



算法分析与设计

Analysis and Design of Algorithm

Lesson 09



要点回顾

- 动态规划算法的设计要点
 - **建模**：目标函数、约束条件
 - **分段**：确定子问题的**边界**
 - **分析**：子问题之间的依赖关系
 - **判断**：**最优子结构性质**
 - **求解**：先定最小子问题（初值），**自底向上**求解
- 实例：矩阵链相乘问题
 - **目标**：加括号求出矩阵链相乘的最小乘法次数
 - 穷举法（二叉树求解，复杂度分析）
 - 动态规划算法的**递归**实现
 - 动态规划算法的**迭代**实现

组合问题中的动态规划法

最长公共子序列

子序列：若给定序列 $X=\{x_1,x_2,\dots,x_m\}$ ，则另一序列 $Z=\{z_1,z_2,\dots,z_k\}$ ，是 X 的子序列是指：存在一个**严格递增**下标序列 $\{i_1,i_2,\dots,i_k\}$ 使得对于所有 $j=1,2,\dots,k$ 有： $z_j=x_{i_j}$ 。

例：序列 $Z=\{B,C,D,B\}$ 是 $X=\{A,B,C,B,D,A,B\}$ 的子序列，相应的递增下标序列为 $\{2,3,5,7\}$ 。

给定两个序列 X 和 Y ，当另一序列 Z 既是 X 的子序列又是 Y 的子序列时，称 Z 是序列 X 和 Y 的**公共子序列**。

最长公共子序列

- **问题：** 给定两个序列 $X=\{x_1, x_2, \dots, x_m\}$ 和 $Y=\{y_1, y_2, \dots, y_n\}$ ，找出 X 和 Y 的最长公共子序列 (Longest Common Subsequence, LCS)

- **实例：**

X: A B C B D A B

Y: B D C A B A

最长公共子序列: B C B A, 长度4

- ➔ 不是唯一的，长度相等情况下，可能有多个不同公共子序列，只需给出一个即可



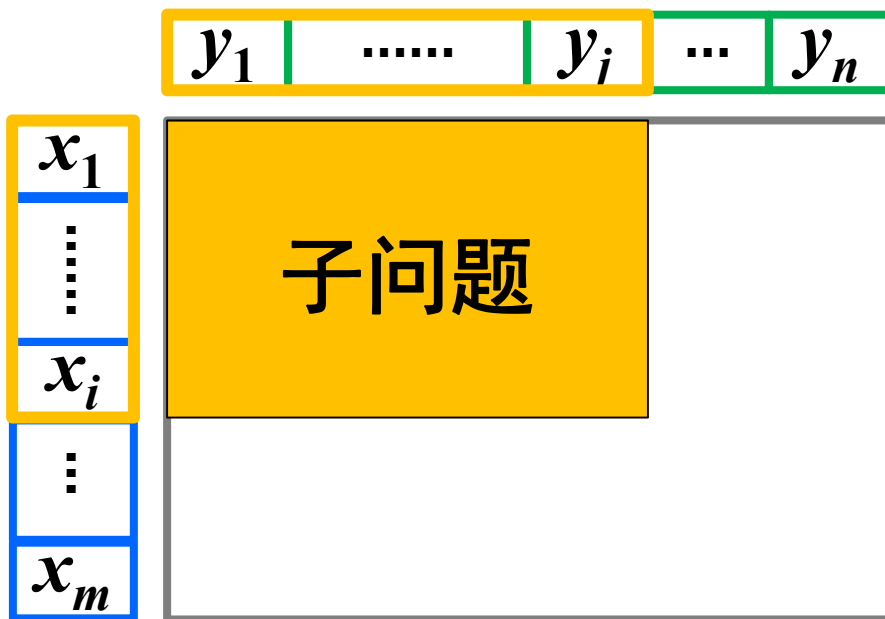
穷举法

- 不妨设 $m \leq n, |X|=m, |Y|=n$
- 算法：依次检查X的每个子序列在Y中是否出现
- 时间复杂度：
 - X有 2^m 个子序列
 - 每个子序列 $O(n)$ 时间

最坏情况下时间复杂度： $O(n2^m)$

子问题界定

- X的终止位置是 i ，Y的终止位置是 j
- $X_i = \{x_1, x_2, \dots, x_i\}$ ， $Y_j = \{y_1, y_2, \dots, y_j\}$
- 参数 i 和 j 界定子问题



子问题间的依赖关系

- 假设两个序列 $X = \{x_1, x_2, \dots, x_m\}$, $Y = \{y_1, y_2, \dots, y_n\}$, $Z = \{z_1, z_2, \dots, z_k\}$ 为 X 和 Y 的 LCS, 那么
 - 若 $x_m = y_n \rightarrow z_k = x_m = y_n$, 且 Z_{k-1} 是 X_{m-1} 与 Y_{n-1} 的 LCS
 - 若 $x_m \neq y_n$, $z_k \neq x_m \rightarrow Z$ 是 X_{m-1} 与 Y 的 LCS
 - 若 $x_m \neq y_n$, $z_k \neq y_n \rightarrow Z$ 是 X 与 Y_{n-1} 的 LCS



满足最优子结构性质和子问题重叠性

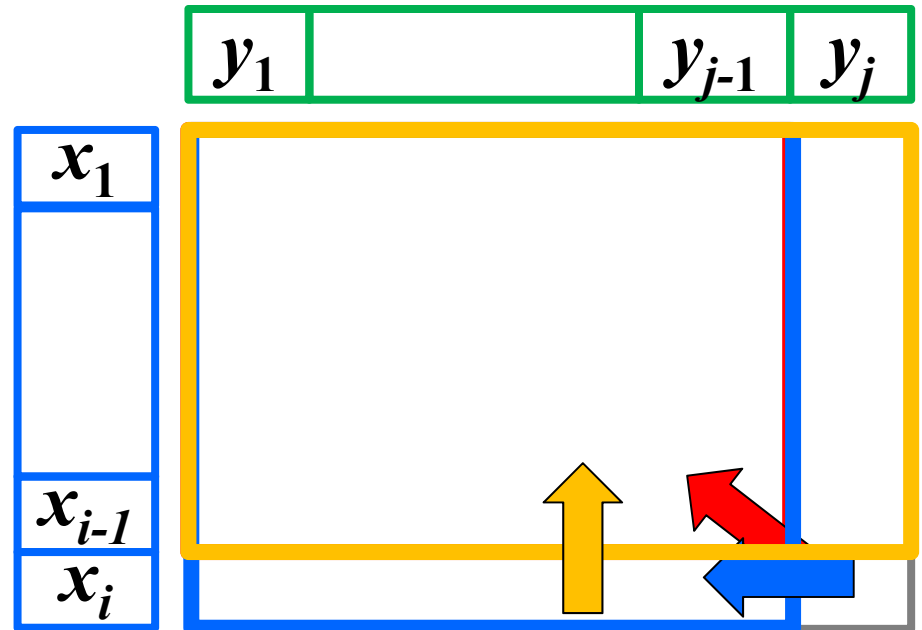
优化函数的递推方程

- 令X和Y的子序列
- $X_i = \{x_1, x_2, \dots, x_i\}$, $Y_j = \{y_1, y_2, \dots, y_j\}$
- $C[i, j]$: X_i 与 Y_j 的LCS的长度

$$C[i, j] = \begin{cases} 0 & \text{若 } i=0 \text{ 或 } j=0 \\ C[i-1, j-1]+1 & \text{若 } i, j > 0, x_i = y_j \\ \max \{C[i, j-1], C[i-1, j]\} & \text{若 } i, j > 0, x_i \neq y_j \end{cases}$$

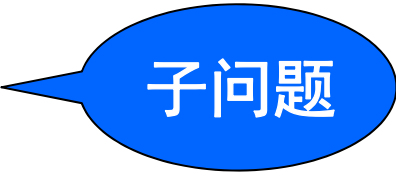

标记函数

- 标记函数： $B[i,j]$ ， 值为 \swarrow 、 \leftarrow 、 \uparrow
- $C[i,j] = C[i-1,j-1] + 1$: \swarrow
- $C[i,j] = C[i,j-1]$: \leftarrow
- $C[i,j] = C[i-1,j]$: \uparrow



算法的伪码

■ 算法 LCS(X, Y, m, n)

1. for $i \leftarrow 1$ to m do $C[i,0] \leftarrow 0$ 
2. for $i \leftarrow 1$ to n do $C[0,i] \leftarrow 0$
3. for $i \leftarrow 1$ to m do
4. for $j \leftarrow 1$ to n do 
5. if $X[i]=Y[j]$
6. then $C[i,j] \leftarrow C[i-1,j-1]+1$
7. $B[i,j] \leftarrow \text{“}\nearrow\text{”}$ 
8. else if $C[i-1,j] \geq C[i,j-1]$
9. then $C[i,j] \leftarrow C[i-1,j]$
10. $B[i,j] \leftarrow \text{“}\uparrow\text{”}$
11. else $C[i,j] \leftarrow C[i,j-1]$
12. $B[i,j] \leftarrow \text{“}\leftarrow\text{”}$

追踪解

- 算法 **StructureSequence(B,i,j)**
 - 输入: $B[i,j]$
 - 输出: X与Y的最长公共子序列
 1. if $i=0$ or $j=0$ then return //序列为空
 2. if $B[i,j]=“↖”$
 3. then 输出 $X[i]$
 4. **StructureSequence(B,i-1,j-1)**
 5. else if $B[i,j]=“↑”$
 6. **then StructureSequence(B,i-1,j)**
 7. **else StructureSequence(B,i,j-1)**

标记函数的实例

- 输入： $X = \langle A, B, C, B, D, A, B \rangle$
 $Y = \langle B, D, C, A, B, A \rangle$

X \ Y	1	2	3	4	5	6
1	B[1,1]=↑	B[1,2]=↑	B[1,3]=↑	B[1,4]=↖	B[1,5]=←	B[1,6]=↖
2	B[2,1]=↖	B[2,2]=←	B[2,3]=←	B[2,4]=↑	B[2,5]=↖	B[2,6]=←
3	B[3,1]=↑	B[3,2]=↑	B[3,3]=↖	B[3,4]=←	B[3,5]=↑	B[3,6]=↑
4	B[4,1]=↑	B[4,2]=↑	B[4,3]=↑	B[4,4]=↑	B[4,5]=↖	B[4,6]=←
5	B[5,1]=↑	B[5,2]=↑	B[5,3]=↑	B[5,4]=↑	B[5,5]=↑	B[5,6]=↑
6	B[6,1]=↑	B[6,2]=↑	B[6,3]=↑	B[6,4]=↖	B[6,5]=↑	B[6,6]=↖
7	B[7,1]=↑	B[7,2]=↑	B[7,3]=↑	B[7,4]=↑	B[7,5]=↑	B[7,6]=↑

解： $X[2], X[3], X[4], X[6]$, 即B, C, B, A

算法的时空复杂度

- 计算优化函数和标记函数
 - 赋初值，为 $O(m)+O(n)$
 - 计算优化、标记函数迭代次 $\Theta(mn)$ ，循环体内常数运算
 - ∴ 总的时间复杂度为 $\Theta(mn)$
- 构造解（追踪）
 - 每步缩小X或Y的长度，时间 $\Theta(m+n)$

算法时间复杂度： $\Theta(mn)$

空间复杂度： $\Theta(mn)$



小结

- 最长公共子序列问题的建模
- 子问题边界的界定
- 递推方程及初值，优化原则判定
- 伪码
- 标记函数与解的追踪
- 时空复杂度

背包问题(Knapsack Problem)

- 一个旅行者随身携带一个背包。可以放入背包的物品有 n 种，每种物品的重量和价值分别为 w_i, v_i 。如果背包的最大承重限制是 b ，每种物品可以放多个。怎么样选择放入背包的物品使得背包所装物品**价值最大**？
- 不妨设上述 w_i, v_i, b 都是正整数。

实例： $n=4, b=10$

$$v_1=1, v_2=3, v_3=5, v_4=9$$

$$w_1=2, w_2=3, w_3=4, w_4=7$$





建模

- 解是 $\langle x_1, x_2, \dots, x_n \rangle$ ，其中 x_i 是装入背包的第 i 种物品个数

目标函数 $\max \sum_{i=1}^n v_i x_i$

约束条件 $\sum_{i=1}^n w_i x_i \leq b, \quad x_i \in N$

线性规划问题： 由线性条件约束的线性函数取最大或最小的问题

整数规划问题： 线性规划问题的变量 x_i 都是非负整数



子问题界定和计算顺序

- **子问题界定：** 由参数 k 和 y 界定
 - k ：考虑物品 $1, 2, \dots, k$ 的选择
 - y ：背包总重量不超过 y
- 原始输入： $k=n, y=b$
- **子问题计算顺序：**
 - $k=1, 2, \dots, n$
 - 对于给定的 $k, y=1, 2, \dots, b$

优化函数的递推方程

$F_k(y)$: 装前 k 种物品, 总重不超过 y
背包达到的**最大价值**

$$F_k(y) = \max\{F_{k-1}(y), F_k(y-w_k)+v_k\}$$

$$F_0(y) = 0, 0 \leq y \leq b, F_k(0) = 0, 0 \leq k \leq n$$

$$F_1(y) = \left\lfloor \frac{y}{w_1} \right\rfloor v_1, \quad F_k(y) = -\infty \quad y < 0$$

标记函数

$i_k(y)$: 装前 k 种物品，总重不超过 y ，背包达到最大价值时装入物品的**最大标号**

$$i_k(y) = \begin{cases} i_{k-1}(y) & F_{k-1}(y) > F_k(y - w_k) + v_k \\ k & F_{k-1}(y) \leq F_k(y - w_k) + v_k \end{cases}$$

$$i_1(y) = \begin{cases} 0 & y < w_1 \\ 1 & y \geq w_1 \end{cases}$$

动态规划求解背包问题

输入: $n=4, b=10$

$v_1=1, v_2=3, v_3=5, v_4=9$

$w_1=2, w_2=3, w_3=4, w_4=7$



$F_k(y)$ 的计算表如下:

$k \backslash y$	1	2	3	4	5	6	7	8	9	10
1										
2										
3										
4										

动态规划求解背包问题

输入: $n=4, b=10$

$v_1=1, v_2=3, v_3=5, v_4=9$

$w_1=2, w_2=3, w_3=4, w_4=7$



$F_k(y)$ 的计算表如下:

$k \backslash y$	1	2	3	4	5	6	7	8	9	10
1	0	1	1	2	2	3	3	4	4	5
2	0	1	3	3	4	6	6	7	9	9
3	0	1	3	5	5	6	8	10	10	11
4	0	1	3	5	5	6	9	10	10	12

输入: $n=4, b=10$

$v_1=1, v_2=3, v_3=5, v_4=9$

$w_1=2, w_2=3, w_3=4, w_4=7$

追踪解

$i_k(y)=$

$k \backslash y$	1	2	3	4	5	6	7	8	9	10
1	0	1	1	1	1	1	1	1	1	1
2	0	1	2	2	2	2	2	2	2	2
3	0	1	2	3	3	3	3	3	3	3
4	0	1	2	3	3	3	4	3	4	4

$$i_4(10)=4 \rightarrow x_4 \geq 1$$

$$i_4(10 - w_4) = i_4(3) = 2 \rightarrow x_2 \geq 1, x_4 = 1, x_3 = 0$$

$$i_2(3 - w_2) = i_2(0) = 0 \rightarrow x_2 = 1, x_1 = 0$$

解: $x_1=0, x_2=1, x_3=0, x_4=1$, 价值12