Advanced topics & contemporary research in
functional programming languages
and
type systems
Part 1: program execution and transformations
Xavier Leroy; 5 lectures

- Operational semantics, environments, closures.
- Compilation to abstract machines.
- Program transformations: continuation-passing style, monads, closure conversion, defunctionalization.
- Functional intermediate representations in optimizing compilers.

Leitmotif: semantic preservation & how to prove it.
Part 2: type systems
Didier Rémy; 8 lectures

- Parametric polymorphism (System F and ML).
- Type soundness: subject reduction and progress theorems.
- Type inference via constraint solving.
- Recursion; imperative features; recursive types.
- Algebraic data types. Existential types.
- Logical relations and representation independence.

Leitmotif: type safety & how to prove it.
Part 3: towards program proof
Yann Régis-Gianas; 6 lectures

- Generalized Algebraic Data Types (GADTs)
- Dependently-typed programming.
- Contracts and hybrid type checking.
- Higher-order Hoare logic for functional programs.

Leitmotif: richer types guarantee stronger properties of programs.
Part 4: mechanization using a proof assistant (Coq)

Xavier Leroy; 1 lecture

- Coq mechanization of the theory of the simply-typed $\lambda$-calculus.

- Towards mechanization of richer type systems: how to handle bound variables and $\alpha$-conversion.
Prerequisites

A taste for programming!

Knowledge of a functional programming language (pref. OCaml or Haskell)

Notions of operational semantics and $\lambda$-calculus.
Evaluation

- Mid-term exam. (1/3)
- Final exam. (1/3)
- Programming project. (1/3)

Note: this course is not “breakable” in half.
Related MPRI courses

2.2 Models of programming languages
2.3.1 Concurrency
2.6 Abstract interpretation
2.7.1 + 2.7.2 Proof systems and proof assistants
2.35.1 Constraint programming
2.36.1 Proofs of programs
## Time Table

<table>
<thead>
<tr>
<th>Date Range</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sep 18, 25; Oct 02, 09, 16</td>
<td>Program execution &amp; transformations</td>
</tr>
<tr>
<td>Oct 23, 30; Nov 06, 13, 20</td>
<td>Type systems</td>
</tr>
<tr>
<td>Nov 27 or Dec 04</td>
<td><em>mid-term exam</em></td>
</tr>
<tr>
<td>Dec 11, 18; Jan 08</td>
<td>Type systems</td>
</tr>
<tr>
<td>Jan 15, 22, 29; Feb 05, 12, 19</td>
<td>Towards proved programs</td>
</tr>
<tr>
<td>Feb 26</td>
<td>A taste of mechanization</td>
</tr>
<tr>
<td>Mar 04 or 11</td>
<td><em>final exam</em></td>
</tr>
<tr>
<td>mid Dec</td>
<td><em>programming project announced</em></td>
</tr>
<tr>
<td>late Feb</td>
<td><em>programming project deadline</em></td>
</tr>
</tbody>
</table>
http://gallium.inria.fr/~xleroy/mpri/2-4/

- Part 1: slides + exercises + OCaml examples
- Part 2: course notes (incl. exercises) + slides
- Part 3: slides (incl. exercises)
- Part 4: slides + Coq examples

Plus: further reading; examples of exams from past years.