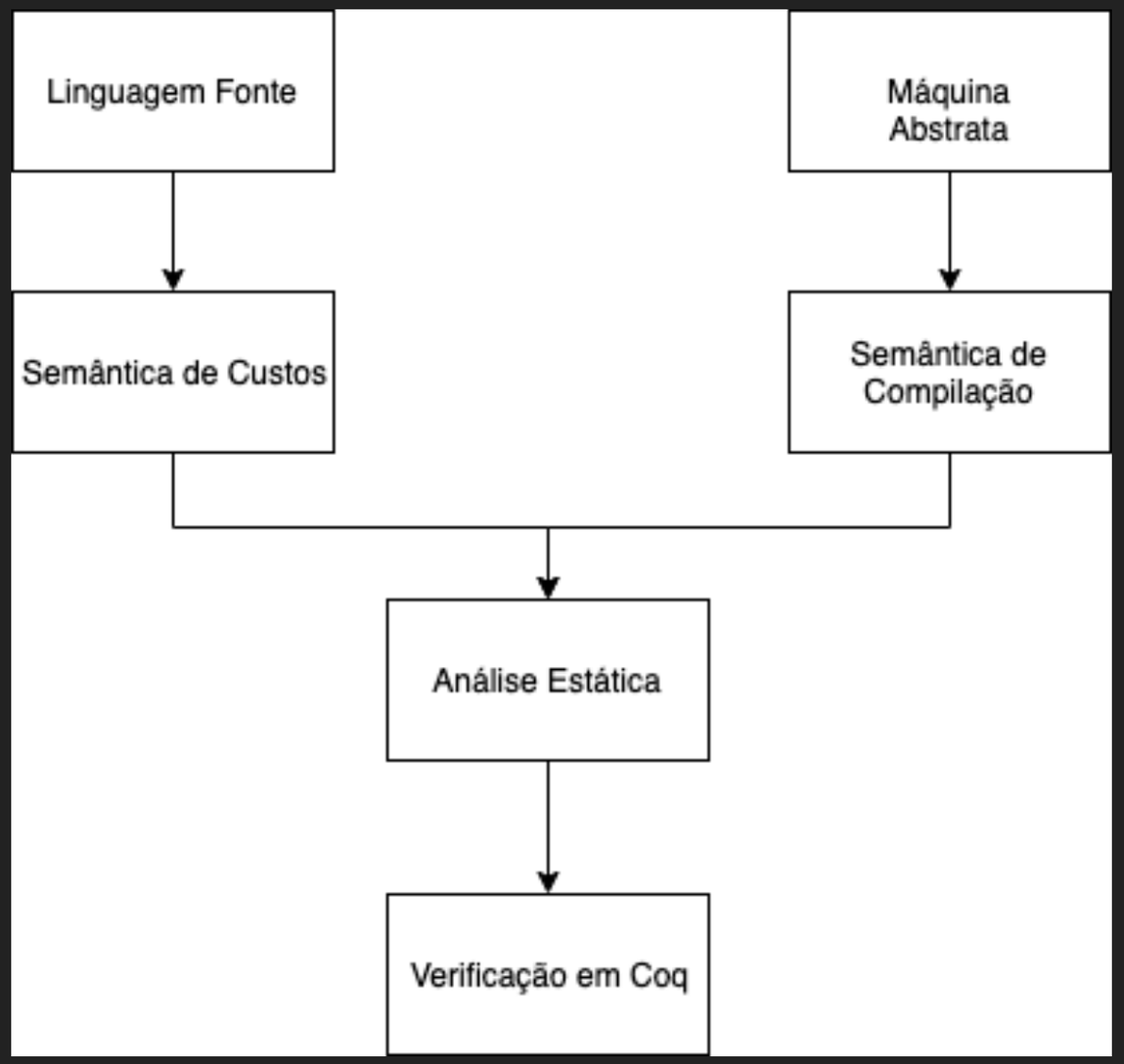
ANÁLISE ESTÁTICA DE PROGRAMAS COM Coq

BRUNO DE FREITAS BONATTO

ÁLVARO MOREIRA, RODRIGO MACHADO

OBJETIVOS DO TRABALHO

- Criação de uma ferramenta para a análise estática de programas L1
- Verificação formal dessa análise usando o Coq



RAML

```
1  type nat = Zero | Succ of nat
2
3
4  let rec add n1 n2 = Raml.tick(1.0);
5    match n1 with
6       | Zero -> n2
7       | Succ n -> Succ (add n n2)
8
9
10 let rec mult n1 n2 =
11   match n1 with
12      | Zero -> Zero
13      | Succ n ->
14        add n (mult n n2)
15
16 let rec fact n =
17   match n with
18      | Zero -> Succ Zero
19      | Succ Zero -> Succ Zero
20      | Succ x -> mult n (fact x)
```

RAML

```
Resource Aware ML, Version 1.4.1, July 2018

Typechecking module ...
  Typecheck successful.
  Stack-based typecheck successful.

Analyzing function add ...

  Trying degree: 3

== add :

  [nat; nat] -> nat

  Non-zero annotations of the argument:
     10.00  <--  ([Succ(*)], [])
      5.00  <--  ([], [])

  Non-zero annotations of result:

  Simplified bound:
     5.00 + 10.00*M
   where
     M is the number of Succ-nodes of the 1st component of the argument
```

RAML

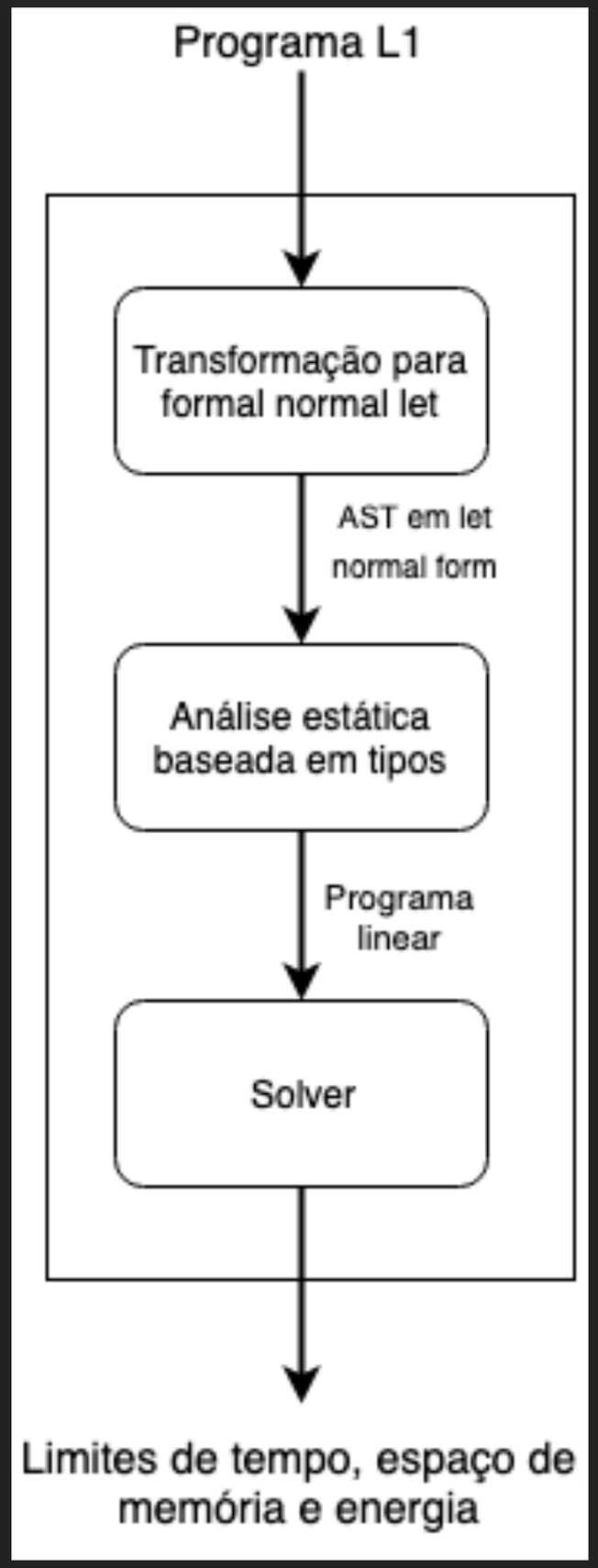
```
Analyzing function mult ...

  Trying degree: 3

== mult :

  [nat; 'a] -> nat

Non-zero annotations of the argument:
   10.00  <--  ([Succ(*); Succ(*)], *)
   14.00  <--  ([Succ(*)], *)
    3.00  <--  ([], *)

Non-zero annotations of result:

Simplified bound:
   3.00 + 9.00*M + 5.00*M^2
 where
   M is the number of Succ-nodes of the 1st component of the argument
```

RAML

```
Analyzing function fact ...

  Trying degree: 3

== fact :

  nat -> nat

  Non-zero annotations of the argument:
   10.00  <--  [Succ(*); Succ(*); Succ(*)]
   24.00  <--  [Succ(*); Succ(*)]
   27.00  <--  [Succ(*)]
    5.00  <--  []

Non-zero annotations of result:

Simplified bound:
   5.00 + 18.33*M + 7.00*M^2 + 1.67*M^3
 where
   M is the number of Succ-nodes of the argument
```

A FERRAMENTA

- Em desenvolvimento
- Escrita em Haskell
- Transforma programa fonte em uma representação intermediária
- Efetua a análise parametrizada pela máquina abstrata escolhida
- A análise gera um programa linear cuja solução é o limite superior da medida em questão



VERIFICAÇÃO USANDO Coq

- Em desenvolvimento
- Implementamos a L1 e seu sistema de tipos em Coq e provamos preservação e progresso
- Vamos implementar a SSM1 e a sua semântica de compilação
- A propriedade que vamos verificar ainda não está definida