The Egel Language

M.C.A. (Marco) Devillers

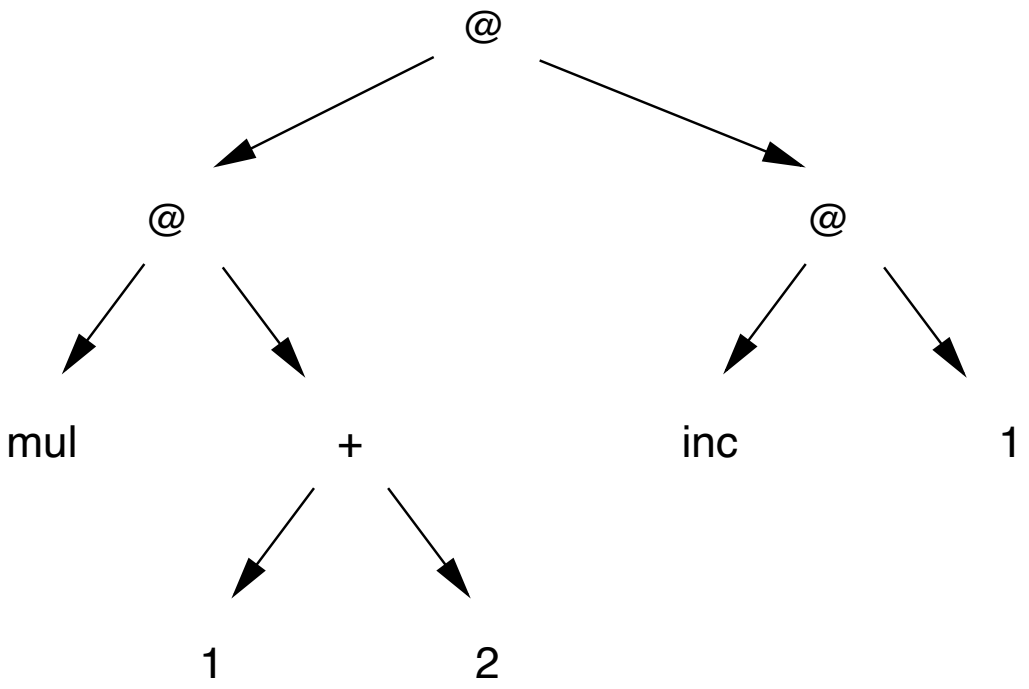
The Egel Language

- untyped concurrent functional scripting language
- based on eager combinator rewriting
- concise but remarkably powerful syntax
- aimed at taking over the world

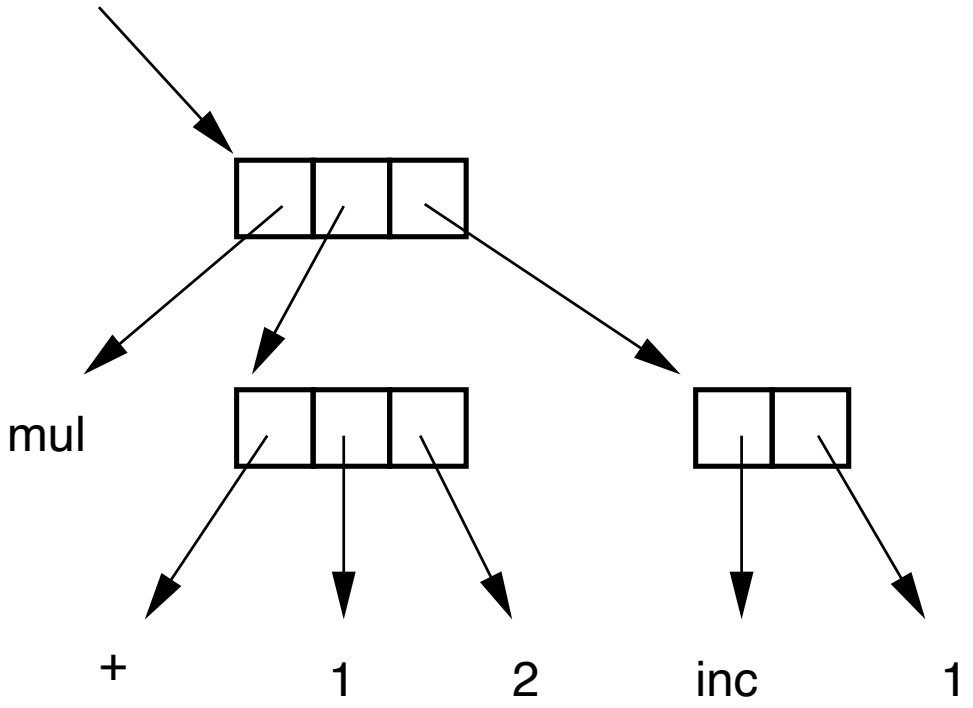
The operational model

- any term forms a directed acyclic graph (DAG), or tree for short
- *no* abstract machine, nothing, nada
- KISS (keep it simple since -you- are stupid) technology

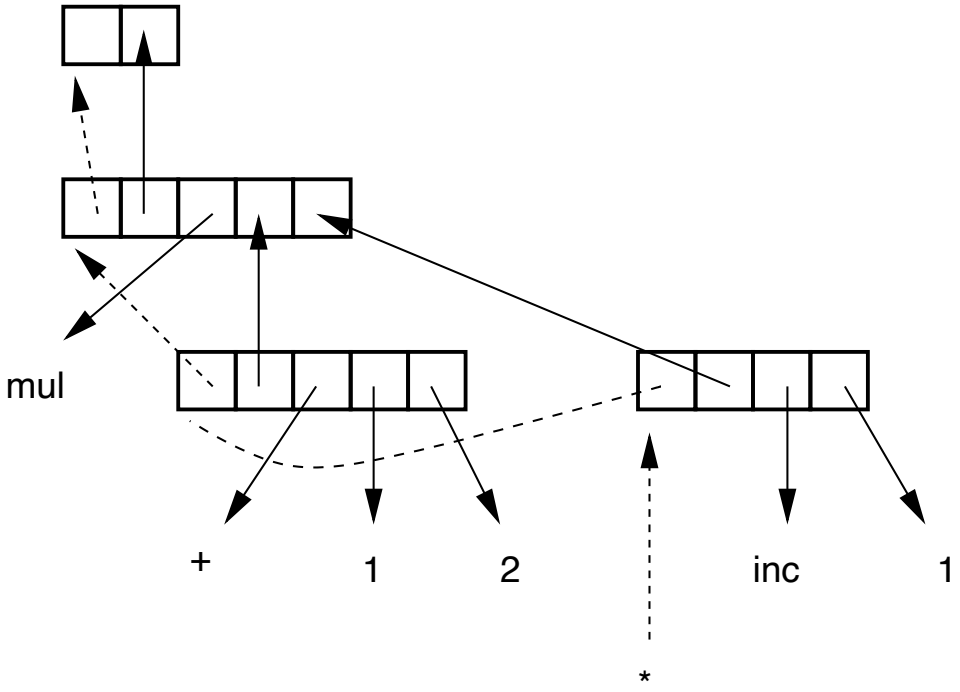
From expression to tree



Computer representation

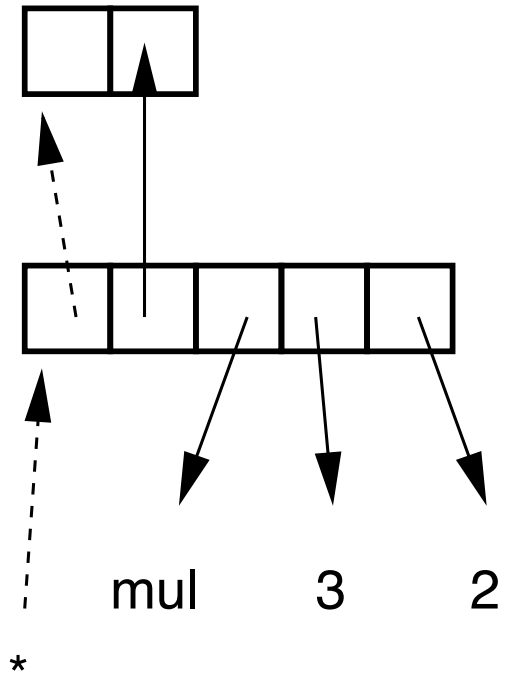


Twisted representation

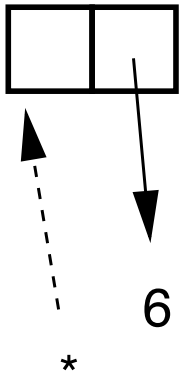


- It's still a directed acyclic graph (DAG)!

Evaluation (1)



Evaluation (2)



Goal of Egel

Fully exploit this computational model

- translate to automatically garbage collected C++ objects
- extend to support exceptions
- allow for 'cheap' concurrency
- allow migrating combinators

Front-end language

```
# A parallel fibonnaci implementation.

import "prelude.eg"

namespace Fibonnaci (
    using System

    def fib =
        [ 0 -> 0
        | 1 -> 1
        | N -> fib (N- 2) + fib (N- 1) ]

    def pfib =
        [ 0 -> 0
        | 1 -> 1
        | X -> [ (F0, F1) -> F0 + F1 ]
            (par [Y -> pfib (X - 1) ] [Z-> pfib (X - 2)]) ]

)

using System

def main = Fibonnaci::pfib (3+2)
```

Front-end language

- reminiscent of a lambda calculus with constants
- scripts are collections of terms
- untyped to fully explore the model
- modules and namespaces
- focus on utility, not guarantees; terms, not types

Interpreter

```
egel [-|--interact] [-I path|--include path] [file]

egel [-I path|--include path] -e command

egel [-h|--help|-v|--version]
```

Support batch, interactive, and read-eval-print (REPL) modes.

0.1 (beta) version

- stand-alone interpreter for Linux/BSD/MacOS systems
- translates terms to C++ objects running bytecode
- capable of automating simple tasks
- tardy but robust
- relatively light-weight (1 MiB) but needs Unicode (libicu)
- can be extended with C++ written modules
- can be integrated into C++ applications
- now working on serialization, transport, and node protocol

Aimed at taking over the world

- see a data center as just a large computer
- start any calculation anywhere
- https://egel-lang.github.io/