Maximum Weight Common Subsequence

Pijus Kumar Sarker
Graduate Student, Faculty of Computer Science
University of Windsor

Problem: The Mutation Sensitive Alignment (MSA) algorithm computes in the first step the MWCS (Maximum Weight Common Subsequence) of the *MUM* sequences *A* and *B* obtained from the genomes *G1* and *G2*, where each *MUM* is assigned a weight (could be its length or something else). As in the slides, we can consider the MUM labels of one, say *A*, to be in canonical order CS[*1..n*] and that of *B* in a permuted order PS[*1..n*]. An MWCS of CS[*1..n*] and PS[*1..n*] is in increasing order and of maximum weight.

Algorithm:

$$MWCS(i,j) = \max \begin{Bmatrix} MWCS[i-1, j-1] + w(A[i])d \\ MWCS[i, j-1] \\ MWCS[i-1, j] \end{Bmatrix}$$

Here *d*=1 if match otherwise *d*=0
Set, *MWCS[0,j]* = 0, *MWCS[i,0]* = 0, for 0 ≤ *i,j* ≤ *n*

Sample Input:
Sequences, A[i] = {1,2,3,4,5} and B[j]={1,5,2,4,3}
Weight: W[i] = {10,2,1,3,20}

Dynamic Programming Table

	-	1	5	2	4	3
-	0	0	0	0	0	0
1	0	↖ 10	10	10	10	10
2	0	↑ 10	10	12	12	12
3	0	↑ 10	10	12	12	13
4	0	↑ 10	10	12	12	15
5	0	10	↖ 30 ←	30 ←	30 ←	30

How to run the program:

1. Use GCC compiler to run this program
2. Command to run c program,
   gcc path/to/file/filename –o output_file_name
3. Then the run/open the output file.

```
/*
 * File:   mwcs.c
 * Author: pijuskumar
 *
 * Created on November 23, 2013, 12:53 PM
 *
 * *****************
 * *  Test Input   *
 * *****************
 *
 * A[i] = {1,2,3,4,5}
*  B[i] = {1,5,2,4,3}
 *
 */


#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <assert.h>

int N, d[1000], T[1000][1000], DP[1000][1000];
char *A[1000], *B[1000];
int dire[1000][1000][3],drd=0, drl=0, drt=0;

int W[] = {10,2,1,3,20};
// A[i] = {1,2,3,4,5}
// B[i] = {1,5,2,4,3}

int main(int argc, char** argv) {
   int m,n,i=0,j;
   char a[1000],b[1000];
   char S;


   printf("******** MAXIMUM WEIGHT COMMON SUBSEQUENCE *********\n\n");
   printf("Enter First Sequence (canonical order) A[i]: ");
   scanf("%s",&a);
   printf("\n\nEnter Second Sequence (permuted order) B[j]: ");
   scanf("%s",&b);

   if(strlen(a)!=strlen(b)){
      printf("\nPlease enter same length sequences as one is in canonical order and another
is in permuted order.");
   }else{
      parseInput(a, b);
   }

   assignMUMWeight();
```

```
    computeMWCS();

    displayDP();

    showResult();
    return 0;
}

int getmax(int d, int l, int t){
    int max=0;
    if(d==l && l==t){
        max = d;
        drd = 1;
        drl = 1;
        drt = 1;
    }else{
        if(d>l && d>t){
            max = d;
            drd = 1;
            drl = 0;
            drt = 0;
        }else if(l>d && l>t){
            max = l;
            drd = 0;
            drl = 1;
            drt = 0;
        }else{
            max = t;
            drd = 0;
            drl = 0;
            drt = 1;
        }

    }

    return max;
}
void computeMWCS(){
    int i,ti,j,tj,k,d,l,t,temp,max;

    for(i=1; i<=N; i++){
        for(j=1; j<=N+1; j++){
            DP[i][j]=0;
        }
    }

    for(ti=1; ti<=N; ti++){
        for(tj=1; tj<=N; tj++){
            i=ti-1;
            j=tj-1;
            l = DP[ti][tj-1];
            t = DP[ti-1][tj];
            d = DP[ti-1][tj-1];

            if(stringToInt(A[i])==stringToInt(B[j])){
                d += W[i];
            }
            max = getmax(d,l,t);
            DP[ti][tj] = max;
            dire[i][j][0] = drd;
            dire[i][j][1] = drl;
            dire[i][j][2] = drt;

        }
    }
}

void showResult(){
    int i,j,k=0,list[1000],m=N,o=0,t;

    printf("\nMAXIMUM WEIGHT : %d\n",DP[N][N]);
    printf("\n\n *****  MAXIMUM WEIGHTED COMMON SUBSEQUENCE *****\n");
    for(i=N; i>-1; i--){
        for(j=m; j>-1; j--){
          if(dire[i][j][0]==1){
              list[o] = i+1;
              o++;
              m=j-1;
              break;
          }else if(dire[i][j][2]==1){
              m=j;
              break;
          }else{

          }
        }
    }
    for(i=0; i<o;i++){
        k=o-1-i;
        if(i < k){
            t= list[i];
            list[i] = list[k];
            list[k] = t;
        }
    }
    printf("\n\n");
    for(i=0; i<o;i++){
        printf("%d ",list[i]);
    }
```

```
}


void assignMUMWeight(){
   int i,rnd;
   for(i=0; i<N; i++){
      rnd = rand() % 15;
      W[i]= 1 + rnd;  // assign random weight
   }

   printf("\n\n ----------------- WEIGHTS -----------------\nMUMs         = ");
   for(i=0; i<N; i++){
      printf("%s  ",A[i]);
   }
   printf("\nWeight, W[i] = ");
   for(i=0; i<N; i++){
      printf("%d  ",W[i]);
   }
   printf("\n");
}

void displayDP(){
   int i,j;
   printf("\n\n------------------------ DP TABLE ------------------------\n");
   for(i=-1; i<=N; i++){
      if(i<0){
         char mm[]="0";
         printf("     ");
      }else{
         if(i==0){
            printf("0    ");
         }else{
            printf("%s    ", B[i-1]);
         }

      }
   }
   printf("\n-----------------------------------------------------------\n");
   i=0;
   for(i=-1; i<N; i++){
      if(i==-1){
         printf("0  | ");
      }else{
         printf("%s  | ",A[i]);
      }

      for(j=0; j<=N; j++){
         printf("%d    ",DP[i+1][j]);
      }
```

```
        printf("\n");
    }

    printf("\n\n");

    for(i=0;i<N;i++){
        for(j=0;j<N;j++){
            printf("%d%d%d    ",dire[i][j][0],dire[i][j][1],dire[i][j][2]);
        }
        printf("\n");
    }
}

void parseInput(char a[1000], char b[1000]){
    int i=0,j=0,sum=0;
     A[i] = strtok(a,",");
    while(A[i]!=NULL) {
      j= ++i;
      A[j] = strtok(NULL,",");
    }
    N=i;
    i=0;
    B[i] = strtok(b,",");

    while(B[i]!=NULL) {
      sum=0;
      j= ++i;
      B[j] = strtok(NULL,",");
    }
}

int stringToInt(char str[]){
    int i=0,sum=0;

    while(str[i]!='\0'){
         if(str[i]< 48 || str[i] > 57){
             printf("Unable to convert it into integer.\n");
             return 0;
         }
         else{
             sum = sum*10 + (str[i] - 48);
             i++;
         }
 }
    return sum;
}
```