

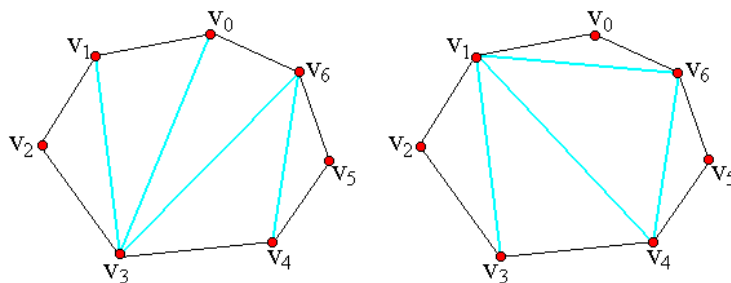


凸多边形最优三角剖分

凸多边形最优三角剖分

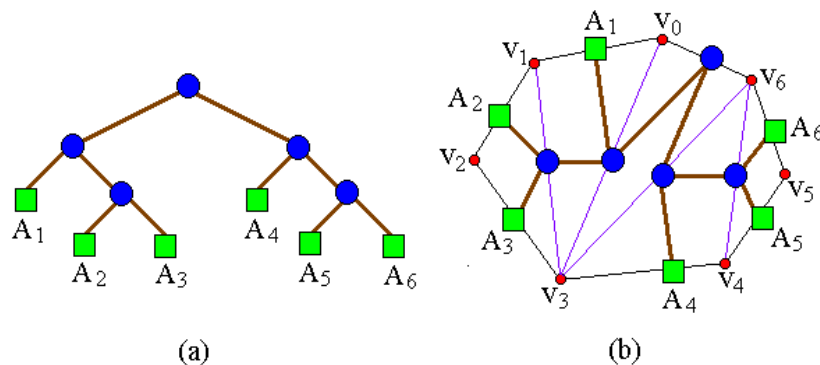
- 用多边形顶点的逆时针序列表示凸多边形，即 $P = \{v_0, v_1, \dots, v_{n-1}\}$ 表示具有 n 条边的凸多边形。
- 若 v_i 与 v_j 是多边形上不相邻的2个顶点，则线段 $v_i v_j$ 称为多边形的一条弦。弦将多边形分割成2个多边形 $\{v_i, v_{i+1}, \dots, v_j\}$ 和 $\{v_j, v_{j+1}, \dots, v_i\}$ 。
- **多边形的三角剖分**是将多边形分割成互不相交的三角形的弦的集合 T 。

给定凸多边形 P ，以及定义在由多边形的边和弦组成的三角形上的权函数 w 。要求确定该凸多边形的三角剖分，使得即该三角剖分中诸三角形上权之和为最小。



三角剖分的结构及其相关问题

- 一个表达式的完全加括号方式相应于一棵完全二叉树，称为表达式的语法树。例如，完全加括号的矩阵连乘积 $((A_1(A_2A_3))(A_4(A_5A_6)))$ 所相应的语法树如图 (a) 所示。
- 凸多边形 $\{v_0, v_1, \dots, v_{n-1}\}$ 的三角剖分也可以用语法树表示。例如，图 (b) 中凸多边形的三角剖分可用图 (a) 所示的语法树表示。
- 矩阵连乘积中的每个矩阵 A_i 对应于凸 $(n+1)$ 边形中的一条边 $v_{i-1}v_i$ 。三角剖分中的一条弦 v_iv_j , $i < j$, 对应于矩阵连乘积 $A[i+1:j]$ 。





最优子结构性质

- 凸多边形的最优三角剖分问题有最优子结构性质。
- 事实上，若凸 $(n+1)$ 边形 $P=\{v_0, v_1, \dots, v_n\}$ 的最优三角剖分 T 包含三角形 $v_0 v_k v_n$ ， $1 \leq k \leq n-1$ ，则 T 的权为3个部分权的和：三角形 $v_0 v_k v_n$ 的权，子多边形 $\{v_0, v_1, \dots, v_k\}$ 和 $\{v_k, v_{k+1}, \dots, v_n\}$ 的权之和。可以断言，由 T 所确定的这2个子多边形的三角剖分也是最优的。因为若有 $\{v_0, v_1, \dots, v_k\}$ 或 $\{v_k, v_{k+1}, \dots, v_n\}$ 的更小权的三角剖分将导致 T 不是最优三角剖分的矛盾。



最优三角剖分的递归结构

- 定义 $t[i][j]$, $1 \leq i < j \leq n$ 为凸子多边形 $\{v_{i-1}, v_i, \dots, v_j\}$ 的最优三角剖分所对应的权函数值, 即其最优值。为方便起见, 设退化的多边形 $\{v_{i-1}, v_i\}$ 具有权值 0。据此定义, 要计算的凸 $(n+1)$ 边形 P 的最优权值为 $t[1][n]$ 。
- $t[i][j]$ 的值可以利用最优子结构性性质递归地计算。
 - 当 $j-i \geq 1$ 时, 凸子多边形至少有 3 个顶点。由最优子结构性性质, $t[i][j]$ 的值应为 $t[i][k]$ 的值加上 $t[k+1][j]$ 的值, 再加上三角形 $v_{i-1}v_kv_j$ 的权值, 其中 $i \leq k \leq j-1$ 。
 - 由于在计算时还不知道 k 的确切位置, 而 k 的所有可能位置只有 $j-i$ 个, 因此可以在这 $j-i$ 个位置中选出使 $t[i][j]$ 值达到最小的位置。由此, $t[i][j]$ 可递归地定义为:

$$t[i][j] = \begin{cases} 0 & i = j \\ \min_{i \leq k < j} \{t[i][k] + t[k+1][j] + w(v_{i-1}v_kv_j)\} & i < j \end{cases}$$



多边形游戏



多边形游戏

多边形游戏是一个单人玩的游戏，开始时有一个由 n 个顶点构成的多边形。每个顶点被赋予一个整数值，每条边被赋予一个运算符“+”或“*”。所有边依次用整数从1到 n 编号。

游戏第1步，将一条边删除。

随后 $n-1$ 步按以下方式操作：

- (1)选择一条边 E 以及由 E 连接着的2个顶点 V_1 和 V_2 ；
- (2)用一个新的顶点取代边 E 以及由 E 连接着的2个顶点 V_1 和 V_2 。将由顶点 V_1 和 V_2 的整数值通过边 E 上的运算得到的结果赋予新顶点。
- (3)所有边都被删除，游戏结束。游戏的得分就是所剩顶点上的整数值。

问题: 对于给定的多边形，计算最高得分。



最优子结构性质

- 在所给多边形中，从顶点 i ($1 \leq i \leq n$) 开始，长度为 j (链中有 j 个顶点) 的顺时针链 $p(i, j)$ 可表示为 $v[i], op[i+1], \dots, v[i+j-1]$ 。
- 如果这条链的最后一次合并运算在 $op[i+s]$ 处发生 ($1 \leq s \leq j-1$)，则可在 $op[i+s]$ 处将链分割为2个子链 $p(i, s)$ 和 $p(i+s, j-s)$ 。
- 设 $m1$ 是对子链 $p(i, s)$ 的任意一种合并方式得到的值，而 a 和 b 分别是在所有可能的合并中得到的最小值和最大值。
- $m2$ 是 $p(i+s, j-s)$ 的任意一种合并方式得到的值，而 c 和 d 分别是在所有可能的合并中得到的最小值和最大值。
- 依此定义有 $a \leq m1 \leq b, c \leq m2 \leq d$
 - (1) 当 $op[i+s] = '+'$ 时，显然有 $a+c \leq m \leq b+d$
 - (2) 当 $op[i+s] = '*'$ 时，有 $\min\{ac, ad, bc, bd\} \leq m \leq \max\{ac, ad, bc, bd\}$
- 换句话说，主链的最大值和最小值可由子链的最大值和最小值得到。



图像压缩



图像压缩

图象的变位压缩存储格式将所给的象素点序列 $\{p_1, p_2, \dots, p_n\}$, $0 \leq p_i \leq 255$ 分割成 m 个连续段 S_1, S_2, \dots, S_m 。第 i 个象素段 S_i 中 ($1 \leq i \leq m$), 有 $l[i]$ 个象素, 且该段中每个象素都只用 $b[i]$ 位表示。设 $t[i] = \sum_{k=1}^{i-1} l[k]$, 则第 i 个象素段 S_i 为

$$S_i = \{p_{t[i]+1}, \dots, p_{t[i]+l[i]}\}$$

设 $h_i = \left\lceil \log \left(\max_{t[i]+1 \leq k \leq t[i]+l[i]} p_k + 1 \right) \right\rceil$, 则 $h_i \leq b[i] \leq 8$ 。因此需要用3位表示 $b[i]$, 如果限制 $1 \leq l[i] \leq 255$, 则需要用8位表示 $l[i]$ 。因此, 第 i 个象素段所需的存储空间为 $l[i] * b[i] + 11$ 位。按此格式存储象素序列 $\{p_1, p_2, \dots, p_n\}$, 需要 $\sum_{i=1}^m l[i] * b[i] + 11m$ 位的存储空间。

图象压缩问题要求确定象素序列 $\{p_1, p_2, \dots, p_n\}$ 的最优分段, 使得依此分段所需的存储空间最少。每个分段的长度不超过256位。



图像压缩

设 $l[i], b[i], 1 \leq i \leq m$, 是 $\{p_1, p_2, \dots, p_n\}$ 的最优分段。显而易见, $l[1], b[1]$ 是 $\{p_1, \dots, p_{l[1]}\}$ 的最优分段, 且 $l[i], b[i], 2 \leq i \leq m$, 是 $\{p_{l[1]+1}, \dots, p_n\}$ 的最优分段。即图象压缩问题满足最优子结构性质。

设 $s[i], 1 \leq i \leq n$, 是象素序列 $\{p_1, \dots, p_i\}$ 的最优分段所需的存储位数。由最优子结构性质易知:

$$s[i] = \min_{1 \leq k \leq \min\{i, 256\}} \{s[i-k] + k * b \max(i-k+1, i)\} + 11$$

其中 $b \max(i, j) = \left\lceil \log \left(\max_{i \leq k \leq j} \{p_k\} + 1 \right) \right\rceil$

算法复杂度分析:

由于算法中对 k 的循环次数不超过 256, 故对每一个确定的 i , 可在时间 $O(1)$ 内完成的计算。因此整个算法所需的计算时间为 $O(n)$ 。



流水作业调度



流水作业调度

n 个作业 $\{1, 2, \dots, n\}$ 要在由2台机器 $M1$ 和 $M2$ 组成的流水线上完成加工。每个作业加工的顺序都是先在 $M1$ 上加工，然后在 $M2$ 上加工。 $M1$ 和 $M2$ 加工作业 i 所需的时间分别为 a_i 和 b_i 。

流水作业调度问题要求确定这 n 个作业的最优加工顺序，使得从第一个作业在机器 $M1$ 上开始加工，到最后一个作业在机器 $M2$ 上加工完成所需的时间最少。

分析：

- 直观上，一个最优调度应使机器 $M1$ 没有空闲时间，且机器 $M2$ 的空闲时间最少。在一般情况下，机器 $M2$ 上会有机器空闲和作业积压2种情况。
- 设全部作业的集合为 $N=\{1, 2, \dots, n\}$ 。 $S \subseteq N$ 是 N 的作业子集。在一般情况下，机器 $M1$ 开始加工 S 中作业时，机器 $M2$ 还在加工其它作业，要等时间 t 后才可利用。将这种情况下完成 S 中作业所需的最短时间记为 $T(S, t)$ 。流水作业调度问题的最优值为 $T(N, 0)$ 。



流水作业调度

设 π 是所给 n 个流水作业的一个最优调度，它所需的加工时间为 $a_{\pi(1)}+T'$ 。其中 T' 是在机器 M_2 的等待时间为 $b_{\pi(1)}$ 时，安排作业 $\pi(2), \dots, \pi(n)$ 所需的时间。

记 $S=N-\{\pi(1)\}$ ，则有 $T'=T(S, b_{\pi(1)})$ 。

证明：事实上，由 T 的定义知 $T' \geq T(S, b_{\pi(1)})$ 。若 $T' > T(S, b_{\pi(1)})$ ，设 π' 是作业集 S 在机器 M_2 的等待时间为 $b_{\pi(1)}$ 情况下的一个最优调度。则 $\pi(1), \pi'(2), \dots, \pi'(n)$ 是 N 的一个调度，且该调度所需的时间为 $a_{\pi(1)}+T(S, b_{\pi(1)}) < a_{\pi(1)}+T'$ 。这与 π 是 N 的最优调度矛盾。故 $T' \leq T(S, b_{\pi(1)})$ 。从而 $T'=T(S, b_{\pi(1)})$ 。这就证明了流水作业调度问题具有最优子结构的性质。

由流水作业调度问题的最优子结构性知，

$$T(N, 0) = \min_{1 \leq i \leq n} \{a_i + T(N - \{i\}, b_i)\}$$

$$T(S, t) = \min_{i \in S} \{a_i + T(S - \{i\}, b_i + \max\{t - a_i, 0\})\}$$



Johnson不等式

对递归式的深入分析表明，算法可进一步得到简化。
设 π 是作业集 S 在机器 M_2 的等待时间为 t 时的任一最优调度。
若 $\pi(1)=i$ ， $\pi(2)=j$ 。则由动态规划递归式可得：

$$\begin{aligned} T(S, t) &= a_i + T(S - \{i\}, b_i + \max\{t - a_i, 0\}) \\ &= a_i + a_j + T(S - \{i, j\}, t_{ij}) \end{aligned}$$

其中，

$$\begin{aligned} t_{ij} &= b_j + \max\{b_i + \max\{t - a_i, 0\} - a_j, 0\} \\ &= b_j + b_i - a_j + \max\{\max\{t - a_i, 0\}, a_j - b_i\} \\ &= b_j + b_i - a_j + \max\{t - a_i, a_j - b_i, 0\} \\ &= b_j + b_i - a_j - a_i + \max\{t, a_i + a_j - b_i, a_i\} \end{aligned}$$

如果作业 i 和 j 满足 $\min\{b_i, a_j\} \geq \min\{b_j, a_i\}$ ，则称作业 i 和 j 满足Johnson不等式。



流水作业调度的Johnson法则

$$t_{ij} = b_j + b_i - a_j - a_i + \max\{t, a_i + a_j - b_i, \textcolor{red}{a_i}\}$$

交换作业i和作业j的加工顺序，得到作业集S的另一调度，它所需的加工时间为 $T'(S, t) = a_i + a_j + T(S - \{i, j\}, t_{ji})$

其中， $t_{ji} = b_j + b_i - a_j - a_i + \max\{t, a_i + a_j - b_j, \textcolor{red}{a_j}\}$

当作业i和j满足Johnson不等式时，有

$$\max\{-b_i, -a_j\} \leq \max\{-b_j, -a_i\}$$

$$a_i + a_j + \max\{-b_i, -a_j\} \leq a_i + a_j + \max\{-b_j, -a_i\}$$

$$\max\{a_i + a_j - b_i, a_i\} \leq \max\{a_i + a_j - b_j, a_j\}$$

$$\max\{t, a_i + a_j - b_i, a_i\} \leq \max\{t, a_i + a_j - b_j, a_j\}$$

由此可见当作业i和作业j不满足Johnson不等式时，交换它们的加工顺序后，不增加加工时间。对于流水作业调度问题，必存在最优调度 π ，使得作业 $\pi(i)$ 和 $\pi(i+1)$ 满足Johnson不等式。进一步还可以证明，调度满足Johnson法则当且仅当对任意 $i < j$ 有

$$\min\{b_{\pi(i)}, a_{\pi(j)}\} \geq \min\{b_{\pi(j)}, a_{\pi(i)}\}$$

由此可知，所有满足Johnson法则的调度均为最优调度。



算法描述

流水作业调度问题的Johnson算法

- (1) 令 $N_1 = \{i \mid a_i < b_i\}$, $N_2 = \{i \mid a_i \geq b_i\}$;
- (2) 将 N_1 中作业依 a_i 的非减序排序; 将 N_2 中作业依 b_i 的非增序排序;
- (3) N_1 中作业接 N_2 中作业构成满足Johnson法则的最优调度。

算法复杂度分析:

算法的主要计算时间花在对作业集的排序。因此, 在最坏情况下算法所需的计算时间为 $O(n \log n)$ 。所需的空间为 $O(n)$ 。