Ants for Dinner

Programming an ants strategy

Laurens van den Brink, Philipp Hausmann, Marco Vassena

30 October 2013

Approach

Design and implement a DSL, which allows to define a strategy in a natural way.

DSL

Our custom language provides imperative features:

- Blocks and statements
- For Loop
- If - Then - Else
- Scoped bindings (variables and local functions)
- Try - Catch
- Top level declarations
- Procedures
- Mutual tail recursion
- Modules

Example

```
import module_a
dir = Left
rec main = {
    try {
        Move;     Turn dir;     PickUp;     main;
    } catch {
        g;
    }
}
rec g = {
    if Home Here && Friend Ahead then {
        let s = { Drop; }
        in {
            s; g;
        }
    } else {
        for x in [Left, Right] {
            Turn x;
        }
        main;
    }
}
```

Parser

- Matches the input file with the grammar of the language
- Constructs the abstract syntax tree (AST).
- Loads and parses recursively any imported module.
- Uses Parsec library.

Compiler

- Compiles the syntax tree into the assembly code
- Inline bindings
- Handles function calls and recursion
- Reports errors

Strategy

Essential strategy:

1. Random walk
2. Pick up food
3. Go back home
4. Drop food

Further Work

- Duplicated code elimination
- More syntactic sugar (while, else-if, switch statement)
- Relax recursion constraints (allow parameters)
- Extend variables