Assignment 3 of Algorithm Design and Analysis

L.J.SHOU 寿林钧
No. **201228013229133**

Nov.07 2012

1 Dynamic Programming(6)

My implementation results are shown as follows using two different scoring functions.

Implementaion details in the Appendix.

$$\delta_1(a,b) = \begin{cases} 1; & \text{a} = \text{b} \\ -1; & \text{a} \neq \text{b} \\ -3; & \text{a} = \text{'-' or b} = \text{'-'} \end{cases} \tag{1}$$

The optimal score we obtain is 618.

S'=

CGCTTCCAGCAGTGACGGAAGGCCTCTTCGGAGTCTCGAGTGGCGAACGGGTGA

GTAACACGTGGGCAATCTGCCCTGCACTTCGGGATAAGCCTGGGAAACTGGGTC

TAATACCGGATAGGACCTCAAGACGCATGTCTTCTGGTGGAAAGCTTTTGCGGT

GTGGGATGGGCCCGCGGCCTATCAGCTTGTTGGTGGGGTGACGGCCTACCAAGG

CGACGACGGGTAGCCGGCCTGAGAGGGTGTCCGGCCACACTGGGACTGAGATAC

GGCCCAGACTCCTACGGGAGGCAGCAGTGGGGAATATTGCACAATGGGCGCAAG

CCTGATGCAGCGACGCCGCGTGGGGGATGACGGCCTTCGGGTTGTAAACCTCTT

TCACCATCGACGAAGGTCCGGGTTTTCTCGGATTGACGGTAGGTGGAGAAGAAG

CACCGGCCAACTACGTGCCAGCAGCCGCGGTAATACGTAGGGTGCGAGCGTTGT

CCGGAATTA–CTGGGCGTAAAGAGCTCGTAGGTGGTTTGTCGCGTTGTTCGTG

AAATCTCACGGCTTAACTGTGAGCGTGCGGGCGATAC–GGGCAGACTAGAGTA

CTGCAGGGGAGACTGGAATTCCTGGTGTAGCGGTGGAATGCGCAGATATCAGGA

GGAACACCGGT–GGCGAAGGCGGGTCTCTGGGCAGTAACTGACGCTGAGGAGC

GAAAGCGTGGGGAGCGAACAGGATTAGATACCCTGGTAG–TCCACGCCGTAaA

CGGTGGGTACTAGGTGTGGGTTTCCTTCCTTGGGATCCGTGCCGTAGCTAACGC

ATTAAGTACCCCG–CCTGGGGAGTACGGCCGCAAGGCTAAA-CTCATGTAA

T’=

-G-TTCCAGCAGTGACGGAAGGCC-CTTCGGGGTCTCGAGTGGCGAACGGGTGA

GTAACACGTGGGTGATCTGCCCTGCACTTTGGGATAAGCCTGGGAAACTGGGTC

TAATACCGAATATGACCACGCGCTTCATGGTGTGTGGTGGAAAGCTTTTGCGGT

GTGGGATGGGCCCGCGGCCTATCAGCTTGTTGGTGGGGTAATGGCCTACCAAGG

CGACGACGGGTAGCCGGCCTGAGAGGGTGACCGGCCACACTGGGACTGAGATAC

GGCCCAGACTCCTACGGGAGGCAGCAGTGGGGAATATTGCACAATGGGCGCAAG

CCTGATGCAGCGACGCCGCGTGAGGGATGACGGCCTTCGGGTTGTAAACCTCTT

TCAATAGGGACGAAGCGCAAG—TGA-CGG-T–ACC-TA–T–AGAAGAAG

GACCGGCCAACTACGTGCCAGCAGCCGCGGTAATACGTAGGGTCCGAG–CGTT

GTCCGGAATTACTGGGCGTAAAGAGCTCGTAGGTGGTTTGTCGCGTTGTTCGTG

AAAACTCACAGCTTAACTGTGG–GCGTGCGGGCGATACGGGCAGACTAGAGTA

CTGCAGGGGAGACTGGAATTCCTGGTGTAGCGGTGGAATGCGCAGATATC–AG

GAGGAACACCGGTGGCGAAGGCGGGTCTCTGGGCAGTAACTGACGCTGAGGAGC

GAAAGCGTGGGGAGCGAACAGGAT–TAGATACCCTGGTAGTCCACGCCGTAAA

CGGTGGGTACTAGGTGTGGGTTTCCTTCCTTGGGATCCGTGCCGTAGCTAAC–

GCATTAAGTACCCCGCCTGGGGAGTACGGCCGCAAGGCTAAAACTCAAATA-

$$\delta_2(a,b) = \begin{cases} 2; & \mathrm{a} = \mathrm{b} \\ -1; & \mathrm{a} \neq \mathrm{b} \\ -2; & \mathrm{a} = \text{'-'} \text{ or } \mathrm{b} = \text{'-'} \end{cases} \tag{2}$$

The optimal score is 1449.

S’=

CGCTTCCAGCAGTGACGGAAGGCCTCTTCGGAGTCTCGAGTGGCGAACGGGT

GAGTAACACGTGGGCAATCTGCCCTGCACTTC–GGGATAAGCCTGGGAAAC

TGGGTCTAATACCGGATAGGACCTCAAGACG-CATG-TCTTCTGGTGGAAAG

CTTTTGCGGTG–TGGGATGGGCCCGCGGCCTATCAGCTTGTTGGTGGGGTG

ACGGCCTACCAAGGCGACGACGGGTAGCCGGCCTGAGAGGG–TGTCCGGCC

ACACTGGGACTGAGATACGGCCCAGACTCCTACGGGAGGCAGCAGTGGGGAA

TATTGCACAATGGGCGCAA–GCCTGATGCAGCGACGCCGCGTGGGGGATGA

CGGCCTTCGGGTTGTAAACCTCTTTCACCATCGACGAAG-GTCCGGGTTTTC

TCGGATTGACGGTAGGTGGAGAAGAAGCACCGGCCAACTACGTGCCAGCAGC

CGCGGTAATACGTAGGGTGCGAGCGTTGTCCGGAATTA–CTGGGCGTAAAG

AGCTCGTAGGTGGTTTGTCGCGTTGTTCGTGAAATCTCACGGCTTAACTGTG

AGCGTGCGGGCGATAC–GGGCAGACTAGAGTACTGCAGGGGAGACTGGAAT

TCCTGGTGTAGCGGTGGAATGCGCAGATATCAGGAGGAACACCGGT–GGCG

AAGGCGGGTCTCTGGGCAGTAACTGACGCTGAGGAGCGAAAGCGTGGGGAGC

GAACAGGATTAGATACCCTGGTAG–TCCACGCCGTAaACGGTGGGTACTAG

GTGTGGGTTTCCTTCCTTGGGATCCGTGCCGTAGCTAACGCATTAAGTACCC

CG–CCTGGGGAGTACGGCCGCAAGGCTAAA-CTCATGTAA

T'=

-G-TTCCAGCAGTGACGGAAGGCC-CTTCGGGGTCTCGAGTGGCGAACGGGT

GAGTAACACGTGGGTGATCTGCCCTGCA–CTTTGGGATAAGCCTGGGAAAC

TGGGTCTAATACCGAATATGACCACGCG-CTTCATGGTGTG-TGGTGGAAAG

CTTTTGCG–GTGTGGGATGGGCCCGCGGCCTATCAGCTTGTTGGTGGGGTA

ATGGCCTACCAAGGCGACGACGGGTAGCCGGCCTGAGA–GGGTGACCGGCC

ACACTGGGACTGAGATACGGCCCAGACTCCTACGGGAGGCAGCAGTGGGGAA

TATTGCACAATGGGCG–CAAGCCTGATGCAGCGACGCCGCGTGAGGGATGA

CGGCCTTCGGGTTGTAAACCTCTTTCAATAGGGACGAAGCG-CAAG—TGA

-CGG-T–ACC-TA–T–AGAAGAAGGACCGGCCAACTACGTGCCAGCAGC

CGCGGTAATACGTAGGGTCCGAG–CGTTGTCCGGAATTACTGGGCGTAAAG

AGCTCGTAGGTGGTTTGTCGCGTTGTTCGTGAAAACTCACAGCTTAACTGTG

G–GCGTGCGGGCGATACGGGCAGACTAGAGTACTGCAGGGGAGACTGGAAT

TCCTGGTGTAGCGGTGGAATGCGCAGATATC–AGGAGGAACACCGGTGGCG

AAGGCGGGTCTCTGGGCAGTAACTGACGCTGAGGAGCGAAAGCGTGGGGAGC

GAACAGGAT–TAGATACCCTGGTAGTCCACGCCGTAAACGGTGGGTACTAG

GTGTGGGTTTCCTTCCTTGGGATCCGTGCCGTAGCTAAC–GCATTAAGTAC

CCCGCCTGGGGAGTACGGCCGCAAGGCTAAAACTCAAATA-

From these two instances, we can see that the optimal alignment and score highly depend on how you design the scoring scheme.

2 Appendix

```c
/*implementation of Needleman_Wunch Algorithm in C by L.J.SHOU */
#include<stdio.h>
#include<stdlib.h>
#include<malloc.h>
#include<string.h>
#define score1 2
#define score2 -1
#define score3 -2
void get_opt_alignment1(int **OPT,int m,int n,char *s)  //backtracking
{
        if(m<1&n<1)
                return;
        if(m<1) //in the first row
        {
                get_opt_alignment1(OPT,m,n-1,s);
                printf("-");
        }
        else if(n<1) // in the first column
        {
                get_opt_alignment1(OPT,m-1,n,s);
                printf("%c",s[m-1]);
        }
        else
        {
                if(OPT[m][n]==OPT[m][n-1]+score3)
                {
                        get_opt_alignment1(OPT,m,n-1,s);
                        printf("-");
                }
                else if(OPT[m][n]==OPT[m-1][n]+score3)
                {
                        get_opt_alignment1(OPT,m-1,n,s);
                        printf("%c",s[m-1]);
                }
                else
                {
                        get_opt_alignment1(OPT,m-1,n-1,s);
                        printf("%c",s[m-1]);
                }
        }
}
void get_opt_alignment2(int **OPT,int m,int n,char *t)
{
        if(m<1&n<1)
                return;
        if(m<1)
        {
                get_opt_alignment2(OPT,m,n-1,t);
                printf("%c",t[n-1]);
        }
        else if(n<1)
        {
                get_opt_alignment2(OPT,m-1,n,t);
                printf("-");
        }
        else
        {
        if(OPT[m][n]==OPT[m-1][n]+score3)
        {
                get_opt_alignment2(OPT,m-1,n,t);
                printf("-");
        }
        else if(OPT[m][n]==OPT[m][n-1]+score3)
        {
                get_opt_alignment2(OPT,m,n-1,t);
                printf("%c",t[n-1]);
        }
        else
        {
                get_opt_alignment2(OPT,m-1,n-1,t);
                printf("%c",t[n-1]);
        }
        }
}
int Needleman_Wunch(char s[],char t[])
{
        int i,j,m,n,**OPT,opt;
        m=strlen(s);
        n=strlen(t);
        OPT=(int **)malloc((m+1)*sizeof(int*));
        for(i=0;i<m+1;i++)
                OPT[i]=(int *)malloc((n+1)*sizeof(int));
        for(i=0;i<m+1;i++)
                OPT[i][0]=score3*i;
        for(j=0;j<n+1;j++)
                OPT[0][j]=score3*j;
        for(i=1;i<m+1;i++)
                for(j=1;j<n+1;j++)
                {

                        if(s[i-1]==t[j-1])
                                OPT[i][j]=OPT[i-1][j-1]+score1;
                        else
                                OPT[i][j]=OPT[i-1][j-1]+score2;
                        if(OPT[i-1][j]+score3>OPT[i][j])
                                OPT[i][j]=OPT[i-1][j]+score3;
                        if(OPT[i][j-1]+score3>OPT[i][j])
                                OPT[i][j]=OPT[i][j-1]+score3;
                }
        printf("S'=\n");
        get_opt_alignment1(OPT,m,n,s);
        printf("\nT'=\n");
        get_opt_alignment2(OPT,m,n,t);
        opt=OPT[m][n];
        //free the assigned memory
        for(i=0;i<m+1;i++)
                free(OPT[i]);
        free(OPT);
        return opt;
}
void main()
{
        FILE *fp;
        char *s,*t;
        char s1[]="urr",t1[]="ocurrance";
//=============Load Gene1_Seq.txt and Gene2_Seq.text
        int m=0,n=0,i;
//m and n denotes the lenth of Gene1_seq and Gene2_seq respectively
        if((fp=fopen("Gene1_Seq.txt","r"))==NULL)
        {
                printf("not open");
                exit(0);
        }
        while (fgetc(fp)!=EOF)
                m++;
        s=(char*)malloc(sizeof(char)*(m+1));    //the last element is '\0'
        rewind(fp);
        i=0;
        while((s[i++]=fgetc(fp))!=EOF);
        s[m]='\0';
        if((fp=fopen("Gene2_Seq.txt","r"))==NULL)
        {
                printf("not open");
                exit(0);
        }
        while (fgetc(fp)!=EOF)
                n++;
        t=(char*)malloc(sizeof(char)*(n+1));
        rewind(fp);
        i=0;
        while((t[i++]=fgetc(fp))!=EOF);
        t[n]='\0';
        fclose(fp);
        printf("\nThe optimal score is %d.\n",Needleman_Wunch(s,t));
}
```