More Logic, Propositional Logic, Theorem Proving

CS156

Chris Pollett

Oct 8, 2014

Outline

- Logic
- Propositional Logic
- Propositional Theorem Proving

Introduction

- Last Monday, we started talking about knowledge-based agents
- These agents are equipped with a knowledge-base (KB) that they can **tell** things as well as **ask** things.
- The basic KB agent returns an action by telling its KB is current percept, asking its KB an action query, and then telling the KB what it did before returning that action.
- We gave an example of a KB agent in action in the Wumpus world setting. Here an agent had to find a bar of gold in a partially observable world, given that a deadly wumpus might be in one room and other rooms might have pits.
- In adjacent to danger squares, the agent's sensors can detect either a breeze or a stench, using these properties the agent can often get the gold without dying, by inferring the actual state of the squares.
- We want to discuss this inference process in more detail so we started talking about logic. We discussed syntax, semantics, model and satisfies.
- Today, we continue to talk about logic.

Entailment

- We covered this slide on Monday, but I'll just review it briefly again.
- We have already said semantics tells us about the truth of a syntactically correct statement in a possible world/model
- We now want to consider logical reasoning. To do this we begin by looking at **entailment** between sentences -- the idea that one sentence follows from another sentence/set of sentences.
- We say $\alpha \models \beta$ iff $M(\alpha) \subseteq M(\beta)$. We read $\models$ as entails
- For example, if we considers worlds satisfying the Wumpus World rules, the fact that we didn't feel a breeze at $(1,1)$ entails that there is no pit in $(1,2)$. On the other hand, when we felt a breeze in $(1,2)$ it didn't entail there was a pit in $(1,1)$.
- The process of using entailment to derive conclusions is called **logical inference**.
- One way to infer $KB \models \alpha$ is to check every single possible model of say $M(KB)$ and verify it is a model of α. This is called **model checking**
- Generally, we would like to come with faster ways to determine entailment.
- We write $KB \vdash_i \alpha$ when we want to express that it was our inference algorithm i that derived α as following from KB.
- An inference algorithm that derives only entailed sentences is called **sound** or **truth-preserving**.
- An inference algorithm is **complete** if it can derive any sentence that is entailed.

Propositional Logic

- We now study a simple but powerful logic called **propositional logic**. We briefly discussed this at the end of last day when I went off notes.
- We want to describe in detail its syntax, its semantics, and entailment.

Syntax

- The syntax of propositional logic defines the allowable sentences.
- The simplest of these are the **atomic sentences** which consist of a single **propositional symbol**, (aka a **variable**).
- Each symbol stands for a proposition that can be true or false.
- We use symbols that start with an uppercase letter followed by other letters or subscripts to denote these symbols. For example, $P, Q, R, W_{1,3}, North$ are each propositional symbols.
- There are two propositional symbols with fixed meanings: True is the always-true proposition, False is the always-false proposition.
- Complex sentences are constructed from simpler sentences using parentheses and **logical connectives**...

Logical Connectives

The logical connectives are:

- $\neg$ (not). We read a sentence such as $\neg W$ is the **negation** of W. We use the term **literal** to mean either an atomic formula (positive literal) W or its negation (negative literal), $\neg W$.
- $\wedge$ (and). A sentence whose main connective is $\wedge$ such as $(W \wedge P)$ is called a **conjunction** and its parts are called **conjuncts**.
- $\vee$ (or). A sentence whose main connective is $\vee$ such as $((W \wedge E) \vee P)$ is called a **disjunction** and its parts are called **disjuncts**. ($\vee$ comes from the Latin "vel" which means or)
- $\Rightarrow$ (implies). A sentence such as $((A \wedge B) \Rightarrow C)$ is called an **implication**. Its **premise** or **antecedent** is $(A \wedge B)$ and its **conclusion** or **consequent** is C. Sometimes implies is written as $\supset$.
- $\Leftrightarrow$ (if and only if) The sentence $(A \Leftrightarrow B)$ is a **biconditional**.
- What constitutes a sentence can be given by the following context-free grammar:

  Sentence := AtomicSentence | ComplexSentence
  AtomicSentence := True | False | P | Q | R |...
  ComplexSentence := (Sentence) | [Sentence]
  | $\neg$ Sentence
  | Sentence $\wedge$ Sentence
  | Sentence $\vee$ Sentence

| Sentence $\Rightarrow$ Sentence
| Sentence $\Leftrightarrow$ Sentence

- Operator precedence $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$ to enforce meaning.

Semantics

- Now that we have defined what is a legal sentence, we now turn to what does a sentence mean? i.e., its semantics.
- In propositional logic, a model fixes the truth value -- true or false -- for every propositional symbol.
- So if the sentence in the KB used the propositional symbols A, B, C, D, then an example model might be: $M = \{A = true, B = false, C = false, D = true\}$
- For four propositional symbols there are $2^4 = 16$ possible models.

Truth of any Sentence in a Model

- The semantics of propositional logic must specify how to compute the truth value of any sentence, given a model.
- We do this inductively. We define True to be true in any model and False to be false in any model. The model itself says the truth or falsity of all AtomicSentences. Suppose we have already defined the truth of φ and ψ. Then:
  $\neg\varphi$ is true in M iff φ is false in M
  $\varphi \wedge \psi$ is true in M iff φ is true in M and ψ is true in M
  $\varphi \vee \psi$ is true in M iff φ is true in M or ψ is true in M
  $\varphi \Rightarrow \psi$ is true in M unless φ is true in M and ψ is false in M
  $\varphi \Leftrightarrow \psi$is true in M iff both φ and ψ are both true or both false

A Simple Knowledge Base

- Now that we have defined the semantics of propositional logic, we can construct a KB for the wumpus world:
- We use the following variables with intended meanings:
  - $P_{x,y}$ *is true if there is a pit in* $[x, y]$
  - $W_{x,y}$ *is true if there is a wumpus in* $[x, y]$ *dead or alive*
  - $B_{x,y}$ *is true if the agent perceives a breeze in* $[x, y]$
  - $S_{x,y}$ *is true if the agent perceives a stench in* $[x, y]$
- We now write some sentences which suffice to derive $\neg P_{1,2}$. We label each sentence R_i so we can refer to them.
  1. *There is no pit in [1,1]:*
     $R_1 : \neg P_{1,1}$
  2. *A square is breezy iff there is a pit in a neighboring square. This has to be stated for each square: for now we include just the relevant squares:*
     $R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$
     $R_3 : B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$
  3. *The preceding sentences are true in all wumpus worlds. Now we include the breeze percept for the first two squares visited in the specific world the agent is in:*
     $R_4 : \neg B_{1,1}$
     $R_5 : B_{2,1}$

A Simple Inference Procedure

- Our goal is to decide whether $KB \models \alpha$ for some sentence α.
- In particular, is $\neg P_{1,2}$ entailed by our KB.
- Our first algorithm for inference is a model-checking approach: enumerate the models, and check that α is true in every model in which KB is true.
- So for our wumpus world example the relevant variables to consider are:
  $B_{1,1}, B_{2,1}, P_{1,1}, P_{1,2}, P_{2,1}, P_{2,2}, \text{ and } P_{3,1}$
- These can take on $2^7 = 128$ values (possible worlds). In three of these the KB is true. In each of these it turns out that $\neg P_{1,2}$, so it does follow
- This is an exponential time algorithm in the number of propositions. It turns out that we don't know if we can do better by being clever: It would imply $NP = co\text{-}NP$.

Theorem Proving

- Rather than determining entailment by model checking, one could instead do it via theorem proving -- applying rules of inference directly to the sentences in our knowledge base to construct a proof of the desired sentence without consulting models.
- To do this we make use of something called **logical equivalence**: two sentences α and β are logically equivalent if they are true in the same set of models. For α, and β sentences we would write this as $\alpha \equiv \beta$.
- For example, $P \wedge Q$ and $Q \wedge P$ are logically equivalent
- The second concept needed for theorem proving is that of **validity**: A sentence is valid if it is true in all models. Valid sentences are also known as **tautologies**.
- The deduction theorem known to the ancient Greeks says that $\alpha \models \beta$ iff the sentence $\alpha \Rightarrow \beta$ is valid.
- So we will be interested in tautologies of the latter form.

Common Logical Equivalences

- $(\alpha \wedge \beta) \equiv (\beta \wedge \alpha)$ commutativety of $\wedge$
- $(\alpha \vee \beta) \equiv (\beta \vee \alpha)$ commutativety of $\vee$
- $((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma))$ associativety of $\wedge$
- $((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma))$ associativety of $\vee$
- $\neg(\neg\alpha) \equiv \alpha$ double-negation elimination
- $\alpha \Rightarrow \beta \equiv \neg\beta \Rightarrow \neg\alpha$ contraposition
- $\alpha \Rightarrow \beta \equiv \neg\alpha \vee \beta$ implication elimination
- $\alpha \Leftrightarrow \beta \equiv \alpha \Rightarrow \beta \wedge \beta \Rightarrow \alpha$ biconditional elimination
- $\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta)$ DeMorgan
- $\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta)$ DeMorgan
- $\alpha \wedge (\beta \vee \gamma) \equiv (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)$ distributivity of $\wedge$ or $\vee$
- $\alpha \vee (\beta \wedge \gamma) \equiv (\alpha \vee \beta) \wedge (\alpha \vee \gamma)$ distributivity of $\vee$ or $\wedge$

Satisfiability

- A sentence is **satisfiable** if it is true in some model.
- For example, $(A \wedge B)$ is satisfiable in the model $\{A = true, B = true\}$
- For propositional logic, the problem of given a propositional formula determining if it is satisfiable, is often written as SAT. It is an example of an NP-complete problem.
- By the deduction theorem, we had $\alpha \models \beta$ iff $\alpha \Rightarrow \beta$ is valid (aka, a tautology).
- The later is equivalent to $\neg\alpha \vee \beta$ and by deMorgan's rule equivalent to $\neg(\alpha \wedge \neg\beta)$.
- So if we could prove $\alpha \Rightarrow \beta$ by checking the unsatisfiability $\alpha \wedge \neg\beta$. This process is sometimes called **reduction ad absurdum**.
- If it is also called proof by **refutation** or proof by **contradiction**.

Inferences and Proofs

- We now look at **inference rules** that can be applied to derive a **proof** -- a chain of conclusions that leads to the establishment of some statement following from the knowledge base.
- The best known such rule is called **Modus Ponens** and it is written as:

$$\frac{\alpha, \alpha \Rightarrow \beta}{\beta}$$

  It states that if we have already derives the two statements α and $\alpha \Rightarrow \beta$ then we are allowed to derive β.
- For example, if we know (WumpusAhead $\wedge$ WumpusAlive) and (WumpusAhead $\wedge$ WumpusAlive) $\Rightarrow$ Shoot, then we are allowed to infer Shoot.
- Another useful rule is **And-Elimination**:

$$\frac{\alpha \wedge \beta}{\alpha}$$

- By considering all the possible true values for α and β one can show both Modus Ponens and And-Elimination are sound.
- One can also make any of our logical equivalences into a rule of inference. For example, biconditional elimination yields the following two inferences:

$$\frac{\alpha \Leftrightarrow \beta}{\alpha \Rightarrow \beta \wedge \beta \Rightarrow \alpha}$$

and

$$\frac{\alpha \Rightarrow \beta \wedge \beta \Rightarrow \alpha}{\alpha \Leftrightarrow \beta}$$

An Example

- Recall earlier we derived $\neg P_{1,2}$ from the following knowledge base statements R_i using model checking.
  1. *There is no pit in [1,1]:*
     $R_1 : \neg P_{1,1}$
  2. *A square is breezy iff there is a pit in a neighboring square. The has to be stated for each square: for now we include just the relevant squares:*
     $R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$
     $R_3 : B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$
  3. *The preceding sentences are true in all wumpus worlds. Now we include the breeze percept for the first two squares visited in the specific world the agent is in:*
     $R_4 : \neg B_{1,1}$
     $R_5 : B_{2,1}$
- Alternatively, we can come up with a (propositional) **proof** of $\neg P_{1,2}$ from the above knowledge base using rules of inferences.
- A proof consists of a sequence of lines $R_1, ...R_m$ where each R_i is either from the knowledge base or follows from some R_j and R_k where $j < i$ and $k < i$ by some rule of (propositional) inference.

The proof of $\neg P_{1,2}$

- Applying Biconditional-elimination to R_2 gives:
  $R_6 : \big(B_{1,1} \Rightarrow \big(P_{1,2} \vee P_{2,1}\big)\big) \wedge \big(\big(P_{1,2} \vee P_{2,1}\big) \Rightarrow B_{1,1}\big)$
- Applying And-Elimination to R_6 gives:
  $R_7 : \big(\big(P_{1,2} \vee P_{2,1}\big) \Rightarrow B_{1,1}\big)$
- Logical equivalence for contraposition gives:
  $R_8 : \big(\neg B_{1,1} \Rightarrow \neg\big(P_{1,2} \vee P_{2,1}\big)\big)$
- Modus Ponens of R_4 and R_8 give:
  $R_9 : \neg\big(P_{1,2} \vee P_{2,1}\big)$
- Applying DeMorgan's rule to R_9 yields
  $R_{10} : \neg P_{1,2} \wedge \neg P_{2,1}$
- The desired statement follows from R_{10} by And-Elimination