Proof Systems

Lecture 3

Logic and Programming Languages
Proof system

• A proof system is a formalized system for proving things.

• Most systems have several components
  1. A set of Axioms. Things that are known to be true without any work
  2. A set if inference rules for deriving larger true statements from smaller true statements
  3. A set of assumptions from which to work

  1. In a mechanized logic, a proof is a data structure that can be checked by a machine
Consistency, completeness, normal forms

• Consistency
  – A system is consistent if falsehood is not provable (from the empty set of assumptions)
  – A system is complete if every theorem is provable from the inference rules of the logic
  – A Normal Form exists if there exists a unique smallest proof for every theorem, and other proofs of the same theorem “reduce” to this proof.
Natural Deduction

- A style of proof with several elements that have become widely used
  1. Introduction rules
  2. Elimination rules
  3. Hypothetical judgements
    1. Reasoning from assumptions

1. Proofs are represented by a tree of “true statements” rooted at the bottom.
Proof trees

• A proof tree has several parts
  1. A statement of what is proven (the root). Drawn below the line
  2. A set of sub trees that represent proofs of the required components. Drawn above the line
  3. A name for the inference rule used. Draw to the left of the line.
  4. A set of premises. Drawn in brackets

\[
\begin{align*}
[p_0 \lor \ p_1] \\
\text{-------------} & \ \lor e_2 \\
& \text{p_1} \ \ [p_2] \\
\text{-------------} & \ \lor i \\
& \text{p_1} \lor \text{p_2} \\
\text{-------------} & \ \sim i \\
\sim (p_1 \lor p_2)
\end{align*}
\]
Introduction rules

• For each connective of the logic, there is an introduction rule, where the root (below the line) has that connective has its outermost form.

\[
\begin{align*}
\frac{A \quad B}{A \land B} & (\land I) \\
\frac{A}{\neg A} & (\neg I) \\
\frac{A}{A \lor B} & (\lor I_1)
\end{align*}
\]
Elimination rules

• For each connective there is a rule that tells how to “consume” a formula with that connective to prove something else. Here the formula with that connective is above the line.

\[
\begin{align*}
\frac{\mathbf{A} \land \mathbf{B}}{\mathbf{A}} & \quad (\land \mathbf{E}_1) \\
\frac{\neg \mathbf{A}}{\mathbf{A}} & \quad (\neg \mathbf{E}) \\
\frac{\mathbf{A} \quad \mathbf{A} \rightarrow \mathbf{B}}{\mathbf{B}} & \quad (\rightarrow \mathbf{E})
\end{align*}
\]
Hypothetical Judgements

• Somethings can be proven using a sort of conditional reasoning.
• We need a way to “temporarily” assume a new condition, and then cut of this assumption when we are done.
  – Assume some formula are true
  – Infer other things follow from these assumptions
    • These are consequences of the assumptions
Natural deduction by the rules

- We will look at each connective, and then study both the introduction and elimination rules for it.
And

\[
\begin{align*}
\frac{A \land B}{A} (\land I) \\
\frac{A \land B}{B} (\land E_2)
\end{align*}
\]
Or

\[
\begin{align*}
\frac{A}{A \lor B} & \quad (\lor I_1) \\
\frac{B}{A \lor B} & \quad (\lor I_2) \\
\frac{A \lor B \quad A \rightarrow C \quad B \rightarrow C}{C} & \quad (\lor E)
\end{align*}
\]
Not

\[
\frac{A \rightarrow \bot}{\neg A} \quad \frac{A \neg A}{\bot} \quad \frac{\bot}{A} \quad \frac{A \neg A}{\neg I} \quad \frac{A}{\neg I} \quad \frac{\neg A}{\neg E} \quad \frac{A}{\neg E}
\]
Implies

\[ A \rightarrow B \]

\[ B \]

\[ (\rightarrow E) \]

\[ \wedge I \]

\[ A \rightarrow B \]

\[ B \]

\[ A \]

\[ \therefore \]

\[ A 

\[ A \rightarrow B \]
Semantics

• The statement below the line is a consequence of the premises, and if it is in a box, the assumption of the box.
• Natural deduction works by maintaining this invariant
• Every step keeps the invariant true
Natural Deduction as a mechanized proof system.

data NatDed n
  = Premise (Prop n)
    | AndI (NatDed n) (NatDed n)
    | AndE1 (NatDed n)
    | AndE2 (NatDed n)
    | Neg2I (NatDed n)
    | Neg2E (NatDed n)
    | ImplyI (Prop n) (NatDed n)
    | ImplyE (NatDed n) (NatDed n)
    | OrI1 (NatDed n) (Prop n)
    | OrI2 (Prop n) (NatDed n)
    | OrE (NatDed n) (NatDed n) (NatDed n)
    | AbsurdE (NatDed n) (Prop n)
    | AbsurdI (NatDed n) (NatDed n)
    | NegI (Prop n) (NatDed n)
Using NatDed

- Building a term of type NatDed is a tree-like structure.
- This tree might be a proof tree. If it maintains the invariant.
- A computer program can “check” if that is the case.
Constructing proof trees

• Constructing proof trees is a lot like programming.
• You are given some premises. These are input to the checker.
• You must build a NatDed data structure that relies only on the given premises.
• Building this tree is a lot like programming. You must build it out if the constructors of NatDed in such a way that the checker will succeed.
Representing the Premises as Data

- data Sequent n = Seq [Prop n] (NatDed n)
Difficulties

• One must think to build a proof tree that will check.
• What pieces do you have?
  – What do they prove?
• What other pieces can you make?
• How can you put them together.
• Sometimes working bottom up helps.
• Mechanized help is useful.
Strategy

- Construct a term.
- Name it.
- Let the system check and print it.
- Does it prove what you expect?
- Did the check complain?
- Make some more terms
- Put them together.
Gentzen style Proofs

• In a Gentzen style proof, we build a tree of hypothetical judgments, instead of a tree of true statements.

• Here the set of assumptions (hypotheses, premises) is an explicit part of the proof.

• $a |- a \land T$
Gentzen approach

• Here we manipulate both the term to the right of the turnstile ( |- ) and the premises to the left of the turnstile.

• This approach is called the sequent calculus
The sequent calculus

• The rules are broken into 4 cases.
• Some of the cases (the last 2) are broken into left and right variants
• The cases
  – Axiom
  – Cut
  – Logical rules
  – Structural rules
Axiom and Cut

Axiom:

\[ A \vdash A \quad (I) \]

Cut:

\[ \Gamma \vdash \Delta, A \quad A, \Sigma \vdash \Pi \]
\[ \Gamma, \Sigma \vdash \Delta, \Pi \quad (Cut) \]
### Logical Rules

<table>
<thead>
<tr>
<th>Left logical rules:</th>
<th>Right logical rules:</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ \Gamma, A \vdash \Delta ] [ \Gamma, A \land B \vdash \Delta ] \hspace{1cm} (\land L_1) ]</td>
<td>[ \Gamma \vdash A, \Delta ] [ \Gamma \vdash A \lor B, \Delta ] \hspace{1cm} (\lor R_1) ]</td>
</tr>
<tr>
<td>[ \Gamma, B \vdash \Delta ] [ \Gamma, A \land B \vdash \Delta ] \hspace{1cm} (\land L_2) ]</td>
<td>[ \Gamma \vdash B, \Delta ] [ \Gamma \vdash A \lor B, \Delta ] \hspace{1cm} (\lor R_2) ]</td>
</tr>
<tr>
<td>[ \Gamma, A \vdash \Delta ] [ \Sigma, B \vdash \Pi ] [ \Gamma, \Sigma, A \lor B \vdash \Delta, \Pi ] \hspace{1cm} (\lor L) ]</td>
<td>[ \Gamma \vdash A, \Delta ] [ \Sigma \vdash B, \Pi ] [ \Gamma, \Sigma \vdash A \land B, \Delta, \Pi ] \hspace{1cm} (\land R) ]</td>
</tr>
<tr>
<td>[ \Gamma \vdash A, \Delta ] [ \Sigma, B \vdash \Pi ] [ \Gamma, \Sigma, A \rightarrow B \vdash \Delta, \Pi ] \hspace{1cm} (\rightarrow L) ]</td>
<td>[ \Gamma, A \vdash B, \Delta ] [ \Gamma \vdash A \rightarrow B, \Delta ] \hspace{1cm} (\rightarrow R) ]</td>
</tr>
<tr>
<td>[ \Gamma \vdash A, \Delta ] [ \Gamma, \neg A \vdash \Delta ] \hspace{1cm} (\neg L) ]</td>
<td>[ \Gamma \vdash A, \Delta ] [ \Gamma \vdash \neg A, \Delta ] \hspace{1cm} (\neg R) ]</td>
</tr>
</tbody>
</table>
### Structural Rules

**Left structural rules:**

\[
\frac{\Gamma \vdash \Delta}{\Gamma, A \vdash \Delta} \quad (WL)
\]

\[
\frac{\Gamma, A, A \vdash \Delta}{\Gamma, A \vdash \Delta} \quad (CL)
\]

\[
\frac{\Gamma_1, A, B, \Gamma_2 \vdash \Delta}{\Gamma_1, B, A, \Gamma_2 \vdash \Delta} \quad (PL)
\]

**Right structural rules:**

\[
\frac{\Gamma \vdash \Delta}{\Gamma \vdash A, \Delta} \quad (WR)
\]

\[
\frac{\Gamma \vdash A, A, \Delta}{\Gamma \vdash A, \Delta} \quad (CR)
\]

\[
\frac{\Gamma \vdash \Delta_1, A, B, \Delta_2}{\Gamma \vdash \Delta_1, B, A, \Delta_2} \quad (PR)
\]
Intuition

• Logical rules introduce new formula either on the left or the right. They maintain a logical invariant just like the Natural Deduction rules.
  – What is the invariant?

• Structural rules manipulate the formula regardless of the shape or connective that the formula have.
Intuition 2

• Think of the rules as instructions for constructing a proof.
• Some of the instructions are ambiguous. There may be many ways to follow them.
• Next time we will study automated methods for finding a proof.