《火花》解题报告

李秉霈
长沙市长郡中学

目录

1　题目大意

- 有一棵 n 个点、根为 1 的树。第 i 个点上有 c_i 个物品，权值均为 v_i。
- 需要先选择 t 个不同的点，对于选的每个点，在它的每个祖先（包括自己）上各取一个物品。
  - **本题保证 $t < c_i$，因此不会存在没有物品可取的情况**。
- 接下来可以取至多 k 个物品，要求所有在这一步被取了物品的点形成一个含根的连通块。
- 最大化整个过程中取的物品权值之和。

2　数据范围

对于所有数据：$1 \le n, k \le 10^4$，$0 \le t \le n$，$\boldsymbol{t < c_i} \le 10^9$，$1 \le v_i \le 10^9$，$nk(t+1) \le 10^7$。

- 子任务 1（10 分）：$n, k, t \le 10$。
- 子任务 2（20 分）：$t = 0$。
- 子任务 3（15 分）：$nk(t+1) \le 10^6$。
- 子任务 4（25 分）：$nk(t+1) \le 5 \times 10^6$。
- 子任务 5（30 分）：无特殊限制。

3 解题过程

3.1 子任务 1

各种高复杂度暴力均可通过。

3.2 子任务 2

$t=0$ 时，问题即为经典的“树上依赖背包”（可参见参考资料）。

在这里我们简述一下该方法。令 $d_{1\cdots n}$ 为原树的任意 DFS 序。考虑按该顺序决策每个点被取的物品数，假设现在考虑到了 d_i：

- 若 d_i 被取了至少一个物品，直接 $i \leftarrow i+1$ 即可。
- 否则，d_i 子树内的点也被强制取 0 个，于是 $i \leftarrow i + siz_{d_i}$。

对上述过程设计 dp：设 $f_{i,j}$ 表示考虑到 d_i，现在还可以至多取 j 个物品，最大的总权值是多少。转移：

1. $f_{i,j} \to f_{i+siz_{d_i},j}$；
2. $f_{i,j} + kv_{d_i} \to f_{i+1,j-k}$，其中 $k \in [1, \min(j, c_{d_i})]$。

对第二类转移使用单调队列优化。总时间复杂度 $\mathcal{O}(nk)$。

3.3 子任务 3

我们希望同时决策每个点是否在一开始的 t 个点中（记为 $g_i = 0/1$）和被取物品的次数。

考虑拓展 $t=0$ 的做法。现在每个点的限制是，被取的次数加子树 g 之和不超过 c_i。这启发我们在欧拉序上考虑决策过程。

欧拉序是指，在 DFS 一棵树的过程中，在进入节点 i 和从节点 i 回溯时各记录一次 i 得到的序列。记 $e_{1\cdots 2n}$ 为任意欧拉序，(ℓ_i, r_i) 分别为该欧拉序上 i 第一次和第二次出现的位置（对应进入 / 回溯）。

我们考虑在 ℓ_i 处决策 i 被取的次数，并在 r_i 处决策 g_i。现在的问题是：在我们决策 i 被取的次数时，其上界是受子树 g 之和影响的，而子树的 g 在之后才被决策。

首先我们先研究当 i 被取了至少一个物品时，该问题怎么解决。考虑这样的想法：分步完成 i 被取次数的决策。

- 首先，先在 i 上取走一个物品来强制满足条件。

- 设 t' 为未来会有多少个 $g=1$。注意到子树 g 之和 $\le t' < c_i$。因此接下来在 ℓ_i 处，我们决策点 i 的前 $c_i - 1 - t'$ 个物品被取了多少个，即预留 t' 个物品用于子树的 g。
- 在 r_i 处，我们完成对没用到的预留物品的决策。容易推出，没用到的物品个数就是 $r_i + 1$ 时刻的 t'。

而 i 没有被取物品时，情况则更加简单：此时只需要决策 i 的子树内选一些点 $g=1$ 即可。设 s_i 表示 i 到根链的 v 之和，那么选择 i 子树内 s 的前若干大显然不劣。

整理上述过程可以写出 dp 转移，设 $f_{i,j,k}$ 表示考虑到 e_i，还能至多取 j 个物品，当前 $t'=k$：

- 若 $i = \ell_{e_i}$：
  - $f_{i,j,k} + xv_{e_i} \to f_{i+1,j-x,k}$，其中 $x \in [1, \min(j, c_{e_i} - k)]$。（$e_i$ 至少取一个物品）
  - 设 $S(u,i)$ 表示 u 子树内前 i 大的 s 之和。$f_{i,j,k} + S(e_i, x) \to f_{r_{e_i}+1,j,k-x}$，其中 $x \in [0,k]$。（e_i 不取物品）
- 否则，$i = r_{e_i}$：
  - $f_{i,j,k} + xv_{e_i} \to f_{i+1,j-x,k}$，其中 $x \in [0, \min(j,k)]$。
  - $f_{i,j,k} + xv_{e_i} + s_{e_i} \to f_{i+1,j-x,k-1}$，其中 $x \in [0, \min(j, k-1)]$。

预处理所有 $S(u,i)$ 可以从下至上合并，相当于跑 $\mathcal{O}(n)$ 次规模 $\mathcal{O}(t)$ 的归并，时间复杂度 $\mathcal{O}(nt)$，可以接受。

除了 $i = \ell_{e_i}$ 中 e_i 不取物品的转移之外，其他转移都可以使用单调队列优化。因此可以做到总时间复杂度 $\mathcal{O}(nkt^2)$。由于 $t \le n$ 且 $nt \le nk(t+1) \le 10^6$，因此有 $t \le 10^3$，可以通过该档子任务。

3.4 子任务 4

我们有如下的树形 dp 做法：设 $f_{i,j,k}$ 表示点 i 的子树中，在第一轮被选了 j 个点，在第二轮一共取了 k 个物品，该子树的最大贡献是多少。

这个 dp 在点 i 上的转移分为两部分：逐步把 i 的所有儿子子树的 dp 合并；决策 i 被取的情况。对于后者，使用与上面类似的单调队列优化即可做到时间复杂度 $\mathcal{O}(kt)$。对于前者，根据 这篇文章，总复杂度可以分析到 $\mathcal{O}(nk^2t)$。

综上，这个做法的时间复杂度为 $\mathcal{O}(nk^2t)$。

设 $N=nkt$，当 $k>N^{1/3}$ 时，有 $t\le\sqrt{nt}\le\sqrt{\frac{N}{k}}\le N^{1/3}$。因此把这个做法和子任务 3 的做法拼起来，即可做到 $\mathcal{O}(N^{4/3})$，可以通过该档子任务。事实上，常数较为优秀的该做法可以直接通过本题。

3.5 子任务 5

考虑直接优化子任务 3 的做法：对于 $i=\ell_{e_i}$ 并且 e_i 不取物品的转移，固定 i,j 后贡献函数 $w(\ell,r)=S(e_i,r-\ell)$ 满足四边形不等式，因此这个转移过程是有决策单调性的。使用经典的分治算法就可以做到 $\mathcal{O}(nkt\log t)$ 并通过本题。

Bonus：使用 SMAWK 算法应该可以做到 $\mathcal{O}(nkt)$，但在本题没必要。

4 参考资料与致谢

参考资料：

- LibreOJ #160. 树形背包 题解。
- JSOI 2025 省集 Round 1 Day 4.5（待更新）

感谢周雨衡同学为本题提供 idea，程楷轩、孙铁铮、张丹诚、肖岱恩、郝心悦、郑悠然同学验题。