

动态规划-矩阵连乘-凸多边形最优三角剖分

likai

2021 年 7 月 27 日

1 基础理论

假设: $A[i:j], 1 \leq i < j \leq n$ 表示矩阵 $A_i * A_{i+1} * \dots * A_j$, $m[i][j]$ 表示矩阵 $A_i * A_{i+1} * \dots * A_j$ 连乘时需要执行的乘法次数, 那么求解矩阵连乘的最优目标转换为求解 $\min(m[i][j])$ 。

1) 当 $i = j, A[i:i] = A_i, m[i][i] = 0$

2) 当 $i < j$, 对于 $\forall k (i \leq k < j)$, 如果有, $m[i][j] < m[i][k] + m[k+1][j]$, 那么 $\min(m[i][j]) = m[i][k] + m[k+1][j] + p_{i-1} * p_k * p_j$, 其中 $p_{i-1} * p_k * p_j$ 为 $A[i:k] * A[k+1:j]$ 相乘时所执行的乘法次数。

因此

$$m[i][j] = \begin{cases} 0 & i \geq j \\ \min(m[i][k] + m[k+1][j] + p_{i-1} * p_k * p_j) & i < k \leq j \end{cases} \quad (1)$$

其中, $p_{i-1} * p_k * p_j$ 的含义表示为 $A[i:k] * A[k+1:j]$ 的运算次数, 且如果 $m[i][j]$ 为最优, 那么 $m[i][j-1]$ 也为最优。证明: $m[i][j] = \min(m[i][j-1] + m[j][j] + p_{i-1} * p_i * p_j)$, 如果存在 $m[i][j-1] = best$ 那么就有 $best + p_{i-1} * p_i * p_j < m[i][j]$ 这与 $m[i][j]$ 是最小值矛盾。故 (1) 式是矩阵连乘的最优子结构。

2 算法分析设计

1) 求解最优子结构

input: $m = [(row, col), (row, col), (row, col)]$

```

def getMAndS(m):
    #初始化m矩阵s,
    M=[[0 for i in range(len(m))] for i in range(len(m))]
    #初始化最优k值矩阵
    s=[[0 for i in range(len(m))] for i in range(len(m))]
    '''先求解子结构,子结构的步长首先由相邻逐步扩展至更远, 因为m[1][3]
    的计算过程中需要使用到m[1][2],m[2][3]'''
    for r in range(1,len(m)):# r表示步长
        for start in range(len(m)-r):#start 表示当前开始位置
            end=r+i#end 表示当前结束位置
            M[start][end]=M[start+1][end]
            +m[start][0]*m[start][1]*m[end][1]
            S[start][end]=start
            #计算最优断点
            for k in range(i+1,j):
                t=M[start][k]+M[k+1][end]
                +m[start][0]*m[k][1]*m[end][1]
                if t < M[start][end]:
                    M[start][end]=t
                    S[start][end]=k
    return S

```

2) 递归求解原问题 $A[1:n]$, 因为

$$A[1:n] = A[1:s[1][n]] * A[s[1][n]+1:n]$$

$$A[1:s[1][n]] = A[1:s[1][s[1][n]]] * A[s[1][s[1][n]]+1:s[1][s[1][n]]]$$

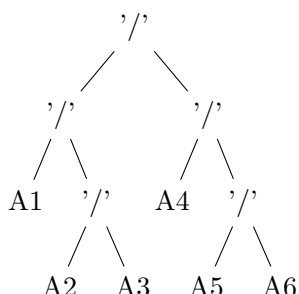
所以构建的递归算法如下:

```

def trick(start,end,s):
    if start>=end:
        return None
    trick(start,s[start][end],s)
    trick(s[start][end]+1,end,s)
    print('(A[',start,',':',s[start][end],')')
    print('(A[',s[start][end]+1,',':',end,')')

```

3 语法树



该语法树对应的则是 $A_1 * A_2 * A_3 * A_4 * A_5 * A_6$ 一种加括号的矩阵连乘方式 $(A_1 * (A_2 * A_3)) * (A_4 * (A_5 * A_6))$ 。

4 凸多边形三角剖分

给定凸多边形的顶点序列 $P = \{v_0, v_1, \dots, v_n\}$, 其中 $v_n = v_0$, 若有 $i < j - 1$, 则称 $v_i v_j$ 为凸多边形的一条弦, 则每一条弦将凸多边形划分为两个多边形, 顶点序列分别为 $P_{ij} = \{v_i, \dots, v_j\}$, $P_{jn} = \{v_j, \dots, v_i\}$

凸多边形三角剖分问题的描述, 将多边形分割成互不相交的三角形的弦的集合。最优凸多边形三角剖分, 弦最多, 互不相交, 由弦和边组成的三角形的权重之和最小。

通过语法树, 可以将凸多边形的三角剖分转换为矩阵连乘问题。

- 1) 多边形的边对应一个矩阵;
- 2) 单条边的权重等于 0;
- 3) 两条相邻边的权重等于 $p_{i-1} * p_i * p_k$;
- 4) 每条弦表示一个小括号

由此 $n + 1$ 条边的最优凸多边形的三角剖分问题则转换为输出 n 个矩阵连乘的最优加括号方式。