EECS 560 Lab 7 Week of March 7

The topic for this lab is heaps. You will do a heapsort on a binary heap experimentally comparing two methods of restructuring the heap after a deletemin. You will need functions to push items up and down the heap. Here’s pseuodcode for push\_up and push\_down to use with buildhelp and heapsort. A is a 1-based array. You must also use 1-based arrays.

push\_up (i)

// move an item up the heap until the correct spot for it is found

//The item being moved up the heap starts in position i

val ← A[i]

position ← i

while ( (val < A[ ⎣position/2⎦ ]) and (position > 1)) do //integer division for find parent

A[position] ← A[ ⎣position/2⎦ ]

position ← ⎣position/2⎦

endwhile

A[position] ← val

push\_down (i)

// moves an item down the heap until the correct spot for it is found

// The item being moved down the heap starts in position i

val ← A[i]

position ← i

repeat the following steps until the correct position is found

Let k be the index of the smaller of A[2\*position] and A[2\*position + 1] //smaller child

If val > A[k] then

A[position] ← A[k]

position ← k

else A[position] ← val // the correct position has been found

Use the buildheap algorithm to construct the initial heap. See the class notes posted February 23 for both buildheap and heapsort.

To verify that the heap has been properly constructed, generate a small set of data (25 items or so), construct the heap, and when finished, print it out and check that the heap properties are satisfied. Then, use that heap to verify that your heapsort is working properly and print out the list after heapsort is finished. Turn in this verification as part of the lab submission.

In order to compare the method discussed in class with the modification described below, you will need two versions of this program. The modified insertion method uses a suggestion from Project 7.7 at the end of Chapter 7 in [A Practical Introduction to Data Structures and Algorithm Analysis 3rd edition (C++ version)](http://people.cs.vt.edu/~shaffer/Book/) by Clifford Shaffer.

Recall heapsort performs a deletemin operation by “remembering” the rightmost leaf (nth position), puts the key that was deleted from the root in position n and then decreases the size of the heap by 1. Starting at the root, the key being remembered is then pushed down the heap. The modification is based on the following observation—when a key X is being pushed down the heap, two comparisons are made. First the smaller of two children is found and then X is compared to the smaller child. This can be improved by omitting the second comparison and always moving the smaller child up to the parent node. When the leaf level is reached, use push\_up to move X to the proper position by comparing with its parent until the its correct spot is found. Usually this should result in fewer comparisons. To test this claim, use a counter to count the number of heap item comparisons done during heapsort. Do not count the comparisons made in buildheap.

For each variation of heapsort perform the operations below ten times. Use an array that holds 2000 values.

1. Generate 2000 random numbers in the range between -1000 and +1000.

2. Use buildheap to turn this list of numbers into a heap.

3. Heapsort the list.

4. Report the number of comparisions made while sorting that list and whether you are using the regular or the modified heapsort

NOTE: Since you must use the same data for both methods, you will need to use a pseudorandom number generator in order to make sure the data used for the modified heapsort is generated in exactly the same way as for the regular heapsort. (Do **not** generate the data and save a copy for use with the other method. The data sets must be generated separately for the two methods.)

To turn in:

1. Your testing that demonstrates both buildheap and heapsort are working properly.

2. All data collected i.e. the number of comparisions for each data set generated, clearly identified by the method being used

3. A discussion of what you learned from the data collected. For example, is the modified method better, and if so is the improvement significant?

All labs are due by noon Sunday.