Projet Minilucy

Paul Patault & Émilien Lemaire

December 14, 2022

ENS Paris-Saclay

Features réalisés

- `when`
- `merge`
- `reset`
- `automates` (en surface uniquement)

Traduction de l'automate

```
node syracuse (i: int) returns (o: int);
let
  automaton
  | Even ->
      o = i -> pre o / 2;
      until (o mod 2 = 1) continue Odd
  | Odd ->
      o = i -> pre o * 3 + 1;
      until (o mod 2 = 0) continue Even
  end
tel
```

Traduction de l'automate

```
type t = Even | Odd

node syracuse(i : int) returns (o : int);
    var state : t; cond__4 : bool; cond__3 : bool;
let
  cond__4 = o mod 2 = 0;
  cond__3 = o mod 2 = 1;
  state = Even ->
           pre (merge state
                  (Even -> if cond__3 then Odd else Even)
                  (Odd -> if cond__4 then Even else Odd));
  o = merge state
        (Even -> i -> pre o / 2)
        (Odd  -> i -> pre o * 3 + 1);
tel
```

Automate (une slide)

```
node syracuse (i: int)
  returns (o: int);
let
  automaton
  | Even ->
      o = i -> pre o / 2;
      until (o mod 2 = 1)
      continue Odd
  | Odd ->
      o = i -> pre o * 3 + 1;
      until (o mod 2 = 0)
      continue Even
  end
tel
```

```
type t = Even | Odd

node syracuse(i : int)
  returns (o : int);
  var state : t;
      cond__4, cond__3 : bool;
let
  cond__4 = o mod 2 = 0;
  cond__3 = o mod 2 = 1;
  state = Even ->
            pre (merge state
                  (Even ->
                    if cond__3 then Odd else Even)
                  (Odd ->
                    if cond__4 then Even else Odd));
  o = merge state
        (Even -> i -> pre o / 2)
        (Odd  -> i -> pre o * 3 + 1);
tel
```

Schéma de compilation