COMP302 Assignment 3

Lucia Eve Berger — 260473661

March 2015

1 Problem 2

In this proof, we will attempt to prove that the two below programs return the same result if applied to the same arguments.

$funpow(n, 0) = 1|pow(n, k) = n * pow(n, k - 1)$

$funpow_tl(n, 0, acc) = acc|pow_tl(n, k, acc) = pow_tl(n, k - 1, n * acc)$

Background:

The two are contrasted because we define $funpow_tl$ as taking an additional parameter, called an accumulator or acc. Another important distinction between the two is that $funpow_tl$ is tail recursive. This means that the last step of evaluation of an application of its argument. This tail recursive is some sense analogous to loops in languages such as Java.

Specification of pow:

Our specification is our generalization about the arguments that are sufficient to establish that function behaves correctly, our general case.

pow: a reasonable specification for pow is

$$ifk >= 0, n >= 1, n^k$$

Note: our assumption is that the argument k, is non-negative, and asserts the application of pow (n) to the k terminates with an expected answer.

Mathematical Induction:

Base Case: To makes sure that our above specification holds, we look at induction on the argument k. We look at both base cases k = 0, assuming it to hold for $k >= 0$, and then subsequently show that this holds for (k+1). The base case k = 0 always evaluates to 1. Note: The k=1 case evaluates to n for all integer numbers.

Assumption: We will assume pow(k) to hold for $k >= 0$. Induction: Now suppose that we look at k+1 for some $k >= 0$. By the inductive hypothesis, we have the pow evaluates to $n^(k + 1)$, that is some $k >= 0$

$$pow(n, k + 1) = n * pow(n, k - 1)$$

$$pow(n, k + 1) = n * pow^(k + 1, -1)$$

$$substitute$$

$$n^k$$

which we can evaluate, thus holding for all other k finishing our induction.

Specification of powtl :

Now lets look at the behavior of fun $\text{pow}_t l$.

$$ifn >= 0, funpow_t l(n, acc) => n^a$$

Note: a is acc

Mathematical Induction:

Our fun powtl is a tail recursive function such that: Pk : for all n, all acc, $pow_t l(n, k, acc) = acc * pow(n, k)$

Lemma: for all k, Pks is true

Base case: Our base case is the same as the specification for the pow. When acc = 1, that is $n^(acc)$, which is: for all n, for all acc, $pow_t l(n, 0, acc) = acc * 1 = acc$

Note: acc * 1 = pow(n,0)=1 from (1)

Lets look at the induction. We want to show that when we manipulate the argument acc, we still have a viable function that is equivalent to the above fun pow.

Assumption: If we assume $Pk - 1$ is true, now we want to prove true for Pk

$$pow_t l(m, k, acc)$$

using induction hypothesis, we can substitute out the acc term. We end up with the below expression.

$$pow_t l(n, k, acc) = acc * pow(n, k)$$

for the case that acc is equal to one, we can show the equivalence of pow and powtl with the same arguments. This concludes our proof.

$$acc = 1 => pow_t l(n, k, 1) = 1 * pow(n, k) = pow(n, k)$$