基于许可图的双机任务调度问题

20071116 高义雄

1 问题概述

1.1 问题模型

有 n 个待安排完成的任务，第 i 个任务需要的时间是 p_i 。

有两台机器，每个任务都需要被连续地安排到某一个机器上完成。

将任务抽象成点，点权为执行任务需要的时间，给定一个许可图 $G=(J,E)$ ，对于两个任务：

1. 在许可图中有边相连，则可以分别在两个机器上同时执行

2. 在许可图中无边相连，执行的时间段必须不能重合。

求一个安排方案，最小化最后一个被做完的任务完成时间。

1.2 现有结论

1. 在许可图是树的情况下，求解此问题的最优解是 **NP-Hard** 的。

2. 在许可图是毛毛虫的情况下，存在 $\mathcal{O}(n)$ 求解优解的算法。

3. 在许可图是环的情况下，存在 $\mathcal{O}(n^2)$ 求解优解的算法。

2 毛毛虫情况下的求解

2.1 定义与符号

毛毛虫（Caterpillars）是一种特殊的树，由一个主干和若干到主干距离为 1 的叶子构成。

定义其顺序 L 为：从左往右依次扫描主干，先加入主干点，再加入连接在该点上的叶子。

例如，图 1 中的顺序 $L=\{a,b,c,d,e,f,g,h,i,j,k,l,m,n,o\}$

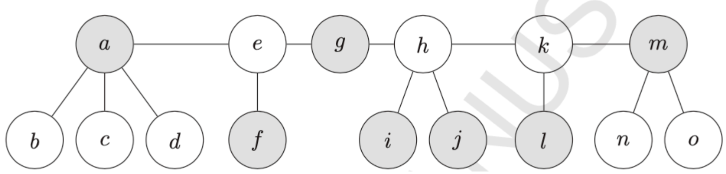

图 1: 一个毛毛虫形许可图示例

含义	符号
任务 i 执行需要的时间	p_i
图 G 的最大权独立集	$I_p(G)$ or S^*
图 G 的最大权独立集的权值和	$\overline{I_p}(G)=\sum_{j\in I_p(G)} p_j$
与点 j 相连的点集	$N(j)$
与集合 J' 相连的点集	$N(J')$
与点 j 相连的叶子集合	$Lv(j)$
任务 j 的开始时间	t_j
任务集合 J' 中最早的开始时间	$t_j(J')=\min_{k\in J'}\{t_k\}$

2.2 求解算法

输入： 毛毛虫形许可图 $CAT=(J,E)$ ，每个任务执行需要的时间 $p_j,\forall j\in J$

输出： 最优求解方案 σ

1. 求解 CAT 的最大权独立集 S^*
2. 移除不在 S^* 中的任意两点之间的边
3. 图变为若干个新的毛毛虫 $CAT_i=(J_i,E_i)$
4. 对于每个新的毛毛虫 CAT_i ，构建其最优方案 σ_i ：
   5. 取 $S_i^*=J_i\cap S^*$
   6. 按照 L_i 的顺序依次安排任务:
   7. 如果任务 $j\in S_i^*$ ，将其安排在机器 1 ，紧接着前一个任务 åå
   8. 如果任务 $j\notin S_i^*$ ，将其安排在机器 2 ，安排在可能的最靠前的位置
9. 将 σ_i 拼接起来得到最优方案 σ

2.3 最优性证明

首先 $\overline{I_p}(G)$ 是问题的下界，我们的目标是证明算法的结果恰好等于 $\overline{I_p}(G)$ ，即证明机器 1 上最后的任务完成时间总是不短于机器 2 上的任务即可。

为了简单描述，我们称在最大权独立集里的点为黑点，其余点为白点。

可以发现断掉白点之间的所有边后，这个新的图的性质是所有的边都是黑-白边。

事实 1. 对于每个新的连通块 CAT_i ，其内的黑点集 S_i^* 仍然是 CAT_i 的最大权独立集

证明： 假设存在其他的 MWIS $I_p(CAT_i)$ 使得 $\overline{I_p}(CAT_i)>\sum_{j\in S_i^*} p_j$ ，那么我们考虑将 CAT_i 这一部分的 MWIS 换成这个新的集合，其他部分的 MWIS 不变，那么还原回仍是原图的一个 IS，而这个新的 IS 比原来的 MWIS 权值还大，所以矛盾了。

$$\sum_{j\in S'} p_j=\overline{I_p}(CAT)-\sum_{j\in S_i^*} p_j+\overline{I_p}\left(CAT_i\right)>\overline{I_p}(CAT)$$

事实 2. 对于 J_i 内任何一个白点子集，其点权和不会超过其邻居黑点的点权和

证明： 假设存在这样的一个白点集 W 满足 $\sum_{j\in W} p_j > \sum_{j\in N(W)} p_j$ ，那么考虑将 S_i^* 换成 $S' = (S_i^* \setminus N(W)) \cup W$ ，易证 S' 也是一个独立集，且比 S_i^* 权值和还要大，矛盾。

$$\sum_{j\in S'} p_j = \sum_{j\in S_i^*} p_j - \sum_{j\in N(W)} p_j + \sum_{j\in W} p_j > \sum_{j\in S_i^*} p_j$$

事实 3. 对于任意白点 β ，其邻居黑点都会被连续地安排在第一个机器上。

证明： 分类讨论：

1. 如果 β 是叶子，则只有一个邻居；
2. 如果 β 不是叶子，假设链上的顺序是 $\alpha-\beta-\gamma$ ，那么黑点顺序是 $\alpha-$ (β 的所有叶子) $-\gamma$ 。

事实 4. 对于任意两个白点 α,β ，如果被连续地安排在了某一个机器上，那么他们一定有公共邻居。

证明： 分类讨论:

1. 若 α 和 β 都是叶子，则两点必然挂在同一个黑点（公共邻居）上，否则不会相邻；
2. 若 α 和 β 一个叶子一个主干，则叶子必定挂在主干上的点的某个邻居黑点上。

事实 5. 对于任意连续安排的白点集，其邻居一定是被连续安排在一个区间内的。

证明： 事实 3 与事实 4 的自然推论。

引理 1. 每一个白点都会被安排在邻居对应的区间里。

证明： 反证法，不符合的就两种情况：

1. $t_\beta < t(N(\beta))$ ： 这种情况不存在，因为算法中每个黑点是连续安排的，如果出现该情况，这个白点会与非邻接的黑点重合，与许可图的要求相冲突。

2. $t_\beta + p_\beta > t(N(\beta)) + \sum_{j\in N(\beta)} p_j$ ： 这种情况不存在，考虑从 β 往前的第一个满足 $t_\alpha = t_{N(\alpha)}$ 的任务 α ，那么从 α 到 β 这一段是连续安排的，由事实 5 ，连续安排的白点集，其邻居一定是被连续安排在一个区间内的，因此白点的区间就是 $[t_\alpha, t_\alpha + \sum_{j\in[\alpha,\beta]} p_j]$ ，黑点的区间就是 $[t(N([\alpha,\beta])), t(N([\alpha,\beta])) + \sum_{j\in N([\alpha,\beta])} p_j]$；又由事实 2 ，对于 J_i 内任何一个白点子集，其点权和不会超过其邻居黑点的点权和，因此有 $t_\alpha + \sum_{j\in[\alpha,\beta]} p_j \le \sum_{j\in N([\alpha,\beta])} p_j$ ，因此 $[\alpha,\beta]$ 这一段的白点终止时间不超过黑点，因此作为最后一个完成的白点 β ，有 $t_\beta + p_\beta \le t(N(\beta)) + \sum_{j\in N(\beta)} p_j$

定理 2. 本算法求出的安排方案为最优解。

证明： 由引理 1 ，每个 σ_i 所需要的时间就是其中黑点所需的时间，即 $\overline{I_p}(CAT_i)$ ，因此总方案 σ 所需的时间 $\sum_i \overline{I_p}(CAT_i) = \overline{I_p}(CAT)$ ，即答案下界。

2.4 复杂度证明

1. 求解最大权独立集，使用动态规划算法，复杂度 $\mathcal{O}(n)$

2. 移除所有的白-白边，只需扫描所有边，因为毛毛虫是特殊的树，因此复杂度为 $\mathcal{O}(n)$

3. 制定序列 L ，只需从左往右依次考虑，每个点只会被扫描一次，因此复杂度为 $\mathcal{O}(n)$

4. 安排任务，只需依次扫描 L_i ，每个点只会被安排一次，因此复杂度为 $\mathcal{O}(n)$

5. 合并安排，因为是直接拼接，复杂度 $\mathcal{O}(n)$

综上，算法复杂度为 $\mathcal{O}(n)$

2.5 扩展与推论

1. 许可图为**一条路径**、**星状图**等均为毛毛虫的特殊情况，均可用此算法求解，复杂度是 $\mathcal{O}(n)$

2. 许可图为**环**，考虑安排是线性的，必定有一个许可边未被使用，因此枚举这条边是谁，然后问题就变成了一条路径，可用此算法求解，复杂度是 $\mathcal{O}(n^2)$

3 待解决的问题

1. 对于许可图为树的情况，证明求解最优解为 **NP-Hard** 的

2. 对于许可图为树的情况，给出低于 $\frac{3}{2}OPT$ 的近似算法