VSTTE'10 Competition Q1 in ProofPower-HOL (v2)

Rob Arthan

18th August 2010

1 Introduction

Isn't anybody in the VSTTE world interested in functional programming? To address the balance, here is an attempt based on primitive recursive functions in ProofPower-HOL executed by rewriting.

This document is a literate script containing the specs, "code" and proof commands. See the theory listing towards the end of the document for a collected summary of the definitions and results.

2 Red Tape

SML

```
│open_theory"hol";
│new_theory "vstte10−1";
│set_merge_pcs ["hol1"];
```

3 The Spec

We need some specs (albeit pretty close to code).

HOL Constant

```
│      NatSum : ℕ LIST → ℕ
├──────────────────────────────
│      NatSum [] = 0
│∧     ∀n ns• NatSum (Cons n ns) = n + NatSum ns
```

HOL Constant

```
│      NatMax : ℕ LIST → ℕ
├──────────────────────────────
│      NatMax [] = 0
│∧     ∀n ns• NatMax (Cons n ns) = if NatMax ns < n then n else NatMax ns
```

4 The Code

Now the code,implemented using the fold function from the library as any functional programmer would (cursing the designer for the order of the last two arguments):

HOL Constant

```
│        SumMax : ℕ LIST → ℕ × ℕ
├──────────────────────────────────────
│ ∀ns•
│        SumMax ns =
│        Fold (λn (s, m)• (n + s, if m < n then n else m)) ns (0, 0)
```

5 Testing, Testing, . . .

ML bindings for the defining theorems:

SML

```
│val nat_sum_def = get_spec ⌜NatSum⌝;
│val nat_max_def = get_spec ⌜NatMax⌝;
│val sum_max_def = get_spec ⌜SumMax⌝;
```

See the theory listing towards the end of the document for the results of the following tests.

First test the specs (Sum is 40, my eye!):

SML

```
│val nat_sum_test_thm = save_thm ("nat_sum_test_thm",
│        rewrite_conv [nat_sum_def]
│        ⌜NatSum [9;5;0;2;7;3;2;1;10;6]⌝);
│val nat_max_test_thm = save_thm ("nat_max_test_thm",
│        rewrite_conv [nat_max_def]
│        ⌜NatMax [9;5;0;2;7;3;2;1;10;6]⌝);
```

Now test the code:

SML

```
│val sum_max_test_thm = save_thm ("sum_max_test_thm",
│        rewrite_conv [sum_max_def, fold_def]
│        ⌜SumMax [9;5;0;2;7;3;2;1;10;6]⌝);
```

6 Correctness Proof

SML

```
│set_goal([], ⌜
│        SumMax [] = (0, 0) ∧
│∀n ns• SumMax (Cons n ns) =
│        (n + Fst (SumMax ns),
│         if Snd(SumMax ns) < n then n else Snd(SumMax ns))
│⌝);
│a(rewrite_tac [sum_max_def, fold_def]);
│val sum_max_prim_rec_thm = save_pop_thm "sum_max_prim_rec_thm";
```

SML

```
set_goal([], ⌜
∀ns•   SumMax ns = (NatSum ns, NatMax ns)
⌝);
a(strip_tac);
a(list_induction_tac ⌜ns⌝ THEN
        asm_rewrite_tac[sum_max_prim_rec_thm, nat_sum_def, nat_max_def]);
val sum_max_correct_thm = save_pop_thm "sum_max_correct_thm";
```

7 Property Proof

SML

```
set_goal([], ⌜
∀ns• NatSum ns ≤ Length ns * NatMax ns
⌝);
a(strip_tac);
a(list_induction_tac ⌜ns⌝ THEN
        asm_rewrite_tac[length_def, nat_sum_def, nat_max_def]);
a(REPEAT strip_tac THEN cases_tac⌜NatMax ns < x⌝
        THEN asm_rewrite_tac[]);
(* *** Goal "1" *** *)
a(POP_ASM_T ante_tac THEN
        rewrite_tac[rewrite_rule[≤_def]
        (pc_rule1 "lin_arith" prove_rule[]
                ⌜∀a b:ℕ • a < b ⇔ a + 1 ≤ b⌝)]
        THEN REPEAT strip_tac
        THEN all_var_elim_asm_tac1);
a(PC_T1 "lin_arith" asm_prove_tac[]);
(* *** Goal "2" *** *)
a(POP_ASM_T ante_tac THEN
        rewrite_tac[rewrite_rule[≤_def]
        (pc_rule1 "lin_arith" prove_rule[]
                ⌜∀a b:ℕ • ¬a < b ⇔ b ≤ a⌝)]
        THEN REPEAT strip_tac);
a(PC_T1 "lin_arith" asm_prove_tac[]);
val nat_sum_nat_max_estimate_thm = save_pop_thm "nat_sum_nat_max_estimate_thm";
```

8 THE THEORY vstte10-1

8.1 Parents

hol

8.2 Constants

NatSum	$\mathbb{N}\ \mathit{LIST} \to \mathbb{N}$
NatMax	$\mathbb{N}\ \mathit{LIST} \to \mathbb{N}$
SumMax	$\mathbb{N}\ \mathit{LIST} \to \mathbb{N} \times \mathbb{N}$

8.3 Definitions

NatSum

$$\begin{array}{l} \vdash \mathit{NatSum}\ [\,] = \mathit{0} \\ \quad \land\ (\forall\ n\ ns \bullet\ \mathit{NatSum}\ (\mathit{Cons}\ n\ ns) = n + \mathit{NatSum}\ ns) \end{array}$$

NatMax

$$\begin{array}{l} \vdash \mathit{NatMax}\ [\,] = \mathit{0} \\ \quad \land\ (\forall\ n\ ns \\ \quad \bullet\ \mathit{NatMax}\ (\mathit{Cons}\ n\ ns) \\ \qquad = (\mathit{if}\ \mathit{NatMax}\ ns < n\ \mathit{then}\ n\ \mathit{else}\ \mathit{NatMax}\ ns)) \end{array}$$

SumMax

$$\begin{array}{l} \vdash \forall\ ns \\ \quad \bullet\ \mathit{SumMax}\ ns \\ \qquad = \mathit{Fold} \\ \qquad\quad (\lambda\ n\ (s,\ m) \bullet\ (n + s,\ (\mathit{if}\ m < n\ \mathit{then}\ n\ \mathit{else}\ m))) \\ \qquad\quad ns \\ \qquad\quad (\mathit{0},\ \mathit{0}) \end{array}$$

8.4 Theorems

nat_sum_test_thm

$$\vdash \mathit{NatSum}\ [\mathit{9};\ \mathit{5};\ \mathit{0};\ \mathit{2};\ \mathit{7};\ \mathit{3};\ \mathit{2};\ \mathit{1};\ \mathit{10};\ \mathit{6}] = \mathit{45}$$

nat_max_test_thm

$$\vdash \mathit{NatMax}\ [\mathit{9};\ \mathit{5};\ \mathit{0};\ \mathit{2};\ \mathit{7};\ \mathit{3};\ \mathit{2};\ \mathit{1};\ \mathit{10};\ \mathit{6}] = \mathit{10}$$

sum_max_test_thm

$$\vdash \mathit{SumMax}\ [\mathit{9};\ \mathit{5};\ \mathit{0};\ \mathit{2};\ \mathit{7};\ \mathit{3};\ \mathit{2};\ \mathit{1};\ \mathit{10};\ \mathit{6}] = (\mathit{45},\ \mathit{10})$$

sum_max_prim_rec_thm

$$\begin{array}{l} \vdash \mathit{SumMax}\ [\,] = (\mathit{0},\ \mathit{0}) \\ \quad \land\ (\forall\ n\ ns \\ \quad \bullet\ \mathit{SumMax}\ (\mathit{Cons}\ n\ ns) \\ \qquad = (n + \mathit{Fst}\ (\mathit{SumMax}\ ns), \\ \qquad\quad (\mathit{if}\ \mathit{Snd}\ (\mathit{SumMax}\ ns) < n \\ \qquad\qquad \mathit{then}\ n \\ \qquad\qquad \mathit{else}\ \mathit{Snd}\ (\mathit{SumMax}\ ns)))) \end{array}$$

sum_max_correct_thm

$$\vdash \forall\ ns \bullet\ \mathit{SumMax}\ ns = (\mathit{NatSum}\ ns,\ \mathit{NatMax}\ ns)$$

nat_sum_nat_max_estimate_thm

$$\vdash \forall\ ns \bullet\ \mathit{NatSum}\ ns \leq \mathit{Length}\ ns * \mathit{NatMax}\ ns$$

9 INDEX