Programming Assignment 3

Class : CS 610-105

Name : Tushar Sharma, UCID : ts362

Contents

Instructions on Running Program

Compile the program

```
$ make
```

Run the program

```
$ make run input=f.txt output=g.txt
```

Here

- f.txt is the input file
- g.txt is output file

Then choose whether to run lsd using pointers or swap

```
Please select technique for sorting:
1        Using pointer array
2        Using Swapping
```

Analysis

Time Complexity

LSD Radix Sort is linear time sorting algorithm. It is a stable sort. I have implemented lsd Radix sort using two different method.

- First Method makes use of pointers to keep track of indices without swapping the values
- Second Method swaps the values in each cycle

Both the algorithms of lsd Radix sort takes O(n) asymptotic time.

LSD Radix Sort Using pointers

n	k	time (s)
8	21	0.000591
16	21	0.000674
32	21	0.000762
64	21	0.000961
128	21	0.001582
256	21	0.002517
512	21	0.004514
1024	21	0.008236
2048	21	0.013947
4096	21	0.0263
5164	21	0.033631

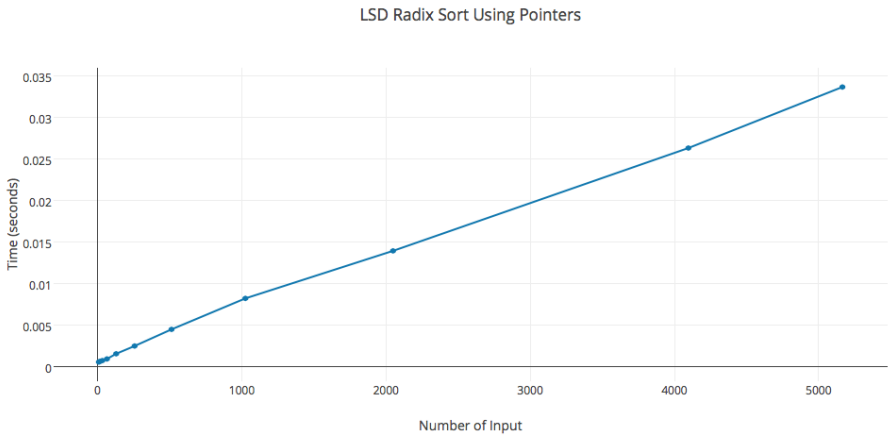

Figure 1: Image of LSD Sort using Pointers

LSD Radix Sort Using Swapping

n	k	time (s)
8	21	0.000773
16	21	0.000715
32	21	0.000798
64	21	0.001096
128	21	0.001924
256	21	0.003028
512	21	0.005486
1024	21	0.010274
2048	21	0.018737
4096	21	0.038562
5164	21	0.045842

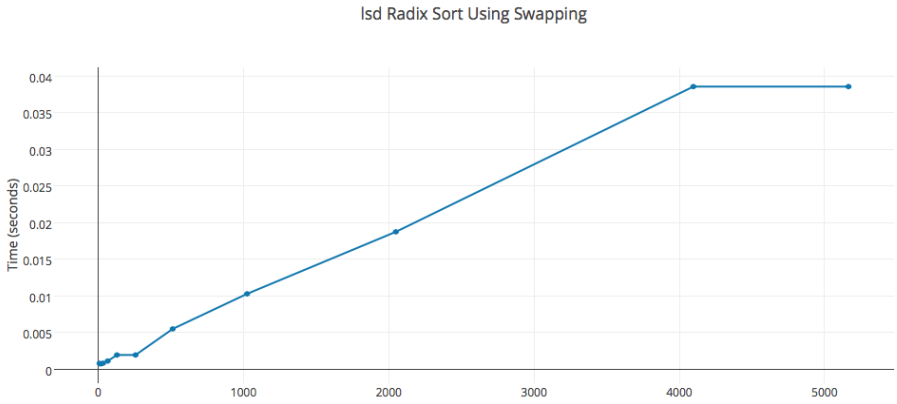

Figure 2: Image of LSD Sort using Pointers

Source Code

```
#include <iostream>
#include <vector>
#include <fstream>
#include <cstring>
#include <cstdlib>
using namespace std;

/* Purpose : Implement Radix Sort using stable sort
 *
 * AUTHOR : Tushar Sharma <ts362.njit.edu>, <tushar.sharma1729.gmail.com>
 */
string input;
string output;
int n;
const int k = 21;
const int R = 256;

void readFromFile(vector<string>&, fstream&, string);
void printToFileNoSwap(char [][k], fstream&, string, int[]);
void getString(char [][k], vector<string>);
void lsdRadixSortNoSwap(char [][k], int []);
void countingSortNoSwap(char [][k], int , int [], int []);

void printToFileSwap(char [][k], fstream&, string);
void lsdRadixSortSwap(char [][k]);
void countingSortSwap(char [][k], int );


int main(int argc, char **argv)
{

    if (argc < 3) {
        input= "ins/f.txt";
        output = "outs/g.txt";
    } else {
        string prefix = "ins/";
        input = prefix + string(argv[1]);
    prefix = "outs/";
    output = prefix + string(argv[2]);
    }


    clock_t time_start, time_stop;
```

```
time_start = clock();

fstream fo, fi;

vector<string> arr;

readFromFile(arr, fi, input);
n = arr.size();

int indexP[n];
memset(indexP, 0, sizeof(int) * n);

char S[n][k];
S[0][0] = ' ';

getString(S, arr);
//inititlize indexP
for (int i = 0; i < n; i++) {
    indexP[i] = i;
}

//print those values of indexP
#ifdef DEBUG
for (int i = 0; i < n; i++) {
    cout<<indexP[i]<<" " ;
}
cout<<endl;
#endif

int choice;
cout<<"\nPlease select technique for sorting: \n1 \t Using pointer array  \n2 \t Using S
cin>>choice;
if ( choice == 1) {
    lsdRadixSortNoSwap(S, indexP);

    #ifdef DEBUG
    //print those values of indexP
    for (int i = 0; i < n; i++) {
        cout<<indexP[i]<<" " ;
    }
    cout<<endl;
    #endif

    printToFileNoSwap(S, fo, output, indexP);
}
```

```
    else {
         lsdRadixSortSwap(S);

     printToFileSwap(S, fo, output);
    }

    fo.close();
    fi.close();

    double duration = ( clock() - time_start ) / (double) CLOCKS_PER_SEC;
    cout<<"\nThis program took "<<duration<<" seconds to execute. Thank you for running.\n"<

    return 0;
}

void readFromFile(vector<string>& arr, fstream& fp, string filename)
{

    string names = "";
    string values;
    char ch;

    fp.open(filename.c_str(), ios::in | ios::binary);


    if (fp.is_open()) {
        while (getline(fp, names)){
        //cout<<names<<endl;
        int index = names.find(" ");
        values = names.substr(0, index);
        //cout<<values<<endl<<endl;
        arr.push_back(values);
    }

    }
}


void printToFileNoSwap(char S[][k], fstream& fo, string filename, int indexP[])
{
    int tempP[n];
    memset(tempP, 0, sizeof(int) * n);


    fo.open(filename.c_str(), ios::out | ios::binary);
    if (!fo) {
```

```
        cout<<"Error opening the file\n";
    exit(-1);
    }
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < k; j++) {
        //cout<<S[i][j];
        fo <<S[indexP[i]][j];
    }
    //cout<<endl;
    fo<<endl;
    }

}

void getString(char S[][k], vector<string> arr)
{

    for (int i = 0; i < n; i++) {
        //string temp = arr[i].substr(0, k);
    int l = arr[i].size();
    for (int j = 0; j < l; j++) {
        if (j < k) {
            S[i][j] = arr[i].at(j);
        //cout<<S[i][j]<<" " ;
          }
    }
        //this is for padding
    if (l < k) {
        for (int j = l ; j < k ; j++) {
            S[i][j] = '\0';
        }
    }
    }
}



void lsdRadixSortNoSwap(char S[][k], int indexP[])
{
    int count[k];
    int prevIndex[n];

   for (int d = 0; d < n ; d++ )
        prevIndex[d] = d;

    for (int d = k - 1; d >=0 ; d--) {
```

```
    countingSortNoSwap(S, d, indexP, prevIndex);
    }

}

void countingSortNoSwap(char S[][k], int j, int indexP[], int prevIndex[])
{
    //here j is the column

    int count[256] = {0};

    char temp[n][k];
    temp[0][0] = ' ';
    int tempCount[n];
    memset(tempCount, 0, sizeof(int) * n);

    for (int i = 0; i < n; i++) {
    int valueChar = (int) S[prevIndex[i]][j];
    #ifdef DEBUG
    cout<<"Value char of "<<S[prevIndex[i]][j]<<" is "<<valueChar<<endl;
    #endif
    count[valueChar + 1]++;
    }

    for (int p = 1; p < 256; p++) {
        count[p] += count[p - 1];
    }

    for (int i = 0; i < n; i++) {
        int valueChar = (int)S[prevIndex[i]][j];
        int index = count[valueChar++]++;

    tempCount[index] =   prevIndex[i];
    #ifdef DEBUG
    cout<<"print valuechar "<<S[prevIndex[i]][j]<<"  index "<<index<<" of "<<tempCount[index
    #endif
    }

    for (int i = 0; i < n; i++) {
        indexP[i] = tempCount[i];
    prevIndex[i] = tempCount[i];
    #ifdef DEBUG
    cout<<tempCount[i]<<" values "<<endl;
    #endif
    }
}
```

```
void printToFileSwap(char S[][k], fstream& fo, string filename)
{
    fo.open(filename.c_str(), ios::out | ios::binary);
    if (!fo) {
        cout<<"Error opening the file\n";
        exit(-1);
    }
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < k; j++) {
            //cout<<S[i][j];
            fo <<S[i][j];
        }
        //cout<<endl;
        fo<<endl;
    }

}

void lsdRadixSortSwap(char S[][k])
{
    int count[k];

    for (int d = k - 1; d >=0 ; d--) {
        countingSortSwap(S, d);
    }

}

void countingSortSwap(char S[][k], int j)
{
    //here j is the column

    int count[256] = {0};
    //char temp[n];

    char temp[n][k];
    temp[0][0] = ' ';


    for (int i = 0; i < n; i++) {
        int valueChar = (int) S[i][j];
        //cout<<"Value char of "<<S[i][j]<<" is "<<valueChar<<endl;
        count[valueChar + 1]++;
    }
```

```
    for (int p = 1; p < 256; p++) {
        count[p] += count[p - 1];
    }

    for (int i = 0; i < n; i++) {
        int valueChar = (int)S[i][j];
        //temp[count[valueChar++]++] = S[i][j];
        int index = count[valueChar++]++;
        //cout<<"index "<<index<<endl;
        //strcpy(temp[index], S[i]);
        for (int p = 0; p < k; p++) {
            temp[index][p] = S[i][p];
            //cout<<temp[index][p]<<endl;
        }
    }

    for (int i = 0; i < n; i++) {
        //S[i][j] = temp[i];
        //cout<<temp[i]<<endl;
        //strcpy(S[i], temp[i]);
        for (int p = 0; p < k; p++) {
            S[i][p] = temp[i][p];
        }
        //cout<<S[i]<<endl;
    }

}
```