Implementing Custom sort and H-index
Lab #6

By
Joshua Matthews

CS 303 Algorithms and Data Structures
October 5, 2015

1. Problem Specification
The goal of this Lab assignment was to implement a custom sort of our own to solve a custom sorting problem. We were also required to find H-index of array.

2. Program Design
This program required two new functions besides the main function and simple maintenance functions that were needed to setup the right conditions to complete the tasks.

The following steps were needed to develop this program:

A. I needed to setup the environment with the correct c++ libraries included and main function to accept arguments that would be passed into the program.
B. Next I setup the code to take the file passed in and extract the values from the file and push them on to a vector to be used for later.
C. I setup a swap function to easily swap variables around inside an array.
D. I then created the custom short which I labeled as short_sort. It accept a reference to a vector of ints and accepted a start position and an end position. If the start and end position variables equal each other the function returns. I iterate from the start to the end positions in the array and check for and set the largest and smallest values and then swap them with the current starting and end positions. Then call the short_sort function again with the starting point incremented and the ending point decremented.
E. I then setup the quick_sort and partition functions to sort the array quickly.
F. I setup the function for find_h to search for the H-index of the array. I create a h_index integer variable and set it to zero. I then iterate through the array that is passed in from the 0th array. I setup an if statement to check if the value at the current index is less then the size of the array minus the h_index to increment the h_index by one. Once this if statement became false it would return the h_index.
G. I setup some print statements around the functions to print out the results and captured the time for the short_sort.

The included libraries were: iostream(for printing out to the screen), fstream(for reading in data from a file), vector(for having a dynamically sizing array), ctime(for calculating program running time in seconds).

3. Testing Plan
The Lab06.cpp file tests our custom sort on the values from whatever file is passed in. The program accepts a file and parsers out each number in the file and pushes them back onto the vector object that was created to hold all of the values. The initial values we retrieve from the files are random numbers between 1 and the number of elements that are in the file inclusively. Once the vector is filled with every value from the file the current time is captured in a variable then the Heap sort is invoked. The vector is passed in by reference so there is no time being wasted on copying the values over to the function and allows the function to have direct manipulation over the object. Once our custom sort is done we capture the current time minus the start time to calculate how long the function took in seconds, then display it to the screen for the user to see.

The H-index is then calculated by passing the array into it's own function and once found saved to a variable and displayed to the screen.

4. Test Cases

Input File	Array Values	Custom Sort	H-index
input_16.txt	unsorted: l30l15l34l17l76l60l4l98l55l5l48l73l43l52l1l16l16l sorted: l1l4l5l15l16l16l17l30l34l43l48l52l55l60l73l76l98l	Short sort time to complete: 2e-06s	H-index = 3

5. Analysis and Conclusion

The custom sort is to show that you can create a sort for any type of problem, one way or another. Some sorts, though they may solve the problem, may not be the fastest sorting algorithm. The H-index is easy to find if the array is already sorted. I was unable to figure out a good way to find the H-index while I was sorting an array or in an un-sorted array.

6. References

The www.cplusplus.com has references and documentation on all of the standard libraries that were used in this program. Input files were supplied by Abhishek Anand.