acm.mipt.ru

олимпиады по программированию на Физтехе

Раздел «Алгоритмы» . LongestCommonSubsequenceCPP :

Наибольшая общая подпоследовательность

***Содержание*:**

Формулировка задачи

Дано две последовательности символов (произвольных объектов). Необходимо найти наибольшую общую подпоследовательность (longest common subsequence), то есть выкинуть некоторое количество элементов из этих последовательностей, чтобы они стали одинаковыми и имели максимальную длину.

Пример:

S1 = **A B** D **E** F F **E** K **A** B C D

S2 = G **A B** C **E** K **E** E **A**

Наибольшая общая подпоследовательность есть

LCS(S1, S2) = **A B E E A**

Идея решения

Идея решения заключается в том, что мы постепенно вычисляем элементы таблицы L, в которой в ячейке (i, j) находится длина наибольшей общей последовательности для S1[1..i] и S2[1..j], где S1[1..i] – первые i элементов последовательности S1, S2[1..j] – первые j элементов последовательности S2. То есть

L[i,j] = Length (LCS (S1[1..i], S2[1..j]))

Легко вычислить первую строчку и первый столбец этой таблицы. Затем, двигаясь по строчкам сверху вниз (а в строчках слева направо) и используя вычисленные значения, последовательно вычисляем всю таблицу. Число в правом нижнем углу – длина искомой последовательности.

После этого по вычисленной таблице нужно восстановить одну из наибольших общих подпоследовательностей.

Код на Си

```
#include <stdio.h>
#define N 1000
#define max(a,b) ((a>b) ? a : b)
int L[N][N];

int main()
{
    char a[N];
    char b[N];
    int i, j, n, m;
    int yes = 0;

    scanf("%s", a);
    scanf("%s", b);
    n = strlen(a);
```


Поиск

Поиск

Раздел «Алгоритмы»

Главная
Форум
Ссылки
El Judge

Инструменты:

Поиск
Изменения
Index
Статистика

Разделы

Информация
Алгоритмы
Язык Си
Язык Ruby
Язык Ассемблера
El Judge
Парадигмы
Образование
Сети
Objective C

Logon>>

```
    m = strlen(b);

    for( j = 0; j < m ; j++ )
    {
       if( b[j] == a[0] ) yes = 1;
       L[0][j] = yes;
    }
    yes = 0;
    for(i = 0; i < n; i++)
    {
       if( a[i] == b[0] ) yes = 1;
       L[i][0] = yes;
    }

    for(i = 1; i < n; i++)
    {
       for(j = 1; j < m ; j++)
       {

          if( a[i] == b[j] )
             L[i][j] = L[i-1][j-1] + 1;
          else
             L[i][j] = max( L[i-1][j], L[i][j-1] );
       }
    }

    printf( "%d\n",  L[n-1][m-1] );
    return 0;
}
```

Задачи для самостоятельного решения

- Как по массиву L восстановить максимальную общую подпоследовательность (одну из максимальных)?
- Напишите программу, которая определяет число различных наибольших общих подпоследовательностей.
- Напишите оптимальный алгоритм в предположении, что элементы последовательностей с большой вероятностью не повторяются (для каждой из последовательностей в отдельности).
- Напишите программу, где нужно найти самую хорошую подпоследовательность. Самая хорошая подпоследовательность имеет не только большую длину, но еще и слабо дробит исходные подпоследовательности на кусочки. То есть нужно максимизировать число очков за общую подпоследовательность C, где число очков Score(C) есть
  - Score(C) = Length(C) - w*(L1 - L2)
    - Length(C) – длина подпоследовательности
    - w – Коэффициент, определяющий величину штрафа
    - L1 и L2 – число непрерывающихся кусочков, на которые подпоследовательность делит первую и вторую подпоследовательность в сумме.

-- ArtemVoroztsov - 23 Oct 2004