https://www.cs.ubc.ca/~jordon/teaching/cpsc322/2019w2/

Lecture 18 - Logic: Intro & Syntax

February 24th, 2020
https://www.cs.ubc.ca/~jordon/teaching/cpsc322/2019w2/lectures/lecture18.pdf

Logic is the language of Mathematics, used to define formal structures (e.g., sets, graphs) and to prove statements about them.

Consider how to represent an environment about which we have only partial (but certain) information. What do we need to represent?:

- **Facts** about the environment
- **Rules** of how the environment works

Propositional Logic

We begin with propositional logics, since this is the starting point for more complex logics. The primitive elements of propositional logic are ***propositions***: Boolean variables that can be $\{true, false\}$.

The goal is to illustrate the basic ideas as a starting point for more complex logics. Boolean nature can be exploited for efficiency.

The proposition symbols $p_1, p_2, \ldots$ etc. are sentences:

- If S is a sentence, then $\neg S$ is a sentence (negation)
- If S_1 and S_2 are sentences, then $S_1 \wedge S_2$ is a sentence (conjunction)
- If S_1 and S_2 are sentences, then $S_1 \vee S_2$ is a sentence (disjunction)
- If S_1 and S_2 are sentences, then $S_1 \Rightarrow S_2$ is a sentence (implication)
- If S_1 and S_2 are sentences, then $S_1 \Leftrightarrow S_2$ is a sentence (biconditional)

Propositional Logics in practice

- Agent is told (perceives) some facts about the world (a set of true propositions)
- Agent is told (already knows / learns) how the world works (a set of logical formulas)
- Agent can answer yes/no questions about whether other facts **must be true**

Using Logics to make inferences...

1. Begin with a task domain.
2. Distinguish those things you want to talk about
3. Choose symbols in the computer to denote propositions
4. Tell the system knowledge about the domain
5. Ask the system whether new statements about the domain are true or false

Propositional Definite Clauses

This is our first representation and reasoning system. We only have two kinds of statements:

- that a **proposition is true**: p_2
- that a proposition is true if one or more other propositions are true $p_2 \leftarrow p_1 \wedge p_3$

Propositional Definite Clauses: Syntax

Def: (atom)
An **atom** is a symbol starting with a lower case letter.

Def: (body)
A **body** is an atom, or is of the form $b_1 \wedge b_2$ where b_1 and b_2 are bodies.

Def: (definite clause)
A **definite clause** is an atom or is a rule of the form $h \leftarrow b$ where h is an atom and b is a body. We read this as "h if b".

Def: (knowledge base)
A **knowledge base** is a set of definite clauses.

Lecture 19 - Logic: Semantics and Bottom-Up Proofs

February 26th, 2020
https://www.cs.ubc.ca/~jordon/teaching/cpsc322/2019w2/lectures/lecture19.pdf

Propositional Definite Clauses Semantics: Interpretation

Semantics allows you to relate the symbols in the logic to the domain you're trying to model. An atom can be T or F.

Def: (Interpretation)
An *interpretation* I assigns a truth value to each atom.

So an interpretation is just a **possible world**

We can use the **interpretation** to determine the truth value of **clauses** and **knowledge bases**.

Def: (truth values of statements)
A body $b_1 \wedge b_2$ is true in I if and only if b_1 is true in I and b_2 is true in I.

Def: (truth values of statements cont')
A rule $h \leftarrow b$ is false in I if and only if b is true in I and h is false in I.

Semantics of a KB:
A **knowledge base KB** is true in I if and only if every clause in KB is true in I.

Def: Model
A **model** of a set of clauses (a Knowledge base) is an interpretation in which all clauses are ***true***.

Logical Consequence

Def: (logical consequence)
If KB is a set of clauses and G is a conjunction of atoms, then G is a logical consequence of KB, written a $KB \models G$, if G is true in every model of KB.

- It can also be said that G logically follows from KB, or that KB entails G.
- In other words, $KB \models G$ if there is no interpretation I in which KB is $true$ and G is $false$.

Soundness and Completeness

Suppose we are told that we have a **proof procedure for PDCL**. What needs to be shown in order for us to trust the given procedure?

Recall:
$K \vdash G$ means that G can be derived by, the given proof procedure, from KB.
$K \models G$ means that G is true in all models of KB

Def: (soundness)
A proof system is sound if $KB \vdash G$ implies that $KB \models G$

Def: (completeness)
A proof procedure is complete if $KB \models G$ implies that $KB \vdash G$

Bottom-up Ground Proof Procedure

One mainly used rule of derivation, which is a generalized form of ***modus ponens*** is defined as follows:

If $h \leftarrow b_1 \wedge \ldots \wedge b_m$ is a clause in the knowledge base, and each b_i has been derived, then h can be derived.

The procedure uses the above rule to find whether G logically follows from the knowledge base. We then have that $KB \vdash G$ if at the end of the procedure $G \subseteq C$:

Algorithm:

$C := \{\}$

repeat:

select clause "$h \leftarrow b_1 \wedge ... \wedge b_m$" in KB such that $b_i \in C$ for all i, and $h \notin C$.
$C := C \cup \{h\}$

until no more clauses can be selected