X-ABALearn: Argumentative Learning with Semantics à la Carte*

Emanuele De Angelis[1] , **Maurizio Proietti**[1] , **Francesca Toni**[2]
[1] CNR-IASI, Rome, Italy
[2]Imperial, London, UK
{emanuele.deangelis, maurizio.proietti}@iasi.cnr.it, ft@imperial.ac.uk

Abstract

ABA Learning is a recent approach for obtaining Assumption-based Argumentation (ABA) frameworks by reasoning with transformation rules from background knowledge and positive/negative examples of concepts of interest. ABA Learning relies on credulous reasoning under a specific semantic notion of extensions for ABA, namely that of *stable* extensions. In this paper, we newly frame the problem in terms of credulous reasoning under *any* semantic notion of extensions for ABA. Focusing on *admissible, complete, grounded, preferred* as well as stable extensions, we present *X-ABALearn*, a novel parametric algorithm (with *X* any ABA semantics) based on variants of the transformation rules of ABA Learning and an implementation thereof in Answer Set Programming. Finally, we explore the use of (our implementation of) *X-ABALearn* on several learning problems, including tabular data and beyond.

1 Introduction

ABA Learning (Proietti and Toni 2024; De Angelis, Proietti, and Toni 2024; De Angelis, Proietti, and Toni 2025) is a recently proposed form of symbolic learning for obtaining Assumption-based Argumentation (ABA) frameworks (Bondarenko et al. 1997; Dung, Kowalski, and Toni 2009; Toni 2014; Cyras et al. 2017) from background knowledge (also in the form of ABA frameworks) and positive and negative examples of concepts of interest. ABA Learning obtains (learns) an ABA framework F expanding the original background knowledge so that there is at least one *stable extension* Δ of F (as defined in (Bondarenko et al. 1997)) with all given positive examples and no negative examples accepted in Δ. In committing to a specific semantics ABA Learning is not alone, within the symbolic learning field, as, for example, ILASP (Law, Russo, and Broda 2014) adopts stable models (Gelfond and Lifschitz 1988) for learning answer set programs and FOLD-RM (Wang, Shakerin, and Gupta 2022) adopts the well-founded model (Gelder, Ross, and Schlipf 1991) for learning (stratified) logic programs. However, reasoning under different (ABA or logic programming) semantics gives different results in general and restricting to a single semantics may limit flexibility and applicability of the output of learning in different contexts.

In this paper we propose *X-ABALearn*, a novel form of ABA Learning that is parametric with respect to and can be instantiated with any semantics *X* of extensions for ABA. Like ABA Learning, *X-ABALearn* is restricted to *flat* ABA frameworks with an underlying atomic language and learns by reasoning with transformation rules. These are Rote Learning, Folding, Assumption Introduction and Fact Subsumption, adapted from transformation rules in ABA Learning (Proietti and Toni 2024; De Angelis, Proietti, and Toni 2024; De Angelis, Proietti, and Toni 2025). We show that only Rote Learning (needed to ground the learning in examples) and Fact Subsumption (needed for simplification) depend on the chosen semantics, whereas Folding (needed for generalisation) and Assumption Introduction (needed for making learnt ABA rules defeasible) are independent of the semantics. Like Brave ABALearn (De Angelis, Proietti, and Toni 2024), *X-ABALearn* embeds a form of credulous reasoning, by seeking to obtain an ABA framework with at least one extension where all positive examples and no negative examples are accepted. In *X-ABALearn* this is achieved by a parameter for credulous acceptance that can be instantiated with any semantics. Overall, *X-ABALearn* is decoupled from the choice of a semantics of reference.

We study properties of *X-ABALearn* that are independent on the semantics and for five specific instances (admissible, complete, grounded, preferred and stable extensions). We also propose an implementation, like *ASP-ABAlearn*$_B$ (De Angelis, Proietti, and Toni 2024) in Answer Set programming (ASP), of these five instances of *X-ABALearn*, obtained by suitably specialising the parameters with ASP encodings. For implementing the credulous acceptance parameter, we adapt the encodings from (Lehtonen, Wallner, and Järvisalo 2021). Finally, we report on a qualitative and quantitative experimental evaluation based on our implementation of *X-ABALearn*, by considering some example ABA learning problems, to explore the versatility of our approach, as well as others drawn from tabular dataset, to stress-test the computational viability of our implementation, in comparison with the baseline *ASP-ABAlearn*$_B$.

In summary, we make the two contributions: (i) we propose *X-ABALearn*, the first semantics-independent symbolic learner, and study properties thereof; (ii) we propose an im-

*Accepted to the *23rd International Conference on Principles of Knowledge Representation and Reasoning* (KR 2026) – special track *KR meets Machine Learning and Explanation*, July 20–23, 2026 – Lisbon, Portugal. Part of FLoC 2026.

plementation of *X-ABALearn* for five semantics and conduct an experimental evaluation, qualitatively and quantitatively exploring potential and limitations of our implementation to inform future work for symbolic learning.

2 Related Work

ABA Learning was first proposed by (Proietti and Toni 2024), followed by two algorithms: *ASP-ABAlearn*$_B$ (De Angelis, Proietti, and Toni 2024) and *brave ABA Learn* (De Angelis, Proietti, and Toni 2025), differing in their use of the Folding strategy, and hence for the type of inductive generalisations they obtain, but both based on stable extensions ((De Angelis, Proietti, and Toni 2025) also consider the grounded extension, but only for stratified ABA frameworks, where grounded and stable extensions coincide). Our main contribution is an ABA Learning algorithm that instead takes ABA semantics as a parameter. To this aim, we also prove that the Rote Learning and Fact Subsumption rules introduced in (Proietti and Toni 2024) depend on the chosen semantics, while Folding and Assumption Introduction do not. The implementation of *ASP-ABAlearn*$_B$ exploits a mapping of flat ABA frameworks into ASP programs in the simple case where the ABA semantics is given by stable extensions. Here, we extend this approach to any ABA semantics that can be encoded in ASP by using the mappings defined by (Lehtonen, Wallner, and Järvisalo 2021).

Due to the mapping between flat ABA and logic programming, ABA Learning is related to *Inductive Logic Programming* – see (Cropper et al. 2022) for a recent survey. Approaches to Inductive Logic Programming include Brave Induction (Sakama and Inoue 2009), ILASP (Law, Russo, and Broda 2014), FastLAS (Law et al. 2020), and FOLD-RM (Wang, Shakerin, and Gupta 2022). Besides the many differences between the learning algorithms, Brave Induction, ILASP and FastLAS learn ASP programs under the stable model semantics (Gelfond and Lifschitz 1988) and FOLD-RM (Wang, Shakerin, and Gupta 2022) learns stratified normal logic programs where stable models and the well-founded model (Gelder, Ross, and Schlipf 1991) coincide. To the best of our knowledge, no extension of these approaches to other semantics has been studied.

The AA-CBR approach of (Cyras, Satoh, and Toni 2016; Cocarascu, Cyras, and Toni 2018; Paulino-Passos and Toni 2021) learns abstract argumentation frameworks (AAF) (Dung 1995) from casebases, where each case is characterised by a set of binary features. Then, the outcome for new cases is predicted by calculating the grounded extension (Dung 1995) of the learnt AAF. AA-CBR can be seen as an instance of ABA Learning (namely, the so called Greedy ABA Learning (De Angelis, Proietti, and Toni 2025)), whereas AAFs can be represented as a proper subclass of ABA frameworks under any semantics. Thus, our work indirectly shows that AA-CBR can be generalised by learning ABA frameworks from casebases and associate with them semantics other than the one based on grounded extensions. This generalisation can be significant when the casebase is incoherent (i.e., two cases are equipped with the same features but a different label).

3 Background

3.1 Assumption-Based Argumentation (ABA)

An *ABA framework* (as originally proposed in (Bondarenko et al. 1997), but presented here following (Dung, Kowalski, and Toni 2009; Toni 2014) and (Cyras et al. 2017)) is a tuple $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\ } \rangle$ such that

- $\langle \mathcal{L}, \mathcal{R} \rangle$ is a deductive system, where $\mathcal{L}$ is a *language* and $\mathcal{R}$ is a set of *(inference) rules* of the form $s_0 \leftarrow s_1, \ldots, s_m$ ($m \geq 0, s_i \in \mathcal{L}$, for $0 \leq i \leq m$);
- $\mathcal{A} \subseteq \mathcal{L}$ is a (non-empty) set of *assumptions*;[1]
- $\overline{\ }$ is a *total mapping* from $\mathcal{A}$ into $\mathcal{L}$, where $\overline{a}$ is the *contrary* of a, for $a \in \mathcal{A}$ (also denoted as $\{a \mapsto \overline{a} \mid a \in \mathcal{A}\}$).

Given a rule $s_0 \leftarrow s_1, \ldots, s_m$, we call s_0 the *head* and $s_1, \ldots, s_m$ the *body*; if $m = 0$ then the body is *empty* (represented as $s_0 \leftarrow$) and the rule is called a *fact*. We focus on *flat* ABA frameworks, where assumptions are not heads of rules. Also, we restrict attention to ABA frameworks where $\mathcal{L}$ is a finite set of ground atoms. However, following (Proietti and Toni 2024; De Angelis, Proietti, and Toni 2024), we use *schemata* for rules, assumptions and contraries, using variables to represent compactly all instances over some underlying universe $\mathcal{U}$, as illustrated next.

Example 1. *For $\mathcal{U}$ some universe, let $\mathcal{R} = \{p(X) \leftarrow q(X), f(X), \quad p(X) \leftarrow r(X), \quad s(X) \leftarrow q(X), \quad f(a) \leftarrow | X \in \mathcal{U}\}$, $\mathcal{L} = \{p(X), q(X), r(X), s(X), f(X) | X \in \mathcal{U}\}$, $\mathcal{A} = \{\underline{q(X)}, r(X) | X \in \mathcal{U}\}$ and, for all $X \in \mathcal{U}$, $\overline{q(X)} = p(X)$, $\overline{r(X)} = s(X)$. Then $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\ } \rangle$ is a flat ABA framework with $\mathcal{L}$ a set of ground atoms.*

By *vars*(E) we denote the set of variables occurring in atom, rule, or rule body E. Also, by *pred*(E) we denote the set of predicate symbols occurring in E, where E is an atom, rule, set thereof, or an ABA framework. We assume that each ABA framework F is such that $\mathcal{L} = \{p(t) | p \in \textit{pred}(F), t$ is a tuple of elements of $\mathcal{U}$ of the same length as the arity of $p\}$.

In the remainder, unless stated otherwise, we assume as given an ABA framework $F = \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\ } \rangle$ restricted as mentioned earlier.

The semantics of flat ABA frameworks can be given by "acceptable" *extensions*, i.e. sets of *arguments* able to successfully deal with *attacks*, in the sense determined by the chosen semantics.[2] Intuitively, arguments are deductions of claims using rules and supported by assumptions, and attacks are directed at the assumptions in the support of arguments. More formally, following (Dung, Kowalski, and Toni 2009; Toni 2014; Cyras et al. 2017):

- *An argument for (the claim) $s \in \mathcal{L}$ supported by $A \subseteq \mathcal{A}$ and $R \subseteq \mathcal{R}$* (denoted $A \vdash_R s$) is a finite tree with nodes labelled by sentences in $\mathcal{L}$ or by $true$ (a sentence not already in $\mathcal{L}$), the root labelled by s, leaves either $true$ or assumptions in A, and non-leaves s' with, as children,

[1] Non-emptiness can always be guaranteed (Toni 2014).

[2] For flat ABA frameworks, the original semantics in terms of sets of assumptions (Bondarenko et al. 1997) is equivalent to the ones given here in terms of sets of arguments (Cyras et al. 2017).

Transformation Rule	$\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\ } \rangle$	$\langle \mathcal{L}', \mathcal{R}', \mathcal{A}', \overline{\ }' \rangle$
R1. *Rote Learning*	$p(t) \in \mathcal{L}$	$\mathcal{R}' = \mathcal{R} \cup \{p(X) \leftarrow X = t\}$
R2. *Folding*	ρ_1: $H \leftarrow Eqs_1, B_1, B_2$, ρ_2: $K \leftarrow Eqs_1, Eqs_2, B_1$ $\rho_1, \rho_2 \in \mathcal{R}$ with $vars(Eqs_2) \cap vars(\rho_1) = \emptyset$	$\mathcal{R}' = (\mathcal{R} \setminus \{\rho_1\}) \cup \{\rho_3\}$ with ρ_3: $H \leftarrow Eqs_2, K, B_2$.
R3. *Assumption Introduction*	$(\rho_1 : H \leftarrow Eqs, B) \in \mathcal{R}$ X is a tuple of variables taken from $vars(\rho_1)$	$\mathcal{R}' = (\mathcal{R} \setminus \{\rho_1\}) \cup \{\rho_2\}$, with $\rho_2 : H \leftarrow Eqs, B, \alpha(X)$ $\mathcal{A}' = \mathcal{A} \cup \{\alpha(X)\}$, $\overline{\alpha(X)}' = c\_\alpha(X)$, and $\overline{\beta}' = \overline{\beta}$ for all $\beta \in \mathcal{A}$
R4. *Fact Deletion*	$(\rho : p(X) \leftarrow X = t) \in \mathcal{R}$	$\mathcal{R}' = \mathcal{R} \setminus \{\rho\}$

Table 1: Transformation rules from $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\ } \rangle$ to $\langle \mathcal{L}', \mathcal{R}', \mathcal{A}', \overline{\ }' \rangle$. We show only the parts that change, and ignore the changes to $\mathcal{L}$.

the elements of the body of some rule in R with head s' (and all rules in R are used in the tree).

- $A_1 \vdash_{R_1} s_1$ *attacks* $A_2 \vdash_{R_2} s_2$ iff $s_1 = \overline{a}$ for some $a \in A_2$.

Example 2. *For $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\ } \rangle$ in Example 1 with rules in $\mathcal{R}$ numbered from 1 to 4 and $\mathcal{U} = \{a\}$, $\{q(a)\} \vdash_{\{1,4\}} p(a)$, $\{q(a)\} \vdash_{\{3\}} s(a)$ and $\{q(a)\} \vdash_{\emptyset} q(a)$ are (amongst the) arguments, with the former attacking itself and the other two.*

Let $Args$ be the set of all arguments of $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\ } \rangle$. We focus on the semantics below, as they are all supported by the system (Lehtonen, Wallner, and Järvisalo 2021) our implementation uses. Given $A, B \subseteq Args$, we say that A *attacks* B if there exist $x \in A$ and $y \in B$ such that x attacks y; A is *conflict-free* iff A does not attack itself; A *defends* B if, for all $C \subseteq Args$ that attack B, A attacks C. Then, a conflict-free extension $A \subseteq Args$ is

- *admissible* (*adm* in short) iff A defends itself;
- *complete* (*com* in short) iff A is admissible and contains every $x \in Args$ such that A defends $\{x\}$;
- *grounded* (*grd* in short) iff A is $\subseteq$-minimally complete;
- *preferred* (*prf* in short) iff A is $\subseteq$-maximally admissible;
- *stable* (*stb* in short) iff A attacks $\{x\}$ for each $x \in Args \backslash A$.

Example 3. *Continuing Example 2, $\{\emptyset \vdash_{\{4\}} f(a), \{r(a)\} \vdash_{\{2\}} p(a), \{r(a)\} \vdash_{\emptyset} r(a)\}$ is admissible, complete, preferred and stable; $\{\emptyset \vdash_{\{4\}} f(a)\}$ is admissible, complete and grounded; there are no other "acceptable" extensions under the semantics we consider.*[3]

For $\sigma \in \{adm, com, grd, prf, stb\}$, we say that an ABA framework is *σ-satisfiable* iff it admits at least one σ-extension, and *σ-unsatisfiable* otherwise. For Δ a σ extension of F and $s \in \mathcal{L}$, we write $F \models_{\Delta} s$ to indicate that s is the claim of an argument in Δ. If there exists a σ extension Δ of F such that $F \models_{\Delta} s$, we say that s is a *credulous σ-consequence* of F, written $F \models_{\sigma} s$.

3.2 Transformation Rules for ABA Learning

To learn ABA frameworks from examples, we follow the approach based on *transformation rules* introduced by (Proietti and Toni 2024; De Angelis, Proietti, and Toni 2024). As in these works, we assume that all rules in $\mathcal{R}$ are written in *normalised* form as follows:

$$p_0(X_0) \leftarrow eq_1, \ldots, eq_k, p_1(X_1), \ldots, p_n(X_n)$$

where $p_i(X_i)$, for $0 \leq i \leq n$, is an atom (whose ground instances are) in $\mathcal{L}$ and eq_i, for $0 \leq i \leq k$, is an equality $t_1^i = t_2^i$, with t_j^i a term whose variables occur in the tuples $X_0, X_1, \ldots, X_n$. In particular, we represent a ground fact $p(t) \leftarrow$ as $p(X) \leftarrow X = t$. The body of a normalised rule can be freely rewritten by using the standard axioms of equality, e.g., $Y_1 = a, Y_2 = a$ can be rewritten as $Y_1 = Y_2, Y_2 = a$. For constructing arguments, we assume that, for any ABA framework, the language $\mathcal{L}$ contains all equalities between elements of the underlying universe $\mathcal{U}$ and $\mathcal{R}$ includes all rules $a = a \leftarrow$, where $a \in \mathcal{U}$. When presenting the transformation rules, we use the following notations: (1) H, K denote heads of rules, (2) Eqs (possibly with subscripts) denotes sets of equalities, (3) B (possibly with subscripts) denotes sets of atoms. We assume that, for all rules $H \leftarrow B \in \mathcal{R}$, $vars(H) \subseteq vars(B)$.

The transformation rules we use are: *Rote Learning*, *Folding*, *Assumption Introduction*, and *Fact Deletion*. Table 1 summarises these rules. They all transform a given ABA framework, say $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\ } \rangle$, into a new ABA framework, say $\langle \mathcal{L}', \mathcal{R}', \mathcal{A}', \overline{\ }' \rangle$. The *Rote Learning* rule can be used to learn ground facts either from positive examples or for contraries of assumptions, as we will show in the next sections. The *Folding* rule is borrowed from logic program transformation (Pettorossi and Proietti 1994) and can be seen as a form of *inverse resolution* (Muggleton 1995), which can be used for generalising a rule by replacing some atoms in its body with their consequence using a rule in $\mathcal{R}$. The *Assumption Introduction* rule adds a (possibly new) assumption to the body of a rule so to make it defeasible. By learning rules for the contrary of the assumption, we may be able to avoid the acceptance of unwanted arguments (i.e., to capture exceptions to default rules). The *Fact Deletion* rule is a generalisation of the *Fact Subsumption* rule considered in (De Angelis, Proietti, and Toni 2024), which consists in deleting a fact that does not affect whether examples are σ-consequences of the learnt ABA framework (see Section 5). In Section 5 we study the soundness with respect to ABA semantics σ of R1–R4. In Section 6 we show how these rules can be combined to form a learning strategy, for any σ.

[3] In terms of assumption-level extensions (Bondarenko et al. 1997; Toni 2014), the first extension in the example corresponds to $\{r(a)\}$ and the second extension corresponds to $\emptyset$.

3.3 Answer Set Programming (ASP)

In Section 6 we use ASP (Gelfond and Lifschitz 1991) consisting of rules of the form

$\texttt{p :- q}_1\texttt{,}\ldots\texttt{,q}_\texttt{k}\texttt{, not q}_{\texttt{k+1}}\texttt{,}\ldots\texttt{, not q}_\texttt{n}\texttt{.}$ or

$\texttt{:- q}_1\texttt{,}\ldots\texttt{,q}_\texttt{k}\texttt{, not q}_{\texttt{k+1}}\texttt{,}\ldots\texttt{, not q}_\texttt{n}\texttt{.}$

where $\texttt{p}$, $\texttt{q}_1$, ..., $\texttt{q}_\texttt{n}$, are atoms, $\texttt{k} \geq 0$, $\texttt{n} \geq 0$, and not denotes negation as failure. We assume that the reader is familiar with the *stable model semantics* for ASP (Gelfond and Lifschitz 1988), and we call *answer set* of P any set of ground atoms assigned to P by that semantics. P is said to be *satisfiable*, denoted *sat*(P), if it has an answer set, and *unsatisfiable* otherwise. An atom p is a *brave consequence* of P if $\texttt{p} \in A$, for some answer set A of P. For flat ABA frameworks with an atomic language, (Lehtonen, Wallner, and Järvisalo 2021) construct a mapping, call it μ_σ, into ASP programs such that, for $\sigma \in \{adm, com, grd, prf, stb\}$, $F \models_\sigma s$ iff supported(s) is a brave consequence of $\mu_\sigma(F)$, where supported is a predicate of $\mu_\sigma(F)$ that encodes acceptance.[4]

4 ABA Learning under σ-Extensions

We give a novel notion of ABA learning, parametric with respect to the ABA semantics and generalising the ABA learning problem of (Proietti and Toni 2024; De Angelis, Proietti, and Toni 2024) which focuses only on stable extensions.

The *background knowledge* is *any* (flat and atomic) ABA framework. *Positive* and *negative examples* are ground atoms of the form $p(c)$, for p a predicate with arity $n \geq 0$ and c a tuple of n ground terms in $\mathcal{U}$. Due to the flatness restriction, examples are not assumptions. Similarly to (De Angelis, Proietti, and Toni 2024), we also assume that an ABA learning problem specifies the set $\mathcal{T}$ of *learnable predicates*, which do not necessarily coincide with the predicates occurring in the (positive and negative) examples.

Definition 1. *An* ABA learning problem *is a triple* $\langle\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\ }\rangle, \langle \mathcal{E}^+, \mathcal{E}^-\rangle, \mathcal{T}\rangle$*, where:*

- $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\ }\rangle$ *is a* σ*-satisfiable ABA framework, called* background knowledge*,*
- $\mathcal{E}^+ \subseteq \mathcal{L}$ *is a set of* positive examples*,*
- $\mathcal{E}^- \subseteq \mathcal{L}$ *is a set of* negative examples*, with* $\mathcal{E}^+ \cap \mathcal{E}^- = \emptyset$*,*
- $\mathcal{T}$ *is a set of* learnable *predicates, such that:* (1) $pred(\mathcal{E}^+ \cup \mathcal{E}^-) \subseteq \mathcal{T}$*,* (2) $\mathcal{T} \cap pred(\mathcal{A}) = \emptyset$*, and* (3) *for every* $p \in \mathcal{T}$ *of arity* n *and* n*-tuple* t *of ground terms in* $\mathcal{U}$*,* $p(t)$ *belongs to* $\mathcal{L}$*.*

In the remainder, unless stated otherwise, we assume as given an ABA learning problem $\langle F, \langle \mathcal{E}^+, \mathcal{E}^-\rangle, \mathcal{T}\rangle$ with $F = \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\ }\rangle$. Without loss of generality, when considering this or other ABA learning problems, we leave the language component of the ABA frameworks implicit, and use, e.g., $\langle \mathcal{R}, \mathcal{A}, \overline{\ }\rangle$ to stand for F.

Next, we define the notion of entailment of sets of positive and negative examples parametrised with ABA semantics.

Definition 2. *For any ABA semantics* σ*:*

- *for a* σ *extension* Δ *of* F*, we write* $F \models_\Delta \langle \mathcal{E}^+, \mathcal{E}^-\rangle$ *if* $\forall e \in \mathcal{E}^+$*,* $F \models_\Delta e$*, and* $\forall e \in \mathcal{E}^-$*,* $F \not\models_\Delta e$*;*
- F σ-entails $\langle \mathcal{E}^+, \mathcal{E}^-\rangle$*, denoted* $F \models_\sigma \langle \mathcal{E}^+, \mathcal{E}^-\rangle$*, if there exists a* σ *extension* Δ *of* F*, such that* $F \models_\Delta \langle \mathcal{E}^+, \mathcal{E}^-\rangle$*.*

The relation $F \models_\sigma \langle \mathcal{E}^+, \mathcal{E}^-\rangle$ cannot be fully expressed in terms of credulous entailment of single claims. Indeed, on one hand, if $F \models_\sigma \langle \mathcal{E}^+, \mathcal{E}^-\rangle$, then, for all $e \in \mathcal{E}^+$, $F \models_\sigma e$, but, for $e \in \mathcal{E}^-$, not necessarily $F \not\models_\sigma e$. On the other hand, if for all $e \in \mathcal{E}^+$, $F \models_\sigma e$ and for all $e \in \mathcal{E}^-$, $F \not\models_\sigma e$, then not necessarily $F \models_\sigma \langle \mathcal{E}^+, \mathcal{E}^-\rangle$.

Now, we define a σ*-solution* of an ABA learning problem as a new framework that extends the background knowledge and σ-entails the examples.

Definition 3. *A* σ-solution *of an ABA learning problem* $\langle\langle \mathcal{R}, \mathcal{A}, \overline{\ }\rangle, \langle \mathcal{E}^+, \mathcal{E}^-\rangle, \mathcal{T}\rangle$ *is an ABA framework* $\langle \mathcal{R}', \mathcal{A}', \overline{\ }'\rangle$ *such that: (i)* $\mathcal{R} \subseteq \mathcal{R}'$*, (ii) for each* $H \leftarrow B \in \mathcal{R}' \setminus \mathcal{R}$*,* $pred(H) \cap pred(\langle \mathcal{R}, \mathcal{A}, \overline{\ }\rangle) \subseteq \mathcal{T}$*, (iii)* $\mathcal{A} \subseteq \mathcal{A}'$*, (iv)* $\overline{\alpha}' = \overline{\alpha}$ *for all* $\alpha \in \mathcal{A}$*, (v)* $\langle \mathcal{R}', \mathcal{A}', \overline{\ }'\rangle$ *is* σ*-satisfiable, and (vi)* $\langle \mathcal{R}', \mathcal{A}', \overline{\ }'\rangle \models_\sigma \langle \mathcal{E}^+, \mathcal{E}^-\rangle$*.*

A σ*-solution is* intensional *when* $\mathcal{R}' \setminus \mathcal{R}$ *consists of rule schemata constructed out of predicates and variables only.*

In Definition 3, condition (ii) requires that the head predicate of a learnt rule is either an "old" predicate (in the background knowledge or in the examples), in which case it needs to be a learnable predicate in $\mathcal{T}$, or a new predicate (not already in the background knowledge, e.g. the contrary of a new assumption in $\mathcal{A}' \setminus \mathcal{A}$), in which case it does not need to be in $\mathcal{T}$. Intensionality captures a notion of generality for the learnt rules, i.e., rules defined in purely conceptual terms that do not make explicit reference to specific elements in the universe $\mathcal{U}$.

Our notion of σ-solution reflects the learning scenario where examples describe *one* of the extensions the learnt ABA framework should admit, i.e., considering positive/negative examples as claims that are accepted/not accepted, respectively, in that extension.

Example 4. *Let us consider the following ABA learning problem* $\langle\langle \mathcal{R}, \mathcal{A}, \overline{\ }\rangle, \langle \mathcal{E}^+, \mathcal{E}^-\rangle, \mathcal{T}\rangle$ *inspired by the classical Nixon diamond problem (Reiter and Criscuolo 1981):*

$\mathcal{R} = \{\ quaker(a) \leftarrow,\ quaker(b) \leftarrow,$
$\quad rep(a) \leftarrow,\ rep(b) \leftarrow\},$
$\mathcal{A} = \{\beta\}$*, where* β *is a bogus assumption with* $\overline{\beta} = c\_\beta$
$\mathcal{E}^+ = \{pacifist(a)\}$*,* $\mathcal{E}^- = \{pacifist(b)\}$*,* $\mathcal{T} = \{pacifist\}$*.*

Solutions include ABA frameworks with sets of rules $\mathcal{R}'_1$ *and* $\mathcal{R}'_2$ *whereby (for* $X \in \{a, b\}$*)*

$\mathcal{R}'_1 \setminus \mathcal{R} = \{pacifist(a) \leftarrow\}$*,*
$\mathcal{R}'_2 \setminus \mathcal{R} = \{pacifist(X) \leftarrow quaker(X), norm\_quaker(X),$
$\quad abnorm\_quaker(X) \leftarrow rep(X),\ norm\_rep(X),$
$\quad abnorm\_rep(X) \leftarrow quaker(X),\ norm\_quaker(X)\ \}$*,*

[4] (Lehtonen, Wallner, and Järvisalo 2021) also compute the *ideal* semantics (Dung, Mancarella, and Toni 2007), but with a mapping onto ASP as a subroutine for computing admissible sets of assumptions. We focus instead on ABA semantics that can be computed by a direct mapping μ_σ onto an ASP encoding. This encoding will be presented in Section 7, where $\mu_\sigma(F)$ will be defined as the union of a set $\texttt{ABA}(F)$ of ASP rules that represent the ABA framework and a set π_σ of ASP rules that encode the ABA semantics σ (see Definition 9).

with $\mathcal{A}' = \mathcal{A} \cup \{norm\_quaker(X), norm\_rep(X)\}$ *and:*
$\overline{norm\_quaker(X)}' = abnorm\_quaker(X)$,
$\overline{norm\_rep(X)}' = abnorm\_rep(X)$.

The ABA framework with rules $\mathcal{R}'_1$ *is a* σ*-solution for* $\sigma \in \{adm, com, grd, stb, prf\}$. *It is not intensional, as the term "a" occurs in the learnt rule pacifist*$(a) \leftarrow$. *The ABA framework with rules* $\mathcal{R}'_2$ *is an intensional* σ*-solution for* $\sigma \in \{adm, stb, com, prf\}$, *but it is not a* σ*-solution for* $\sigma \in \{grd\}$. *Indeed, it admits both a stable and a complete extension, and hence an admissible, extension* Δ *such that pacifist*(a) *is accepted in* Δ *and pacifist*(b) *is not. However, its grounded extension does not accept any pacifist claim, and hence it is not a grd-solution of the given problem.*

Finally, if we consider the ABA learning problem $\langle\langle \mathcal{R} \setminus \{rep(a) \leftarrow\}, \mathcal{A}, \overline{\ }\rangle, \langle \mathcal{E}^+, \mathcal{E}^-\rangle, \mathcal{T}\rangle$, *with* $\mathcal{R}, \mathcal{A}, \overline{\ }, \mathcal{E}^+, \mathcal{E}^-$, *and* $\mathcal{T}$ *defined as above, then the following set of rules, together with* $\mathcal{A}'$ *and* $\overline{\ }'$ *as above, constitutes an intensional* σ*-solution for* $\sigma \in \{adm, com, grd, stb, prf\}$*:*
$\mathcal{R}'_3 \setminus \mathcal{R} = \{$*pacifist*$(X) \leftarrow quaker(X), norm\_quaker(X)$,
$abnorm\_quaker(X) \leftarrow rep(X)\}$.

The following example shows that the choice of the set $\mathcal{T}$ in the definition of an ABA Learning problem is relevant for the existence of a σ-solution, bsides the choice of σ.

Example 5. *Let us now consider the ABA learning problem* $\langle\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\ }\rangle, \langle \mathcal{E}^+, \mathcal{E}^-\rangle, \mathcal{T}\rangle$ *where:*
$\mathcal{R} = \{p(X) \leftarrow q(X), not\_p(X), \quad r(a) \leftarrow\}$,
$\mathcal{A} = \{not\_p(X)\}, \quad \overline{not\_p(X)} = p(X)$,
$\mathcal{E}^+ = \{q(a)\}, \quad \mathcal{E}^- = \{q(b)\}, \quad \mathcal{T} = \{q\}$.
An intensional σ*-solution, for* $\sigma \in \{adm, grd, com, prf\}$ *is obtained by adding the rule* $q(X) \leftarrow r(X)$.
This is not an stb-solution. In fact, it is easy to see that no stb-solution exists. However, for the same ABA learning problem except that $\mathcal{T} = \{p, q\}$, *by adding the rule* $p(X) \leftarrow$, *we get an intensional stb-solution.*

We say that an ABA semantics σ is stronger than an ABA semantics σ' if every σ-extension is also a σ'-extension.

Proposition 1. *If* F *is a* σ*-solution of an ABA Learning problem and* σ *is stronger than* σ', *then* F *is also a* σ'*-solution of the ABA Learning problem.*

For instance, since every stable, or preferred, or grounded extension is also complete, and hence admissible, if an ABA Learning problem admits a *stb*/*prf*/*grd*-solution, then it also admits a *com*-solution, and hence an *adm*-solution.

5 Transformation Rules under σ-extensions

Now we prove some properties of the transformation rules presented in Section 3.2 with respect to σ-extensions. These properties will be used in Section 6 to prove the soundness and completeness of the *X-ABALearn* algorithm.

The goal of Rote Learning is to get a (non-intensional) σ-solution. The possibility of finding such a solution may depend on the chosen ABA semantics σ. For instance, for the ABA Learning problem of Example 5, with $\mathcal{T} = \{q\}$, by Rote Learning $q(a)$ we get a *grd*-solution, while there is no *stb*-solution. Thus, in order to design an algorithm, we introduce a function that computes the suitable facts to be be (rote) learnt, parametrically with respect to σ.

Definition 4. *For a set* G *of ground atoms, we denote* $\mathcal{R}(G) = \{p(X) \leftarrow X = t \mid p(t) \in G\}$. *Given an ABA Learning problem* $\langle\langle \mathcal{R}, \mathcal{A}, \overline{\ }\rangle, \langle \mathcal{E}^+, \mathcal{E}^-\rangle, \mathcal{T}\rangle$ *and an ABA semantics* σ, *a* rote learning solver Φ_σ *returns a set* G *of ground atoms such that* $\langle \mathcal{R} \cup \mathcal{R}(G), \mathcal{A}, \overline{\ }\rangle$ *is a* σ*-solution, if any* σ*-solution exists, and* $\bot$, *otherwise.*

We will show in Section 7 how Φ_σ can be realised for each $\sigma \in \{adm, stb, grd, com, prf\}$ by exploiting a mapping of ABA frameworks with any of these semantics into ASP (Lehtonen, Wallner, and Järvisalo 2021).

Example 6. *Let us consider the ABA Learning problem* $\langle\langle \mathcal{R}, \mathcal{A}, \overline{\ }\rangle, \langle \mathcal{E}^+, \mathcal{E}^-\rangle, \mathcal{T}\rangle$ *where* $X, Y \in \{$*alex, bob, carol, david, frank, john, mary*$\}$ *(the* ρ_i*'s denote rule identifiers), describing a legal context with the goal to learn rules that predict conviction:*
$\mathcal{R} = \{$ ρ_1 : *convicted*$(X) \leftarrow$ *w_con*(X, Y), *reliable*(Y),
ρ_2 : *reliable*$(X) \leftarrow$ *person*(X), *not_unreliable*(X),
ρ_3 : *unreliable*$(X) \leftarrow$ *person*(X), *not_reliable*(X),
ρ_4 : *w_con*$($*mary, alex*$) \leftarrow$, ρ_5 : *w_con*$($*david, carol*$) \leftarrow$,
ρ_6 : *w_con*$($*frank, ella*$) \leftarrow$, ρ_7: *w_con*$($*john, carol*$) \leftarrow$,
ρ_8: *caught_in_the_act*$($*bob*$) \leftarrow$, ρ_9 : *trustworthy*$($*alex*$) \leftarrow$,
ρ_{10} : *trustworthy*$($*carol*$) \leftarrow$, ρ_{11} : *trustworthy*$($*ella*$) \leftarrow$,
ρ_{12} : *unadmissible*$($*alex*$) \leftarrow$, ρ_{13} : *person*$(X) \leftarrow\}$
$\mathcal{A} = \{$ *not_reliable*(X), *not_unreliable*$(X)\}$,
$\overline{$*not_reliable*$(X)} =$ *reliable*(X),
$\overline{$*not_unreliable*$(X)} =$ *unreliable*(X),
$\mathcal{E}^+ = \{$*convicted(bob), convicted(david), convicted(john)*$\}$,
$\mathcal{E}^- = \{$*convicted(mary)*$\}$,
$\mathcal{T} = \{$*convicted, reliable, unreliable*$\}$
Rule ρ_1 *states that a defendant is convicted if s/he has a witness against* (w_con) *and this witness is reliable. Rules* ρ_2, ρ_3, *state that a witness cannot be reliable and unreliable at the same time. The other rules are facts about individuals, for instance unadmissible*$(alex) \leftarrow$ *means that alex' testimony was obtained unlawfully, e.g. through coercion.*

Let us consider the stb semantics. A rote learning solver Φ_{stb} *may introduce the new fact:*
ρ_{14}: *convicted*$(X) \leftarrow X =$ *bob*
The resulting ABA framework with rules $\mathcal{R} \cup \{\rho_{11}\}$ *is a non-intensional stb-solution. However, this solution might not be deemed satisfactory because it not only admits a stable extension that accepts all positive example and no negative examples, but also stable extensions where, for instance, mary is convicted, while neither david nor john are.*

In applications where we need definite predictions, we may want to avoid semantics admitting multiple extensions and, for instance, the grounded semantics may be a more suitable choice. In our example, Φ_{grd} *would allow Rote Learning to introduce, besides* ρ_{14}, *the new facts:*
ρ_{15}: *reliable*$(X) \leftarrow X =$ *carol*
ρ_{16}: *reliable*$(X) \leftarrow X =$ *ella*
The ABA framework with rules $\mathcal{R} \cup \{\rho_{14}, \rho_{15}, \rho_{16}\}$ *is a non-intensional grd-solution, and in its (unique) grd-extension bob, david, and john are convicted and mary is not.*

Rote Learning returns non-intentional solutions. We now show how they can be transformed into intensional ones. Folding allows the generalisation of learnt rules and the

derivation of intensional rules, but, as shown below, this transformation may affect the σ-entailment of the examples.

Example 7. *By folding rules ρ_{14} and ρ_{15} of Example 6 using rules ρ_8 and ρ_{10} (after normalisation), we get two intensional rules:*

ρ_{17}*: convicted*$(X) \leftarrow$ *caught_in_the_act*(X)

ρ_{18}*: reliable*$(X) \leftarrow trustworthy(X)$

The resulting ABA framework with rules $\mathcal{R} \cup \{\rho_{17}, \rho_{18}, \rho_{16}\}$ is not a grd-solution, as its grounded extension accepts the negative example convicted$(mary)$.

Assumption Introduction can be applied to render defeasible the rules obtained by Folding, for which the rote learning solver Φ_σ associated with σ computes a suitable set of facts representing exceptions. Then, by Rote Learning those facts, we recover a (non-intensional) σ-solution.

Example 8. *The ABA framework obtained in Example 7 can be transformed into a grd-solution by first applying Assumption Introduction, thereby getting*

ρ_{19}*: reliable*$(X) \leftarrow trustworthy(X), \alpha(X)$

and then applying Rote Learning, thereby getting:

ρ_{20}*:* $c\_\alpha(X) \leftarrow X = alex$.

The ABA framework with rules $\mathcal{R} \cup \{\rho_{17}, \rho_{19}, \rho_{20}, \rho_{16}\}$, is a (non-intensional) grd-solution for the problem of Example 6.

The following proposition generalises Proposition 2 in (De Angelis, Proietti, and Toni 2024) from *stb*-extensions to any ABA semantics σ. It guarantees that any ABA framework obtained by applying Folding to a σ-solution can be transformed, by Assumption Introduction and Rote Learning, into a (non-intensional) σ-solution.

Proposition 2. *Let $\langle \mathcal{R}_1, \mathcal{A}_1, {}^{-1} \rangle$ be a σ-solution of the ABA Learning problem $Prob = (\langle \mathcal{R}_0, \mathcal{A}_0, {}^{-0} \rangle, \langle \mathcal{E}^+, \mathcal{E}^- \rangle, \mathcal{T})$.*

Let $\mathcal{R}_2 = (\mathcal{R}_1 \setminus \{\rho_1\}) \cup \{\rho_3\}$ be obtained by Folding ρ_1, ρ_2 (as in R2, Table 1), to obtain $H \leftarrow Eqs_2, K, B_2$.

Let $\langle \mathcal{R}_3, \mathcal{A}_3, {}^{-3} \rangle$ be the ABA framework obtained by applying Assumption Introduction (as in R3, Table 1), where $\mathcal{R}_3 = (\mathcal{R}_2 \setminus \{\rho_3\}) \cup \{\rho_4\}$, $\rho_4 = H \leftarrow Eqs_2, K, B_2, \alpha(X)$, α is a new predicate symbol (i.e. not in $\langle \mathcal{R}_3, \mathcal{A}_3, {}^{-3} \rangle$ already), $vars(\{Eqs_1, B_1\}) \subseteq X = vars(\rho_4)$, $\mathcal{A}_3 = \mathcal{A}_1 \cup \{\alpha(X)\}$, and ${}^{-3} = {}^{-1} \cup \{\alpha(X) \mapsto c\_\alpha(X)\}$.

Then there exists a set G of atoms with predicate c_α, such that $G = \Phi_\sigma(\langle \mathcal{R}_3, \mathcal{A}_3, {}^{-3} \rangle, \langle \mathcal{E}^+, \mathcal{E}^- \rangle, \mathcal{T})) \neq \bot$ and $\langle \mathcal{R}_3 \cup \mathcal{R}(G), \mathcal{A}_3, {}^{-3} \rangle$ is a σ-solution of $Prob$.

Finally, we introduce a restricted version of Fact Deletion preserving σ-entailment of $\langle \mathcal{E}^+, \mathcal{E}^- \rangle$.

Definition 5. *Let $\rho : p(X) \leftarrow X = t \in \mathcal{R}$ be such that $\langle \mathcal{R} \setminus \{\rho\}, \mathcal{A}, {}^{-} \rangle \models_\sigma \langle \mathcal{E}^+, \mathcal{E}^- \rangle$. Then, we say that Fact Deletion of ρ (see R4 in Table 1)* preserves the σ-entailment of $\langle \mathcal{E}^+, \mathcal{E}^- \rangle$.

Note that Fact Subsumption in (De Angelis, Proietti, and Toni 2024) preserves stb-entailment of $\langle \mathcal{E}^+, \mathcal{E}^- \rangle$.

Example 9. *Continuing Example 8, fact ρ_{16} can be deleted, as $\langle \mathcal{R}', \mathcal{A}', {}^{-\prime} \rangle \models_{grd} \langle \mathcal{E}^+, \mathcal{E}^- \rangle$, for $\mathcal{R}' = \mathcal{R} \cup \{\rho_{17}, \rho_{19}, \rho_{20}\}$ and $\langle \mathcal{E}^+, \mathcal{E}^- \rangle$ as in Example 6. Now, by a final Folding using rule ρ_{12}, from rule ρ_{20}, we get:*

ρ_{21}*:* $c\_\alpha(X) \leftarrow$ *unadmissible*(X).

The ABA framework with rules $\mathcal{R} \cup \{\rho_{17}, \rho_{19}, \rho_{21}\}$ is an intensional grd-solution of the problem in Example 6.

6 The *X-ABALearn* Algorithm

Now we introduce the *X-ABALearn* algorithm (Algorithm 1), to guide the application of the transformation rules towards the computation of intensional σ-solutions of a given ABA Learning problem. *X-ABALearn* is parametric with respect to: (i) a rote learning solver Φ_σ (see Definition 4) and (ii) a σ-entailment relation $\models_\sigma$ (see Definition 2), which depend on the chosen ABA semantics σ. (In Algorithm 1, we highlight in yellow the parts depending on σ.)

X-ABALearn makes use of an *applyFolding* function, which is defined as follows.

Definition 6. *Let $\mathcal{R}$ be a set of rules. applyFolding is a function that, for any rule ρ, applyFolding$(\rho, \mathcal{R})$ is a rule ρ_f obtained by repeatedly applying the Folding transformation rule to ρ using rules in $\mathcal{R}$.*

X-ABALearn also uses the notion of *assumption relative to*, in order to avoid introducing unnecessary assumptions.

Definition 7. *Suppose that $\rho = H \leftarrow B, \alpha(X)$ is a rule in $\mathcal{R}$ where $\alpha(X) \in \mathcal{A}$ is the only assumption occurring in the rule. Then, we say that $\alpha(X)$ is an* assumption relative to B.

Similarly to *ASP-ABAlearn$_B$*, *X-ABALearn* is the composition of the following two procedures *RoLe* and *Gen*.

- *RoLe* computes a non-intensional σ-solution of the input ABA Learning problem $(\langle \mathcal{R}_0, \mathcal{A}_0, {}^{-0} \rangle, \langle \mathcal{E}^+, \mathcal{E}^- \rangle, \mathcal{T})$. This is done by first applying Φ_σ, at line 4, to compute a suitable set G of atoms, and then by applying Rote Learning, at line 5, to add all facts in $\mathcal{R}(G)$ (see Definition 4).

  Directly from Definition 4, we get the following soundness and completeness result for *RoLe*.

  Theorem 1. *For any ABA Learning problem $\langle \langle \mathcal{R}, \mathcal{A}, {}^{-} \rangle, \langle \mathcal{E}^+, \mathcal{E}^- \rangle, \mathcal{T} \rangle$ and ABA semantics σ, the RoLe procedure does not fail iff $\langle \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, {}^{-} \rangle, \langle \mathcal{E}^+, \mathcal{E}^- \rangle, \mathcal{T} \rangle$ has a σ-solution. If RoLe does not fail, its output is a set $\mathcal{R}_l$ of rules such that the ABA framework $\langle \mathcal{R} \cup \mathcal{R}_l, \mathcal{A}, {}^{-} \rangle$ is a σ-solution.*

- *Gen* derives an intensional σ-solution by transforming each non-intensional rule ρ learnt by *RoLe* as follows.

  At line 9, ρ is (implicitly) removed by Fact Deletion if this preserves σ-entailment (indeed, ρ is not added to the set $\mathcal{R}_t$ to be transformed, if $\langle \mathcal{R}_t, \mathcal{A}, {}^{-} \rangle \models_\sigma \langle \mathcal{E}^+, \mathcal{E}^- \rangle$). This step may lead to the removal of many facts when, after iterating, Gen gets suitably generalised rules.

  At line 10, *Gen* repeatedly applies Folding to ρ (if it belongs to $\mathcal{R}_t$), using the rules in the background knowledge $\mathcal{R}_0$ and the rules learnt so far, so as to get a new intensional rule ρ_f. To apply Folding, *Gen* uses the *applyFolding* function.

  If the ABA framework obtained by Folding is not a σ-solution of the given ABA Learning problem (line 11), because there is no σ-extension that accepts all positive examples and no negative example, *Gen* applies Assumption Introduction (lines 12–21), followed by Rote Learning (lines 22–24). At line 13, the use of an assumption $\alpha(X)$ already present in $\mathcal{A}$ and appearing in a rule of the form $H \leftarrow B, \alpha(X)$ (formally, an *assumption relative to*

Algorithm 1: *X-ABALearn*

Input: $(\langle \mathcal{R}_0, \mathcal{A}_0, {}^{-_0}\rangle, \langle \mathcal{E}^+, \mathcal{E}^-\rangle, \mathcal{T})$: ABA learning problem; σ: ABA semantics
Output: $\langle \mathcal{R}, \mathcal{A}, {}^{-}\rangle$: intensional σ-solution

1 $\mathcal{R} := \mathcal{R}_0$; $\mathcal{A} := \mathcal{A}_0$; ${}^{-} := {}^{-_0}$; $\mathcal{R}_l := \emptyset$;
2 *RoLe*(); *Gen*(); **return** $\langle \mathcal{R}, \mathcal{A}, {}^{-}\rangle$;
3 **Procedure** *RoLe*()
4 $G := \Phi_\sigma((\langle \mathcal{R}, \mathcal{A}, {}^{-}\rangle, \langle \mathcal{E}^+, \mathcal{E}^-\rangle, \mathcal{T}))$;
/* - Rote learning - */
5 **if** $G = \bot$ **then** fail; **else** $\mathcal{R}_l := \mathcal{R}_l \cup \mathcal{R}(G)$;
6 **Procedure** *Gen*()
7 **foreach** $\rho : (p(X) \leftarrow X = t) \in \mathcal{R}_l$ **do**
8 $\mathcal{R}_l := \mathcal{R}_l \setminus \{\rho\}$; $\mathcal{R}_t := \mathcal{R} \cup \mathcal{R}_l$;
/* - Fact Deletion - */
9 **if** $\langle \mathcal{R}_t, \mathcal{A}, {}^{-}\rangle \not\models_\sigma \langle \mathcal{E}^+, \mathcal{E}^-\rangle$ **then**
/* - Folding - */
10 $\rho_f := \mathit{applyFolding}(\rho, \mathcal{R})$;
/* - Assumption Introduction - - - - - - - - - - */
11 **if** $\langle \mathcal{R}_t \cup \{\rho_f\}, \mathcal{A}, {}^{-}\rangle \not\models_\sigma \langle \mathcal{E}^+, \mathcal{E}^-\rangle$ **then**
12 let $\rho_f = (H \leftarrow B)$; $X := \mathit{vars}(B)$;
13 **if** $\exists\, \alpha(X) \in \mathcal{A}$ *relative to* B **then**
14 $\mathcal{R} := \mathcal{R} \cup \{H \leftarrow B, \alpha(X)\}$;
15 **if** $\langle \mathcal{R}, \mathcal{A}, {}^{-}\rangle \not\models_\sigma \langle \mathcal{E}^+, \mathcal{E}^-\rangle$ **then**
16 fail;
17 **end**
18 **else** /* introduce a new assumption */
19 for $\alpha(X) \notin \mathcal{A}$, $\mathcal{A} := \mathcal{A} \cup \{\alpha(X)\}$;
20 $\mathcal{R} := \mathcal{R} \cup \{H \leftarrow B, \alpha(X)\}$;
21 ${}^{-} := {}^{-} \cup \{\alpha(X) \mapsto c\_\alpha(X)\}$;
/* - Rote Learning - - - - - - - - - - - */
22 $F := \langle \mathcal{R}, \mathcal{A}, {}^{-}\rangle$;
23 $G := \Phi_\sigma\ ((F, \langle \mathcal{E}^+, \mathcal{E}^-\rangle, \{c\_\alpha\}))$;
24 $\mathcal{R}_l := \mathcal{R}_l \cup \mathcal{R}(G)$;
25 **end**
26 **end**
27 **end**
28 **end**

B – see Definition 7), in the case where an assumption has to be added to a new rule with the same body B, ensures the termination of *X-ABALearn* under suitable hypotheses discussed below. Proposition 2 guarantees that these applications of Assumption Introduction and Rote Learning derive a new σ-solution by finding suitable exceptions to the rule learnt by Folding. *Gen* is iterated until all learnt rules are intensional, or a failure is reported.

We say that *X-ABALearn terminates with failure* for a given ABA Learning problem if it halts by executing the "fail" statement at line 16. If it halts without executing "fail", then it *terminates with success*.

In the following definition we indicate the properties of the *applyFolding* function used for proving the soundness and termination of *X-ABALearn* below.

Definition 8. *Given a set $\mathcal{R}$ of rules, applyFolding is*

1. complete *if, for all $\rho \in \mathcal{R}$, applyFolding$(\rho, \mathcal{R})$ is an intensional rule.*
2. bounded *if there exists $k \geq 0$ such that, for any fact $p(X) \leftarrow X = t$, size(applyFolding$(\rho, \mathcal{R})) \leq k$.*

Let us now establish the termination and soundness of *X-ABALearn* for any ABA semantics σ.

Theorem 2. *For all ABAlearn problems,* X-ABALearn *with any ABA semantics σ and a complete and bounded applyFolding function terminates. If it terminates with success, then it returns an intensional σ-solution.*

7 *X-ABALearn* Implementation

ASP-ABAlearn$_B$ (De Angelis, Proietti, and Toni 2024) relies upon an ASP implementation (in a combination of Clingo (Gebser et al. 2019) and SWI-Prolog (Wielemaker et al. 2012)) of the stable extension semantics at object level. This means that an ABA Learning problem is encoded directly as an ASP program P by using negation-as-failure to represent assumptions, thereby reducing $\models_{stb}$ to checking satisfiability of P, and Φ_{stb} to computing answer sets of P.

We have realised a proof-of-concept implementation of *X-ABALearn* (again, in a combination of Clingo and SWI-Prolog) by lifting *ASP-ABAlearn$_B$*[5] from the native ASP encoding of the stable extension semantics and adding an interface module to plug-in suitable implementations of $\models_\sigma$ and Φ_σ, for each σ. Neither the transformation rules nor the transformation strategy required changes to enable learning ABA frameworks under a different ABA semantics. Even if the implementations of $\models_\sigma$ and Φ_σ do not need, in principle, to be ASP-based, in this work we have leveraged on the ASP encoding $\texttt{ABA}(F)$ of an ABA framework F introduced by (Lehtonen, Wallner, and Järvisalo 2021) and the implementations π_σ of the semantics (see Figure 1 for $\sigma = adm$). This meta-level approach immediately gave us a working implementation for experimenting *X-ABALearn* under various semantics.

Given any ABA framework F, in the ASP encoding $\texttt{ABA}(F)$, we use the additional predicate `dom(t)`, which holds for all tuples `t` of constants in $\mathcal{L}$. For instance, rule ρ_1 of Example 6 is encoded as follows.

```
head(id(1,X,Y),convicted(X)) :- dom(X),dom(Y).
body(id(1,X,Y),w_con(X,Y)) :- dom(X),dom(Y).
body(id(1,X,Y),reliable(Y)) :- dom(X),dom(Y).
assumption(not_reliable(X)) :- dom(X).
contrary(not_reliable(X),reliable(X)) :- dom(X).
```

where `id(1,X,Y)` is a unique identifier for rules, and `dom` is defined by `dom(alex). ... dom(john).` (ASP facts).

Definition 9. *For an ABA semantics σ, we denote by $\Pi_\sigma(\langle F, \langle \mathcal{E}^+, \mathcal{E}^-\rangle, \mathcal{T}\rangle)$ the union of (a) $\texttt{ABA}(F)$, (b) π_σ, and the following sets of ASP rules:*

(c) $\{$ `:- not supported(e).` $\mid$ `e` $\in \mathcal{E}^+$ $\}$
(d) $\{$ `:- supported(e).` $\mid$ `e` $\in \mathcal{E}^-$ $\}$
(e)
```
{  head(id(n,X),p(X)) :- new_p(X).,
   { new_p(X) } :- dom(X).,
   #minimize{1,X : new_p(X)}  | p ∈ T}
```

[5] https://github.com/ABALearn/aba_asp

```
% π_common
in(X) :- assumption(X), not out(X).
out(X) :- assumption(X), not in(X).
supported(X) :- assumption(X), in(X).
supported(X):- head(R,X), supported(Y): body(R,Y).
defeated(X) :- supported(Y),contrary(X,Y).
:- in(X), defeated(X).
% Δ_adm
derived_from_undefeated(X) :- assumption(X),
  not defeated(X).
derived_from_undefeated(X) :- head(R,X),
  triggered_by_undefeated(R).
triggered_by_undefeated(R) :- head(R,_),
  derived_from_undefeated(X) : body(R,X).
attacked_by_undefeated(X) :- contrary(X,Y),
  derived_from_undefeated(Y).
:- in(X), attacked_by_undefeated(X).
```

Figure 1: ASP encoding π_{adm}.

with n a new positive integer that does not occur in any `id` *term of* `head` *atoms in* $\texttt{ABA}(F)$.

Hence, we have the following mapping into ASP of the σ-entailment relation $\models_\sigma$:

$F \models_\sigma \langle \mathcal{E}^+, \mathcal{E}^- \rangle$ iff $sat(\Pi_\sigma(\langle F, \langle \mathcal{E}^+, \mathcal{E}^- \rangle, \emptyset \rangle))$,

and of the rote learning solver Φ_σ:

$\Phi_\sigma(\langle F, \langle \mathcal{E}^+, \mathcal{E}^- \rangle, \mathcal{T} \rangle) =$

G if G is an answer set of $\Pi_\sigma(\langle F, \langle \mathcal{E}^+, \mathcal{E}^- \rangle, \mathcal{T} \rangle)$,

$\bot$ if $\Pi_\sigma(\langle F, \langle \mathcal{E}^+, \mathcal{E}^- \rangle, \mathcal{T} \rangle)$ is unsatisfiable.

`new_p` distinguishes new facts from the atoms `p(X)` that are already consequences of the ASP rules (a)–(d). This is realized in Clingo by using (i) the *choice rule* (Gebser et al. 2012) $\{\texttt{new\_p(X)}\} :- \texttt{dom(X)}$, non-deterministically selecting a subset of $\{\texttt{new\_p(t)} \mid \texttt{dom(t)}\}$, and (ii) the directive $\#\texttt{minimize}\{1, \texttt{X} : \texttt{new\_p(X)}\}$, enforcing the computation of answer sets with *minimal* subsets of `new_p` atoms, and hence the addition of a minimal set of new facts by Φ_σ.

The following properties of $\Pi_\sigma(\langle F, \langle \mathcal{E}^+, \mathcal{E}^- \rangle, \mathcal{T} \rangle)$ guarantee the soundness of our ASP implementation.

Theorem 3. *Let F be an ABA framework. $F \models_\sigma \langle \mathcal{E}^+, \mathcal{E}^- \rangle$ iff $\Pi_\sigma(\langle F, \langle \mathcal{E}^+, \mathcal{E}^- \rangle, \emptyset \rangle))$ is satisfiable.*

Theorem 4. *(1) There exists a solution of the ABA learning problem $\langle F, \langle \mathcal{E}^+, \mathcal{E}^- \rangle, \mathcal{T} \rangle$ iff $\Pi_\sigma(\langle F, \langle \mathcal{E}^+, \mathcal{E}^- \rangle, \mathcal{T} \rangle))$ is satisfiable. (2) If G is an answer set of $\Pi_\sigma(\langle F, \langle \mathcal{E}^+, \mathcal{E}^- \rangle, \mathcal{T} \rangle))$, then $\langle \mathcal{R} \cup \mathcal{R}(G), \mathcal{A}, \overline{\ } \rangle$ is a σ-solution of $\langle F, \langle \mathcal{E}^+, \mathcal{E}^- \rangle, \mathcal{T} \rangle$.*

8 Experimental evaluation

The goal of this section is twofold. First, in Section 8.1, we explore the usefulness of the parametric nature of *X-ABALearn* and its implementation, by considering some example ABA learning problems which, albeit simple, require interesting reasoning patterns. Then, in Section 8.2, we stress-test the computational viability of our implementation, in comparison with the baseline *ASP-ABAlearn$_B$*, on the ABA learning problems drawn from the tabular datasets of the UCI ML repository (Kelly, Longjohn, and Nottingham 2023) already considered in (De Angelis, Proietti, and Toni 2024). While the first part shows that adopting different semantics gives high flexibility and may lead to different ABA learning solutions suitable for different settings, the second part shows a high computational overhead for the implementation, compared to *ASP-ABAlearn$_B$*, deriving from our reliance on meta-interpretation in ASP.

We focus our experimental evaluation on instances of *X-ABALearn* and *ASP-ABAlearn$_B$* as a representative ILP system focusing on a single (stable) semantics, ignoring others such as the aforementioned ILASP and FastLAS. However, our results here can be combined with those reported in (De Angelis, Proietti, and Toni 2024), between *ASP-ABAlearn$_B$* and these other ILP systems, for an indirect comparison.

8.1 Qualitative Evaluation

Let us first define how to make predictions from a σ-solution $F = \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\ } \rangle$ of a given $\langle F_0, \langle \mathcal{E}^+, \mathcal{E}^- \rangle, \mathcal{T} \rangle$.

Definition 10. *Let $e \in \mathcal{L}$. We say that*

- F credulously infers e (under σ) *if $F \models_\sigma e$ (see Sect. 3);*
- F positively infers e (under σ) *if $F \models_\sigma \langle \mathcal{E}^+ \cup \{e\}, \mathcal{E}^- \rangle$.*

Credulous inference is a straightforward adaptation of *credulous reasoning* (Dung, Mancarella, and Toni 2002) to the context of ABA Learning: (a new example) e is credulously inferred in F if there exists a σ-extension of F that accepts e. Positive inference is more restrictive: it requires the existence of an extension that accepts all examples in $\mathcal{E}^+ \cup \{e\}$ and no example in $\mathcal{E}^-$. Thus, for any σ, if F positively infers e under σ, then F credulously infers e under σ, but not vice versa.

The notion of positive inference is consistent with the one used by *ASP-ABAlearn$_B$* and ILASP for $\sigma = stb$. Positive inference is easily implemented by representing examples via constraints as shown at Points (c) and (d) of Definition 9.

We now present and discuss inference for some examples under various semantics.

Example 10. *A classical example taken from (Dimopoulos and Kakas 1995) is as follows (where png and spng stand for penguin and superpenguin, respectively):*

$\mathcal{R} = \{$*bird*$(b1) \leftarrow$, *bird*$(b2) \leftarrow$, *png*$(p1) \leftarrow$, *png*$(p2) \leftarrow$,
spng$(sp1) \leftarrow$, *spng*$(sp2) \leftarrow$,
bird$(X) \leftarrow$ *png*(X), *png*$(X) \leftarrow$ *spng*$(X)\}$,

$\mathcal{E}^+ = \{$*flies*$(b1)$,*flies*$(sp1)\}$ $\mathcal{E}^- = \{$*flies*$(p1)\}$.

For $\sigma = adm$, X-ABALearn *computes, as a first result, a σ-solution F_{adm} with the following rules:*

$\mathcal{R}_{adm} = \{$*flies*$(X) \leftarrow bird(X), \alpha_1(X)\} \cup \mathcal{R}$

with no rule for the contrary $c\_\alpha_1(X)$ of $\alpha_1(X)$. F_{adm} positively (hence credulously) infers flies(X) for all $X \in \{b1, b2, p1, p2, sp1, sp2\}$, thus showing a poor predictive performance. In particular, F_{adm} positively infers flies$(p2)$, which in $\mathcal{R}$ has the same characterisation of the negative example $p1$ (they are both penguins).

For $\sigma = grd$, X-ABALearn *learns F_{grd} with rules:*

$\mathcal{R}_{grd} = \{\ c\_\alpha_1(X) \leftarrow png(X), \alpha_2(X),$
$c\_\alpha_2(X) \leftarrow spng(X)\} \cup \mathcal{R}_{adm}$

F_{grd} positively (and credulously) infers flies(X) for $X \in \{b1, b2, sp1, sp2\}$, thus showing optimal prediction accuracy. Note that F_{grd} is also an stb-, and by Proposition 1 a com- and prf-solution. In contrast, F_{adm} is not a solution under the other semantics we consider here.

Example 11. *Let us consider the ABA Learning problem of Example 4. For $\sigma = grd$,* X-ABALearn *fails to find a solution (indeed, no grd-solution exists). For $\sigma \in \{adm, stb, com, prf\}$,* X-ABALearn *computes a σ-solution F' with rules $\mathcal{R}'_2$. Note that, by construction, F' does not positively infer the negative example pacifist(b), while it credulously infers this example.*

Example 12. *Let us consider again the ABA Learning problem of Example 6. For $\sigma \in \{adm, stb, com, prf\}$,* X-ABALearn *computes only the rule (see Example 7):*

ρ_{17}*: convicted$(X) \leftarrow$ caught_in_the_act(X)*

Thus, the ABA framework F' with rules $\mathcal{R} \cup \{\rho_{17}\}$ is an intensional σ-solution for all $\sigma \in \{adm, stb, com, prf\}$. X-ABALearn *also computes the grd-solution F'' with rules $\mathcal{R} \cup \{\rho_{17}, \rho_{19}, \rho_{21}\}$ shown in Examples 7–9.*

It is interesting to compare the two solutions F' and F'' in terms of their predictive behaviour. If we take any new individual b for whom caught_in_the_act(b) holds, then both ABA frameworks positively infer convicted(b). However, if we take a new individual with the same characteristics of $mary$, that is, an m such that w_con(m, a), trustworthy(a), and unadmissible(a) hold, then m is positively inferred by F' (unlike mary), while m is not positively inferred by F''.

Example 13. *Let us consider the following ABA Learning problem based on a variant of the barber paradox.*

$\mathcal{R} = \{$ *shaves$(f, X) \leftarrow$ shavable(X), not_shaves(X, X),*
person$(a) \leftarrow$, person$(b) \leftarrow$, person$(f) \leftarrow$,
beardless$(b) \leftarrow$ $\}$

$\mathcal{A} = \{$*not_shaves(X)*$\}$ with $\overline{\text{not\_shaves}(X)} =$ *shaves(X),*
$\mathcal{E}^+ = \{$*shavable(a)*$\}$, $\mathcal{E}^- = \{$*shavable(b)*$\}$, $\mathcal{T} = \{$*shavable*$\}$.*
This has no stb-solution, simply because the background knowledge is not stb-satisfiable. However, X-ABALearn *can compute an intensional prf-solution, by learning the rules:*

shavable$(X) \leftarrow person(X), \alpha_1(X)$
$c\_\alpha_1(X) \leftarrow$ beardless(X)

This is also an adm-, grd-, and com-solution.

Our small, yet non-trivial, examples show the versatility and modularity of our approach: by plugging in $\models_\sigma$ and Φ_σ for different σ, *X-ABALearn* obtain solutions with a wide variety of predictive behaviour with different semantics.

Note that credulous inference is often not very satisfactory, as it may predict more false positives than positive inference (see, e.g., Example 11). Indeed, when the learnt ABA framework admits more than one σ-extension, for the credulous inference of a new example e it is enough that e is accepted in one of these extensions, say Δ, without requiring, as positive inference does, that Δ also accepts all positive and does not accept any negative examples.

Also, the use of admissible semantics can be the source of poor predictive accuracy (see, e.g., Example 10), even though, by Proposition 1, finding an *adm*-solution may be easier than finding other solutions. This limitation is inherent to the admissible semantics itself, which allows "too many" extensions (recall that all *com*-, *grd*-, *prf*-, or *stb*-extensions are *adm*). Further empirical evidence against the admissible semantics is shown in Section 8.2.

Dually, the uniqueness of *grd* extensions may improve predictive accuracy, when we are able to find solutions under this semantics. Finally, we have presented cases where *grd*-solutions exist and *stb*-solutions do not (Example 13), and also cases where the opposite holds (Example 11).

8.2 Quantitative Evaluation

We used the three tabular datasets for binary classification considered in (De Angelis, Proietti, and Toni 2024) (encoding values of their tuples as facts of the background knowledge, and examples as positive or negative according to their class) to evaluate the overhead of the meta-level interpretation of ABA semantics in *X-ABALearn*.

We ran *X-ABALearn* under each semantics on the datasets. Each run considers a subset of the examples: 100%, 50%, and 15%, respectively, obtained from a 5-fold cross validation process (4 folds used for learning the ABA framework, and 1 fold for inference). We also ran *ASP-ABAlearn$_B$* as a baseline for stable extensions. We conducted the experiments on an Intel Core i7-1185G7 processor with 32GB RAM, using a timeout of 900s per ABA learning problem. Results reported in Table 2 show that *X-ABALearn* under admissible semantics can learn faster than any other instance of *X-ABALearn* and *ASP-ABAlearn$_B$*. However, as already seen in Example 10, it learns solutions that, at prediction time, perform poorly in terms of accuracy and precision (see Table 3), compared to solutions learnt under other semantics. In fact, the ABA frameworks learnt under all other semantics agree as they are 'stratified' and admit a single extension under semantics other than *adm*.

Table 2 also shows that the time taken by *X-ABALearn* under grounded semantics is very high. This is mainly due to the fixed-point computation of grounded assumption sets within the ASP encoding of the semantics done by (Lehtonen, Wallner, and Järvisalo 2021). Finally, the results of *X-ABALearn* under stable semantics compared to *ASP-ABAlearn$_B$* show that the overhead cost of the meta-level interpretation is very significant, leading *X-ABALearn* to timeout on large problems, such as *autism*, even if we consider only 50% of the dataset. This is mainly due to the specific representation choices underpinning the ABA solver implemented at the meta-level (splitting each ABA rule into separate representations of heads and elements of bodies by using ASP facts), which cause high grounding costs for Clingo. For example, in the case of 'breast-w', with 50% of the dataset, the 544.44s taken by *X-ABALearn* (see Table 2) breaks down into the following components: 7.42s are needed to apply the transformation rules for ABA learning; 535.19s is the Clingo wall-clock for preprocessing and grounding the ASP program before the solving phase; 1.83s is the amount of time taken by Clingo on solving. Partial evaluation of the ASP interpreter for stable model semantics, specifically, the 'supported' predicate, w.r.t. the facts for predicates 'head' and 'body', can speed-up the ASP grounding phase, and therefore improve scalability. However, to the best of our knowledge, off-the-shelf partial evaluators for ASP do not exist, and we leave to future work the task of studying program specialization techniques, such as those presented in (De Angelis et al. 2022), for their application to our context. Another source of inefficiency is the choice rule used in Φ_σ: instantiating it over the whole do-

	dataset	acute	breast-w	autism
	#BK	494	649	674
	#features	6	9	17
	$\langle \lvert\mathcal{E}^+\rvert, \lvert\mathcal{E}^-\rvert \rangle$	$\langle 61, 59 \rangle$	$\langle 458, 241 \rangle$	$\langle 189, 515 \rangle$
100%	π_{adm}	0.06	1.76	1.94
	π_{com}	0.41	timeout	timeout
	π_{prf}	0.40	timeout	timeout
	π_{grd}	121.60	timeout	timeout
	π_{stb}	0.32	timeout	timeout
	`asp-stb`	0.17	18.21	59.52
50%	π_{adm}	0.04	0.54	0.55
	π_{com}	0.21	727.55	timeout
	π_{prf}	0.21	727.54	timeout
	π_{grd}	16.55	timeout	timeout
	π_{stb}	0.18	544.44	timeout
	`asp-stb`	0.11	5.37	20.03
15%	π_{adm}	0.03	0.09	0.09
	π_{com}	0.08	7.01	9.96
	π_{prf}	0.08	7.14	10.10
	π_{grd}	0.44	705.66	timeout
	π_{stb}	0.07	5.09	7.27
	`asp-stb`	0.06	0.70	0.86

Table 2: Results of the experimental evaluation with the *acute*, *breast-w* and *autism* UCI datasets ('#BK' gives the number of rules in the background knowledge, '#features' the number of predicates in BK, and '$\langle \lvert\mathcal{E}^+\rvert, \lvert\mathcal{E}^-\rvert \rangle$' the number of positive and negative examples). In each run we consider 100%, 50%, and 15% of the datasets, and report the learning times (wall-clock in seconds, averaged on the iterations of the 5-fold cross validation process) grouped by run by *X-ABALearn* and *ASP-ABAlearn*$_B$ (`asp-stb`)

	accuracy	*precision*	*recall*	*F1 score*
π_{adm}	0.49	0.49	1.00	0.66
others	1.00	1.00	1.00	1.00

Table 3: Performance results with 100% of the *acute* dataset, under π_{adm} and any other semantics (*others*, all giving the same results)

main (see point *(e)* of Definition 9) does not scale up with a naive "generate-and-test" approach for generating subsets of contraries. To tackle this issue we could adopt an approach similar to that used in (De Angelis, Proietti, and Toni 2024), where the grounding of the choice rule is restricted to the constants of the domain of the corresponding assumption.

9 Conclusions

We defined and implemented *X-ABALearn*, a novel, semantics-agnostic symbolic learning method. Our main message is that there is no need of rethinking the learning method from scratch for each chosen semantics. It is enough to develop strongly delimited semantics-dependent modules (the entailment relation and the rote learning solver), while keeping the overall structure of the *X-ABALearn* algorithm.

Our experimental evaluation shows potential for a wide variety of predictive behaviours with different semantics. However, for the learning task, the meta-level interpretation approach followed in our proof-of-concept implementation works well only for small ABA learning problems. In order to scale up, it may be necessary to resort to program optimisation techniques such as partial evaluation (Leuschel and Bruynooghe 2002), which could reduce the meta-interpretation overhead, and more sophisticated ASP implementation approaches (Weinzierl, Taupe, and Friedrich 2020), which could make the grounding phase of the ASP process much more efficient

Acknowledgments

De Angelis and Proietti are members of INdAM-GNCS. Toni was partially funded by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant 101020934, ADIX) and by EPSRC (grant UKRI3928, NeSyDebates).

AI Declaration

The authors have not employed any Generative AI tools.

References

Bondarenko, A.; Dung, P.; Kowalski, R.; and Toni, F. 1997. An abstract, argumentation-theoretic approach to default reasoning. *Artif. Intell.* 93:63–101.

Cocarascu, O.; Cyras, K.; and Toni, F. 2018. Explanatory predictions with artificial neural networks and argumentation. In *Proc. IJCAI/ECAI Workshop on Explainable Articial Intelligence (XAI-18)*.

Cropper, A.; Dumancic, S.; Evans, R.; and Muggleton, S. H. 2022. Inductive logic programming at 30. *Mach. Learn.* 111(1):147–172.

Cyras, K.; Fan, X.; Schulz, C.; and Toni, F. 2017. Assumption-based argumentation: Disputes, explanations, preferences. *FLAP* 4(8).

Cyras, K.; Satoh, K.; and Toni, F. 2016. Abstract argumentation for case-based reasoning. In Baral, C.; Delgrande, J. P.; and Wolter, F., eds., *Proc. KR 2016*, 549–552. AAAI Press.

De Angelis, E.; Fioravanti, F.; Gallagher, J. P.; Hermenegildo, M. V.; Pettorossi, A.; and Proietti, M. 2022. Analysis and transformation of constrained Horn clauses for program verification. *Theory and Practice of Logic Programming* 22(6):974–1042.

De Angelis, E.; Proietti, M.; and Toni, F. 2024. Learning brave assumption-based argumentation frameworks via ASP. In Endriss, U.; Melo, F. S.; Bach, K.; Diz, A. J. B.; Alonso-Moral, J. M.; Barro, S.; and Heintz, F., eds., *Proc. ECAI 2024*, volume 392 of *Frontiers in Artificial Intelligence and Applications*, 3445–3452. IOS Press.

De Angelis, E.; Proietti, M.; and Toni, F. 2025. Greedy ABA learning for case-based reasoning. In *Proc. AAMAS 2025*, 556–564.

Dimopoulos, Y., and Kakas, A. C. 1995. Learning non-monotonic logic programs: Learning exceptions. In *Proc. ECML 1995*, 122–137.

Dung, P.; Kowalski, R.; and Toni, F. 2009. Assumption-based argumentation. In *Argumentation in Artificial Intelligence*. Springer. 199–218.

Dung, P. M.; Mancarella, P.; and Toni, F. 2002. Argumentation-based proof procedures for credulous and

sceptical non-monotonic reasoning. In Kakas, A. C., and Sadri, F., eds., *Computational Logic: Logic Programming and Beyond, Essays in Honour of Robert A. Kowalski, Part II*, Lecture Notes in Computer Science, 289–310. Springer.

Dung, P. M.; Mancarella, P.; and Toni, F. 2007. Computing ideal sceptical argumentation. *Artif. Intell.* 171(10-15):642–674.

Dung, P. M. 1995. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.* 77(2):321–358.

Gebser, M.; Kaminski, R.; Kaufmann, B.; and Schaub, T. 2012. *Answer Set Solving in Practice*. Morgan & Claypool Publishers.

Gebser, M.; Kaminski, R.; Kaufmann, B.; and Schaub, T. 2019. Multi-shot ASP solving with clingo. *TPLP* 19(1):27–82.

Gelder, A. V.; Ross, K. A.; and Schlipf, J. S. 1991. The well-founded semantics for general logic programs. *J. ACM* 38(3):620–650.

Gelfond, M., and Lifschitz, V. 1988. The stable model semantics for logic programming. In *ICLP*, 1070–1080.

Gelfond, M., and Lifschitz, V. 1991. Classical negation in logic programs and disjunctive databases. *New Generation Computing* 9:365–385.

Kelly, M.; Longjohn, R.; and Nottingham, K. 2023. The UCI machine learning repository. https://archive.ics.uci.edu.

Law, M.; Russo, A.; Bertino, E.; Broda, K.; and Lobo, J. 2020. Fastlas: Scalable inductive logic programming incorporating domain-specific optimisation criteria. In *Proc. AAAI 2020*, 2877–2885. AAAI Press.

Law, M.; Russo, A.; and Broda, K. 2014. Inductive learning of answer set programs. In *Proc. JELIA 2014*, LNCS 8761, 311–325.

Lehtonen, T.; Wallner, J. P.; and Järvisalo, M. 2021. Declarative algorithms and complexity results for assumption-based argumentation. *J. Artif. Intell. Res.* 71:265–318.

Leuschel, M., and Bruynooghe, M. 2002. Logic program specialisation through partial deduction: Control issues. *Theory and Practice of Logic Programming* 2(4&5):461–515.

Muggleton, S. 1995. Inverse entailment and Progol. *New generation computing* 13(3-4):245–286.

Paulino-Passos, G., and Toni, F. 2021. Monotonicity and noise-tolerance in case-based reasoning with abstract argumentation. In *Proc. KR 2021*, 508–518.

Pettorossi, A., and Proietti, M. 1994. Transformation of logic programs: Foundations and techniques. *J. Log. Program.* 19/20:261–320.

Proietti, M., and Toni, F. 2024. Learning assumption-based argumentation frameworks. In *ILP 2022*, Lecture Notes in Computer Science (LNAI 13779), 100–116. Springer.

Reiter, R., and Criscuolo, G. 1981. On interacting defaults. In *Proc. IJCAI 1981*, 270–276. William Kaufmann.

Sakama, C., and Inoue, K. 2009. Brave induction: A logical framework for learning from incomplete information. *Mach. Learn.* 76(1):3–35.

Toni, F. 2014. A tutorial on assumption-based argumentation. *Argument & Computation* 5(1):89–117.

Wang, H.; Shakerin, F.; and Gupta, G. 2022. FOLD-RM: A scalable, efficient, and explainable inductive learning algorithm for multi-category classification of mixed data. *Theory and Practice of Logic Programming* 22(5):658–677.

Weinzierl, A.; Taupe, R.; and Friedrich, G. 2020. Advancing lazy-grounding asp solving techniques–restarts, phase saving, heuristics, and more. *Theory and Practice of Logic Programming* 20(5):609–624.

Wielemaker, J.; Schrijvers, T.; Triska, M.; and Lager, T. 2012. SWI-Prolog. *Theory and Practice of Logic Programming* 12(1-2):67–96.

A Proof of Proposition 2

Proof. By the definition of a σ-solution of an ABA Learning problem, $\langle \mathcal{R}_1, \mathcal{A}_1, {}^{-1} \rangle$ admits a σ-extension Δ such that, for all $e \in \mathcal{E}^+$, $\langle \mathcal{R}_1, \mathcal{A}_1, {}^{-1} \rangle \models_\Delta e$ and, for all $e \in \mathcal{E}^-$, $\langle \mathcal{R}_1, \mathcal{A}_1, {}^{-1} \rangle \not\models_\Delta e$. Let us consider the rule ρ_4': $H \leftarrow Eqs_1, Eqs_2, K, B_1, B_2, \alpha(X)$ obtained from ρ_1 by adding $Eqs_2, K, \alpha(X)$ to its body, where *vars*$(\{Eqs_1, B_1\}) \subseteq X$, and let $R_4' = (R_1 \setminus \{\rho_1\}) \cup \{\rho_4'\}$. By induction on the argument structure, for all s, $A_1 \vdash_{R_1} s \in \Delta$ iff $A_1 \cup \{\alpha(X)\} \vdash_{R_4'} s \in \Delta$ (note, in particular, that no rule for the contrary $c\_\alpha(X)$ of $\alpha(X)$ occurs in R_4'). Let $B_1 = s_1, \ldots, s_k$, let $G = \Phi_\sigma(\langle \mathcal{R}_3, \mathcal{A}_3, {}^{-3} \rangle, \langle \mathcal{E}^+, \mathcal{E}^- \rangle, \mathcal{T}\rangle) = \{c\_\alpha(t) \mid Eqs_1\{X/t\}$ *is a set of identities and* $\exists i \in \{1, \ldots, k\}. A_1 \cup \{\alpha(X)\} \not\models_{R_4'} s_i\}$, and let $R_4 = R_3 \cup \mathcal{R}(G)$ (that is, R_4 is obtained from R_4' by dropping Eqs_1, B_1 from the body of ρ_4' and adding to the resulting set of rules the facts for the contrary of $\alpha(X)$). Now, we can construct a new σ-extension Δ' such that, for all claims $s \neq c\_\alpha(X)$, $A_1 \vdash_{R_1} s \in \Delta$ iff $A_1' \vdash_{R_3} s \in \Delta'$, with $A_1' \subseteq (A_1 \cup \{\alpha(X)\}) \cap \overline{G}$. Let $\mathcal{F}_4 = \langle \mathcal{R}_4, \mathcal{A}_1 \cup \{\alpha(X)\}, {}^{-1} \cup \{\alpha(X) \mapsto c\_\alpha(X)\} \rangle$. Then, Δ' is a σ-extension such that, for all $e \in \mathcal{E}^+$, $\mathcal{F}_4 \models_{\Delta'} e$ and, for all $e \in \mathcal{E}^-$, $\mathcal{F}_4 \not\models_{\Delta'} e$, and hence $\mathcal{F}_4$ is a σ-solution of $(\langle \mathcal{R}_0, \mathcal{A}_0, {}^{-0} \rangle, \langle \mathcal{E}^+, \mathcal{E}^- \rangle, \mathcal{T})$. □

Proof of Theorem 2

Proof. (Sketch) The termination (with success or failure) of *X-ABALearn* is a consequence of the finiteness of the language $\mathcal{L}$, the boundedness of *applyFolding* (see Definition 8), and the *relative to* condition at line 13 of the *X-ABALearn* algorithm. Indeed, these conditions enforce that there exists a finite set $\mathcal{B}$ of atoms such that, for any ρ_f obtained from ρ by any sequence of applications of *fold*, $bd(\rho_f) \subseteq \mathcal{B}$ (up to a renaming of variables). Thus, the *relative to* condition guarantees that only a finite number of new assumptions can be introduced and a finite number of new rules can be learnt.

If *X-ABALearn* terminates with success, then each learnt rule is of the form $H \leftarrow B, \alpha(X)$, where $H \leftarrow B$ is obtained by *applyFolding* and $\alpha(X)$ may be absent. By the completeness of *applyFolding* (see Definition 8), $H \leftarrow B, \alpha(X)$ is intensional. □

Proofs of Theorems 3 and 4

Proof. (Sketch) Both follow from the correctness of the encoding of the semantics σ as $\texttt{ABA}(F) \cup \pi_\sigma$ (Lehtonen, Wallner, and Järvisalo 2021). □