Challenge

Spare Time Teaching

July 29, 2015

Problem

Let's play with some coinduction. Implement, in Coq, two functions `filter_out` n xst and `filter_in` n xst which both take a natural number (not zero) and a stream, one returns the stream after filter out every $(n+1)$th element, and the other returns the stream if we filter in every $(n+1)$th element. Notice they are each others opposites.

```
Require Import List.

CoInductive stream_ℕ : Type :=
| Cons : ℕ → stream_ℕ → stream_ℕ.

Fixpoint prefix n xst :=
  match n with
    | 0 ⇒ nil
    | S n' ⇒
      match xst with
        | Cons x xst' ⇒ x :: prefix n' xst'
      end
  end.

CoFixpoint ℕs n := Cons n (ℕs (S n)).

(* Filter out *)

(* ... *)

Compute prefix 10 (filter_out 2 (ℕs 0)).
(*
     = 0 :: 1 :: 3 :: 4 :: 6 :: 7 :: 9 :: 10 :: 12 :: 13 :: nil
     : list ℕ
 *)

(* Filter in *)
```

```
(* ... *)

Compute prefix 10 (filter_in 2 (ℕs 0)).
(*
     = 2 :: 5 :: 8 :: 11 :: 14 :: 17 :: 20 :: 23 :: 26 :: 29 :: nil
     : list ℕ
 *)
```