots 16]$ that does not contain $\mathtt{saturate}$. As Δ is an exception to p, either $A = \{\mathtt{a} \mid \mathtt{t(a)} \in A^*\}$ does not extend e_{pi} or it is not an answer set of $p_{as} \cup \Delta \cup top(B \cup e_{ctx})$. By the facts in A', A must contain all of the inclusions of e, but if it contains any exclusion $\mathtt{a}$, then both $\mathtt{t(a)}$ and $\mathtt{f(a)}$ would be in A^*, contradicting that $\mathtt{saturate} \notin A^*$. Hence, A must not be an answer set of $p_{as} \cup \Delta \cup top(B \cup e_{ctx})$. $\Pi[7]$ ensures that A is a model of $(p_{as} \cup \Delta \cup top(B \cup e_{ctx}))^A$. The $\mathtt{prec}$ relation represents a total ordering $\prec$ over the atoms in A such that every atom in A (that is not proven by Δ or the bottom program) can be proved using only "lower" atoms (according to $\prec$); hence, no subset of A can also be a model, meaning that A is an answer set of $p_{as} \cup \Delta \cup top(B \cup e_{ctx})$. Contradiction!
- Let A be an answer set of $\Pi \cup \mathcal{M}_{except}(e, p)$. Assume for contradiction that $\Delta = \{\delta \mid \mathtt{delta}(\delta) \in A\}$ is not an exception of p. As $\Delta \lhd p_{pi}$ (p_{pi}^{exc} is empty and p_{pi}^{inc} is in the program as facts), there must be an answer set of A' of $p_{as} \cup \Delta \cup top(B \cup e_{ctx})$ that extends e_{pi}. As A' is an answer set, there must be a total ordering $\prec$ of the atoms in A' s.t. for each $\mathtt{a} \in A'$ there is at least one rule R in $p_{as} \cup \Delta \cup top(B \cup e_{ctx})$ whose body is satisfied by A' and s.t. for every positive body literal $\mathtt{b}$ in R, $\mathtt{b} \prec \mathtt{a}$. Hence, there will be an answer set A^* of $\Pi[1 \ldots 16] \cup \mathcal{M}_{except}(e, p)$ s.t. $A' = \{\mathtt{a} \mid \mathtt{t(a)} \in A^*\}$ and $\{(\mathtt{a}, \mathtt{b}) \mid \mathtt{a} \prec \mathtt{b}, \mathtt{a}, \mathtt{b} \in A'\} = \{(\mathtt{a}, \mathtt{b}) \mid \mathtt{prec(a, b)} \in A^*\}$ (this answer set cannot contain $\mathtt{saturate}$, because $\mathtt{prec}$

represents a total ordering and there is no way to prove $\texttt{f(a)}$ for any atom $\texttt{a} \in A'$). As $A \subset A^*$, A cannot be an answer set of $\Pi[1 \dots 16] \cup \mathcal{M}_{except}(e, p)$, and therefore cannot be an answer set of $\Pi \cup \mathcal{M}_{except}(e, p)$, contradicting the original assumption. Hence, $\Delta = \{\delta \mid \texttt{delta}(\delta) \in A\}$ is an exception to p.

$\square$

Given the result of the theorem, we can use Clingo (Gebser et al. 2016) to find the answer sets of the encoding and post process them to get the set of all exceptions. However, the algorithm does not consider all exceptions, it only needs the minimal exceptions. By adding the line "`#heuristic delta(A).[1@1, false]`" to the ASP program and running Clingo with the arguments "`--enum-mode=domRec heursitic=domain`", we direct Clingo to only return answer sets which are subset minimal over the `delta` atoms, meaning that the answer sets it returns correspond exactly to the minimal exceptions.

Searching for positive fixes

Definition 9. *Let e be a WCDPI, p be a partial possibility and Δ be an exception to p. The program $\mathcal{M}_{fix}(e, p, \Delta)$ consists of the following:*

- *$top(B \cup e_{ctx})$;*
- *For each $\texttt{a} \in e^{inc}_{pi}$, $\{\texttt{:- not a.}\}$;*
- *For each $\texttt{a} \in e^{exc}_{pi}$, $\{\texttt{:- a.}\}$;*
- *For each $\texttt{a} \in Ab$, $\{\texttt{a :- delta(a). 0\{delta(a)\}1.}\}$;*
- *For each $\texttt{a} \in p^{as}_{pi} \cup \Delta$, $\{\texttt{a.}\}$.*

Theorem 5. *For any WCDPI e, any partial possibility p and any exception Δ to p, the set of all positive fixes of Δ is equal to the set $\{\{\delta \mid \texttt{delta}(\delta) \in A\} \mid A \in AS(\Pi \cup \mathcal{M}_{fix}(e, p, \Delta))\}$.*

Proof.

Assume $\Delta^+_{fix} \subseteq Ab$ is a positive fix of Δ.

$\Leftrightarrow \exists A \in AS(p_{as} \cup \Delta \cup \Delta^+_{fix} \cup top(B \cup e_{ctx}))$ s.t. A extends e_{pi}

$\Leftrightarrow \exists A \in AS(p_{as} \cup \Delta \cup \left\{ \begin{array}{l} \texttt{a :- delta(a).} \\ \texttt{0\{delta(a)\}1.} \end{array} \middle| \texttt{a} \in Ab \right\} \cup top(B \cup e_{ctx}))$ s.t. $\{\delta \mid \texttt{delta}(\delta) \in A\}$ s.t. A extends e_{pi}.

$\Leftrightarrow \exists A \in AS(p_{as} \cup \Delta \cup \{\texttt{:- not a.} \mid \texttt{a} \in e^{inc}_{pi}\} \cup \{\texttt{:- a.} \mid \texttt{a} \in e^{exc}_{pi}\} \cup \left\{ \begin{array}{l} \texttt{a :- delta(a).} \\ \texttt{0\{delta(a)\}1.} \end{array} \middle| \texttt{a} \in Ab \right\} \cup top(B \cup e_{ctx}))$ s.t. $\{\delta \mid \texttt{delta}(\delta) \in A\}$

$\Leftrightarrow \exists A \in AS(\mathcal{M}_{fix}(e, p, \Delta))$ s.t. $\{\delta \mid \texttt{delta}(\delta) \in A\}$ $\square$

Given the result of the theorem, we can use Clingo (Gebser et al. 2016) to find the answer sets of the encoding and post process them to get the set of all fixes. Similarly to in the previous section, we use Clingo's heuristics to restrict the answer sets to those which correspond to minimal positive fixes.

Encoding used in the final solve stage

Before defining the encoding, we ask the reader to recall (from (Law et al. 2020a), Proposition 1) that an "observational" CDPI e is covered by a hypothesis $H \subseteq S_M$ if and only if for each $\texttt{a} \in e^{inc}_{pi}$, H contains at least one rule that is a subrule of a rule in $\mathcal{C}(T, \texttt{a}, e)$ and for each $\texttt{a} \in e^{exc}_{pi}$, H contains no rule that is a subrule of any rule in $\mathcal{C}(T, \texttt{a}, e)$. This also applies to possibilities in this paper (as they are observational examples). For any possibility p and any $\texttt{a} \in p^{inc}_{pi} \cup p^{exc}_{pi}$, we write $\mathcal{C}^*(T, O, \texttt{a}, p)$ to denote the rules $R \in O$ s.t. R is a subrule of at least one rule in $\mathcal{C}(T, \texttt{a}, p)$.

Definition 10. *Let T be the ILP^+_{LAS} task $\langle B, M, E^+\rangle$, let O be a hypothesis space. $\mathcal{M}_{solve}(T, O)$ is the program consisting of the following components:*

1. *For each $h \in O$, the rule $\texttt{0\{in\_h(h}_{\texttt{id}}\texttt{)\}1.}$ and the weak constraint $:\sim \texttt{in\_h(h}_{\texttt{id}}\texttt{).[|h|@1, in\_h(h}_{\texttt{id}}\texttt{)]}$*
2. *For each example $e \in E^+$*
   - *If e has a finite penalty, the weak constraint: $:\sim \texttt{not cov(e}_{\texttt{id}}\texttt{).[e}_{\texttt{pen}}\texttt{@1, noise(e}_{\texttt{id}}\texttt{)]}$*
   - *If e has an infinite penalty, the constraint: $\texttt{:- not cov(e}_{\texttt{id}}\texttt{).}$*
   - *For each $p \in poss^*(T, e)$:*
     - *The rule $\texttt{cov(e}_{\texttt{id}}\texttt{) :- not n\_poss(p}_{\texttt{id}}\texttt{).}$*
     - *For each $a \in p^{inc}_{pi}$, the rule: $\texttt{n\_poss(p}_{\texttt{id}}\texttt{) :-} \bigwedge_{\texttt{h} \in \mathcal{C}^*\texttt{(T,O,a,p)}} \texttt{not in\_h(h}_{\texttt{id}}\texttt{).}$*
     - *For each $a \in p^{exc}_{pi}$, for each $h \in \mathcal{C}^*(T, O, a, p)$ the rule: $\texttt{n\_poss(p}_{\texttt{id}}\texttt{) :- in\_h(h}_{\texttt{id}}\texttt{).}$*

Given a task T we say that $H \subseteq O$ is an optimal inductive solution of T in O if $H \in ILP^+_{LAS}(T)$ and $\nexists H' \subseteq O$ st $H' \in ILP^+_{LAS}(T)$ and $\mathcal{S}_{len}(H', T) < \mathcal{S}_{len}(H', T)$.

Theorem 6. *Let T be the ILP^{OPL}_{LAS} task $\langle B, M, E^+\rangle$, O be a hypothesis space. Let AS be the optimal answer sets of $\mathcal{M}_{solve}(T, O)$. The set of all optimal inductive solutions of T in O is equal to $\{\{h \in O \mid \texttt{in\_h(h}_{\texttt{id}}\texttt{)} \in A\} | A \in AS\}$.*

Proof. Let P be the program containing all rules in $\mathcal{M}_{solve}(T, O)$ other than the hard constraints. Clearly there is a one-to-one mapping between the answer sets of P and the subsets of O. This is characterised as follows:

$$AS(P) = \left\{ \begin{array}{c} \{\texttt{in\_h(h}_{\texttt{id}}\texttt{)} \mid h \in H\} \cup \\ \left\{ \texttt{n\_poss(p}_{\texttt{id}}\texttt{)} \middle| \begin{array}{c} e \in E^+, \\ p \in poss^*(T, e), \\ H \text{ does not accept } p \end{array} \right\} \\ \cup \{\texttt{cov(e}_{\texttt{id}}\texttt{)} \mid e \in E^+, H \text{ accepts } e\} \end{array} \middle| H \subseteq O \right\}.$$

The explanation is as follows. The second set (defining the `n_poss` atoms) follows because $\texttt{n\_poss(p}_{\texttt{id}}\texttt{)}$ holds if and only if there is either an inclusion in p^{inc}_{pi} for which H contains no subrules of every rule in $\mathcal{C}^*(T, O, \texttt{a}, e)$ or there is an exclusion in p^{exc}_{pi} for which H contains at least one subrule of a rule in $\mathcal{C}^*(T, O, \texttt{a}, e)$. Hence, $\texttt{n\_poss(p}_{\texttt{id}}\texttt{)}$ holds iff p is accepted by H. The third set defines `cov` atoms for those examples for which at least one minimal possibility is accepted. Hence (by Theorem 1) `cov(e)` holds iff e is accepted by H.

Due to the hard constraints, the answer sets of $\mathcal{M}_{solve}(T, O)$ are the answer sets corresponding to hypotheses that cover every example with an infinite penalty. Hence, $\{\{h \in O \mid \texttt{in\_h}(\texttt{h}_{\texttt{id}}) \in A\} | A \in AS(\mathcal{M}_{solve}(T, O))\}$ is equal to the set of inductive solutions that are contained in O. The penalty of each answer set A (at priority 1) is equal to the score $\mathcal{S}_{len}(\{h \in O \mid \texttt{in\_h}(\texttt{h}_{\texttt{id}}) \in A\}, T)$. Hence, the set of all optimal inductive solutions of T in O is equal to $\{\{h \in O \mid \texttt{in\_h}(\texttt{h}_{\texttt{id}}) \in A\} | A \in AS\}$. $\square$

Note that as FastNonOPL uses FastLAS to construct an OPT-sufficient subset of the hypothesis space (as shown by Theorem 3), and then uses $\mathcal{M}_{solve}(T, O)$ to find an optimal solution of T in O, Theorem 6 shows that FastNonOPL is sound and complete w.r.t. the optimal solutions of any non-recursive ILP^{+}_{LAS} task.

Flags for evaluation

The flags used for FastLAS and ILASP in the experiments were as follows:

- For the agent experiments:
  - ILASP: "`--version=2i -ml=4`"
  - FastLAS (used by FastNonOPL to build the OPT-sufficient subset of the hypothesis space): no flags given.
- For the CAVIAR experiments
  - ILASP: "`--version=4 -nsp -ng --restarts`"
  - FastLAS (when run directly): no flags given.
  - FastLAS (used by FastNonOPL to build the OPT-sufficient subset of the hypothesis space): no flags given.

References

Gebser, M.; Kaminski, R.; Kaufmann, B.; Ostrowski, M.; Schaub, T.; and Wanko, P. 2016. Theory solving made easy with clingo 5. In *Technical Communications of the 32nd International Conference on Logic Programming (ICLP 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Law, M.; Russo, A.; Bertino, E.; Broda, K.; and Lobo, J. 2020a. FastLAS: Scalable inductive logic programming incorporating domain-specific optimisation criteria. In *AAAI*. Association for the Advancement of Artificial Intelligence.

Law, M.; Russo, A.; Bertino, E.; Broda, K.; and Lobo, J. 2020b. Proofs of the theorems in FastLAS: Scalable inductive logic programming incorporating domain-specific optimisation criteria. `https://github.com/spike-imperial/FastLAS/blob/master/proofs_document.pdf`.

Lifschitz, V., and Turner, H. 1994. Splitting a logic program. In *ICLP*, volume 94, 23–37.