7.4.1 Syntax

The **syntax** of propositional logic defines the allowable sentences. The **atomic sentences** consist of a single **proposition symbol**. Each such symbol stands for a proposition that can be true or false. We use symbols that start with an uppercase letter and may contain other letters or subscripts, for example: P, Q, R, H71,3 and North. The names are arbitrary but are often chosen to have some mnemonic value—we use W1,3 to stand for the proposition that the wumpus is in [1,3). (Remember that symbols such as W1,3 are **atomic**, i.e., W, 1, and 3 are not meaningful parts of the symbol.) There are two proposition symbols with fixed meanings: **True** is the always-true proposition and **False** is the always-false proposition.

**Complex sentences** are constructed from simpler sentences, using parentheses and **logical connectives**. There are five connectives in common use:

- **(not).** A sentence such as —W1,3 is called the negation of W1,3. A **literal** is either an atomic sentence (a **positive literal**) or a negated atomic sentence (a **negative literal**).
- **A (and).** A sentence whose main connective is **A**, such as W1,3 A P3,1 is called a **conjunction**; its parts are the conjuncts. (The A looks like an "A" for "And."
- **V (or).** A sentence using V, such as (W1,3 A P3,1) V W2,2, is a **disjunction** of the disjuncts (W1,3 A P31) and W2,2. (Historically, the V comes from the Latin “vet,” which means "or" For most people, it is easier to remember V as an upside-down A.)
- **implies.** A sentence such as (W1,3 A P3,1) —W2,2 is called an **implication** (or conditional). Its **premise** or **antecedent** is (W1,3 A P3,1), and its **conclusion** or **consequent** is —W2,2. Implications are also known as **rules** or **if then** statements. The implication symbol is sometimes written in other books as i or \(\iff\) (if and only if. The sentence W1,3 \(\iff\) W2,2 is a **biconditional**. Some other books write this as

<table>
<thead>
<tr>
<th>Sentence</th>
<th>AtomicSentence</th>
<th>ComplexSentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>AtomicSentence</td>
<td>Tree False P Q</td>
<td></td>
</tr>
<tr>
<td>ComplexSentence</td>
<td>Sentence</td>
<td>[Sentence]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sentence</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sentence V Sentence</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sentence Sentence</td>
</tr>
<tr>
<td>OPERATOR PRECEDENCE</td>
<td>:</td>
<td>A, V, (\iff)</td>
</tr>
</tbody>
</table>

Figure 7.7 A **ENS (Backus—Naur Form)** grammar of sentences in propositional logic, along with operator precedences, from highest to lowest.
Section 7.4. Propositional Logic: A Very Simple Logic

Figure 7.7 gives a formal grammar of propositional logic; see page 1060 if you are not familiar with the BNF notation. The BNF grammar by itself is ambiguous; a sentence with several operators can be parsed by the grammar in multiple ways. To eliminate the ambiguity we define a precedence for each operator. The "not" operator \( \neg \) has the highest precedence, which means that in the sentence \( \neg A \land B \) the binds most tightly, giving us the equivalent of \( (\neg A) \land B \) rather than \( A \land B \). (The notation for ordinary arithmetic is the same: \(-2 + 4\) is 2, not 6.) When in doubt, use parentheses to make sure of the right interpretation. Square brackets mean the same thing as parentheses; the choice of square brackets or parentheses is solely to make it easier for a human to read a sentence.

### 7.4.2 Semantics

Having specified the syntax of propositional logic, we now specify its semantics. The semantics defines the rules for determining the truth of a sentence with respect to a particular model. In propositional logic, a model simply fixes the truth value—true or false—for every proposition symbol. For example, if the sentences in the knowledge base make use of the proposition symbols \( P_1, P_2, P_3 \), then one possible model is

\[
M_1 = \{ P_{1,2} = \text{false}, P_{2,2} = \text{false}, P_{3,1} = \text{true} \}
\]

With three proposition symbols, there are \( 2^3 = 8 \) possible models—exactly those depicted in Figure 7.5. Notice, however, that the models are purely mathematical objects with no necessary connection to wumpus worlds. \( P_{1,2} \) is just a symbol; it might mean "there is a pit in \([1,2]\)" or "I'm in Paris today and tomorrow."

The semantics for propositional logic must specify how to compute the truth value of any sentence, given a model. This is done recursively. All sentences are constructed from atomic sentences and the five connectives; therefore, we need to specify how to compute the truth of atomic sentences and how to compute the truth of sentences formed with each of the five connectives. Atomic sentences are easy:

- True is true in every model and False is false in every model.
- The truth value of every other proposition symbol must be specified directly in the model. For example, in the model nil given earlier, \( P_{1,2} \) is false.

For complex sentences, we have five rules, which hold for any subsentences \( P \) and \( Q \) in any model \( M \) (here "iff" means "if and only if"):

- \( \neg P \) is true iff \( P \) is false in \( M \).
- \( P \land Q \) is true iff both \( P \) and \( Q \) are true in \( M \).
- \( P \lor Q \) is true iff either \( P \) or \( Q \) is true in \( M \).
- \( P =r Q \) is true unless \( P \) is true and \( Q \) is false in \( M \).
- \( P \Leftrightarrow Q \) is true iff \( P \) and \( Q \) are both true or both false in \( M \).

The rules can also be expressed with truth tables that specify the truth value of a complex sentence for each possible assignment of truth values to its components. Truth tables for the five connectives are given in Figure 7.8. From these tables, the truth value of any sentence can be computed with respect to any model \( M \) by a simple recursive evaluation. For example,