Foundations of Computer Science
Lecture 12: Recapping and Real World Use!

Anil Madhavapeddy
1st Nov 2023
Goals of Programming

- to describe a computation so that it can be done *mechanically*:
  - expressions compute values
  - commands cause effects
- to do so *efficiently and correctly*, giving right answers *quickly*
- to allow *easy modification* as our needs change
  - through an orderly *structure* based on *abstraction* principles
  - programmer should be able to predict effects of changes
Why Program in OCaml?

• It is interactive.

• It has a flexible notion of data type.

• It hides the underlying hardware: no crashes.

• Programs can easily be understood mathematically.

• It distinguishes naming from updating memory.

• It manages storage in memory for us.
Language

Static type checking

Parametric Polymorphism

Type Inference

Algebraic Data Types

Pattern Matching

First Class Functions

Abstraction
# let x = "1" + 1 ;;

Error: This expression has type string but an expression was expected of type int
1A Object Oriented Programming

Prof Rob Harle

```
# let x = "1" + 1 ;;
Error: This expression has type string but an expression was expected of type int
```
# type 'a tree =
| Lf
| Br of 'a * 'a tree * 'a tree


```ocaml
# let fn l = List.map (fun (a,b) -> string_of_int a ^ b) l;;

val fn : (int * string) list -> string list = <fun>
```
1B Concepts in Programming Languages

1B Further Java

II Types
## Language

- Static type checking
- Parametric Polymorphism
- Type Inference
- Algebraic Data Types
- Pattern Matching
- First Class Functions

```ocaml
# type vehicle =
|   | Car of bool
|   | Motorbike of int
|   | Bicycle
```
# type vehicle =
   | Car of bool
   | Motorbike of int
   | Bicycle

# match v with
   | Car false -> "car"
   | Car true  -> "reliant robin"
   ...

1B Semantics of Programming Languages

# type vehicle =
  | Car of bool
  | Motorbike of int
  | Bicycle

# match v with
  | Car false -> "car"
  | Car true  -> "reliant robin"
  ...

Language
- Static type checking
- Parametric Polymorphism
- Type Inference
- Algebraic Data Types
- Pattern Matching
- First Class Functions
Upcoming Courses:

1A Operating Systems
1B Compiler Construction
1B Programming in C/C++
OCaml: a system

**Runtime**
- Fast Foreign Functions
- Static Linking
- Garbage Collection
- Fast Native Code
- Multiarchitecture
- Portable Bytecode

**Language**
- Pattern Matching
- Algebraic Data Types
- Type Inference
- First Class Functions
- Static type checking
- Parametric Polymorphism
OCaml (& ML): Influences

- **Runtime**
  - Fast Foreign Functions
  - Static Linking
  - Garbage Collection
  - Fast Native Code
  - Multiarchitecture
  - Portable Bytecode

- **Language**
  - Pattern Matching
  - Algebraic Data Types
  - Type Inference
  - First Class Functions
  - Static type checking
  - Parametric Polymorphism

- **Influenced**
  - JavaScript
  - Common Lisp
  - F#
  - C#
  - C++
  - Rust
  - Elm
  - Java
  - Scala
  - Python
  - OCaml (ML): Influences

Influences

OCaml: scaling

Single Core

Execution sequential. Why bother with immutable values?
OCaml: scaling

Values are shared in memory and so can be seen by all cores
OCaml: scaling
1B Concurrent & Distributed Systems

Single Core

Multiple Core

Values are shared in memory and so can be seen by all cores

Must copy variables between machines!

Multiple Machines
OCaml: Web Programming

**Runtime**
- Fast Foreign Functions
- Static Linking
- Garbage Collection
- Fast Native Code
- Multiarchitecture
- Portable Bytecode

**Language**
- Pattern Matching
- Algebraic Data Types
- Type Inference
- First Class Functions
- Static type checking
- Parametric Polymorphism

**Flexibility**
- JavaScript
- Wasm

https://rescript-lang.org

ReScript is a robustly typed language that compiles to efficient and human-readable JavaScript. It comes with a lightning fast compiler toolchain that scales to any codebase size.
OCaml: Building Hardware

HardCaml is a structural hardware design DSL embedded in OCaml. The library can be used for front end design tasks up to the synthesis stage where a VHDL or Verilog netlist is generated. Libraries for fast simulation using LLVM, waveform viewing and co-simulation with Icarus Verilog are provided.

HardCaml-RiscV is a simple pipelined RV32I core, targetted towards a FPGA implementation and built with HardCaml.
Jane Street

Been using OCaml for twenty years or so, with ~30 million lines of code.

~2000 employees, many of whom code in OCaml, with ~600 fulltime developers.

Much of the core source code is available as open source code: realworldocaml.org

And a really fun podcast at: https://signalsandthreads.com/
OCaml: Operating Systems

Runtime
Language
Flexibility

A programming framework for building type-safe, modular systems

MirageOS is a library operating system that constructs unikernels for secure, high-performance network applications across a variety of cloud computing and mobile platforms. Code can be developed on a normal OS such as Linux or MacOS X, and then compiled into a fully-standalone specialised unikernel that runs under a Xen or KVM.

Recent Updates all
- MirageOS running on the ESP32 embedded chip (26 Jan 2018)
- MirageOS Winter 2017 hack retreat roundup (23 Dec 2017)

Fast Native Code
First Class Functions
Parametric Polymorphism

Unix
Mobile
Unikernels
Containers

https://mirage.io
Docker
The most popular way to share and extend software distributions.

13m+ developers use Docker for Desktop daily.
7m+ applications developed.
13 billion monthly image downloads.

At the heart of desktop integration on Windows and Mac, there are services written in OCaml that process every byte of traffic.

https://github.com/moby/vpnkit

Find out more in Part II Cloud Computing!
OCaml: Safety Critical

- Runtime
- Language
- Garbage Collection
- Fast Native Code
- Portable Bytecode
- Fast Foreign Functions
- Static Linking
- Multiarchitecture
- Pattern Matching
- Algebraic Data Types
- Type Inference
- Static type checking
- Parametric Polymorphism
- Static Linking
- Static Analysis

FLOW IS A STATIC TYPE CHECKER FOR JAVASCRIPT.

The Coq Proof Assistant

https://coq.inria.fr
Creating safe robots with Imandra

Kostya Kanishev
Jul 9, 2018 · 3 min read

From self-driving cars to medical surgeons, robots have become ubiquitous. Ensuring they operate safely and correctly is evermore important. The most popular middleware for robotics is the open-sourced Robot OS. We have begun work on developing an Imandra interface to Robot OS, opening up the world of robotics to the latest advancements in automated reasoning. In this post, we showcase our early results, discuss our roadmap and our submission for a talk at the upcoming ROSCon 2018 (Madrid, Spain).
OCaml: Data Science

Runtime
- Fast Foreign Functions
- Static Linking
- Garbage Collection
- Fast Native Code
- Multiarchitecture
- Portable Bytecode

Language
- Pattern Matching
- Algebraic Data Types
- Type Inference
- First Class Functions
- Static type checking
- Parametric Polymorphism

Flexibility

OCaml Scientific Computing

ocaml.xyz
Goals of Programming

• to **describe a computation** so that it can be done **mechanically**:
  
  • expressions compute values
  
  • commands cause effects

• to do so **efficiently and correctly**, giving right answers **quickly**

• to allow **easy modification** as our needs change
  
  • through an orderly **structure** based on **abstraction** principles
  
  • programmer should be able to predict effects of changes