Programming Languages

Second Phase Report

José António Ribeiro da Silva Lopes
ist1103938

Behavior of product and union types

Product and union types are just ways of aggregating other types inside the same structure and associating them with labels. In the case of product types, which are called `structs` in this implementation, the type has "access" to all of it's inner fields, that are accessed via the dot (`.`) operator. As for union types, the concrete instance can only hold one of the possible labels. For a more thorough use of union types, the `match` construct is the ideal solution. It allows for the destructuring of the type's labels, while being exhaustive in this same labels.

Sub-Typing and type checking recursive types

Sub-typing was implemented as a default method, part of the `ASTType` interface. Only the types with special sub-typing rules, like `ASTTStruct`, `ASTTArrow`, etc. overrode the default method. For all the concrete implementations of sub-typing, the first thing done was check if the other type was an instance of `ASTTId`. If so, that type would get unrolled and then passed again to the sub-type method. With this sort of "lazy unrolling", recursive type checking is also achieved, because there will come a point where the type being compared is not a sub-type of the recursive type at all, or where it already is, or where future unrolls might result in this relation being observed.