Logic: recap

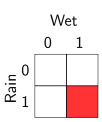
Review: ingredients of a logic

Syntax: defines a set of valid **formulas** (Formulas)

Example: Rain $\wedge$ Wet

Semantics: for each formula f, specify a set of **models** $\mathcal{M}(f)$ (assignments / configurations of the world)

Example:



Inference rules: given KB, what new formulas f can be derived?

Example: from Rain $\wedge$ Wet, derive Rain

- Logic provides a formal language to talk about the world.
- The valid sentences in the language are the logical formulas, which live in syntax-land.
- In semantics-land, a model represents a possible configuration of the world. An interpretation function connects syntax and semantics. Specifically, it defines, for each formula f, a set of models $\mathcal{M}(f)$.

Review: inference algorithm

Inference algorithm:

$$\text{KB} \xrightarrow{\text{(repeatedly apply inference rules)}} f$$

Definition: modus ponens inference rule

$$\frac{p_1, \quad \cdots \quad, p_k, \quad (p_1 \wedge \cdots \wedge p_k) \rightarrow q}{q}$$

Desiderata: soundness and completeness

entailment (KB $\models f$) derivation (KB $\vdash f$)

- A knowledge base is a set of formulas we know to be true. Semantically the KB represents the conjunction of the formulas.
- The central goal of logic is inference: to figure out whether a query formula is entailed by, contradictory with, or contingent on the KB (these are semantic notions defined by the interpretation function).
- The unique thing about having a logical language is that we can also perform inference directly on syntax by applying **inference rules**, rather than always appealing to semantics (and performing model checking there).
- We would like the inference algorithm to be both sound (not derive any false formulas) and complete (derive all true formulas). Soundness is easy to check, completeness is harder.

Review: formulas

Propositional logic: any legal combination of symbols

$(\text{Rain} \wedge \text{Snow}) \rightarrow (\text{Traffic} \vee \text{Peaceful}) \wedge \text{Wet}$

Propositional logic with only Horn clauses: restricted

$(\text{Rain} \wedge \text{Snow}) \rightarrow \text{Traffic}$

- Whether a set of inference rules is complete depends on what the formulas are. Last time, we looked at two logical languages: propositional logic and propositional logic restricted to Horn clauses (essentially formulas that look like $p_1 \wedge \cdots \wedge p_k \rightarrow q$), which intuitively can only derive positive information.

Review: tradeoffs

Formulas allowed	Inference rule	Complete?
Propositional logic	modus ponens	no
Propositional logic (only Horn clauses)	modus ponens	yes
Propositional logic	resolution	yes

- We saw that if our logical language was restricted to Horn clauses, then modus ponens alone was sufficient for completeness. For general propositional logic, modus ponens is insufficient.
- In this lecture, we'll see that a more powerful inference rule, **resolution**, is complete for all of propositional logic.

Summary

Propositional logic	**First-order logic**
model checking	n/a
$\Leftarrow$ propositionalization	
modus ponens (Horn clauses)	modus ponens++ (Horn clauses)
resolution (general)	resolution++ (general)

++: unification and substitution

Key idea: variables in first-order logic

Variables yield compact knowledge representations.

- To summarize, we have presented propositional logic and first-order logic. When there is a one-to-one mapping between constant symbols and objects, we can propositionalize, thereby converting first-order logic into propositional logic. This is needed if we want to use model checking to do inference.
- For inference based on syntactic derivations, there is a neat parallel between using modus ponens for Horn clauses and resolution for general formulas (after conversion to CNF). In the first-order logic case, things are more complex because we have to use unification and substitution to do matching of formulas.
- The main idea in first-order logic is the use of variables (not to be confused with the variables in variable-based models, which are mere propositional symbols from the point of view of logic), coupled with quantifiers.
- Propositional formulas allow us to express large complex sets of models compactly using a small piece of propositional syntax. Variables in first-order logic in essence takes this idea one more step forward, allowing us to effectively express large complex propositional formulas compactly using a small piece of first-order syntax.
- Note that variables in first-order logic are not same as the variables in variable-based models (CSPs). CSPs variables correspond to atomic formula and denote truth values. First-order logic variables denote objects.