10 Learning and Non-Monotonic Knowledge Representations

In deciding how to represent and reason with (learned) knowledge so far, we have mostly relied on classical logic, where the addition of new pieces of knowledge in an existing theory cannot reduce the set of its inferences. We glimpsed an alternative in Chapter 6 and in Chapter 8, where we demonstrated how the particular semantics that we considered for multi-layered policies effectively led to a form of non-monotonic reasoning. But, as we discussed in Section 6.3, "common sense" knowledge in particular often concerns generalities and "rules-of-thumb": conclusions that we presume to hold in general, but that may fail under exceptional circumstances. Competent use of such knowledge demands non-monotonic reasoning. We saw that reasoning with examples implicitly achieves such non-monotonic inferences in Section 6.4, but sometimes explicit, declarative knowledge is also provisional in the same way. Moreover, as we noted in Example 7.11, some prominent kinds of common-sense knowledge, notably the frame axioms, are not witnessed whenever they produce useful inferences; our techniques for learning from partial observations therefore could not capture such common-sense knowledge in any useful way. We therefore need an alternative to classical logic that allows for such provisional, "defeasible" inferences. We revisit the knowledge representation question in this chapter, by considering a generic and flexible form of non-monotonic reasoning, based on a formalization of the process of argumentation. We will first introduce the basic structure of arguments abstractly in Section 10.1, and present algorithms for drawing inferences about which arguments are acceptable at the abstract level in Section 10.2. We'll then consider interpreting rules (in the style of the chaining proof system, from Section 5.2) as arguments, and how we can efficiently decide what conclusions to draw from these arguments in a given context in Section 10.3. Finally, we will examine how the learning process can be extended to enable the design of efficient algorithms that learn to make these richer inferences in Section 10.4.

10.1 Abstract argumentation semantics.

The chapter discusses semantics for non-monotonic reasoning via *arguments*. Roughly speaking, an argument is an association between different concepts, with the truth of some of them being offered as a reason in support of or against the claimed truth of another concept. Thus, the argument that "Bob needs to take an umbrella because it

is raining outside" associates the concept of "Bob needs to take an umbrella" with the concept of "it is raining outside," and offers the latter as a reason in support of the former claim. As in classical reasoning, the interesting questions arise in theories with multiple pieces of knowledge. If, for instance, we have a second argument that "Bob does not need to take an umbrella, because Bob will not leave the house", then this offers the reason that "Bob will not leave the house" in support of the negation of the claim that "Bob needs to take an umbrella". What should we conclude if we are in a situation where both "it is raining outside" and "Bob will not leave the house" are true?

If we were to interpret both arguments as classical implications, then we end up with an inconsistent theory, as we infer both that "it is raining outside" and its negation. Under an argumentation semantics, we would gracefully handle the co-existence of the two conflicting arguments by stipulating how their conflict needs to be resolved. For instance, we could state that the second argument is stronger than, or has priority over, the first one, and thus, in case of conflict, the inference of the second one prevails. Argument strength is, in some cases, obvious, as in the celebrated pair of arguments that birds generally fly, but penguins do not. The concept of penguinhood being, in this case, a subset (i.e., a more restricted form) of the concept of birdhood, carries with it an inherent priority in supporting its associated claim that this particular bird does not fly. In the general case, however, priorities are not inherent, and are assumed to be provided by some external process. More interesting for us is that argument priorities can also be learned, along with the involved arguments themselves.

Even with given arguments and given priorities, however, there is no unique way of deciding what claims are supported when there are multiple arguments involved. It is exactly this question that formal argumentation seeks to answer: What are candidate "reasonable" semantics to determine what claims are supported by a set of arguments and priorities? The question remains meaningful even if we abstract away the details of how individual arguments are represented, and simply consider the set A of all arguments (the basic objects here, whatever they are) that are relevant in a particular context. Certain pairs of arguments can be in conflict with each other, in the sense that they cannot both be accepted to hold at the same time in that context. We specify that one argument has priority over another via a "defeat relation":

Definition 10.1 (Defeat relation and conflicting arguments) A *defeat relation* $R \subseteq A \times A$ on the set of arguments A indicates that one argument *defeats* (has priority over) the other, i.e., $(a, b) \in R$ means a defeats b. We say that the pairs in R are the *conflicting* arguments. We denote the set of all attackers of $a \in A$ by $R^{-1}(a) := \{ b \in A : (b, a) \in R \}$.

Note that for two conflicting arguments with incomparable priorities, we can include both orderings of the pair in R.

Definition 10.2 (Abstract Argumentation Framework) The tuple $\langle A, R \rangle$ of a set of arguments A and defeat relation R is an *abstract argumentation framework (AAF)*. A *semantics* σ of an abstract argumentation framework is a set of subsets of A that are said to be *acceptable*.

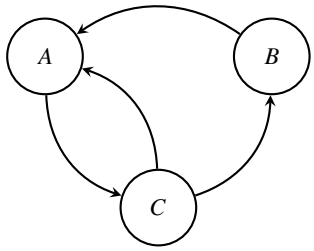


Figure 10.1 An abstract argumentation framework comprising three arguments and defeat relations between them, visualized as a directed graph.

Example 10.1: Consider a dialogue between two Athenian generals, arguing whether to wage war on Sparta. The first general puts forward the argument that (A) Sparta is a threat to Athens, and thus Athens should attack Sparta. The second general counter-argues that (B) if Athens attacks Sparta, then Thebes will consider Athens a threat, and they will attack Athens. The first general attempts to support his original argument by counter-arguing that (C) Athens can defend against an attack by Thebes by forming a coalition with one of Thebes' enemies, pointing out that Sparta is an enemy of Thebes and, therefore, a potential ally of Athens. The second general proceeds to point out that the two arguments put forth by the first general are in conflict, since attacking Sparta will preclude it from becoming an ally of Athens. The resulting AAF includes, therefore, three arguments, whose defeat relation can be visualized via the directed graph in Figure 10.1. Looking at the graph, one can naturally ask: What subsets of arguments can jointly be considered acceptable?

The example above already identifies a property that is key across different argumentation semantics: no acceptable subset of arguments in any semantics can include any pair of arguments that are related through the defeat relation, and hence are "in conflict". We shall call this the *conflict-free* criterion. There is a second, perhaps more subtle, property that follows from the example above: arguments in an acceptable subset of arguments can be "attacked" by other arguments, as long as this "attack" can, somehow, be "defended" by an argument in the acceptable subset. Building on these two properties, we define a first AAF semantics.

Definition 10.3 (Admissible Extension) Given an AAF $\langle A, R \rangle$, a subset $E \subseteq A$ of arguments is an *admissible extension* of $\langle A, R \rangle$ if E is conflict-free, and for every argument $a \in E$ such that $(b, a) \in R$ (i.e., a is "attacked by" b), there exists $c \in E$ such that $(c, b) \in R$ (i.e., c "defends" a by "attacking" b).

Example 10.2: With respect to our Athenian generals AAF, continuing Example 10.1, argument A cannot be part of an admissible extension, since it is attacked by argument B, and the only way to defend it is by including argument C in the extension, which would violate conflict-freeness. Analogously, argument B cannot be part of an admissible extension, either. However, argument C is part of an extension,

since its attack by argument A is defended by C itself. Thus, $\{C\}$ is an admissible extension of the AAF. Along with the empty subset of arguments, which also trivially meets the requirements of admissibility, this AAF has two admissible extensions.

Even before considering other ways of defining semantics for AAFs, we already see that even for a fixed semantics, an AAF can admit multiple admissible (or more generally, acceptable) extensions. These extensions could be seen to resemble the assignments that satisfy a classical theory. Much in the same way that in classical reasoning we determine entailment of a formula by a theory by checking whether the formula is satisfied by all the assignments that satisfy the theory, so in argumentation we can define an analogous notion of entailment.

Definition 10.4 (Skeptical and Credulous Entailment) Given an AAF $\langle A, R\rangle$, and a semantics σ determining the acceptable extensions of $\langle A, R\rangle$, an argument $a \in A$ is

- *skeptically entailed* by $\langle A, R\rangle$ (under the chosen semantics) if a belongs to *every*[1] acceptable extension in σ and
- *credulously entailed* by $\langle A, R\rangle$ (under the chosen semantics) if a belongs to *at least one* extension in σ.

In our Athenian generals example (Example 10.1–10.2), recall that the admissible extensions are $\emptyset$ and $\{C\}$. Argument C is therefore credulously entailed, but not skeptically entailed under the admissible extension semantics. Arguments A and B are neither credulously nor skeptically entailed.

The distinction between skeptical and credulous entailment highlights a central tension in non-monotonic reasoning: should we accept as conclusions only those claims that are *inevitable* across all possible acceptable views (skeptical), or is it reasonable to accept claims that are supported in *some* coherent views (credulous)? Different applications may call for different stances. For example, safety-critical decision-making often leans toward skeptical entailment, while exploratory reasoning may allow for credulous acceptance. In classical logic, entailment is effectively defined through a skeptical perspective.

Now, one could consider alternative ways of defining acceptability; i.e., different ways of determining under an argumentative reasoning semantics what constitutes the analogue of a satisfying assignment of a theory under the classical reasoning semantics. Before we do that, we introduce an auxiliary notion.

Definition 10.5 (Characteristic Function) Given an AAF $\langle A, R\rangle$, the *characteristic function* $F : \mathcal{P}(A) \to \mathcal{P}(A)$ (where $\mathcal{P}(A)$ denotes the set of all subsets of A) is defined for a set of arguments $E \subseteq A$ as

$$F(E) = \left\{ a \in A \ : \ \forall b \in R^{-1}(a)\ \exists c \in E \cap R^{-1}(b) \right\}.$$

Consequently, $F(E)$ is the set of all arguments that are defended by E.

[1] Under semantics that could have no extensions, the skeptical entailment definition can be refined to exclude the vacuous skeptical entailment of all arguments when no extensions exist.

With the use of the characteristic function, we can more concisely state the definition of the admissible extensions: Given an AAF $\langle A, R\rangle$, a subset $E \subseteq A$ of arguments is an *admissible extension* of $\langle A, R\rangle$ if E is conflict-free and $E \subseteq F(E)$. Varying the condition on $F(E)$ gives rise to other semantics.

Definition 10.6 (Complete, grounded, and preferred extensions) Given an AAF $\langle A, R\rangle$ with characteristic function F, the set of extensions E satisfying $E = F(E)$ are the *complete extensions*. The subset-least complete extension is the *grounded extension*. The subset-maximal admissible extensions are the *preferred extensions*.[2]

Strictly speaking, we should only define grounded extensions to be minimal extensions, since we have not yet shown that the grounded extension is unique. We will obtain this as a consequence of Theorem 10.9

Observe that the complete, grounded, and preferred extensions are all admissible. The grounded extension is a "skeptical" semantics, in the sense that it only includes the set of arguments that *must* be in *any* complete extension. By contrast, we say that the preferred extensions are "credulous," in that *every* conflict-free admissible set of arguments is included in some preferred extension.

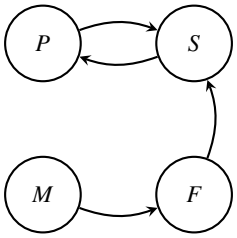
The semantics discussed above all rely on the same principle, which they inherit from their admissible nature: an extension is acceptable if its arguments can defend themselves (and possibly other arguments) against attacks. That is, essentially, a "tit-for-tat" approach, that says nothing about arguments outside the extension that are not a direct threat to the extension. One could consider, however, a more "offense-first" approach to defining semantics.

Definition 10.7 (Stable extensions) Given an AAF $\langle A, R\rangle$, a subset $E \subseteq A$ of arguments is a *stable extension* of $\langle A, R\rangle$ if E is conflict-free and every argument in $A \setminus E$ is attacked by some argument in E.

Thus, a stable extension E attacks every foreign argument, not merely defending against attackers.

To appreciate the differences among these semantics, we consider the *frame problem*, a classical problem from knowledge representation capturing the commonsense law of inertia: things persist unless there is a reason not to. A compact solution to the frame problem avoids an explosion of explicit axioms by introducing a generic schema — the *frame axiom* — that encodes this law of inertia. Argumentation provides a natural way to capture this compactness, by treating persistence as a default that can be overridden by exceptions. We will return to this point later in the chapter when discussing structured argumentation, but for now we illustrate how such reasoning can be encoded with abstract arguments.

[2] The presentation here follows the canonical definitions. Equivalently, one can characterize the grounded extension as the (unique) subset-minimal complete extension, and the preferred extensions as the subset-maximal complete extensions. This alternative view highlights a useful symmetry between grounded and preferred semantics.



Semantics	Extensions
Admissible	$\emptyset$, $\{P\}$, $\{M\}$, $\{P, M\}$, $\{S, M\}$
Complete	$\{M\}$, $\{P, M\}$, $\{S, M\}$
Grounded	$\{M\}$
Preferred	$\{P, M\}$, $\{S, M\}$
Stable	$\{P, M\}$, $\{S, M\}$

Figure 10.2 The AAF for the frame axiom example, along with its extensions.

Example 10.3: [Frame Axiom Arguments] The frame axiom can be illustrated with an AAF $\langle A, R\rangle$ containing the following set A of arguments:

- P: the light is on at t+1 because it was on at t (persistence).
- S: the light is off at t+1 because the switch was toggled (exception).
- F: the switch was faulty, so toggling does not work (defeater of exception).
- M: maintenance was performed, so the switch works (defeater of fault).

and defeat relation $R = \{(P, S), (S, P), (F, S), (M, F)\}$, illustrated in Figure 10.2 along with its extensions under different semantics.
Note that $\{F\}$ is not admissible, as it cannot defend itself against M, and $\{S\}$ is not admissible, as it cannot defend itself against F. Among the admissible extensions, $\emptyset$ and $\{P\}$ are not complete, since they (trivially) defend M but fail to include it. The subset-least complete extension is $\{M\}$, which is therefore the grounded extension. The two subset-maximal admissible extensions, $\{P, M\}$ and $\{S, M\}$, are the preferred extensions. In this example, these two preferred extensions also qualify as stable, since each attacks all arguments outside it. This illustrates how *complete* refines *admissible* by enforcing closure under defense, how *grounded* provides a cautious, unique choice, and how *preferred* admits multiple coherent viewpoints. Although preferred extensions in this particular case happen to coincide with stable extensions, we note that this is not typical.

In general, every AAF admits at least one admissible, complete, grounded, and preferred extension, though stable extensions may fail to exist. The grounded extension is always unique, whereas admissible, complete, preferred, and stable extensions can be multiple. In terms of inclusions, every stable extension is preferred, every preferred extension is complete, and every complete extension is admissible, with the grounded extension contained in every preferred extension. When AAFs are *well-founded* (i.e., contain no infinite defeat chains; or equivalently, in finite AAFs, contain no directed cycles), all these semantics coincide and yield a unique extension (the grounded one), rendering the distinction between skeptical and credulous entailment immaterial.

Acceptance Semantics	Extension Existence	Credulous Acceptance	Skeptical Acceptance
Admissible	Always (P)	NP-complete	coNP-complete
Complete	Always (P)	NP-complete	coNP-complete
Grounded	Always (P)	P	P
Preferred	Always (P)	NP-complete	coNP-complete
Stable	NP-complete	NP-complete	coNP-complete

Table 10.1 Computational complexity under different argumentation semantics.

10.2 The complexity of acceptability.

The various semantics defined for abstract argumentation provide a rich landscape of possible extensions, but they also raise fundamental computational questions. Given an AAF, how hard is it to determine whether a particular argument is entailed under a given semantics? How hard is it to compute one (or all) extensions? These questions are not merely theoretical; they are intimately tied to our broader aim of investigating learnability in a non-monotonic reasoning setting, since tractable learnability presupposes tractable reasoning.

Table 10.1 summarizes the known results for the central decision problems in abstract argumentation: existence of extensions, credulous acceptance (membership in at least one extension), and skeptical acceptance (membership in all extensions). While it is beyond our scope to establish all these complexity results, we will discuss the tractability of deciding existence across semantics, and then focus on the entailment problem under the grounded semantics. For simplicity of exposition, we assume that the AAFs are finite, although with more work the results can be extended to infinite AAFs as well.

Let us first consider an algorithm that constructs the unique grounded extension, and prove, therefore, the first few results in Table 10.1: that the (unique) grounded extension always exists, and that it can be reasoned with efficiently.

Algorithm 22 operates by iteratively assigning labels to arguments in a three-valued logic. The following lemma shows that these labels correspond to iterative application of the characteristic function.

Lemma 10.8 (Coupling to Characteristic Function) *Let E_t be the set of arguments labeled* IN *after the t-th full pass of Algorithm 22 (with $E_0 = O_0 = \emptyset$) and O_t be the set of arguments labeled* OUT. *Then, for all $t \geq 0$, $E_{t+1} = F(E_t)$. Moreover, $O_t \cap E_t = \emptyset$, each E_t is conflict-free, $E_t \subseteq E_{t+1}$, and $O_t \subseteq O_{t+1}$.*

Proof We establish the equality $E_{t+1} = F(E_t)$ via two inclusions.

($\subseteq$) If $a \in E_{t+1}$, the IN-rule fired for a in pass t+1, so every $b \in R^{-1}(a)$ was labeled OUT by the end of pass t. By the OUT-rule, each such b has some attacker c_b labeled IN by the end of pass t, i.e., $c_b \in E_t \cap R^{-1}(b)$. Hence $a \in F(E_t)$.

Input: A finite AAF $\langle A, R\rangle$
Output: Grounded extension $G \subseteq A$

Initialize: $E \leftarrow \emptyset$; // current IN snapshot
$O \leftarrow \emptyset$; // current OUT snapshot
while *true* **do**
  // Phase 1: compute next OUT from the current IN
  $O' \leftarrow \{ a \in A : \exists b \in R^{-1}(a)$ with $b \in E \}$;
  // Phase 2: compute next IN from the newly computed OUT′
  $E' \leftarrow \{ a \in A : R^{-1}(a) \subseteq O' \}$;
  // Convergence test (before commit)
  if $E' = E$ ***and*** $O' = O$ **then**
    break
  end
  // Synchronous commit of both labels
  $E \leftarrow E'$; $O \leftarrow O'$;
end
return $G \leftarrow E$;

Algorithm 22: Synchronous labeling for the grounded extension

($\supseteq$) If $a \in F(E_t)$, then for every $b \in R^{-1}(a)$ there exists $c_b \in E_t \cap R^{-1}(b)$. Thus, after processing pass t+1, no attacker of a can remain IN; consequently all attackers of a are OUT and the IN-rule sets a to IN. Hence $a \in E_{t+1}$.

Now, by induction on t, we have the remaining properties: observe that they trivially hold for E_0 and O_0. We first observe that since $E_{t-1} \subseteq E_t$, $O_t \subseteq O_{t+1}$. In turn, therefore, $E_t \subseteq E_{t+1}$. Now, since E_t is known to be conflict-free, and O_{t+1} is the arguments attacked by E_t, $O_{t+1} \cap E_t = \emptyset$. Furthermore, every $b \in O_{t+1}$ cannot be in E_{t+1} since it has an attacker in E_t (and therefore not in O_{t+1}), so $O_{t+1} \cap E_{t+1} = \emptyset$. For $a \in E_{t+1}$, all its attackers b were in O_{t+1}, and for any c attacked by a, since a was not in O_{t+1}, c is not added to E_{t+1}. Thus, E_{t+1} remains conflict-free. □

Based on the lemma, we can now establish Algorithm 22 computes the unique grounded extension.

Theorem 10.9 *In any finite AAF $\langle A, R\rangle$, Algorithm 22 returns the grounded extension in polynomial time. Moreover, the grounded extension is given uniquely by the intersection of all complete extensions.*

Proof By Lemma 10.8, $E_0 \subseteq E_1 \subseteq \cdots$ and labels never revert; in a finite AAF there are at most $|A|$ strict increases, so the algorithm halts. Since each pass considers at most indeg(a) attackers for each argument, then at most $|R|$ attacks are inspected overall, which establishes the polynomial running time.

We next show that $G = E_T$, the terminal IN-set for $t = T$, is, in fact, a complete extension. Indeed, by Lemma 10.8, $E_{t+1} = F(E_t)$ for all t, whence

$$F(G) = F(E_T) = E_{T+1} = E_T = G.$$

The claim follows by the conflict-freeness of G from Lemma 10.8.

To conclude the proof, it remains to establish that G is a subset of any complete extension $E = F(E)$. We show by induction on t that $E_t \subseteq E$ for all t. The base $E_0 = \emptyset \subseteq E$ is trivial. For the inductive step, using the coupling,

$$E_{t+1} = F(E_t) \subseteq F(E) = E,$$

where monotonicity of F gives $F(E_t) \subseteq F(E)$ and completeness gives $F(E) = E$. Hence $E_t \subseteq E$ for all t, and at termination $G = E_T \subseteq E$. Thus, G is contained in every complete extension, i.e., it is the subset-least complete extension. □

As an immediate corollary, the unique grounded extension G always exists and can be computed efficiently—actually, it can be computed in linear time (see Exercise 10.3). Consequently, skeptical and credulous entailment coincide, and both reduce to the efficient testing of membership in G.

The existence of the grounded extension G also yields existence for other semantics. Since G is both *complete* and *admissible*, complete and admissible extensions exist. In finite AAFs, *preferred* extensions exist as the subset-maximal admissible sets. By contrast, *stable* extensions need not exist in general. Moreover, for semantics that may admit multiple extensions, the entailment problem can be computationally harder. We do not establish the intractability of these problems here.

10.3 Structured arguments and reasoning.

While abstract argumentation offers a clean and general account of how arguments interact, it remains silent on where arguments come from, and, consequently, on the differences between different modes of attack on arguments.

The following example illustrates three distinct modes of attack that, once abstracted, collapse to the same AAF: During a debate on whether a certain winged animal, Tweety, can fly, a layperson observes that Tweety is a bird and, appealing to the commonsense knowledge that birds typically fly, concludes that Tweety can fly; call this argument a. On the other side of the debate stand three experts, each offering a distinct reason why Tweety cannot fly:

- b_1: Tweety is in a cage, and caged animals do not fly. This directly supports the opposite conclusion of a, constituting a *rebuttal* attack.
- b_2: The general rule "birds typically fly" does not apply without qualification, since some circumstances (e.g., young) might lead to exceptions. This blocks the inference step in a, without challenging its premise or endorsing the opposite conclusion, constituting an *undercut*.

- b_3: Reliable evidence shows that Tweety is, in fact, a bat. This challenges the premise that Tweety is a bird, without addressing whether birds (or bats) fly, thus *undermining* a.

Structurally, the three expert arguments attack a in different ways: rebutting its conclusion, undercutting its inference, or undermining its premise. If we abstract away this structure, all three reduce to the same AAF: two arguments in mutual defeat, giving us no basis for deciding which argument is stronger.

Explicating the internal structure of arguments, on the other hand, enables learning to resolve defeats: a learned priority that prefers specifics (penguins) over generalizations (birds) could resolve the rebuttal case, a learned contextual applicability condition could resolve the undercut case, and a learned reliability score for sources (testimony vs. observation) could resolve the undermine case.

Different frameworks for structured argumentation exist, differing mainly in what kinds of knowledge they allow and how they treat conflicts. Rather than survey these systematically, we next present a particular instantiation that will serve as our working model for the specific connection to learning we consider.

The setting that we adopt is that of Chapters 7 and 8, where hypotheses are applied on a partial observation to infer the values of masked attributes. Here, we restrict our focus to hypotheses expressible as rules.

Definition 10.10 (Defeasible rules) A *(defeasible) rule* is an expression of the form $\varphi \rightsquigarrow \tau$, whose *body* φ is a finite conjunction of literals, and whose *head* τ is a literal over the attributes in ρ. Two rules are *conflicting* if their head literals are each other's negation. A *(defeasible) knowledge base* $\kappa = \langle \mathtt{R}, \succ \rangle$ comprises a finite collection $\mathtt{R}$ of rules, and a *priority relation* $\succ \subseteq \mathtt{R} \times \mathtt{R}$ over pairs of conflicting rules that is

- *Irreflexive:* for every $r \in \mathtt{R}$, $r \not\succ r$ and
- *Antisymmetric:* if $r \succ r'$ then $r' \not\succ r$.

The intuitive reasoning semantics of a rule $\varphi \rightsquigarrow \tau$ is as usual: if φ is satisfied in a partial observation, then the value of the attribute that corresponds to τ is inferred to be such that it satisfies τ. The key departure from reasoning with classical implications is that rules acknowledge from the start that their inference might be questioned by other rules, and never materialize. Recall that we had encountered a light version of such interference between hypotheses in Chapter 8, where predictions in earlier layers prevailed over those in later layers.

Following Chapter 8, we take information in a partial observation ρ as indisputable. By contrast, inferences coming from any rule (or combination thereof) are defeasible. Defeats against a rule r are realized by arguments concluding the negation of its head. Although, strictly speaking, this corresponds only to a rebuttal type of attack, it effectively also captures undermining attacks by rebutting the conclusion of some rule used to establish a premise of r, and undercutting attacks by rebutting an implicit guard condition of r; cf. Figure 10.3.

What constitutes an argument in our setting? Given a partial observation ρ and a knowledge base $\kappa = \langle \mathtt{R}, \succ \rangle$, we first interpret literals that are satisfied in ρ as facts,

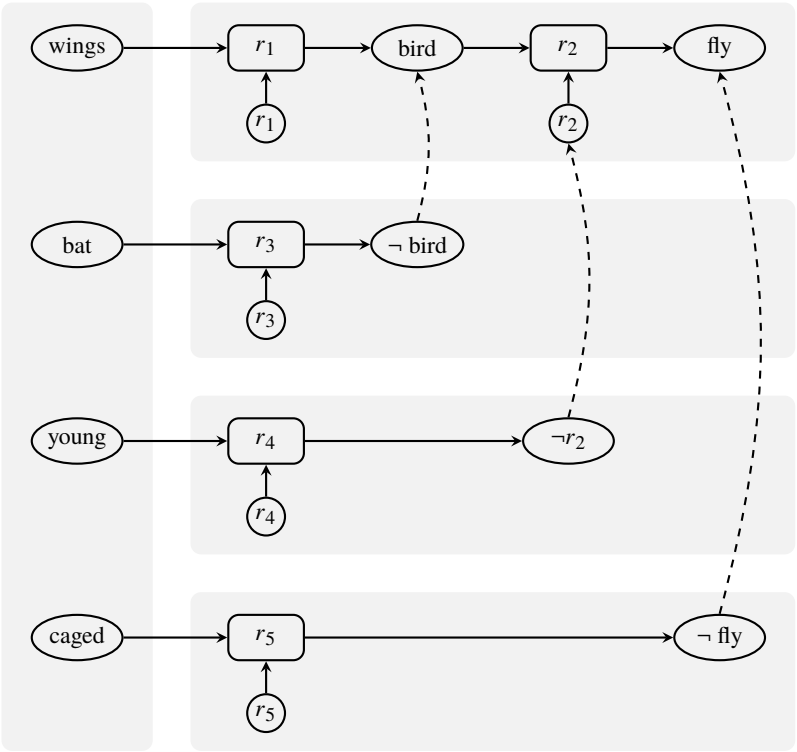


Figure 10.3 Observations on the left support the construction of chains (DAGs) of rules, corresponding to arguments. Even from a single, internally consistent set of observations, it is possible to derive arguments that defeat each other. The second argument *undermines* the top one, the third *undercuts* it, and the fourth *rebuts* it. In our formalism, all defeats are represented as rebuttals of some literal: a rule premise, the guard of a rule determining its applicability, or a rule conclusion. Rule guards are assumed to be true (which can be implemented by including a rule with a tautological premise that concludes each guard), unless defeated by some other argument. This extension can be done implicitly.

and rules in $\mathtt{R}$ as classical implications, and we draw classical inferences by repeatedly applying chaining (Modus Ponens), as in the system of Definition 5.6.

Definition 10.11 (Chaining argument) Given a partial observation ρ and a knowledge base $\kappa = \langle \mathtt{R}, \succ \rangle$, an *argument* for a literal τ in ρ under $\kappa = \langle \mathtt{R}, \succ \rangle$ (or, under $\mathtt{R}$) is the pair consisting of

1. a minimal subset of literals satisfied by ρ and
2. a minimal subset of $\mathtt{R}$ that lead thus to the inference of τ.

That is, the subset of literals and rules appearing in a chaining proof in which every literal (except for the conclusion) and rule is used in a subsequent step of the proof. In an argument for τ, a rule with head τ (whenever one exists) is unique and it is the argument's *crown* rule.

What constitutes an attack in our setting?

Definition 10.12 (Chaining argument attack) Given two arguments in a common partial observation ρ under a common knowledge base $\kappa = \langle \mathtt{R}, \succ \rangle$, the first argument for literal τ *attacks* the second argument on the latter's rule r_2 with head $\neg\tau$ if either

1. *exogenous* / external attack: τ is satisfied in ρ , or
2. *endogenous* / internal attack: the former's crown rule r_1 is such that $r_2 \not\succ r_1$

Thus, we classify defeats into exogenous (attacks from observed literals) and endogenous (attacks between rules modulated by $\succ$). Aligned with the perception of partial observations as indisputable, exogenous defeats lead to indefensible attacks.

We now have the necessary components of an abstract argumentation framework:

Definition 10.13 (Contextualized argumentation system) We say that a knowledge base κ instantiates a *contextualized argumentation system* $\langle A_\kappa, R_\kappa \rangle$, where A_κ is a mapping from any partial observation ρ to the set $A_\kappa(\rho)$ of chaining arguments in ρ under κ, and R_κ is a mapping from any partial observation ρ to the set $R_\kappa(\rho)$ of attacks between arguments in $A_\kappa(\rho)$.

Intuitively, $\langle A_\kappa, R_\kappa \rangle$ defines, for each partial observation ρ, the argumentation framework that emerges when the knowledge base κ is applied in that specific context.

An abstract argumentation semantics gives rise to a corresponding structured nonmonotonic reasoning semantics in the following way:

Definition 10.14 (Structured entailment) Given an argumentation semantics σ, a knowledge base κ applied on a partial observation ρ *entails* a literal τ, written $\kappa(\rho) \models \tau$, if there exists an argument for τ that is entailed by the AAF $\langle A_\kappa(\rho), R_\kappa(\rho) \rangle$ under the argumentation semantics σ, where $\langle A_\kappa, R_\kappa \rangle$ is the contextualized argumentation system instantiated by κ.

As discussed earlier, we commit to the *grounded semantics* for argumentation. This resolves the issue of skeptical versus credulous entailment, since the grounded extension is unique. The benefit of uniqueness is further supported both by psychological evidence on human reasoning and by the need for computational efficiency.[3]

We now return to the example of the frame axiom, to see how it can be compactly represented by our structured arguments.

Example 10.4: [Compact Frame Axiom] We revisit a temporal setting in which a family of propositional variables $X_{i,0}, X_{i,1}, \ldots, X_{i,t}, \ldots$ represent, respectively, whether or not the proposition X_i holds at time t. In the classical planning setting considered in Section 5.6 (and more concretely earlier, in Section 2.1) such a proposition is called a *fluent*. Classical planning required a frame axiom, asserting that the value of the fluent X_i does not change from time t to time $t + 1$ unless it is changed by an action. Classically, we needed to represent this "frame axiom" by a rule that exhaustively considers all of the possible actions that could change the value

[3] Such a set of rules is also called a *logic program*, and the grounded extension was independently discovered and known as the *well-founded semantics* in this setting: each literal is either true (in the grounded extension), false (attacked by the grounded extension), or undefined. See Exercise 10.5.

of X_i, and including in the body of that rule that none of these actions occurred at time t. Unfortunately, this approach to representing inertia runs into the qualification problem discussed in Section 6.3. By contrast, consider the rules of the form $X_{i,t} \rightsquigarrow X_{i,t+1}$ for each $t = 0, 1, 2, \ldots$. These rules enable a defeasible inference that $X_{i,t+1}$ holds whenever $X_{i,t}$ does, in the absence of evidence to the contrary—where observing an action at time t that causes $X_{i,t+1}$ to become false is an attack that constitutes such evidence to the contrary. We can similarly include the rules $\neg X_{i,t} \rightsquigarrow \neg X_{i,t+1}$ that enable the inference that $X_{i,t+1}$ does not hold if $X_{i,t}$ did not, again in the absence of evidence to the contrary. Indeed, these are the frame axioms. Thus we see that our non-monotonic logic allows us to represent the frame axioms in a way that does not require an exhaustive enumeration of the possible exceptional circumstances. In the ground semantics, if we observe $X_{i,t}$ in ρ, then the appropriate frame axiom $X_{i,t} \rightsquigarrow X_{i,t+1}$, together with $X_{i,t}$, constitutes an argument for $X_{i,t+1}$ that, if unchallenged by another argument, is in all of the complete extensions and hence in the grounded extension. The grounded semantics therefore entails $X_{i,t+1}$ in this case, as desired.

Example 10.5: [Qualification of effects] Continuing in the setting of classical models of actions and change of Section 5.6, recall that we had modeled the effects of actions as a collection of rules of the form $[A_t] \supset \ell_{t+1}$, i.e., that if the action A is taken at time t, then the literal ℓ holds at time $t + 1$. For example, A could be the action of flipping a light switch to the "off" position and ℓ could indicate that the light is off, as in Example 10.3. This simple rule does not allow for the possibility that the switch is broken or any other myriad possible exceptional circumstances, as we illustrated previously. Again, as discussed in Section 6.3, the problem of anticipating and modeling all of the possible exceptions is essentially hopeless.

By contrast, if we represent the effect as a defeasible rule $A_t \rightsquigarrow \ell_{t+1}$, this allows for the possibility of initially unanticipated exceptions. We can model a faulty switch with a rule $F_t, A_t, \neg\ell_{t+1} \rightsquigarrow \neg\ell_{t+1}$ with a higher priority than $A_t \rightsquigarrow \ell_{t+1}$. This restores persistence of the state $\neg\ell$ (i.e., the frame axiom) when the faulty switch is flipped off: Then $F_t, A_t, \neg\ell_{t+1} \rightsquigarrow \neg\ell_{t+1}$ defeats $A_t \rightsquigarrow \ell_{t+1}$, so when F holds A no longer causes ℓ. In the grounded extension, if $F_t, A_t, \neg\ell_{t+1} \rightsquigarrow \neg\ell_{t+1}$ is not itself challenged (e.g., by a rule concerning maintenance of the switch, etc.) then $A_t \rightsquigarrow \ell_{t+1}$ cannot be included and $F_t, A_t, \neg\ell_{t+1} \rightsquigarrow \neg\ell_{t+1}$ is included. So, the grounded semantics entails $\neg\ell_{t+1}$ and not ℓ_{t+1} in this case.

Of course, we similarly need the effect rule $A_t \rightsquigarrow \ell_{t+1}$ to have a higher priority than the frame axiom $\neg\ell_t \rightsquigarrow \neg\ell_{t+1}$ since these rules are also conflicting, and the action is an "exception" that defeats the persistence of the state $\neg\ell$. If F_t is not satisfied by ρ but A_t and $\neg\ell_t$ are, then no argument with $F_t, A_t, \neg\ell_{t+1} \rightsquigarrow \neg\ell_{t+1}$ exists under ρ (since F_t is not available to apply the rule), but the arguments $(\{\neg\ell_t\}, \{\neg\ell_t \rightsquigarrow \neg\ell_{t+1}\})$ and $(\{A_t\}, A_t \rightsquigarrow \ell_{t+1})$ are present. Assuming $A_t \rightsquigarrow \ell_{t+1}$ has higher priority than $\neg\ell_t \rightsquigarrow \neg\ell_{t+1}$, the complete extensions must include

$(\{A_t\}, A_t \rightsquigarrow \ell_{t+1})$ and cannot include the defeated argument $(\{\neg\ell_t\}, \{\neg\ell_t \rightsquigarrow \neg\ell_{t+1}\})$. Thus, ℓ_{t+1} is entailed under the grounded semantics.

Regarding efficiency, recall that Theorem 10.9 guarantees that the grounded extension can be computed in time polynomial in $|A_\kappa(\rho)|$ and $|R_\kappa(\rho)|$. While this suffices for efficient reasoning once the set of arguments is given, it does not yet yield efficiency in terms of the size of the knowledge base κ and the partial observation ρ, since $|A_\kappa(\rho)|$ and $|R_\kappa(\rho)|$ can be exponentially larger than $|\kappa|$ and $|\rho|$; cf. Figure 10.4.

Fortunately, the grounded extension of the AAF $\langle A_\kappa(\rho), R_\kappa(\rho)\rangle$ can be computing without enumerating $A_\kappa(\rho)$ and $R_\kappa(\rho)$, by working directly with rules. To describe this process, we need to first introduce some auxiliary notation and definitions.

Definition 10.15 (Dual representation) Let $\mathtt{g}_\kappa(\rho)$ denote the grounded extension of the AAF $\langle A_\kappa(\rho), R_\kappa(\rho)\rangle$. The *dual representation* of $\mathtt{g}_\kappa(\rho)$ is defined to be the set of all rules in any argument in $\mathtt{g}_\kappa(\rho)$, denoted by $\langle \rho, \widetilde{\mathtt{g}_\kappa}\rangle$.

The next result shows that the use of dual representations preserves not only the entailed literals, but also their supporting explanations. For ease of exposition, let $\mathtt{args}(\langle \rho, \widetilde{\mathtt{g}_\kappa}\rangle)$ denote the set of all arguments in ρ under $\widetilde{\mathtt{g}_\kappa}$.

Theorem 10.16 (Grounded extension duality) *Given a knowledge base κ and a partial observation ρ, consider the AAF $\langle A_\kappa(\rho), R_\kappa(\rho)\rangle$ and its grounded extension $\mathtt{g}_\kappa(\rho)$. Then, the grounded extension $\mathtt{g}_\kappa(\rho)$ can be (re)constructed from its dual representation $\langle \rho, \widetilde{\mathtt{g}_\kappa}\rangle$ as the set of all arguments in ρ under $\widetilde{\mathtt{g}_\kappa}$; i.e., $\mathtt{g}_\kappa(\rho) = \mathtt{args}(\langle \rho, \widetilde{\mathtt{g}_\kappa}\rangle)$.*

Proof We establish the claim through two set inclusions.

(⊆) If an argument α belongs to $\mathtt{g}_\kappa(\rho)$, then by definition every rule it uses belongs in $\widetilde{\mathtt{g}_\kappa}$. Hence, α is an argument in ρ under $\widetilde{\mathtt{g}_\kappa}$, and thus, $\alpha \in \mathtt{args}(\langle \rho, \widetilde{\mathtt{g}_\kappa}\rangle)$.

(⊇) Let $\alpha \in \mathtt{args}(\langle \rho, \widetilde{\mathtt{g}_\kappa}\rangle)$. Consider any argument α_1 that attacks α, if one exists. By definition (all defeats realized as rebuttals of rule heads), α_1 attacks α on some rule r used by α. Since α is constructed under $\widetilde{\mathtt{g}_\kappa}$, this rule r appears in an argument $\alpha_2 \in \mathtt{g}_\kappa(\rho)$. The attack on r applies equally to α_2, so α_1 also attacks α_2. Because $\mathtt{g}_\kappa(\rho)$ is admissible, there exists an argument $\alpha_3 \in \mathtt{g}_\kappa(\rho)$ that attacks α_1, thereby defending α. Thus every attacker of α, if one exists, is counter-attacked by some argument in $\mathtt{g}_\kappa(\rho)$, so α is defended by $\mathtt{g}_\kappa(\rho)$. Since the grounded extension is a complete extension, any argument that is defended by $\mathtt{g}_\kappa(\rho)$ must belong to it. Hence $\alpha \in \mathtt{g}_\kappa(\rho)$. □

The exponential growth in Figure 10.4 stems from enumerating all arguments and defeats in $\langle A_\kappa(\rho), R_\kappa(\rho)\rangle$. By Theorem 10.16, the grounded extension can instead be recovered directly from the set of rules that survive grounded reasoning in context ρ, namely $\widetilde{\mathtt{g}_\kappa}$. This shifts computation from the argument level to the rule level, eliminating the need to explore an exponentially large AAF, and allowing us, instead, to operate on a set of rules of size at most $|\kappa|$. The dual representation $\langle \rho, \widetilde{\mathtt{g}_\kappa}\rangle$ thus

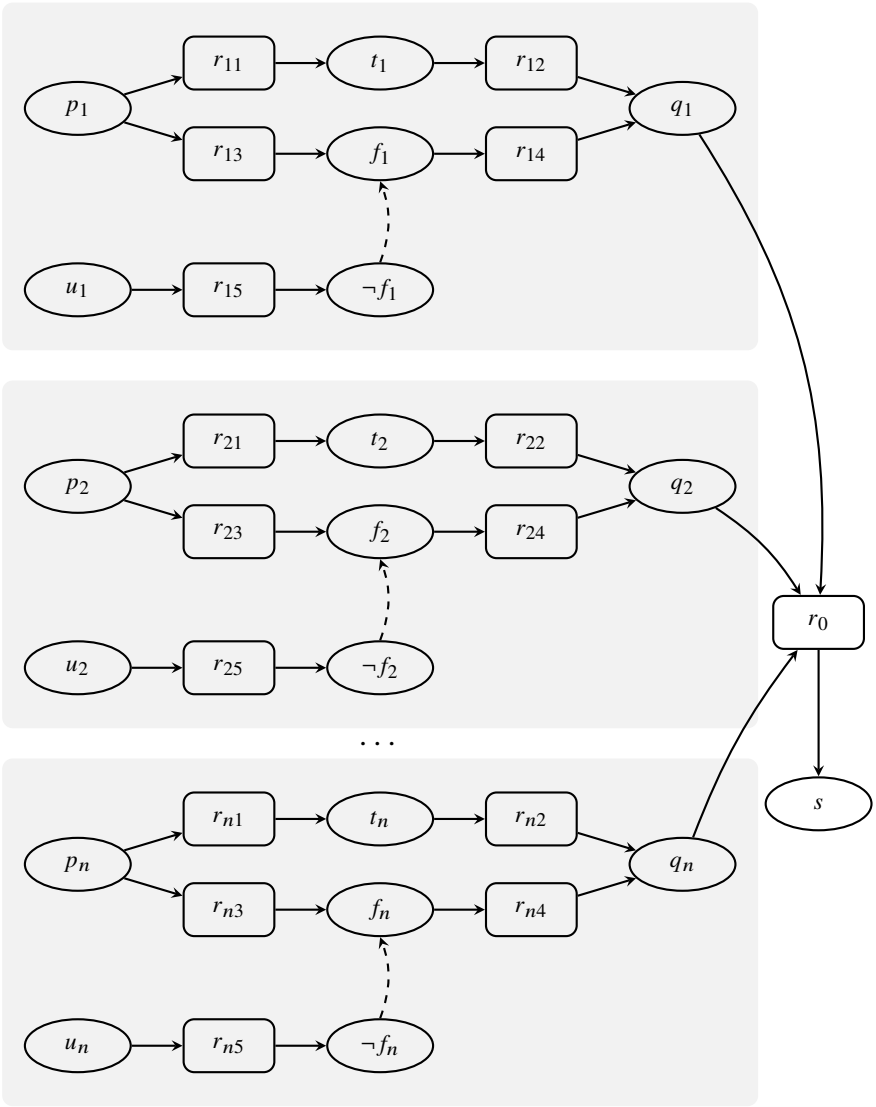


Figure 10.4 A knowledge base κ with $5n{+}1$ rules and a partial observation ρ with $2n$ observed literals give rise to an AAF $\langle A_\kappa(\rho), R_\kappa(\rho)\rangle$ with 2^n arguments for s (plus arguments for other literals), and defeats for $2^n - 1$ of those arguments (plus defeats of arguments for other literals).

compactly captures both the conclusions of the grounded extension and their justifying rules. We proceed next to show how to compute the dual representation.

Intuitively, Algorithm 23 mirrors the structure of Algorithm 22, but operates directly at the rule and literal level. It iteratively eliminates rules and literals defeated by already accepted ones, and admits those whose attackers have all been defeated — thus identifying, in the limit, precisely the rules and literals that survive grounded reasoning. Our next result establishes the efficacy and efficiency of Algorithm 23.

Input: Knowledge base $\kappa = \langle \mathtt{R}, \succ \rangle$ and partial observation ρ
Output: Dual representation $\langle \rho, \widetilde{\mathcal{G}_\kappa} \rangle$ of the grounded extension of the AAF $\langle A_\kappa(\rho), R_\kappa(\rho) \rangle$

Initialize:

- MarkedLs $\leftarrow \rho$; RemainLs $\leftarrow$ all literals in R and ρ
- MarkedRs $\leftarrow \emptyset$; RemainRs $\leftarrow$ R

repeat

```
// Drop literals defeated by marked literals
RemainLs ← RemainLs \ {¬ℓ : ℓ ∈ MarkedLs};
// Drop rules whose head has been defeated
RemainRs ← RemainRs \ { r : head(r) ∉ RemainLs };
// Retain only crowns of formable arguments
RemainRs ← { r ∈ RemainRs :
  r is the crown of some argument in RemainLs under RemainRs };
// Mark dominating rules with marked bodies
MarkedRs ← MarkedRs ∪ { r1 ∈ RemainRs : body(r1) ⊆
  MarkedLs ∧ ∄r2 ∈ RemainRs(head(r2) = ¬head(r1) ∧ r1 ⊁ r2) };
// Mark literals in heads of marked rules
MarkedLs ← MarkedLs ∪ { head(r) : r ∈ MarkedRs };
```

until *no set changes in this round*;
return $\langle \rho, \widetilde{\mathcal{G}_\kappa} \rangle$ *where* $\widetilde{\mathcal{G}_\kappa} \leftarrow$ *MarkedRs*;
Algorithm 23: Rule-level grounded computation with dual representation

Theorem 10.17 (Algorithm 23 computes the grounded extension dual) *For any knowledge base κ and partial observation ρ, Algorithm 23 returns the dual representation $\langle \rho, \widetilde{\mathcal{G}_\kappa} \rangle$ of the grounded extension of the AAF $\langle A_\kappa(\rho), R_\kappa(\rho) \rangle$ in polynomial time.*

Proof We proceed to establish termination, soundness, completeness, and efficiency.
Termination: *algorithm halts and final set* MarkedRs *is well-defined*. Across iterations, either (i) some literal is removed from RemainLs, (ii) some rule is removed from RemainRs, or (iii) some rule or literal is added to MarkedRs or MarkedLs. All four sets evolve monotonically — no element is reintroduced once removed or marked. Since both the set of literals and the set of rules are finite, the algorithm halts after finitely many iterations.
Completeness: *every rule in the dual gets marked*. We will actually prove, more generally, that for every argument in the grounded extension for ρ under κ, the crown rule r^* is included in MarkedRs (and thus for all $r^* \in \widetilde{\mathcal{G}_\kappa}$, r^* is in MarkedRs), and for every argument attacked by the grounded semantics, some rule is removed from RemainRs. We prove this by induction on the depth of the argument in the DAG given by the attack relation (maximum length of a path to the argument from a source).

Consider an argument at some depth d. (The base case, $d = 0$, consists of arguments with no higher-priority attacker.) We first consider arguments in the grounded extension. Suppose we order the rules of the argument topologically, as in the corresponding chaining proof. We show by induction along this order that the premises of each rule become marked, and then the rule itself is marked and its literal is included in markedLs.

- *Base case:* Observed premises are in MarkedLs initially. Since their complements are removed immediately, exogenous defeats cannot block them.
- *Inductive step:* Suppose all premises of a rule r in the argument are marked. The crown filter ensures that r is retained in RemainRs (its body literals are available, cycles are excluded, and the argument is formable). Any conflicting argument that could defeat r has depth at most $d - 1$, and so by induction hypothesis, some rule used by the argument is removed from RemainRs. Since all of the arguments using the same (higher priority) crown rule r' attack this argument and are defeated, none of these arguments are formable using RemainRs, and hence the crown rule r' of these attacks is also removed from RemainRs. Therefore, r passes the domination test and is marked, and its head literal is subsequently marked.

Thus, eventually, the crown rule of the argument is included in MarkedRs and the literal is included in markedLs, as needed.

Similarly, for arguments attacked by the grounded extension, we find that the attackers have depth at most $d - 1$, so by induction hypothesis their literals are marked. Hence, the attacked rule r is removed from RemainRs because it conflicts with a marked literal.

Soundness: *only rules in the dual are marked*. Maintain the invariant that:

1. every $r \in$ MarkedRs is the crown of some argument formable from MarkedLs (ensured by the crown filter and the condition $\texttt{body}(r) \subseteq$ MarkedLs);
2. r has no undefeated attacker among the still-possible rules (enforced by the "no remaining conflicting r_2 with $r_1 \not\succ r_2$" test);
3. MarkedLs contains heads of MarkedRs and no complementary literals (by removal of complements in the first step).

Thus, inductively, whenever a rule r is marked, it therefore belongs to an argument constructible under the marked rules and literals, such that no attacks exist using the remaining rules. Hence, this argument is defended by $\mathtt{g}_\kappa(\rho)$ and therefore must be included because the grounded extension is a complete extension. In turn, r, being one of its rules, belongs to $\widetilde{\mathtt{g}_\kappa}$.

Efficiency: *polynomial time complexity*. Let $\#R := |\mathtt{R}|$ and let $\#L$ be the number of distinct literals in $\mathtt{R}$ or ρ. Clearly, $\#R \leq |\kappa|$ and $\#L \leq |\kappa| + |\rho|$. Per iteration:

1. dropping complements in RemainLs takes $O(\#L)$;
2. removing rules with unavailable heads takes $O(\#R)$;

3. the crown filter (checking for formable arguments) builds the rule–literal dependency restricted to Remain sets and performs strongly-connected component detection and a forward justification pass, costing $O(\#R{+}\#L)$;
4. marking dominating rules requires checking $\texttt{body}(r) \subseteq$ MarkedLs in total $O(\#R{+}\#L)$ and testing for undefeated attackers via indexed conflict lists, for $O(\#R^2)$ worst case;
5. propagating heads of marked rules takes $O(\#R)$.

Thus, one iteration costs $O(\#R^2 + \#R + \#L)$. Each iteration strictly grows MarkedLs $\cup$ MarkedRs or shrinks RemainLs$\cup$RemainRs, so there are at most $\#R{+}\#L$ iterations. The overall time complexity is therefore $O\big((\#R{+}\#L)(\#R^2{+}\#R{+}\#L)\big) = \text{poly}(|\kappa|{+}|\rho|)$. □

It follows immediately that entailment under the grounded semantics for the AAF $\langle A_\kappa(\rho), R_\kappa(\rho)\rangle$ can be computed in time polynomial in the size of κ and ρ.

10.4 Learning through machine coaching.

We now turn to the problem of learning structured argumentation frameworks. In particular, we focus on learning contextualized argumentation systems (CAS) as instantiated by a knowledge base following the conceptualization presented earlier.

The learning problem is considerably challenging for several reasons. First, learning may need to resolve the semantic ambiguity of reasoning, by identifying not only the underlying CAS but also its entailment semantics. Second, even for a fixed semantics – especially one that may admit multiple acceptable extensions – the learning task may not be that of learning a function, but rather a many-to-many relation. Third, even disregarding defeasibility, the chaining of rules already encodes highly complex mappings, despite individual rules being simple. Finally, even conceding the use of a fixed semantics that yields a unique acceptable extension, and even under severely restricted rule chaining, learning may still need to uncover the priorities among rules – which induce the defeat relation of the argumentation framework – and this can itself become a difficult combinatorial search problem.

There is yet another learning challenge that is not semantic or computational, but rather explanatory. A key motivation for moving from abstract to structured argumentation was the desire to make explicit the supporting reasons for the entailed conclusions. One might hope, therefore, that learning would go beyond identifying a functionally equivalent argumentation framework – one that entails the same conclusions – and aim instead to recover a structurally equivalent one, which reproduces the same chains of reasoning and supporting explanations. This added explanatory goal substantially raises the bar for what counts as successful learning, since even small structural deviations may preserve conclusions while distorting their justificatory basis.

One approach to address these intertwined challenges, is *"machine coaching"*: an interactive learning paradigm in which a teacher provides feedback on the learner's reasoning rather than on its outputs. Unlike traditional supervised learning, which seeks to approximate input-output mappings, machine coaching aims to align the learner's

process of reasoning with that of the teacher. In what follows, we formalize this view of learning, redefining success so that the objective shifts from functional correctness to structural conformity, while retaining the essential features of PAC learnability.

In machine coaching, the supervised learning problem is extended to accommodate:

(i) a sequential, bilateral interaction between the learner and the teacher, where given an observation, the learner makes a prediction and then receives advice;
(ii) the possibility of arbitrary advice in response to an unacceptable prediction, without requiring the teacher to specify a correct one; and
(iii) a learning objective that is not to infer the teacher's advice itself (i.e., given an observation and a prediction, predict the advice), but rather to conform to it—that is, to produce predictions that eventually elicit no further advice.

Before we present the machine coaching process, let us first introduce some terminology and notation. We suppose that the learner's task is to produce a mapping from a space of *inputs* I to a space of *outputs* O, represented by a *hypothesis* function $h : I \to O$ mapping inputs to outputs. A *feedback* (or *advice*) function $f : I \times O \to A$ maps input-output pairs to elements of an *advice space* A. We require that A includes a distinguished "no-advice" element $\checkmark$, and that for every input $x \in I$ there exists at least one output $y \in O$ such that $f(x, y) = \checkmark$.

Definition 10.18 (Machine coaching) The *machine coaching process* for a distribution D over I is the following protocol between a *learner* and *teacher* that proceeds in sequential *rounds*. In each round,

1. A *learning example* input x is drawn from D and provided to the learner.
2. Given x, the learner selects an output *prediction* $y \in O$ using its current hypothesis h and provides it to the teacher.
3. Given x and y, the teacher chooses *advice* $a \in A$ to provide to the learner.
4. Using x and a, the learner chooses a new hypothesis to use in the next round.

A learner that retains the same hypothesis whenever $a = \checkmark$ is said to be *conservative*.

As alluded to above, the learner's objective in machine coaching is to find a hypothesis that the teacher deems adequate, analogous to finding a valid definition of a property in the PAC semantics. We'll use the following analogue of approximate validity:

Definition 10.19 (Approximate conforming) A hypothesis h is said to be $(1 - \epsilon)$-*approximately conformant* under a distribution D over I with respect to advice function f if, when an input x is drawn from D, we have $f(x, h(x)) = \checkmark$ except with probability at most ϵ.

The learning task is then given below:

Definition 10.20 (Conformant Learning) An algorithm is a *(probably approximately) conformant learner* for a feedback class F (with input, output, and advice spaces I, O, and A, respectively), using a hypothesis class H (with input and output spaces I and

O) is as follows. For all real values $\delta, \epsilon \in (0, 1]$, every distribution D over I with representation size n, and every feedback function $f \in F$ with representation size s, the algorithm, when given the parameters δ, ϵ, and F, "halts" and returns a hypothesis $h \in H$ after at most $t(1/\delta, 1/\epsilon, n, s)$ steps of the machine coaching process. With probability at least $1 - \delta$, the output h is $(1 - \epsilon)$-approximately conformant under D against f. If t grows polynomially in its parameters, the learner is said to be *efficient*.

We'll actually consider learning algorithms that can meet a more demanding standard, that eventually conform perfectly:

Definition 10.21 (Mistake-bounded learning) An algorithm is a *mistake-bounded learner* for a feedback class F (with input, output, and advice spaces I, O, and A, respectively), using a hypothesis class H (with input and output spaces I and O), if for every representation size n for I and representation size s for F, there is an integer *mistake bound* $m(n, s)$ such that for every feedback function $f \in F$ and every possible infinite sequence of examples of size n, the algorithm (given F) obtains the advice $\checkmark$ except in at most $m(n, s)$ rounds of the machine coaching process with f.

It is sufficient to find a mistake-bounded learner because, as we show next, conservative learners with a polynomial mistake bound can be converted to conformant learners; and mistake-bounded learners can easily be converted to conservative learners.

Lemma 10.22 (On-line to batch conversion) *If there is a mistake-bounded learner for a feedback class F (with input, output, and advice spaces I, O, and A, respectively), using a hypothesis class H (with input and output spaces I and O) with a polynomial mistake bound, then there is an efficient probably approximately conformant learner for F using H.*

Proof Let Alg be a mistake-bounded learner with associated polynomial mistake bound $m(n, s)$. Our conformant learner will be the following algorithm: given the parameters δ, ϵ, and F, we run the mistake-bounded learner and count the number of consecutive examples that receive the advice $\checkmark$ from the teacher. That is, our algorithm's counter is initially 0 and we use Alg to obtain an initial hypothesis h. In each round of the machine coaching process, if the advice is $\checkmark$, we increment the counter, and otherwise we use Alg to compute a new hypothesis h and reset the counter to 0. Furthermore, in each round our algorithm will check if the counter is at least $\frac{1}{\epsilon} \ln \frac{m(n,s)}{\delta}$. If so, the algorithm halts and outputs the current hypothesis h.

Observe that when the algorithm halts, this final hypothesis h received the advice $\checkmark$ on $\frac{1}{\epsilon} \ln \frac{m(n,s)}{\delta}$ examples drawn independently from D. For a hypothesis $\tilde{h}$ that is not $(1 - \epsilon)$-approximately conformant, the probability of receiving $\checkmark$ in each round is at most $1 - \epsilon$, and hence the probability that such a $\tilde{h}$ receives $\checkmark$ for $\frac{1}{\epsilon} \ln \frac{m(n,s)}{\delta}$ examples independently drawn from D is at most

$$(1 - \epsilon)^{\frac{1}{\epsilon} \ln \frac{m(n,s)}{\delta}} \leq e^{-\ln \frac{m(n,s)}{\delta}} \leq \frac{\delta}{m(n, s)}$$

as usual. Now, observe that since Alg is mistake-bounded, any sequence of $\frac{m(n,s)}{\epsilon} \ln \frac{m(n,s)}{\delta}$ examples contains at most $m(n, s)$ rounds in which the advice is

not ✓. Hence, our algorithm considers at most $m(n, s)$ distinct hypotheses, and by a union bound, the probability that any hypothesis that is not $(1 - \epsilon)$-approximately conformant receives ✓ in $\frac{1}{\epsilon} \ln \frac{m(n,s)}{\delta}$ rounds is at most δ. On the other hand, by the pigeonhole principle, at least one of the hypotheses must receive ✓ in at least $\frac{1}{\epsilon} \ln \frac{m(n,s)}{\delta}$ out of the $\frac{m(n,s)}{\epsilon} \ln \frac{m(n,s)}{\delta}$ rounds. Therefore, with probabilty $1 - \delta$, the algorithm halts and outputs a $(1 - \epsilon)$-approximately conformant hypothesis within $\frac{m(n,s)}{\epsilon} \ln \frac{m(n,s)}{\delta}$ rounds. Since this is a polynomial in n, s, $\frac{1}{\epsilon}$ and $\frac{1}{\delta}$, it is indeed an efficient conformant learner. □

The power of conformant learning ultimately depends on the feedback class F: some feedback schemes reduce to ordinary PAC learning — both in the target classes that are efficiently learnable and in that the learning concerns only their functional behavior; others enable the efficient learning of richer target classes, while also supporting the recovery of their underlying reasoning structures. We illustrate the latter below.

Consider a teacher with access to a target $\kappa^* = \langle \mathtt{R}^*, \succ^* \rangle$, and a learner maintaining a hypothesis $\kappa = \langle \mathtt{R}, \succ \rangle$. For a given context x, the learner produces an explanation $y = \widetilde{\mathtt{g}_\kappa}$ corresponding to the rules in its current dual grounded extension $\langle x, \widetilde{\mathtt{g}_\kappa} \rangle$, and the teacher replies with a piece of structured advice $f_{\kappa^*}(x, y)$ from the advice space

$$A \;=\; \{\checkmark\} \cup (\{\mathit{incomplete}, \mathit{indefensible}\} \times 2^{\mathtt{R}^*}),$$

meeting the following criteria:

Definition 10.23 (Chaining argument teacher) A *chaining argument teacher* for a target contextualized argument system $\kappa^* = \langle \mathtt{R}^*, \succ^* \rangle$ is an advice function $f_{\kappa^*}(x, y)$ where x is a context and y is a set of rules (the dual grounded extension of $\kappa = \langle \mathtt{R}, \succ \rangle$ as obtained via Theorem 10.16) such that

- f only returns advice $\langle \mathit{incomplete}, S \rangle$ if a non-empty set $S \subseteq \widetilde{\mathtt{g}_{\kappa^*}} \setminus y$ of rules belongs to an argument in $\mathtt{g}_{\kappa^*}(x)$ that is not present under y but would be under $y \cup S$. Hence, if the learner omits rules S that are in the teacher's grounded extension and which would yield additional arguments, the teacher may respond that y is incomplete.
- f only returns advice $\langle \mathit{indefensible}, S \rangle$ if there exists an argument in x under subset-minimal $S \subseteq \mathtt{R}^*$ that attacks an argument in x under y, but is not defended by any argument in $\mathtt{g}_{\kappa^*}(x)$.[4] Hence, if some learner argument is attacked by a teacher-constructible, undefeated argument, the teacher may report that the attack from S is indefensible.
- f only returns advice ✓ if neither of the above conditions are met. Hence, if the teacher does not object to the learner's prediction y, then it responds that it has no advice to give.

[4] Although the indefensibility condition is not strictly required for establishing conformant learnability, it ensures that the attacking argument itself lies within the teacher's grounded extension, rather than among the remaining (undecided or defeated) arguments. This relieves the teacher from having to reason about or respond using arguments outside the teacher's accepted set, simplifying the feedback protocol.

This structured feedback protocol thus enables the learner to iteratively align its hypothesis κ with the teacher's κ^*, converging toward conformant reasoning behavior.

Theorem 10.24 (Efficient conformant learning under structured advice) *Assume a target knowledge base $\kappa^* = \langle R^*, \succ^* \rangle$ whose priority relation $\succ^*$ totally orders every pair of conflicting rules. There exists an efficient conformant learner for the class of chaining argument teachers.*

Proof The learner proceeds as follows: It initializes $\kappa^0 = \langle \mathtt{R}^0, \succ^0 \rangle = \langle \emptyset, \emptyset \rangle$. For every round $t = 0, 1, 2, \ldots$, it draws a context x_t from D; it computes the dual grounded extension $\langle x_t, \widetilde{\mathsf{G}_{\kappa^t}} \rangle$ using Algorithm 23 on input $(\kappa = \langle \mathtt{R}, \succ \rangle_t, x_t)$; it sets $y_t = \widetilde{\mathsf{G}_{\kappa^t}}$; and it queries the teacher and receives $f_{\kappa^*}(x_t, y_t) \in A$. Then:

- If the advice is $\langle \mathit{incomplete}, S \rangle$, it sets $\mathtt{R}^{t+1} \leftarrow \mathtt{R}^t \cup S$ and keeps $\succ^{t+1} = \succ^t$.
- If the advice is $\langle \mathit{indefensible}, S \rangle$, it sets $\succ^{t+1} \leftarrow \succ^t \cup \{(r', r)\}$ for a conflicting rule pair (r', r) with $r' \in S$ and $r \in y_t$, such that $r' \succ^* r$. It also sets $\mathtt{R}^{t+1} = \mathtt{R}^t \cup \{r'\}$.

Let us first observe that for all t, $\mathtt{R}^t \subseteq \mathtt{R}^*$, since only rules in $\mathtt{R}^*$ are added in $\mathtt{R}^t$. Given advice $\langle \mathit{indefensible}, S \rangle$, we wish to identify the requested rule pair (r', r). Since an argument in x_t under $S \subseteq \mathtt{R}^*$ attacks some learner argument in x_t under y_t, the attack targets a certain rule $r \in y_t$ via some conflicting rule $r' \in S$, such that $r \not\succ^* r'$. By subset-minimality of S, r' is the crown rule of the attacking argument in x_t under S. Since $r \not\succ^* r'$, since $r \in y_t \subseteq \mathtt{R}^t \subseteq \mathtt{R}^*$, and because of the total ordering of conflicing rules under the priority relation $\succ^*$, it follows that $r' \succ^* r$, as required. As a consequence, for all t, $\succ^t \subseteq \succ^*$, since only priorities in $\succ^*$ are added in $\succ^t$.

To prove termination, we will be using the following potential function:

$$\Phi_{\kappa^*}(\kappa^t) \;=\; \underbrace{\left|\mathtt{R}^* \setminus \mathtt{R}^t\right|}_{\text{missing rules}} \;+\; \underbrace{\left|\succ^* \setminus \succ^t\right|}_{\text{misordered pairs}} .$$

Note that if the teacher returns a non-✓ advice at round t, then $\Phi_{\kappa^*}(\kappa^{t+1}) \leq \Phi_{\kappa^*}(\kappa^t) - 1$. Indeed, for advice $\langle \mathit{incomplete}, S \rangle$, $\left|\mathtt{R}^* \setminus \mathtt{R}^t\right|$ decreases by $|S| \geq 1$ and $\left|\succ^* \setminus \succ^t\right|$ remains fixed, and for advice $\langle \mathit{indefensible}, S \rangle$, $\left|\succ^* \setminus \succ^t\right|$ decreases by 1, and $\left|\mathtt{R}^* \setminus \mathtt{R}^t\right|$ does not increase. Hence, there are at most $|\mathtt{R}^*| + |\succ^*| = O(|\kappa^*|)$ rounds with non-✓ advice.

We have thus established that the learner is mistake-bounded with a mistake bound of $O(|\kappa^*|)$. We can therefore obtain a conformant learner by Lemma 10.22. Since each mistake causes an update of the learner's hypothesis that runs in polynomial time (by Theorem 10.17), the efficiency of learning follows. □

The efficiency of this learning process rests directly on the efficiency of reasoning established earlier. Machine coaching thus demonstrates a tight interplay between reasoning and learning: tractable learning presupposes tractable reasoning, but structured advice allows the learner to leverage external guidance to bypass its own computational limits. In this way, coaching makes it possible to efficiently learn argumentative knowledge structures that would otherwise be beyond autonomous discovery.

Chapter notes

TO DO

Exercises

10.1 *Computing and contrasting the different argumentation semantics.* Consider the abstract argumentation framework $\langle A, R\rangle$ with $A = \{a, b, c, d, e, f\}$ and

$$R = \{(a, b), (b, a), (c, d), (d, e), (e, c), (a, c), (f, c), (b, f)\}.$$

(a) Draw the directed graph of this abstract argumentation framework, and list the attackers of each argument.
(b) Compute the grounded extension, all complete extensions, all preferred extensions, and all stable extensions.
(c) Use your answer to part (b) to demonstrate that the semantics are distinct in this example. In particular, identify a complete extension different from the grounded extension, an extension that is complete but not preferred, and an extension that is preferred but not stable.

10.2 *Alternative argument semantics are intractable.* In this exercise we will show that deciding entailment of arguments under most semantics is intractable, substantiating the remaining entries of Table 10.1. Recall that credulous versus skeptical acceptance are defined in Definition 10.4.
(a) We first consider credulous acceptance under the admissible semantics from Definition 10.3. The corresponding NP search problem is: given an AAF $\langle A, R\rangle$ and some argument $a \in A$, find an extension E with $a \in E$ that is admissible, i.e., that is conflict-free and such that $E \subseteq F(E)$: every argument a in E that is attacked by some argument b is also defended by some $c \in E$ that attacks b. Prove that this problem is NP-complete.
(b) Next, we consider skeptical acceptance under the admissible semantics. In this case, the corresponding NP search problem concerns finding evidence that an argument is *not* entailed, i.e., a reason to reject the argument. (Classically, "reasons to reject" were distinguished from "reasons to accept", where problems of the first form are called coNP for "complement NP" and the second form NP.) Precisely: given an AAF $\langle A, R\rangle$ and some argument $a \in A$, find an extension E with $a \notin E$ that is admissible. Prove that this problem is NP-complete.
(c) Now consider credulous acceptance under the complete semantics (Definition 10.6): extensions E such that $E = F(E)$. Define an NP search problem corresponding to deciding credulous entailment under the complete semantics, and prove that it is NP complete.
(d) Following the discussion of part (b) regarding NP search problems corresponding to skeptical entailment, define an NP search problem corresponding to deciding skeptical entailment under the complete semantics and prove that it is NP complete.
(e) Define an NP search problem corresponding to deciding credulous entailment under the preferred semantics (Definition 10.6) and prove that it is NP complete.

(f) Define an NP search problem corresponding to deciding skeptical entailment under the preferred semantics (Definition 10.6) and prove that it is NP complete.

10.3 *Computing grounded extensions in linear time.* Show how Algorithm 22 for computing the grounded extension can be implemented in linear time ($O(|A|+|R|)$ for the AAF $\langle A, R\rangle$). ***(Hint:*** *track the number of attackers of each argument that have not yet been added to O.)*

10.4 *Default logics as arguments.* While we have considered prioritized defeasible rules as our primary representation of interest, simpler nonmonotonic logics have been proposed previously. Perhaps the simplest is *default logic*. In a simple "normal" default logic, a set of defeasible rules is given without an associated priority relation. An extension of a set of defeasible rules KB is a set E of literals such that $\ell \in E$ if and only if it is the head τ of some defeasible rule $\varphi \rightsquigarrow \tau \in KB$ such that $\neg\tau \notin E$ and φ is satisfied in E. Prove that for the chaining arguments (Definition 10.11) corresponding to the set of defeasible rules KB, there is an attack relation such that for one of the standard semantics (skeptical or credulous admissible, complete, grounded, or preferred), for each extension E of KB in the sense of normal default logic, there is an extension of the resulting AAF such that E is the set of literals entailed by the AAF (in the sense of Definition 10.14).

10.5 *Logic programs as arguments.* Another widely-studied formalism for nonmonotonic logic is logic programs. Here, we use the following form of defeasible rules: in a rule $\varphi \rightsquigarrow \tau$, the body φ is a conjunction that may contain, in addition to any set of literals, literals of the form $\sim \ell$ for any classical literal ℓ (i.e., we may have $\sim X_i$ or $\sim \neg X_i$ for any attribute X_i). For a set of literals L, we let $\sim L$ denote the set $\{\sim \ell : \ell \in L\}$. Intuitively, $\sim \ell$ is meant to be interpreted as "ℓ is not provable." (In the original formalisms, φ consisted only of literals of the form X_i or $\sim X_j$, and so $\sim$ is known as "negation-as-failure".)

A set of rules KB of this form is called a *logic program*. The interpretation of a logic program is given by a pair of sets of literals L^+ and L^- satisfying the following. We consider the set of rules $KB/(L^+, L^-)$ such that for each rule $\varphi \rightsquigarrow \tau$ in KB such that the only literals of the form $\sim \ell$ contained in φ are in L^-, $KB/(L^+, L^-)$ contains a rule $(\varphi \setminus \sim L^-) \rightsquigarrow \tau$ (that is, omitting the literals $\sim L^-$ from the body φ), and contains no other rules. Then L^+ contains all of the literals provable by chaining from $KB/(L^+, L^-)$ and L^- contains only literals that are not provable. If $L^+ \cup L^-$ includes every literal, then (L^+, L^-) is a *stable model* of the logic program. A pair (L^+, L^-) such that L^+ is subset-minimal and L^- consists of precisely the literals that do not appear in the head of any rule in $KB/(L^+, L^-)$ is a *well-founded model*.

(a) Give an example of a logic program that does not have a stable model. (Prove that no stable model exists for your example.)

(b) Suppose we interpret KB as a set of arguments. The natural attack relation on KB is that $\varphi \rightsquigarrow \tau$ attacks any other rule $\varphi' \rightsquigarrow \tau'$ such that $\sim \tau$ is in the body φ' for the second rule. Prove that for the grounded extension E of this AAF, the knowledge base KB' obtained by omitting all literals of the form $\sim \ell$ from the body of rules in E is

equal to $KB/(L^+, L^-)$ for a well-founded model (L^+, L^-). (And therefore we can also easily compute (L^+, L^-) from E.)
(c) Prove that in a well-founded model (L^+, L^-), L^+ contains only literals provable in every stable model and L^- contains only literals that are not provable in any stable model.

10.6 establish intractability of PAC learning CAS under various restrictions.
TO DO

10.7 would a more aggressive learner that simply updates its hypothesis to include the entire S and also makes all rules higher priority, still work?
TO DO