today's topics:
- propositional logic

Introduction
- "Weak" (search-based) problem-solving does not scale to real problems.
- To succeed, problem solving needs domain specific knowledge.
- In search, knowledge = heuristic.
- We need to be able to represent knowledge efficiently.
- One way to do this is to use logic.

What is a Logic?
- When most people say "logic", they mean either propositional logic or first-order predicate logic.
- However, the precise definition is quite broad, and literally hundreds of types of logics have been studied by philosophers, computer scientists and mathematicians.
- Any "formal system" can be considered a logic if it has:
  - a well-defined syntax;
  - a well-defined semantics; and
  - a well-defined proof theory.

- The syntax of a logic defines the syntactically acceptable objects of the language, which are properly called well-formed formulae (wff). (We shall just call them formulae.)
- The semantics of a logic associate each formula with a meaning.
- The proof theory is concerned with manipulating formulae according to certain rules.
Propositional Logic

• The simplest, and most abstract, logic we can study is called propositional logic.

• Definition: A proposition is a statement that can be either true or false; it must be one or the other, and it cannot be both.

• EXAMPLES. The following are propositions:
  – the reactor is on;
  – the wing-flaps are up;
  – Marvin K Mooney is president.

  whereas the following are not:
  – are you going out somewhere?
  – 2 + 3

• It is possible to determine whether any given statement is a proposition by prefixing it with: 
  It is true that . . . 
  and seeing whether the result makes grammatical sense.

• We now define atomic propositions. Intuitively, these are the set of smallest propositions.

• Definition: An atomic proposition is one whose truth or falsity does not depend on the truth or falsity of any other proposition.

• So all the above propositions are atomic.

Now, rather than write out propositions in full, we will abbreviate them by using propositional variables.

• It is standard practice to use the lower-case roman letters 
  \(p, q, r, \ldots\)

  to stand for propositions.

  Sometimes, Greek letters are also used, e.g.:
  \(\phi, \Phi, \psi, \pi\)

• If we do this, we must define what we mean by writing something like:

  Let \(p\) be Marvin K Mooney is president.

• Another alternative is to write something like \(marvin\text{\_is\text{\_president}}\), so that the interpretation of the propositional variable becomes obvious.

The Connectives

• Now, the study of atomic propositions is pretty boring. We therefore now introduce a number of connectives which will allow us to build up complex propositions.

• The connectives we introduce are:

  \(\wedge\) and \((&\text{ or }\cdot)\)
  \(\lor\) or \((\mid\text{ or }\+)
  \neg\) not \((\sim)\)
  \(\Rightarrow\) implies \((\supset\text{ or }\rightarrow)\)
  \(\Leftrightarrow\) iff \((\leftrightarrow)\)

  (Some books use other notations; these are given in parentheses.)
And

- Any two propositions can be combined by the word “and” to form a third proposition called the conjunction of the original propositions.
- **Definition:** If \( p \) and \( q \) are arbitrary propositions, then the conjunction of \( p \) and \( q \) is written

\[
p \land q
\]

and will be true iff both \( p \) and \( q \) are true.

Or

- Any two propositions can be combined by the word “or” to form a third proposition called the disjunction of the originals.
- **Definition:** If \( p \) and \( q \) are arbitrary propositions, then the disjunction of \( p \) and \( q \) is written

\[
p \lor q
\]

and will be true iff either \( p \) is true, or \( q \) is true, or both \( p \) and \( q \) are true.

- We can summarise the operation of \( \land \) in a truth table. The idea of a truth table for some formula is that it describes the behavior of a formula under all possible interpretations of the primitive propositions that are included in the formula.
- If there are \( n \) different atomic propositions in some formula, then there are \( 2^n \) different lines in the truth table for that formula. (This is because each proposition can take one of 2 values—i.e., true or false.)
- Let us write \( T \) for truth, and \( F \) for falsity. Then the truth table for \( p \land q \) is:

<table>
<thead>
<tr>
<th>( p )</th>
<th>( q )</th>
<th>( p \land q )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( F )</td>
<td>( F )</td>
<td>( F )</td>
</tr>
<tr>
<td>( F )</td>
<td>( T )</td>
<td>( F )</td>
</tr>
<tr>
<td>( T )</td>
<td>( F )</td>
<td>( F )</td>
</tr>
<tr>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
</tr>
</tbody>
</table>

- The operation of \( \lor \) is summarised in the following truth table:

<table>
<thead>
<tr>
<th>( p )</th>
<th>( q )</th>
<th>( p \lor q )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( F )</td>
<td>( F )</td>
<td>( F )</td>
</tr>
<tr>
<td>( F )</td>
<td>( T )</td>
<td>( T )</td>
</tr>
<tr>
<td>( T )</td>
<td>( F )</td>
<td>( T )</td>
</tr>
<tr>
<td>( T )</td>
<td>( T )</td>
<td>( T )</td>
</tr>
</tbody>
</table>

- Note that this “or” is a little different from the usual meaning we give to “or” in everyday natural language.
If... Then...

- Many statements, particularly in mathematics, are of the form:

  \( \text{if } p \text{ is true then } q \text{ is true.} \)

  Another way of saying the same thing is to write:

  \( p \) implies \( q. \)

- In propositional logic, we have a connective that combines two propositions into a new proposition called the conditional, or implication of the originals, that attempts to capture the sense of such a statement.

  \( p \Rightarrow q \)

  - The truth table for \( \Rightarrow \) is:

    | \( p \) | \( q \) | \( p \Rightarrow q \) |
    |-------|-------|-----------------|
    | \( F \) | \( F \) | \( T \)           |
    | \( F \) | \( T \) | \( T \)           |
    | \( T \) | \( F \) | \( F \)           |
    | \( T \) | \( T \) | \( T \)           |

  - The \( \Rightarrow \) operator is the hardest to understand of the operators we have considered so far, and yet it is extremely important.
  
  - If you find it difficult to understand, just remember that the \( p \Rightarrow q \) means "if \( p \) is true, then \( q \) is true". If \( p \) is false, then we don’t care about \( q \), and by default, we make \( p \Rightarrow q \) evaluate to \( T \) in this case.

  - Terminology: if \( \phi \) is the formula \( p \Rightarrow q \), then \( p \) is the antecedent of \( \phi \) and \( q \) is the consequent.

Another common form of statement in maths is:

\( p \) is true if, and only if, \( q \) is true.

- The sense of such statements is captured using the biconditional operator.

  \( p \iff q \)

  - Another common form of statement in maths is:

    \( p \) is true if, and only if, \( q \) is true.

    - The sense of such statements is captured using the biconditional operator.

    - Definition: If \( p \) and \( q \) are arbitrary propositions, then the biconditional of \( p \) and \( q \) is written:

      \( p \iff q \)

      and will be true iff either:

      1. \( p \) and \( q \) are both true; or
      2. \( p \) and \( q \) are both false.
• The truth table for $\iff$ is:

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$p \iff q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F$</td>
<td>$F$</td>
<td>$T$</td>
</tr>
<tr>
<td>$F$</td>
<td>$T$</td>
<td>$F$</td>
</tr>
<tr>
<td>$T$</td>
<td>$F$</td>
<td>$F$</td>
</tr>
<tr>
<td>$T$</td>
<td>$T$</td>
<td>$T$</td>
</tr>
</tbody>
</table>

• If $p \iff q$ is true, then $p$ and $q$ are said to be logically equivalent. They will be true under exactly the same circumstances.

• All of the connectives we have considered so far have been binary: they have taken two arguments.
• The final connective we consider here is unary. It only takes one argument.
• Any proposition can be prefixed by the word ‘not’ to form a second proposition called the negation of the original.

• We can nest complex formulae as deeply as we want.
• We can use parentheses i.e., $\, )\,$, to disambiguate formulae.
• EXAMPLES. If $p, q, r, s$ and $t$ are atomic propositions, then all of the following are formulae:

  - $p \land q \Rightarrow r$
  - $p \land (q \Rightarrow r)$
  - $(p \land (q \Rightarrow r)) \lor s$
  - $((p \land (q \Rightarrow r)) \lor s) \land t$

  whereas none of the following is:

  - $p \land$
  - $p \land q$
  - $p \neg$
Tautologies & Consistency

● Given a particular formula, can you tell if it is true or not?
● No — you usually need to know the truth values of the component atomic propositions in order to be able to tell whether a formula is true.

● Definition: A valuation is a function which assigns a truth value to each primitive proposition.
● In C, we might write:

```c
short Val( AtomicProp *p ) {
    if ( *p )
        return( 1 ); // true
    else
        return( 0 ); // false
}
```
● Given a valuation, we can say for any formula whether it is true or false.
● A valuation is also known as an interpretation.

EXAMPLE. Suppose we have a valuation \( \nu \), such that:
\[
\nu(p) = F
\]
\[
\nu(q) = T
\]
\[
\nu(r) = F
\]
Then the truth value of \( (p \lor q) \Rightarrow r \) is evaluated by:
\[
(\nu(p) \lor \nu(q)) \Rightarrow \nu(r)
\]
\[
= (F \lor T) \Rightarrow F
\]
\[
= T \Rightarrow F
\]
\[
= F
\]
Line (3) is justified since we know that \( F \lor T = T \).
Line (4) is justified since \( T \Rightarrow F = F \).
If you can’t see this, look at the truth tables for \( \lor \) and \( \Rightarrow \).

When we consider formulae in terms of interpretations, it turns out that some have interesting properties.

● Definition:
1. A formula is a tautology iff it is true under every valuation;
2. A formula is consistent iff it is true under at least one valuation;
3. A formula is inconsistent iff it is not made true under any valuation.
● A tautology is said to be valid.
● A consistent formula is said to be satisfiable.
● An inconsistent formula is said to be unsatisfiable.

Theorem: \( \phi \) is a tautology iff \( \neg \phi \) is unsatisfiable.
● Now, each line in the truth table of a formula corresponds to a valuation.
● So, we can use truth tables to determine whether or not formulae are tautologies.
● If every line in the truth table has value \( T \), the the formula is a tautology.
● Also use truth-tables to determine whether or not formulae are consistent.
To check for consistency, we just need to find one valuation that satisfies the formula.

If this turns out to be the first line in the truth-table, we can stop looking immediately: we have a certificate of satisfiability.

To check for validity, we need to examine every line of the truth-table. No short cuts.

The lesson? Checking satisfiability is easier than checking validity.

Syntax

We have already informally introduced propositional logic; we now define it formally.

To define the syntax, we must consider what symbols can appear in formulae, and the rules governing how these symbols may be put together to make acceptable formulae.

Definition: Propositional logic contains the following symbols:

1. A set of primitive propositions, \( \Phi = \{ p, q, r, \ldots \} \).
2. The unary logical connective ‘\( \neg \)’ (not), and binary logical connective ‘\( \lor \)’ (or).
3. The punctuation symbols ‘\( ) \)’ and ‘\( \)’.

The primitive propositions will be used to represent statements such as:

- I am in Brooklyn
- It is raining
- It is Friday 11th November 2005.

These are primitive in the sense that they are indivisible; we cannot break them into smaller propositions.

The remaining logical connectives (\( \land, \Rightarrow, \Leftrightarrow \)) will be introduced as abbreviations.

We now look at the rules for putting formulae together.

Definition: The set \( WFF \), of (well formed) formulae of propositional logic, is defined by the following rules:

1. If \( p \in \Phi \), then \( p \in WFF \).
2. If \( \phi \in WFF \), then:
   \[ \neg \phi \in WFF \]
   \[ (\phi) \in WFF \]
3. If \( \phi \in WFF \) and \( \psi \in WFF \), then \( \phi \lor \psi \in WFF \).
• The remaining connectives are defined by:
  \[
  \phi \land \psi = \neg (\neg \phi \lor \neg \psi)
  \]
  \[
  \phi \Rightarrow \psi = (\neg \phi) \lor \psi
  \]
  \[
  \phi \iff \psi = (\phi \Rightarrow \psi) \land (\psi \Rightarrow \phi)
  \]

• These connectives are interpreted:
  \[
  \land \quad \text{And}
  \]
  \[
  \Rightarrow \quad \text{Implies (if... then...)}
  \]
  \[
  \iff \quad \text{If, and only if}
  \]

• This concludes the formal definition of syntax.

---

Semantics

• We now look at the more difficult issue of semantics, or meaning.

• What does a proposition mean?
  • That is, when we write
    \[
    \text{It is raining.}
    \]
    what does it mean?
  • From the point of view of logic, this statement is a proposition: something that is either \( \top \) or \( \bot \).
  • The meaning of a primitive proposition is thus either \( \top \) or \( \bot \).
  • In the same way, the meaning of a formula of propositional logic is either \( \top \) or \( \bot \).

• QUESTION: How can we tell whether a formula is \( \top \) or \( \bot \)?
  • For example, consider the formula
    \[
    (p \land q) \Rightarrow r
    \]
    Is this \( \top \)?
  • The answer must be: possibly. It depends on your interpretation of the primitive propositions \( p, q \) and \( r \).
  • The notion of an interpretation is easily formalised.

• Definition: An interpretation for propositional logic is a function
  \[
  \pi : \Phi \mapsto \{ T, F \}
  \]
  which assigns \( T \) (true) or \( F \) (false) to every primitive proposition.

• But an interpretation only gives us the meaning of primitive propositions; what about complex propositions — arbitrary formulae?

• We use some rules which tell us how to obtain the meaning of an arbitrary formulae, given some interpretation.

• Before presenting these rules, we introduce a symbol: \( \models \). If \( \pi \) is an interpretation, and \( \phi \) is a formula, then the expression
  \[
  \pi \models \phi
  \]
will be used to represent the fact that \( \phi \) is \( \top \) under the interpretation \( \pi \).
  • Alternatively, if \( \pi \models \phi \), then we say that:
    - \( \pi \) satisfies \( \phi \); or
    - \( \pi \) models \( \phi \).
  • The symbol \( \models \) is called the semantic turnstile.
• The rule for primitive propositions is quite simple. If \( p \in \Phi \) then 
\[ \pi \vdash p \text{ iff } \pi(p) = T. \]

• The remaining rules are defined recursively.

• The rule for \( \neg \):
\[ \pi \vdash \neg \phi \text{ iff } \pi \not\vdash \phi \]
(\( \not\vdash \) means ‘does not satisfy’.)

• The rule for \( \lor \):
\[ \pi \vdash \phi \lor \psi \text{ iff } \pi \vdash \phi \text{ or } \pi \vdash \psi \]

• Since these are the only connectives of the language, these are the only semantic rules we need.

• Since:
\[ \phi \Rightarrow \psi \]
is defined as:
\[ (\neg \phi) \lor \psi \]
it follows that:
\[ \pi \vdash \phi \Rightarrow \psi \text{ iff } \pi \not\vdash \phi \text{ or } \pi \vdash \psi \]

• And similarly for the other connectives we defined.

• If we are given an interpretation \( \pi \) and a formula \( \phi \), it is a simple (if tedious) matter to determine whether \( \pi \vdash \phi \).

• We just apply the rules above, which eventually bottom out of the recursion into establishing if some proposition is true or not.

• So for:
\[ (p \lor q) \land (q \lor r) \]
we first establish if \( p \lor q \) or \( q \lor r \) are true and then work up to the compound proposition.

Proof Theory

• What is logic used for? A number of things, but most importantly, it is a language for representing the properties of things.

• But also, we hope it will give us a method for establishing the properties of things.

• To see how logic may be used to establish the properties of things, it helps to look at its history.

• Logic was originally developed to make the notion of an argument precise.
(We do not mean argument as in fighting here!)
Here is a classic argument:

All men are mortal
Socrates is a man

\[ \therefore \text{Socrates is mortal} \]

This example serves to illustrate a number of features of arguments:

- The argument has a number of premises — these are the statements that appear before the horizontal line;
- The argument has a conclusion — this is the statement that appears after the horizontal line;
- The argument has the form

\[ \text{If} \quad \text{you accept that} \quad \text{the premises are true} \]
\[ \text{then} \quad \text{you must accept that} \quad \text{the conclusion is true.} \]

In mathematics, we are concerned with when arguments are sound.

To formalise the notion of a sound argument, we need some extra terminology...

**Definition:** If \( \phi \in \text{WFF} \), then:

1. if there is some interpretation \( \pi \) such that

\[ \pi \models \phi \]

then \( \phi \) is said to be satisfiable, otherwise \( \phi \) is unsatisfiable.

2. if

\[ \pi \models \phi \]

for all interpretations \( \pi \), then \( \phi \) is said to be valid.

Valid formulae of propositional logic are called tautologies.

**Theorem:**

1. If \( \phi \) is a valid formula, then \( \neg \phi \) is unsatisfiable;
2. If \( \neg \phi \) is unsatisfiable, then \( \phi \) is valid.

We indicate that a formula \( \phi \) is valid by writing

\[ \models \phi \]

We can now define the logical consequence.
• Definition: If \( \{ \phi_1, \ldots, \phi_n \} \subseteq WFF \) then \( \phi \) is said to be a logical consequence of \( \{ \phi_1, \ldots, \phi_n \} \) iff \( \phi \) is satisfied by all interpretations that satisfy 
\[ \phi_1 \land \cdots \land \phi_n. \]

• We indicate that \( \phi \) is a logical consequence of \( \phi_1, \ldots, \phi_n \) by writing 
\[ \{ \phi_1, \ldots, \phi_n \} \models \phi. \]

• An expression like this is called a semantic sequent.

• Theorem: \[ \{ \phi_1, \ldots, \phi_n \} \models \phi. \] iff \[ \models (\phi_1 \land \cdots \land \phi_n) \Rightarrow \phi. \]

• So we have a method for determining whether \( \phi \) is a logical consequence of \( \phi_1, \ldots, \phi_n \): we use a truth table to see whether \( \phi_1 \land \cdots \land \phi_n \Rightarrow \phi \) is a tautology. If it is, then \( \phi \) is a logical consequence of \( \phi_1, \ldots, \phi_n \).

• Our main concern in proof theory is thus to have a technique for determining whether a given formula is valid, as this will then give us a technique for determining whether some formula is a logical consequence of some others.

• EXAMPLE. Show that \( p \land q \models p \lor q \).

To do this, we construct a truth-table for 
\[ (p \land q) \Rightarrow (p \lor q). \]

Here it is:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>( p \land q )</th>
<th>( p \lor q )</th>
<th>( (p \land q) \Rightarrow (p \lor q) )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( p )</td>
<td>( q )</td>
<td>( p \land q )</td>
<td>( p \lor q )</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>

Since 
\[ (p \land q) \Rightarrow (p \lor q). \]

is true under every interpretation, we have that \( p \lor q \) is a logical consequence of \( p \land q \).

• The notion of logical consequence we have defined above is acceptable for a definition of a sound argument, but is not very helpful for checking whether a particular argument is sound or not.

• The problem is that we must look at all the possible interpretations of the primitive propositions. While this is acceptable for, say, 4 primitive propositions, it will clearly be unacceptable for 100 propositions, as it would mean checking \( 2^{100} \) interpretations. (Moreover, for first-order logic, there will be an infinite number of such interpretations.)

• What we require instead is an alternative version of logical consequence, that does not involve this kind of checking. This leads us to the idea of syntactic proof.
‘Syntactic’ Proof

• The idea of syntactic proof is to replace the semantic checking to determine whether a formula is valid by a procedure that involves purely syntactic manipulation.

• The kinds of techniques that we shall use are similar to those that we use when solving problems in algebra.

• The basic idea is that to show that \( \phi \) is a logical consequence of \( \phi_1, \ldots, \phi_n \), we use a set of rules to manipulate formulae.

  If we can derive \( \phi \) from \( \phi_1, \ldots, \phi_n \) by using these rules, then \( \phi \) is said to be proved from \( \phi_1, \ldots, \phi_n \), which we indicate by writing

\[
\phi_1, \ldots, \phi_n \vdash \phi.
\]

• The symbol \( \vdash \) is called the syntactic turnstile.

• An expression of the form

\[
\phi_1, \ldots, \phi_n \vdash \phi.
\]

is called a syntactic sequent.

• A rule has the general form:

\[
\vdash \phi_1, \ldots, \vdash \phi_n \text{ rule name} \quad \vdash \phi
\]

Such a rule is read:

If \( \phi_1, \ldots, \phi_n \) are proved
then
\( \phi \) is proved.

• EXAMPLE. Here is an example of such a rule:

\[
\frac{\vdash \phi \vdash \psi}{\vdash \phi \land \psi} \text{ \( \land \)-I}
\]

This rule is called and introduction. It says that if we have proved \( \phi \), and we have also proved \( \psi \), then we can prove \( \phi \land \psi \).

• EXAMPLE. Here is another rule:

\[
\frac{\vdash \phi \land \psi}{\vdash \phi} \quad \text{\( \land \)-E}
\]

This rule is called and elimination. It says that if we have proved \( \phi \land \psi \), then we can prove both \( \phi \) and \( \psi \); it allows us to eliminate the \( \land \) symbol from between them.

• Let us now try to define precisely what we mean by proof.

• Definition: (Proof) If

\[
\{ \phi_1, \ldots, \phi_n, \phi \} \subseteq \text{WFF}
\]

then there is a proof of \( \phi \) from \( \phi_1, \ldots, \phi_n \) iff there exists some sequence of formulae

\[
\psi_1, \ldots, \psi_n
\]

such that \( \psi_n = \phi \) and each formula \( \psi_k \), for \( 1 \leq k < n \) is either one of the formulae \( \phi_1, \ldots, \phi_n \) or else is the conclusion of a rule whose antecedents appeared earlier in the sequence.
• If there is a proof of $\phi$ from $\phi_1, \ldots, \phi_m$, then we indicate this by writing:
  \[ \phi_1, \ldots, \phi_m \vdash \phi. \]

• It should be clear that the symbols $\vdash$ and $|$ are related. We now have to state exactly how they are related.

• There are two properties of $\vdash$ to consider:
  – soundness;
  – completeness.
  – Intuitively, $\vdash$ is said to be sound if it is correct, in that it does not let us derive something that is not true.
  – Intuitively, completeness means that $\vdash$ will let us prove anything that is true.

---

**A Proof System**

• There are many proof systems for propositional logic; we shall look at a simple one.

• First, we have an unusual rule that allows us to introduce any tautology.

  **RULE** $\vdash \phi$ TAUIf $\phi$ is a tautology

• Because a tautology is true there is no problem bringing it into the proof.

---

**Definition:** (Soundness) A proof system $\vdash$ is said to be sound with respect to semantics $|$ iff
  \[ \phi_1, \ldots, \phi_m \vdash \phi \]
  implies
  \[ \phi_1, \ldots, \phi_m \vdash \phi. \]

**Definition:** (Completeness) A proof system $\vdash$ is said to be complete with respect to semantics $|$ iff
  \[ \phi_1, \ldots, \phi_m \vdash \phi \]
  implies
  \[ \phi_1, \ldots, \phi_m \vdash \phi. \]

---

**Next, rules for eliminating connectives.**

  \[
  \frac{\vdash \phi \land \psi; \quad \vdash \phi}{\vdash \phi \land \psi} \ 
  \]

  \[
  \frac{\vdash \phi_1 \lor \cdots \lor \phi_n; \quad \phi_i \vdash \phi_i; \quad \cdots; \quad \phi_n \vdash \phi}{\vdash \phi} \ 
  \]
• An alternative ∨ elimination rule is:

\[ \vdash \phi \lor \psi ; \quad \lor\text{-E} \]
\[ \vdash \phi \Rightarrow \chi ; \quad \lor\text{-E} \]
\[ \vdash \psi \Rightarrow \chi \]

\[ \vdash \chi \]

• Next, a rule called modus ponens, which lets us eliminate ⇒.

\[ \vdash \phi \Rightarrow \psi ; \quad \Rightarrow\text{-E} \]
\[ \vdash \phi \quad \Rightarrow\text{-E} \]
\[ \vdash \psi \]

Proof Examples

• In this section, we give some examples of proofs in the propositional calculus.

• Example 1:

\[ p \land q \vdash q \land p \]

1. \( p \land q \) Given
2. \( p \) From 1 using \( \land\text{-E} \)
3. \( q \) 1, \( \land\text{-E} \)
4. \( q \land p \) 2, 3, \( \land\text{-I} \)

• Next, rules for introducing connectives.

\[ \vdash \phi_1, \ldots, \vdash \phi_n ; \quad \land\text{-I} \]
\[ \vdash \phi_1 \land \cdots \land \phi_n \quad \land\text{-I} \]
\[ \vdash \phi_1 \lor \cdots \lor \phi_n \]

• We have a rule called the deduction theorem. This rule says that if we can prove \( \psi \) from \( \phi \), then we can prove that \( \phi \Rightarrow \psi \).

\[ \phi \vdash \psi \]
\[ \vdash \phi \Rightarrow \psi \]

• There are a whole range of other rules, which we shall not list here.

Example 2:

\[ p \land q \vdash p \lor q \]

1. \( p \land q \) Given
2. \( p \) 1, \( \land\text{-E} \)
3. \( p \lor q \) 2, \( \lor\text{-I} \)
Example 3:
\[ p \land q \Rightarrow r \]
1. \( p \land q \) \hspace{1em} \text{Given}
2. \( p \) \hspace{1em} 1, \land \text{-E}
3. \( p \Rightarrow r \) \hspace{1em} \text{Given}
4. \( r \) \hspace{1em} 2, 3, \Rightarrow \text{-E}

Example 4:
\[ p \Rightarrow q, q \Rightarrow r \Rightarrow p \Rightarrow r \]
1. \( p \Rightarrow q \) \hspace{1em} \text{Given}
2. \( q \Rightarrow r \) \hspace{1em} \text{Given}
3. \( p \) \hspace{1em} \text{Ass}
4. \( q \) \hspace{1em} 1, 3, \Rightarrow \text{-E}
5. \( r \) \hspace{1em} 2, 4, \Rightarrow \text{-E}
6. \( p \Rightarrow r \) \hspace{1em} 3, 5, \Rightarrow \text{-I}

Example 5:
\[ (p \land q) \Rightarrow r \Rightarrow (p \Rightarrow (q \Rightarrow r)) \]
1. \( (p \land q) \Rightarrow r \) \hspace{1em} \text{Given}
2. \( p \) \hspace{1em} \text{Ass}
3. \( q \) \hspace{1em} \text{Ass}
4. \( p \land q \) \hspace{1em} 2, 3, \land \text{-I}
5. \( r \) \hspace{1em} 1, 4, \Rightarrow \text{-I}
6. \( q \Rightarrow r \) \hspace{1em} 3-5, \Rightarrow \text{-I}
7. \( p \Rightarrow (q \Rightarrow r) \) \hspace{1em} 2-6, \Rightarrow \text{-I}

Example 6:
\[ p \Rightarrow (q \Rightarrow r) \Rightarrow (p \land q) \Rightarrow r \]
1. \( p \Rightarrow (q \Rightarrow r) \) \hspace{1em} \text{Given}
2. \( p \land q \) \hspace{1em} \text{Ass}
3. \( p \) \hspace{1em} 2, \land \text{-E}
4. \( q \) \hspace{1em} 2, \land \text{-E}
5. \( q \Rightarrow r \) \hspace{1em} 1, 3, \Rightarrow \text{-E}
6. \( r \) \hspace{1em} 4, 5, \Rightarrow \text{-E}
7. \( (p \land q) \Rightarrow r \) \hspace{1em} 2-6, \Rightarrow \text{-I}
Example 7:
\[ p \Rightarrow q, \neg q \vdash \neg p \]

1. \( p \Rightarrow q \) Given
2. \( \neg q \) Given
3. \( p \) Ass
4. \( q \) 1, 3, \( \Rightarrow \)-E
5. \( q \land \neg q \) 2, 4, \( \land \)-I
6. \( \neg p \) 3, 5, \( \neg \)-I

Example 8:
\[ p \Rightarrow q \vdash \neg (p \land \neg q) \]

1. \( p \Rightarrow q \) Given
2. \( p \land \neg q \) Ass
3. \( p \) 2, \( \land \)-E
4. \( \neg q \) 2, \( \land \)-E
5. \( q \) 1, 3, \( \Rightarrow \)-E
6. \( q \land \neg q \) 4, 5, \( \land \)-I
7. \( \neg (p \land \neg q) \) 6, \( \neg \)-I

Example 9:
Jim will party all night and pass AI? That must be wrong. If he works hard he won’t have time to party. If he doesn’t work hard he’s not going to pass AI.

Let:

\[ p \] Jim will party all night
\[ q \] Jim will pass AI
\[ r \] Jim works hard

Formalisation of argument:
\[ r \Rightarrow \neg p, \neg r \Rightarrow \neg q \vdash \neg (p \land q) \]

1. \( r \Rightarrow \neg p \) Given
2. \( \neg r \Rightarrow \neg q \) Given
3. \( p \land q \) Ass
4. \( r \) Ass
5. \( \neg p \) 1, 4, \( \Rightarrow \)-E
6. \( p \) 3, \( \land \)-I
7. \( p \land \neg p \) 5, 6, \( \land \)-I
8. \( \neg r \) 4, 7, \( \neg \)-I
9. \( \neg q \) 2, 9, \( \Rightarrow \)-E
10. \( q \) 3, \( \land \)-E
11. \( q \land \neg q \) 9, 10, \( \land \)-I
12. \( \neg (p \land q) \) 3, 11, \( \neg \)-I
Summary

- This lecture started to look at logic from the standpoint of artificial intelligence.
- The main use of logic from this perspective is as a means of knowledge representation.
- We introduced the basics of propositional logic, and talked about some of the properties of sentences in logic.
- We also looked at a formal definition of syntax and semantics, and the semantic approach to inference.
- We also looked at proof theory, and gave examples of a number of different kinds of proof.
- Next lecture we will go on to look at predicate logic.