Proof 1

October 18, 2013

In this document we present a proof of

$$
\begin{aligned}
INDXS = \quad & \forall A : \texttt{Type}. \\
& \forall x : A. \\
& \quad \forall lst : \texttt{List}(A). \\
& \quad\quad \exists indxs : \texttt{List}(\mathbb{N}). \\
& \quad\quad\quad \forall i : \mathbb{N}. \quad i \in_{\mathbb{N}} indxs \\
& \quad\quad\quad\quad\quad\quad \iff \\
& \quad\quad\quad\quad\quad\quad i < ||lst|| \ \wedge \ x = lst[i] \in A
\end{aligned}
$$

that generates the following program:

```
let rec indexes x lst =
  match lst with
    [] -> []
  | h :: t ->
     let l = List.map (fun x -> x + 1) (indexes x t)
     in if x = h then 0 :: l else l ;;
```

We prove our goal by induction on `L`. We get two subgoals: (1) the base case, and (2) the induction case.

```
*- PRF : vr_test_indx-1 @edd.markb-al @gateway
# top
[-]
∀[A:Type]. ∀deq:EqDecider(A). ∀x:A. ∀L:A List.  (∃indxs:{ℕ List| (∀i:ℕ. ((i ∈ indxs) ⟺ (i < ||L||) ∧ (L[i] = x)))})

# BY (InductionOnList THEN Reduce 0)

# 1

[-]
1. [A] : Type
2. deq : EqDecider(A)@i
3. x : A@i
4. L : A List
[-]
⊢ ∃indxs:{ℕ List| (∀i:ℕ. ((i ∈ indxs) ⟺ (i < 0) ∧ (bottom() = x)))}

# BY

# 2

[+]A,deq,x,L,u,v,,
[-]
⊢ ∃indxs:{ℕ List| (∀i:ℕ. ((i ∈ indxs) ⟺ (i < (||v|| + 1)) ∧ ([u / v][i] = x)))}

# BY
```

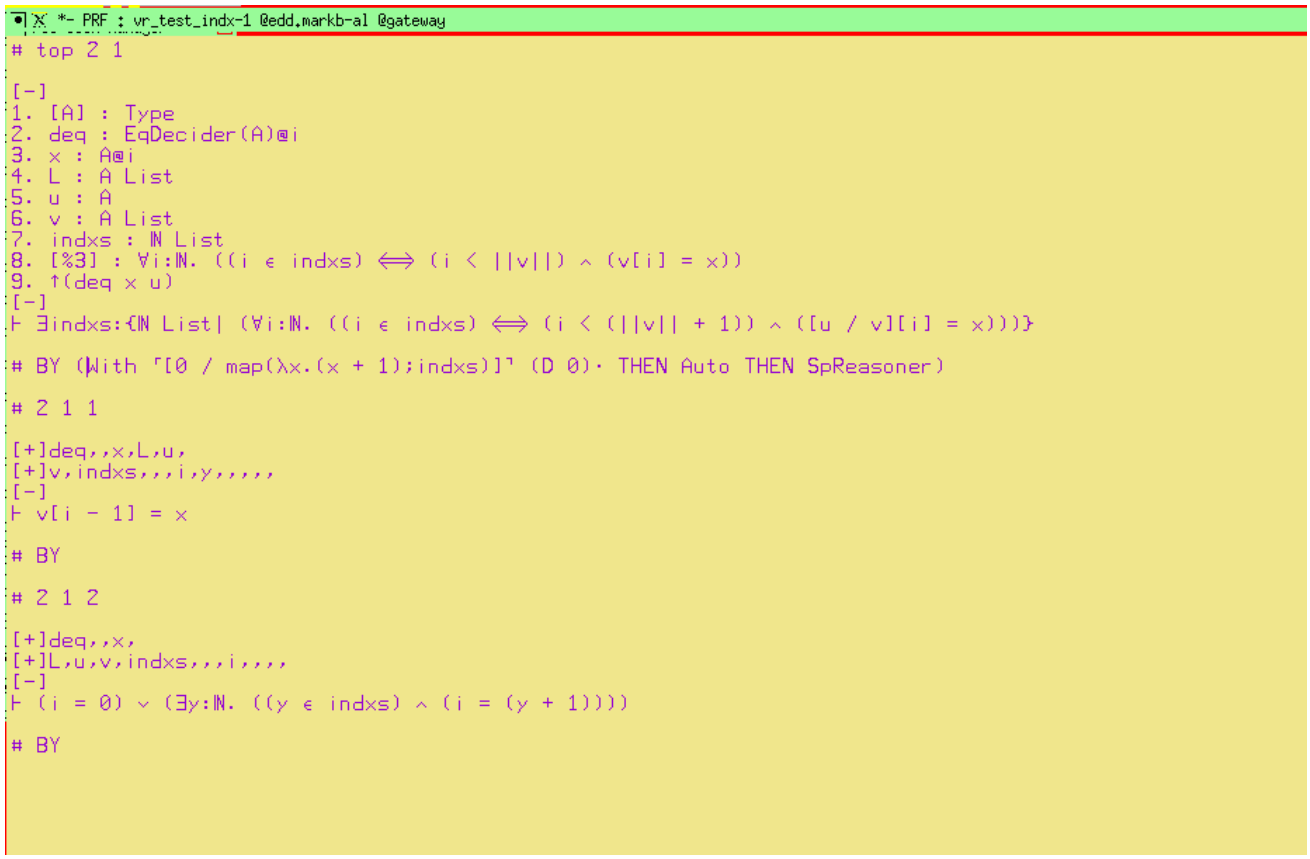
The base case is trivial, the list of indexes is the empty list `[]`:

```
*- PRF : vr_test_indx-1 @edd.markb-al @gateway
* top 1

[-]
1. [A] : Type
2. deq : EqDecider(A)@i
3. x : A@i
4. L : A List
[-]
⊢ ∃indxs:{ℕ List| (∀i:ℕ. ((i ∈ indxs) ⟺ (i < 0) ∧ (bottom() = x)))}

* BY (With '[]' (D 0)· THEN Auto)
```

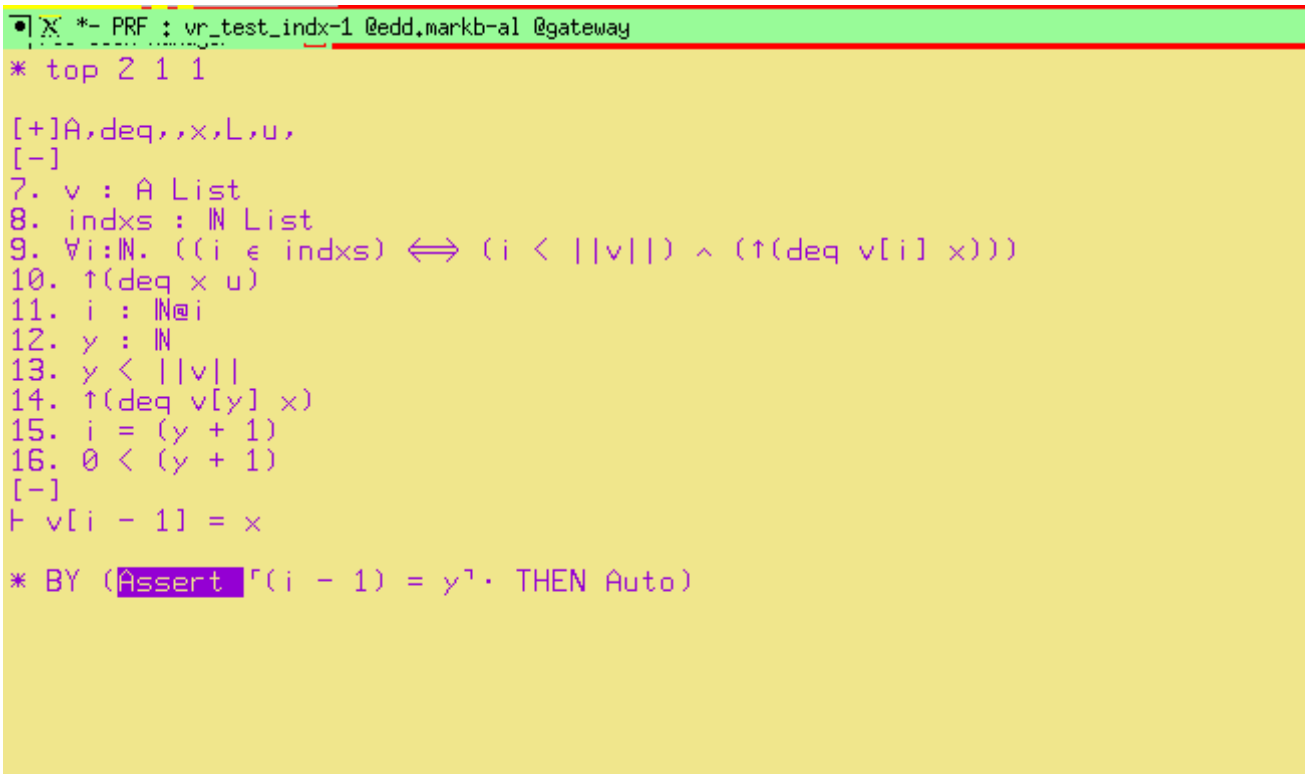
In the induction case, we first check whether or not `x` is equal to `u`, the head of our list `L`. We obtain two subgoals. In the first branch, we get to assume that `x` and `u` are equal and in the other branch we get to assume that they are different.

```
*- PRF : vr_test_indx-1 @edd.markb-al @gateway
# top 2

[-]
1. [A] : Type
2. deq : EqDecider(A)@i
3. x : A@i
4. L : A List
5. u : A
6. v : A List
7. ∃indxs:{ℕ List| (∀i:ℕ. ((i ∈ indxs) ⟺ (i < ||v||) ∧ (v[i] = x)))}
[-]
⊢ ∃indxs:{ℕ List| (∀i:ℕ. ((i ∈ indxs) ⟺ (i < (||v|| + 1)) ∧ ([u / v][i] = x)))}

# BY (D (-1) THEN (VrBoolCase 'deq x u'· THENA Auto))

# 2 1

[+]indxs,,,
[-]
⊢ ∃indxs:{ℕ List| (∀i:ℕ. ((i ∈ indxs) ⟺ (i < (||v|| + 1)) ∧ ([u / v][i] = x)))}

# BY

# 2 2

[+]indxs,,,
[-]
⊢ ∃indxs:{ℕ List| (∀i:ℕ. ((i ∈ indxs) ⟺ (i < (||v|| + 1)) ∧ ([u / v][i] = x)))}

# BY
```

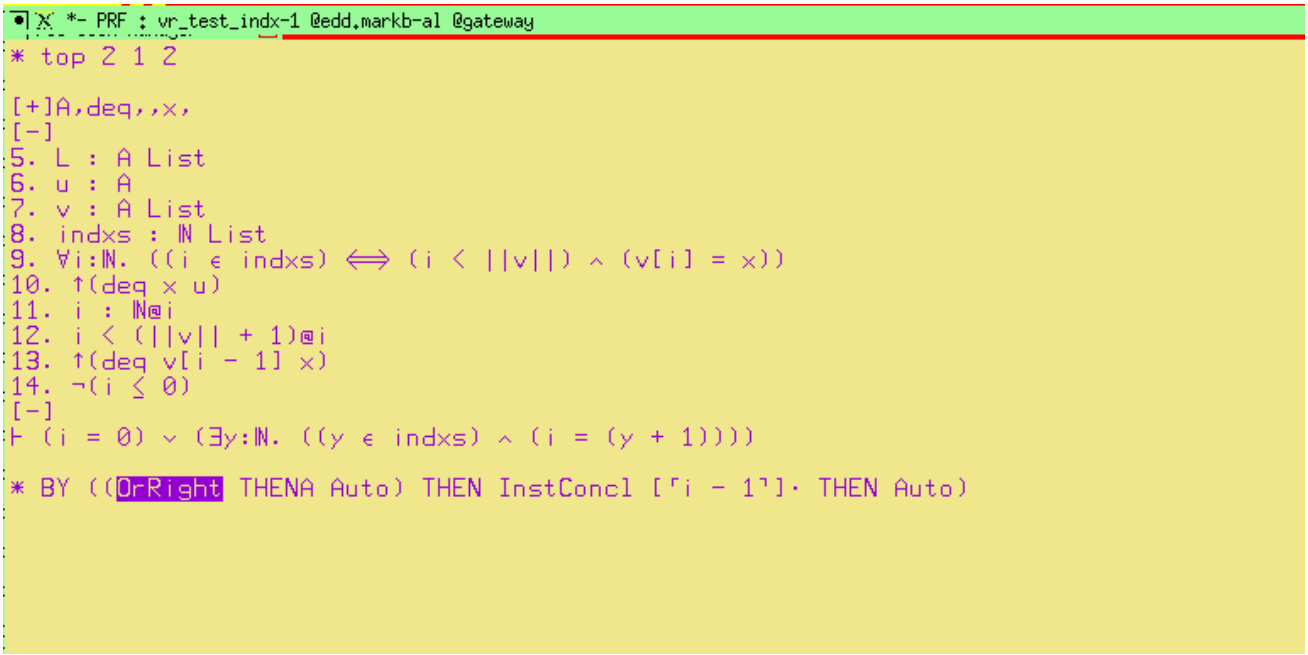
If `x` is equal to `u` then the list of indexes is `0 :: map (fun x => x + 1) indxs`, where `indxs` is the list of positions of `x` in the tail of `L`. The `SpReasoner` tactic proves several of the simple subgoals. We are left with two subgoals in this branch. Both of them are trivial.



In the first subgoal we just need to know that `i - 1` is equal to `y` to finish the proof:

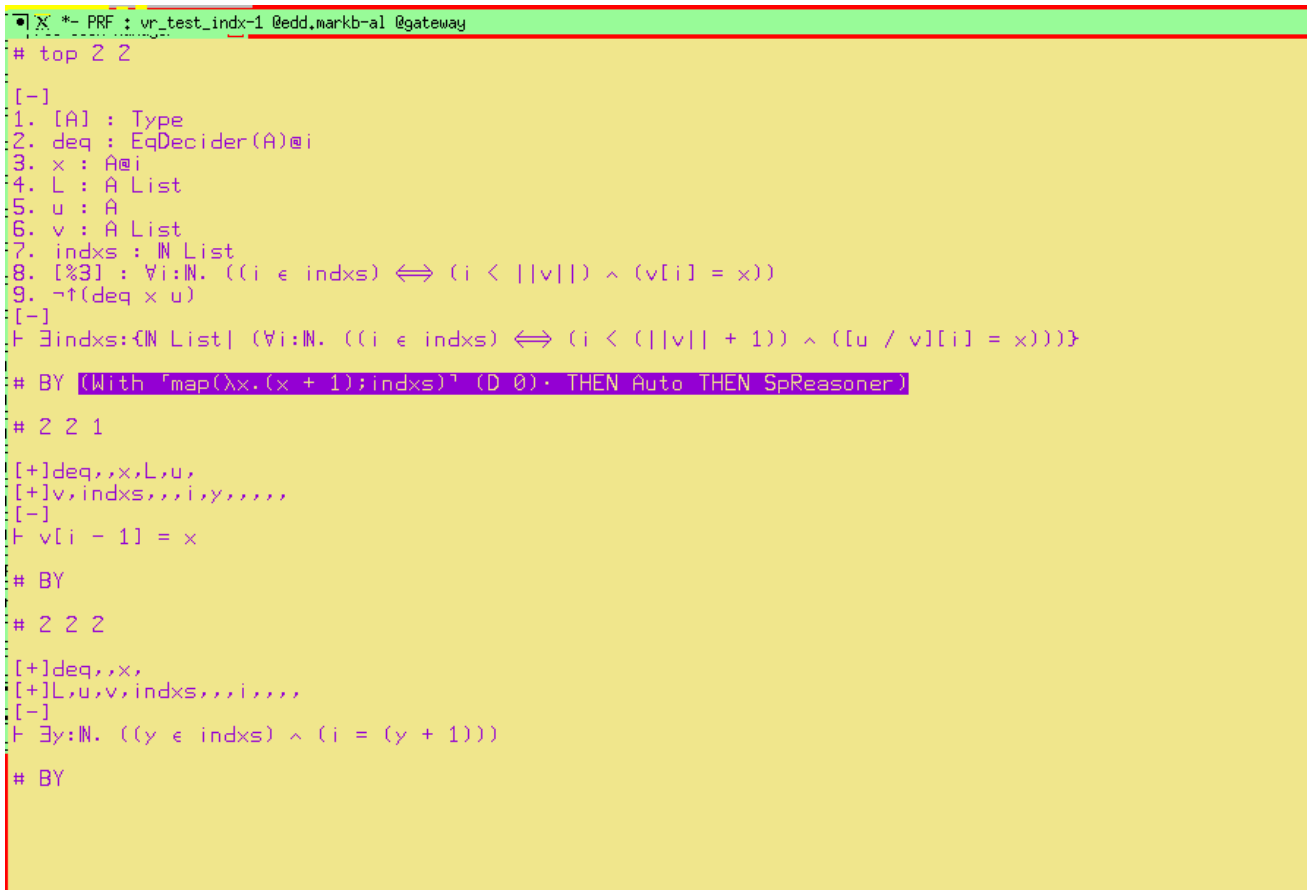


The second subgoals comes from having to prove that `i` is a member of `0 :: map (fun x => x + 1) indxs` given that `i` is less than the length of `0 :: map (fun x => x + 1) indxs` and that the `i`'s element of that list is `x`. Proving that `i` is a member of `0 :: map (fun x => x + 1) indxs` boils down to proving that either `i` is `0` or that `i - 1` is in `indxs`:



Now, if `x` is not equal to `u` then the list of indexes is `map (fun x => x + 1) indxs`, where `indxs` is the list

of positions of `x` in the tail of `L`. Again, our `SpReasoner` tactic proves several of the simple subgoals, and we are left with two trivial subgoals (see below).



```
*- PRF : vr_test_indx-1 @edd.markb-al @gateway
* top 2 2 1

[+]A,deq,,x,L,u,
[-]
7. v : A List
8. indxs : ℕ List
9. ∀i:ℕ. ((i ∈ indxs) ⟺ (i < ||v||) ∧ (↑(deq v[i] x)))
10. ¬↑(deq x u)
11. i : ℕ@i
12. y : ℕ
13. y < ||v||
14. ↑(deq v[y] x)
15. i = (y + 1)
16. 0 < (y + 1)
[-]
⊢ v[i - 1] = x

* BY (Assert ⌜(i - 1) = y⌝· THEN Auto)
```

```
*- PRF : vr_test_indx-1 @edd.markb-al @gateway
* top 2 2 2

[+]A,deq,,x,
[-]
5. L : A List
6. u : A
7. v : A List
8. indxs : ℕ List
9. ∀i:ℕ. ((i ∈ indxs) ⟺ (i < ||v||) ∧ (v[i] = x))
10. ¬↑(deq x u)
11. i : ℕ@i
12. i < (||v|| + 1)@i
13. ↑(deq v[i - 1] x)
14. ¬(i ≤ 0)
[-]
⊢ ∃y:ℕ. ((y ∈ indxs) ∧ (i = (y + 1)))

* BY (InstConcl [⌜i - 1⌝]· THEN Auto)
```