15-212-ML

Lecture 1: Types and Evaluation

Frank Pfenning

Aug 25, 1998

1. Administrative Issues
2. Course Goals and Outline
3. Types and Evaluation

Teaching Staff

- John Lafferty (in 2 weeks)
- Frank Pfenning (me!)
- Sec F, 12:30-1:20, Mark Plesko
- Sec G, 1:30-2:20, Philip Wickline
- Sec H, 2:30-3:20, Doug Fearing
- Sec I, 3:30-4:20, Adam Megacz
- `http://www.cs.cmu.edu/~fp/15-212-ML/`
- **No recitation this week**

Textbooks

- Lawrence C. Paulson, ML for the Working Programmer, 2nd ed, CUP, 1996.
- Robert Harper, Programming in Standard ML, on-line, 1998.

Course Requirements

- Attend lectures and recitations
- Do (and hand in!) homework assignments (50%)
- Midterm (20%), in class, closed book
- Final (30%), 3 hour, open book and notes

Assignments

- 6 assignments (first next week, others 2 weeks each)
- Handed out in recitation
- Written and programming parts
- Programs due Wednesday at **Noon**
  (Please correct syllabus)
- A program is like an essay!
- No joint assignments

Course Goals

- 211: Data structures and algorithms
- 212: Advanced programming techniques
  - Decomposing problems
  - Combining solutions
  - Reason about program correctness

Concepts

- Induction and recursion, loop invariants
- Data abstraction
  - data types
  - representation invariants
- Control abstraction
  - higher-order functions
  - exceptions and continuations
- Types and modularity
- Mutable data structures, objects
- Streams, demand-driven programming

Further Topics

- Symbolic computation
- Game search
- Grammars and parsing
- Type-checking and evaluation
- Computability
- Concurrency

The ML Language

- Carrier for concepts
- Supports (in the language):
  - recursion
  - user-declared data types
  - concrete and abstract types
  - higher-order functions
  - exceptions and continuations
- Advanced module system

Characteristics of ML

- Statically typed
- "*Well-typed programs cannot go wrong*"
- Mathematically defined via *evaluation* of *expressions* to *values*
- Much later and infrequent: *effects*
- Computation with symbolic values via *pattern matching*

Defining ML (Effect-Free Fragment)

- Types t
- Expressions e
- Values v (subset of expressions)
- $e : t$ *e has type t*
- $e \stackrel{1}{\Longrightarrow} e'$ *e reduces to e′ in one step*
- $e \Longrightarrow e'$ *e reduces to e′ in 0 or more steps*
- $e \Longrightarrow v$ *e evaluates to v*

Types `int`

Values $\ldots, \texttt{\~{}1}, \texttt{0}, \texttt{1}, \ldots,$

that is, $\overline{n}$ for every integer n.

Expressions e_1 `+` e_2, e_1 `-` e_2, e_1 `*` e_2,

e_1 `div` e_2, e_1 `mod` e_2, *etc.*

Typing Rules

$\overline{n}$: `int`

e_1 `+` e_2 : `int`

if e_1 : `int` and e_2 : `int`

similar for other operations.

Evaluation Rules

$$e_1 \;\mathsf{+}\; e_2 \stackrel{1}{\Longrightarrow} e_1' \;\mathsf{+}\; e_2 \quad \text{if } e_1 \stackrel{1}{\Longrightarrow} e_1'$$

$$\overline{n_1} \;\mathsf{+}\; e_2 \stackrel{1}{\Longrightarrow} \overline{n_1} \;\mathsf{+}\; e_2' \quad \text{if } e_2 \stackrel{1}{\Longrightarrow} e_2'$$

$$\overline{n_1} \;\mathsf{+}\; \overline{n_2} \stackrel{1}{\Longrightarrow} \overline{n_1 + n_2}$$

Types `bool`

Values `true` and `false`

Expressions `if` e_1 `then` e_2 `else` e_3.

Typing Rules

`if` e_1 `then` e_2 `else` e_3 : t

if e_1 : `bool`

and $e_2 : t$

and $e_3 : t$

Evaluation Rules

$$\texttt{if } e_1 \texttt{ then } e_2 \texttt{ else } e_3 \stackrel{1}{\Longrightarrow} \texttt{if } e_1' \texttt{ then } e_2 \texttt{ else } e_3$$

$$\text{if } e_1 \stackrel{1}{\Longrightarrow} e_1'$$

$$\texttt{if true then } e_2 \texttt{ else } e_3 \stackrel{1}{\Longrightarrow} e_2$$

$$\texttt{if false then } e_2 \texttt{ else } e_3 \stackrel{1}{\Longrightarrow} e_3$$

Types t_1 `*` t_2 for any type t_1 and t_2.

Values `(`v_1`,` v_2`)` for values v_1 and v_2.

Expressions `(`e_1`,` e_2`)`, `#1` e, `#2` e

Typing Rules

$\texttt{(}e_1\texttt{,}\ e_2\texttt{)} : t_1\ \texttt{*}\ t_2$

if $e_1 : t_1$

and $e_2 : t_2$

$\texttt{\#1}\ \ e : t_1$

if $e : t_1 \texttt{*} t_2$ for some t_2.

$\texttt{\#2}\ \ e : t_2$

if $e : t_1 \texttt{*} t_2$ for some t_1.

Evaluation Rules

$$
\begin{array}{rcll}
(e_1\texttt{,}\ e_2) & \stackrel{1}{\Longrightarrow} & (e_1'\texttt{,}\ e_2) & \text{if } e_1 \stackrel{1}{\Longrightarrow} e_1' \\
(v_1\texttt{,}\ e_2) & \stackrel{1}{\Longrightarrow} & (v_1\texttt{,}\ e_2') & \text{if } e_2 \stackrel{1}{\Longrightarrow} e_2' \\
\texttt{\#1}\ e & \stackrel{1}{\Longrightarrow} & \texttt{\#1}\ e' & \text{if } e \stackrel{1}{\Longrightarrow} e' \\
\texttt{\#1}\ (v_1\texttt{,}\ v_2) & \stackrel{1}{\Longrightarrow} & v_1 & \\
\texttt{\#2}\ e & \stackrel{1}{\Longrightarrow} & \texttt{\#2}\ e' & \text{if } e \stackrel{1}{\Longrightarrow} e' \\
\texttt{\#2}\ (v_1\texttt{,}\ v_2) & \stackrel{1}{\Longrightarrow} & v_2 &
\end{array}
$$