Assignment 1

COS30023 - Languages in Software Development

Daniel Parker - 971328X

August 16, 2014

1. Proof Tree

2. Bubble Sort

2.1. bubble/2

In planning this predicate, I rationalised the result I wanted as being the sorted result of the head of the current list and the head of the tail of the current list. Then I would keep the smaller of the two, and recursively bubble the bigger part as the head of the remaining tail.

2.2. bubble_sort/2

The bubble_sort predicate makes use of three other predicates; bubble/2, reverse/2, and remove/3. This predicated is again recursive in nature. A sorted list is created by bubbling, reversing the result, removing the top most element (head), reversing it back again and calling the bubble sort on the remaining tail.

2.2.1. Source

```
% Bubble list from left to right
%
% LOGIC:
%       Check the head against the head of the tail
%       swapping the heads if the first head is bigger than the second head.
%       recursively bubble the remaining tail with the correct swapped head.
%
bubble([],[]).
bubble([X],[X]).
bubble([X,Y|T], [Y|Z]) :- X > Y, bubble([X|T], Z).
bubble([X,Y|T], [X|Z]) :- X =< Y, bubble([Y|T], Z).
```

```
% Remove the head
remove([H|T],H,T).

% Bubble sort the whole list
%
% LOGIC:
%       Bubble the list, reverse it and remove the head.
%       Bubble the tail of the resulting list. Repeat until all parts are sor
bubble_sort([],[]).
bubble_sort([X], [X]).
bubble_sort(X,[H|L]) :- bubble(X, Y),
                                                  reverse(Y, Z),
                                                  remove(Z,H,T),
                                                  reverse(T,C),
                                                  bubble_sort(C, L).
```

3. Logical Circuits

Data