北大合宿 2019 Day1 G 問題
トレジャーハンター (Treasure Hunter)

原案: tsutaj
問題文: tsutaj
解答: tsutaj
解説: tsutaj

2019 年 7 月 14 日

問題

トレジャーハンター (Treasure Hunter)

- 各頂点に価値 p_i 、各辺に重み w_i がついた木 $T = (V, E)$ がある
  - 頂点 u と v を結ぶ辺を (u, v)、その重み w を $\mathrm{weight}((u, v)) = w$ と表記
- 与えられた木の部分木 $T' = (V', E'), V' \subseteq V, E' \subseteq E$ であって、以下の条件を満たすものの中で、部分木に含まれる頂点の価値の合計を最大化せよ
  - T' は連結
  - $\sum_{e \in E'} \mathrm{weight}(e) \leq W$
- 制約
  - $1 \leq N \leq 10^2$
  - $1 \leq W \leq 10^5$
  - $1 \leq p_i \leq 10^9$
  - $1 \leq w_i \leq 10^5$
  - 与えられるグラフは木である

想定誤解法

TLE 解法: 普通の木 DP

- 適当に根を決めて根付き木にする
- $\mathrm{dp}[i][j]$ (以下の状態を管理)
  - i 番目の頂点までで
  - 切った辺の重みの総和が j であるときの価値最大
- 親側と子側の情報をマージするとき、重みの総和について 2 重ループが必要
- 全体で $O(NW^2)$ 必要となり TLE
  - map などを用いて値が入ってるところだけ覚えても通らないと思います

想定解法

想定解法: 再帰動的計画法 (DFS しながら DP)

再帰動的計画法

$[X_t, X_f] = \mathrm{recDP}(v, X)$

- 引数 (input)
  - v (頂点 id)
  - X (関数を呼んだ時点でのナップザックの状態を表す配列)
- 返り値 (output)
  - X_t (頂点 v を使用した時のナップザックの状態を表す配列)
  - X_f (頂点 v を使用しなかった時のナップザックの状態を表す配列)

- 適当に根を決めて根付き木にする
- 根から、その時点までのナップザックの状態を持って DFS
  - 子について DFS を呼んで得られた結果をもとに、ナップザックの状態を更新
  - その頂点を使用する / 使用しない ときにできる状態更新遷移が異なるので、それに注意しながら実装

出典

- このテクニックは昨年出た論文 “Linear Pseudo-Polynomial Factor Algorithm for Automaton Constrained Tree Knapsack Problem” Link に記載されているものです
  - 著者である tmaehara 先生が Qiita に解説記事を書いています (さわりの部分を理解するには良い資料) Link
  - これの Example 3 (Connectivity Constrained Problem) を参考に作りました
    - 辺に制約があるのでちょっと工夫しないと答えが合わないので注意
  - 独立集合制約・優先順位制約・連結制約・k 個の連結制約などなど・・・汎用的に使えるテクニックのようです
- これを実装すると $O(N^{\log_2 3}W)$ でこの問題が解け、AC できます
  - 重心分解をすると $O(NW \log W)$ も達成可能のようです
- 去年の ICPC 国内予選 H でも似たような問題が出ているので出してみました

Writer 解・統計

- Writer 解
  - tsutaj (C++・101 行・2862 bytes)
- 統計
  - AC / tried: 2 / 34 (5.9 %)
  - First AC
    - On-site: なし
    - On-line: ushitapunichiakun (126 min 51 sec)