Foundations of Computer Science – Problem Sheet 3

Heidi Howard

October 24, 2014

This supervision is all about binary trees, functions as values, partial application and streams. Please use multiple argument functions whereever possible, instead of tuples.

This supervision will cover the material from lectures 7 to 9.

Exercise 1:

Draw and write the binary search tree in that arises from successively inserting the following pairs into the empty tree: (Alice, 6), (Tobias, 2), (Gerald, 8), (Lucy, 9). Then repeat this task using the order (Gerald, 8), (Alice, 6), (Lucy, 9), (Tobias, 2). Why are results different?

[adapted from exercise 7.1 in the course notes]

Exercise 2:

Describe an algorithm for deleting an entry from a binary search tree. Comment on the suitability of your approach. Now code this algorithm, I have provided some familiar functions and example binary search trees in the `sup3-helper.sml` file.

[adapted from exercise 7.4 and 7.5 in the course notes]

Exercise 3:

Write a function to remove the first element from a functional array. All the other elements are to have their subscripts reduced by one. The cost of this operation should be linear in the size of the array.

[adapted from exercise 7.8 in the course notes]

Exercise 4:

Code the curried function exf, which takes as arguments the function f and the list l. The result must consist of those elements x of l such that f(x) is also a member of l. The elements of the result must be distinct from each other but may appear in any order. For example, if f(x) = x + 1 and l = [9, 3, 2, 2, 8] then the result should be [2, 8] or [8, 2].

[taken from 2000 P1 Q1]

Exercise 5:

The type option, declared below, can be viewed as a type of lists having at most one element. (It is typically used as an alternative to exceptions.) Write a new function that combines both map and filter, e.g. when the function given returns None then remove the element (like filter) and when the function given returns Some x then put x into the list (like map).

```
datatype 'a option = None | Some of 'a
val mapfilter = fn: ('a -> 'b option) -> 'a list -> 'b list
```

[taken from exercise 8.4 in the course notes]

Exercise 6:

This is past exam question, try (at least at first) to complete the question within 30 mins.

(a) The polymorphic curried function delFirst takes two arguments, a predicate (boolean-valued function) p and a list xs. It returns a list identical to xs except that the first element satisfying p is omitted; if no such element exists, then it raises an exception. Code this function in ML.

(b) Use the function delFirst to express the polymorphic function delFirstElt, where delFirstElt x xs returns a list identical to xs except that it omits the first occurrence of x.

(c) Carefully explain the polymorphic types of these two functions, paying particular attention to currying and equality

(d) A list ys is a permutation of another list xs if ys is obtained by rearranging the elements of xs. For example, [2,1,2,1] is a permutation of [2,2,1,1]. Code an ML function to determine whether one list is a permutation of another.

(e) A list ys is a generalised permutation of xs if ys is obtained by rearranging the elements of xs, where one element of xs is specially treated: it may appear any number of times (including zero) in ys. For example, [1,2,1] is a generalised permutation of [1,2] but [1,2,2,1] is not because two elements (1 and 2) appear the wrong number of times in it. Code an ML function to determine whether one list is a generalised permutation of another.

[taken from tripos question 2009 P1 Q1]

Exercise 7:

For the first supervision we wrote a function which takes a list and returns a new list containing only elements at even indexes, e.g. given [a, b, c, d] it should return [b, d]. Now write the equivilent for lazy lists.

[adapted from exercise 3.3 from the course notes]

Exercise 8:

A lazy binary tree is either empty or is a branch containing a label and two lazy binary trees, possibly to infinite depth

(a) Present an ML datatype to represent lazy binary trees

(b) Present an ML function that accepts a lazy binary tree and produces a lazy list that contains all of the trees labels.

[taken from exercise 9.4 in the course notes]

Exercise 9:

Write an analogue of map for sequences.

[taken from exercise 9.1 in the course notes]