Compiler Design
Functional Programming Languages

Ondřej Janošík (xjanos12@stud.fit.vutbr.cz)
Marek Kidoň (xkidon00@stud.fit.vutbr.cz)

Abstract

This lecture focuses on describing basic features of functional programming languages like high-order functions, function polymorphism and lazy-evaluation and how these features can be implemented in compiler.

In runtime functional programs have to be able to work with functions as first-class citizens. They can be unevaluated, partially applied or passed as parameters to other functions. First topic is about basic purely functional languages in general, their properties and common data structures.

Due to polymorphic function properties we need to know function types at compile time. Although it is possible to specify function type signature explicitly, we mostly want to let the compiler infer type signatures instead. In second part we discuss how the compiler uses type inference to deduce function types and how to select appropriate implementation in case of polymorphic functions.