Compiler Construction

Caml

Abstract

Caml is a dialect of ML programming language developed at INRIA and ENS. The language is statically typed and strictly evaluated. Its syntax is very similar to Haskell, although Caml doesn't belong to purely functional programming languages. It offers functional pattern matching as well as some imperative features such as modifiable variables, arrays, and several varieties of loops. A lot of information about data types are automatically inferred by the compiler. Error recovery tools such as exception handling are also included. Caml provides basic data types as well as more sophisticated types (tuples, arrays, lists, sets, hash tables, queues, stacks, etc.). New complex types can be easily defined (e.g. records or enumerations). Caml contains a polymorphic type system, meaning that some undetermined types can be represented by variables that might be later instantiated at will. Thanks to its functional origin, functions can be passed as arguments to other functions and can be returned as a result of a function. As language derived from ML, garbage collector takes care of memory management. The allocation and deallocation of data structures is performed implicitly. More known modification of Caml is its object-oriented variant called Ocaml.

Martin Žouželka (xzouze00)

Jiří Zajíc (xzajic10)