Recall from last time

**Conclusion**
LF, the dependently typed logical framework
One corner of the $\lambda$-cube.
No imprepdicativity, no induction principles thus adequate emcondings possible.
Canonical forms inductively defined.
All implemented in the Twelf system.

**Homework**
Complete one case of the adequacy theorem proof for negE in one direction, and negE $D_1 \ D_2 \uparrow$ true $B$ in the other.
Example

Judgments: \( N \) even, \( N \) odd, \( N + M = K \)

Evidence:

\[
\begin{align*}
\text{ev0} \quad & z \text{ even} \\
N \text{ odd} \quad & s \quad N \text{ even} \\
\text{evN} \quad & s \quad N \text{ odd} \\
N \text{ even} \quad & o\text{dN} \\
\text{pl0} \quad & 0 + M = M \\
N + M = K \quad & (s \quad N) + M = s \quad K \\
\text{plN} \quad &
\end{align*}
\]
Example in Twelf

Let’s look at even.elf
Theorem: If $O_1 :: N \text{ odd and } O_2 :: M \text{ odd then }$

$\mathcal{P} :: N + M = K \text{ and } \mathcal{E} :: K \text{ even.}$

Proof: by induction on $O_1$.

Case: $O_1 = \begin{array}{l}
\text{ev0} \\
\text{odN}
\end{array}$

- $z \text{ even}$

$\begin{array}{l}
\mathcal{P}_1 :: z + M = M \\
\mathcal{P}_2 :: (s\ z) + M = (s\ M)
\end{array}$

$\mathcal{E} :: (s\ M) \text{ even}$

by pl0

by plN on $\mathcal{P}_1$

by evN $O_2$
Theorem

\[ \mathcal{O}_1 \]:: \( N \) odd
\[ s \, N \] even
\[ \mathcal{O}_1 \]:: \( \mathcal{O}_1 = \mathcal{O}_1' \) odd by \( \text{odN} \) on \( \mathcal{O}_2 \)

Case: \( \mathcal{O}_1 \) =
\[ s \, (s \, N) \] odd
\[ s \, N \] even
\[ s \, (s \, N) \] odd
\[ \mathcal{P}_1 \]:: \( N + M = K \) and \( \mathcal{E}_1 \):: \( K \) even by \( \text{ind. hyp. on} \ \mathcal{O}_1', \ \mathcal{O}_2 \)
\[ \mathcal{P}_2 \]:: \( (s \, N) + M = (s \, K) \) by \( \text{plN} \) on \( \mathcal{P}_1 \)
\[ \mathcal{P}_3 \]:: \( (s \, (s \, N)) + M = (s \, (s \, K)) \) by \( \text{plN} \) on \( \mathcal{P}_2 \)
\[ \mathcal{E}_1 \]:: \( (s \, K) \) odd by \( \text{odN} \) on \( \mathcal{E}_1 \)
\[ \mathcal{E}_1 \]:: \( (s \, (s \, K)) \) even by \( \text{evN} \) on \( \mathcal{O}_3 \)
Elaboration forms

Case analysis and recursion

Termination,
Coverage,
Hypothetical judgments.

\[
\text{thm} : \quad \forall M : \text{nat} \quad \forall N : \text{nat} \quad \forall O_1 : \text{odd} \quad M \quad \forall O_2 : \text{odd} \quad N \\
\quad \exists K : \text{nat} \quad \exists P : \text{plus} \quad M \quad N \quad K \quad \exists E : \text{even} \quad K
\]

\[
\text{fun thm (odN ev0)} \quad O_2 = (\text{plN pl0}) \quad (\text{evN} \quad O_2) \\
\mid \text{thm (odN (evN} \quad O_1)) \quad O_2 = \\
\quad \text{let} \\
\quad \quad (P, E) = \text{thm} \quad O_1 \quad O_2 \\
\quad \text{in} \\
\quad \quad (\text{plN} \quad (\text{plN} \quad P), \text{evN} \quad (\text{odN} \quad E)) \\
\text{end}
\]
LF functions vs. meta-functions

LF functions

\[ \text{thm} : \ \Pi M : \text{nat}. \ \Pi N : \text{nat}. \ \Pi O_1 : \text{odd} \ M. \ \Pi O_2 : \text{odd} \ N. \ \Pi K : \text{nat}. \ \Pi P : \text{plus} \ M \ N \ K. \ \Pi E : \text{even} \ K. \]

Meta-functions

\[ \text{thm} : \ \forall M : \text{nat}. \forall N : \text{nat}. \forall O_1 : \text{odd} \ M. \forall O_2 : \text{odd} \ N. \ \exists K : \text{nat}. \exists P : \text{plus} \ M \ N \ K. \ \exists E : \text{even} \ K. \]

Assessment

- \( \Sigma \) not part of LF (otherwise typing not unique)
- No cases in LF (otherwise no adequacy)
Relational encoding

Judgment:

\[ \mathcal{O}_1 \quad \mathcal{O}_2 \quad \mathcal{P} \quad \mathcal{E} \]

\[ \text{thm (} N \text{ odd ) (} M \text{ odd ) = (} N + M = K \text{ ) (} K \text{ even )} \]

Philosophical We use exactly the same technology as before

Problem Meta theoretical justification that this is a proof lies outside the formal system.

In other words How do we know that we define the evidence for this judgment correctly?
Relational encoding

Evidence: (for the base case)

\[
\begin{align*}
\text{thm} \left( \frac{z \text{ even}}{s \ z \ \text{odd}} \right) (\ M \text{ odd } ) &= \left( \frac{z + M = M}{(s \ z) + M = (s \ M)} \right) (\ \frac{M \text{ odd}}{((s \ M) \text{ even})} )
\end{align*}
\]
Evidence: (for the inductive case)

\[
\begin{align*}
\text{thm} \left( N \text{ odd} \right) & \left( M \text{ odd} \right) = \left( N + M = K \right) \left( K \text{ even} \right) \\
\text{thm} \left( \frac{(s \ N)}{(s \ (s \ N)) \text{ odd}} \right) & \left( M \text{ odd} \right) = \left( \frac{(s \ N) + M = (s \ K)}{(s \ (s \ N)) + M = (s \ (s \ K))} \right) \left( \frac{(s \ K) \text{ odd}}{(s \ (s \ K)) \text{ even}} \right)
\end{align*}
\]

Carsten Schürmann
Logical- and Meta-Logical Frameworks Lecture 3
Example in Twelf

Representation of judgment

\text{thm} : \text{odd } N \rightarrow \text{odd } M \rightarrow \text{plus } N \ M \ K \rightarrow \text{even } K \rightarrow \text{type.}

Representation of base case

\text{b} : \text{thm } (\text{odN } \text{ev0}) \ 02 \ (\text{plN } \text{pl0}) \ (\text{evN } 02).

Representation of inductive case

\text{i} : \text{thm } 01 \ 02 \ P \ E

\rightarrow \text{thm } (\text{odN } (\text{evN } 01)) \ 02 \ (\text{plN } (\text{plN } P)) \ (\text{evN } (\text{odN } E)).
Use ideas of judgment and evidence to express meta theorems.

Recall example: Find evidence $\mathcal{D} :: A \supset \neg \neg A$ true.

Proving a meta theorem: Find evidence

$$
\mathcal{D} :: \text{thm}\ ( N \text{ odd } )\ ( M \text{ odd } ) = ( N + M = K ) ( K \text{ even } )
$$

Thus search for derivations is important.
Overview search techniques

Recall from Lecture 1:

**Bottom-Up** (backward-chaining) Consider rules that *match* the conclusion.

**Top-Down** (forward-chaining) Consider rules that *matches* premisses

**Mixed** A little bottom-up, a little top-down.

**Remark** The search techniques are independent from the logic. They depend on how to *match judgments*.

**Remark** In LF: Given $\Gamma$, given $A$, find $M$, s.t. $\Gamma \vdash M \uparrow A$.

**Technique** Logic Programming: The sublanguage is called Elf.
Elf’s search semantics

Propositions

\[ P ::= a \ M_1 \ldots \ M_n \]

Goal formulas

\[ G ::= P \mid \Pi x : A. \ G \mid D \rightarrow G \]

- \( x : A \) are \textit{universally} quantified parameters
- \( D \) are dynamic extension of the signature
- Note relation to \( \lambda \)Prolog

Definite clauses

\[ D ::= P \mid \Pi x : A. \ D \mid G \rightarrow D \]

- \( x : A \) are \textit{existentially} quantified parameters
- \( D \) are subgoals
- Note relation to \( \lambda \)Prolog
Elf’s search semantics (cont’d)

Logic Variables Notation: $\hat{X}, \hat{D}, \hat{E}$

Unification $\Gamma \vdash M = N : A$ and $\Gamma \vdash A = B : K$
  ▶ Make $M$ and $N$ equal.
  ▶ Higher-order unification undecidable.
  ▶ Pattern unification + constraints.

Goal search $\Gamma \vdash G \Rightarrow M$
  ▶ Construct $M : G$ from $G$.

Immediate entailment $\Gamma \vdash D \gg G \Rightarrow M$
  ▶ Construct $M : G$ from $G$, by focusing on $D$. 
Elf’s search semantics (cont’d)

How to search for evidence.

\[
\begin{align*}
\Gamma, x : D \in \Gamma, \Sigma & \quad \Gamma \vdash x : D \ggg M : G \\
\hline
\Gamma & \vdash M : P \\
\Gamma \vdash P = Q : \text{type} & \quad \Gamma \vdash M = N : P \\
\hline
\Gamma & \vdash M : P \ggg N : Q \\
\end{align*}
\]

\[
\begin{align*}
\Gamma, x : A & \vdash M : G \\
\hline
\Gamma & \vdash \lambda x : A. M : \Pi x : AG \\
\Gamma, D & \vdash M : G \\
\hline
\Gamma & \vdash \lambda x : D. M : D \to G \\
\end{align*}
\]

\[
\begin{align*}
\Gamma & \vdash M \hat{X} : [\hat{X}/x]G \ggg N : Q \\
\Gamma & \vdash M : \Pi x : A. G \ggg N : Q \\
\Gamma & \vdash D \ggg M : Q \\
\Gamma & \vdash N : G \\
\hline
\Gamma & \vdash G \to D \ggg MN : Q \\
\end{align*}
\]
Example Find evidence \( D \) of \( A \supset \neg \neg A \) true.

Example Find evidence that since 5 is odd and 7 is odd, 12 is even, using our logic program.

Problem How to control the non-determinism?

Twelf Let’s let it run in Twelf. (see even-el.elf)
Elf’s search semantics (cont’d)

- Existential variables.
- Back-tracking.
- Embedded implications.
  + Works with higher-order encodings.
  + Same syntax as LF signatures.
  - No user control on search.
  * No extra logical constants.
When is a Elf program a proof?

If it is a realizer (total function).
Curry-Howard correspondance.

Difficult:
- There are so many logical programs.
- Instantiation of logic variables is not local.
- The hypotheses.

Solution:
1. Mode correctness
2. [World correctness]
3. Termination correctness
4. Coverage correctness
Mode correctness

**Definition**  
*Mode criterion* During execution, ground inputs are being mapped onto output ground outputs.  
[Rohwedder, Pfenning]

**Twelf syntax**  
\%mode thm +O₁ +O₂ -P -E.

**Algorithm**  
Traverse the constructor type.

- Show that the overall output is ground assuming the overall input and the output of or subgoals are ground.
- Show that all inputs to the subgoals are ground assuming the overall input to be ground.

**Demonstration**  
even-meta.elf.
World correctness

Definition [World criterion] During execution the local context is always regular formed. [Schürmann]

Twelf syntax \%worlds () thm +0₁ +0₂ -P -E.

Algorithm Traverse the constructor type.

- Show that each collection of negative occurrences fall within the world schema defined beforehand.

Demonstration even-meta.elf.
Termination correctness

Definition [Termination criterion] The execution will eventually terminate.

[Rohwedder, Pfenning, Pientka]

Twelf syntax \%terminates O1 (thm O1 O2 P E).

Algorithm Traverse the constructor type

- Check if the argument in each recursive call gets smaller.

Properties

- In general undecidable.
- Well-founded subterm ordering.
- Lexicographic and simultaneous extensions.

Demonstration even-meta.elf.
Meta Theory

**Definition:** [Coverage criterion] The execution will always make progress.

[Schürmann, Pfenning]

**Twelf syntax**

- \(\%\text{covers } (\text{thm } +01 +02 -P -E)\). Input coverage.
- \(\%\text{total } 01 (\text{thm } 01 02 P E)\). Includes output coverage.

**Algorithms**

- Traverse relevant parts of the signature
  - Compute set of coverage candidates.
  - Try to cover
  - Interpret failure
  - Refine set of coverage candidate

**Properties**

- In general undecidable. [Coquand]
- Algorithms always terminates.
- Open for 10 years.
Conclusion

Twelf is meta logical framework.
Logic Programming Semantics give raise to new function arrow.
Totality = Modes + World + Termination + Coverage.

Homework

Prove that if $n$ is odd and $m$ is even, then $n + m$ is odd.
Extra: Implement the double negation theorem.