会津合宿 2019 Day3 G 問題
Restricted DFS

原案: tsutaj
問題文: tsutaj
解答: tsutaj・rsk0315
解説: tsutaj

2019 年 9 月 20 日

問題

Restricted DFS

- 木が与えられ、それぞれの頂点 i には整数 A_i が定められている
- 頂点 i から DFS することを考える
  - 頂点番号が若い順に子を見る
  - ある頂点を訪れようとする際に A_i の値が 0 になっていれば、そこで DFS 打ち切り
  - A_i が正ならば、その頂点を訪れ A_i をデクリメントし、次の探索に進む
- それぞれの頂点から DFS をはじめたときのステップ数を求める

制約

- $1 \leq N \leq 2 \times 10^5$
- $0 \leq A_i \leq 10^9$

想定誤解法

- 普通に毎回 DFS をする
  - 1 回の DFS に $O(N)$ かかり、それを各頂点に関してやるので・・・
  - $O(N^2)$ かかり間に合わない
- なんらかの形で値を再利用しないと間に合わなさそう

想定解法

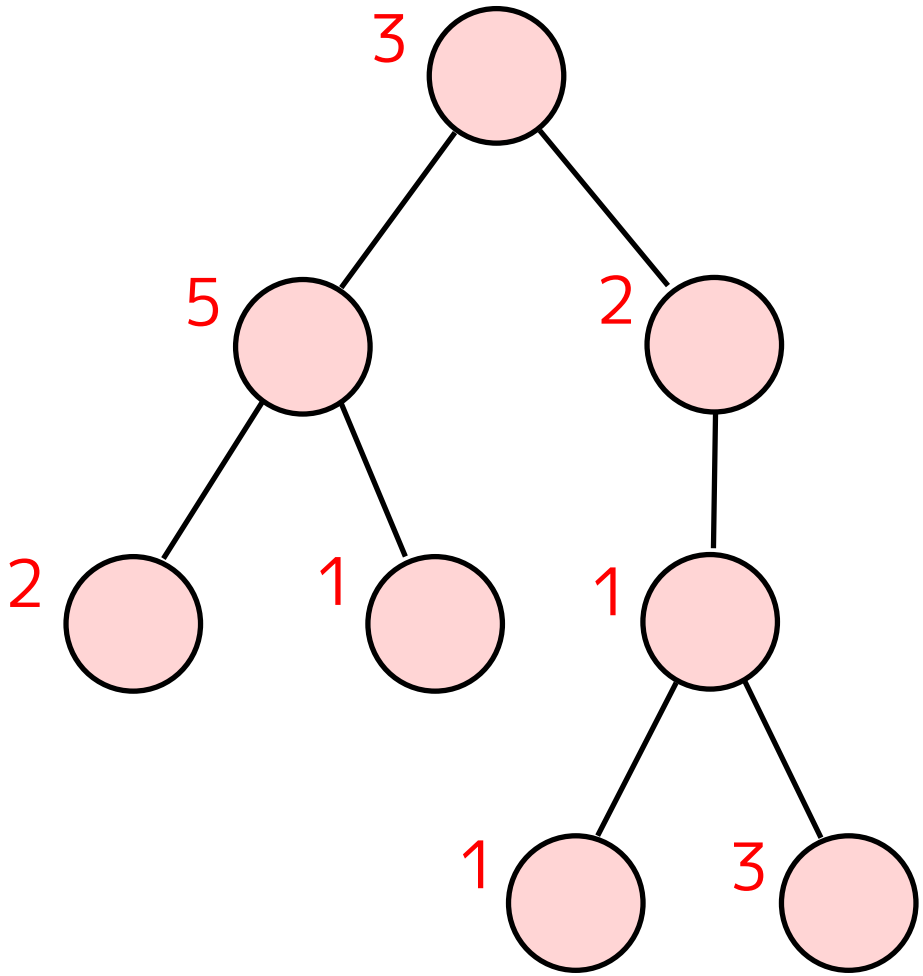
想定解法: 全方位木 DP

- 最初に適当な頂点を根とし、各頂点について以下を求める
  - $v1$: その頂点を根とする部分木でかかるステップ数
  - $v2$: その頂点から DFS をスタートして、途中で探索が失敗するか
- $v1$ と $v2$ の更新に気をつけて、rerooting

次に例を示します

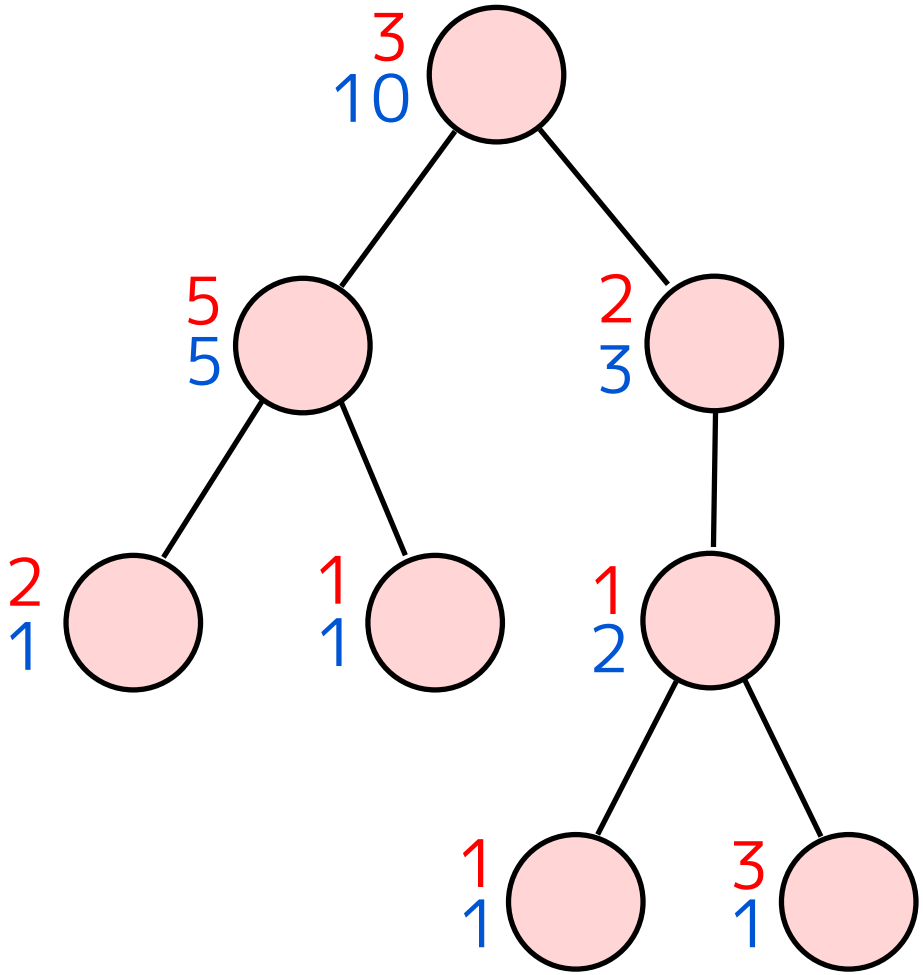
想定解法

以下のような木があったとする
赤文字は各頂点に割り当てられた A_i の値を表す



想定解法

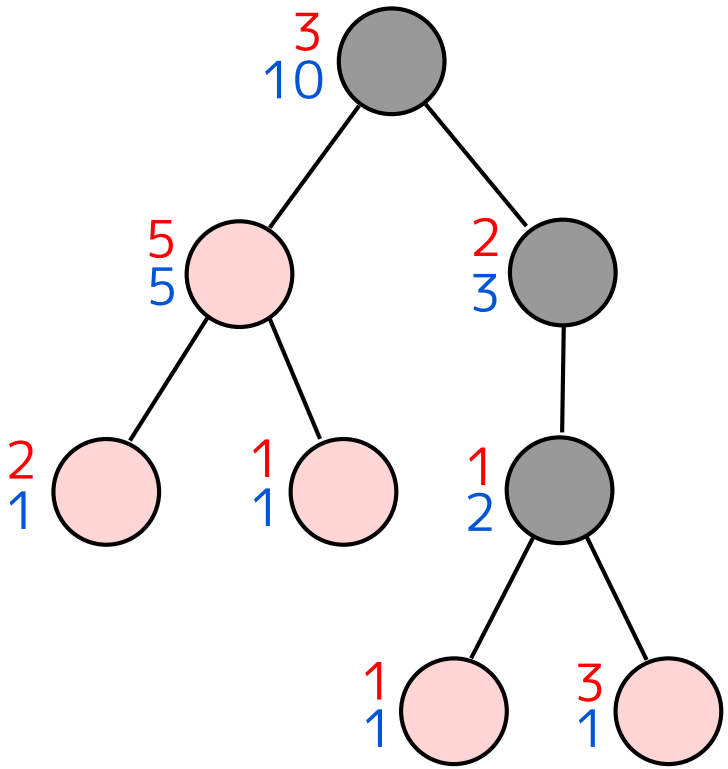
それぞれの頂点について $v1$ を求める (青文字)



想定解法

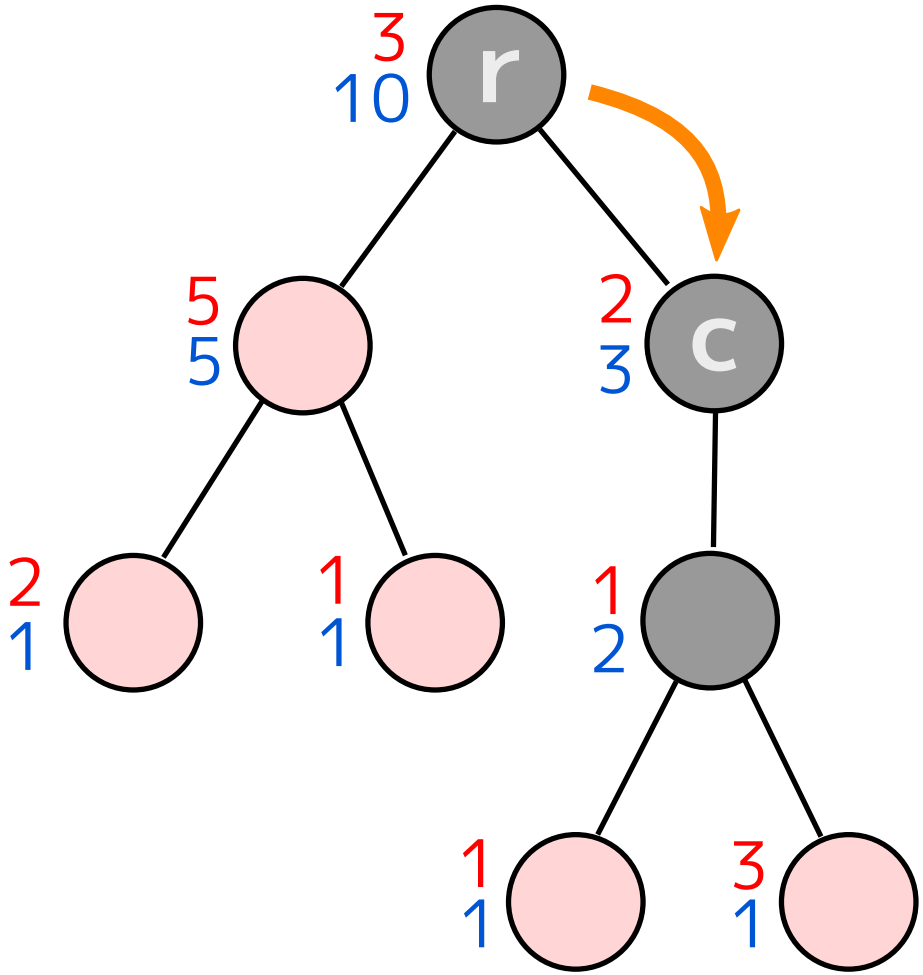
さらに $v2$ も求める (黒で塗ったものが、$v2$ が true である頂点)

- 子が m 個あるような頂点は $m+1$ 回デクリメントされるため、$A_i < m+1$ となる頂点は true となる
- 自分の子であって $v2$ が true となる頂点があるとき、自分自身に戻ってこれないため true となる



想定解法

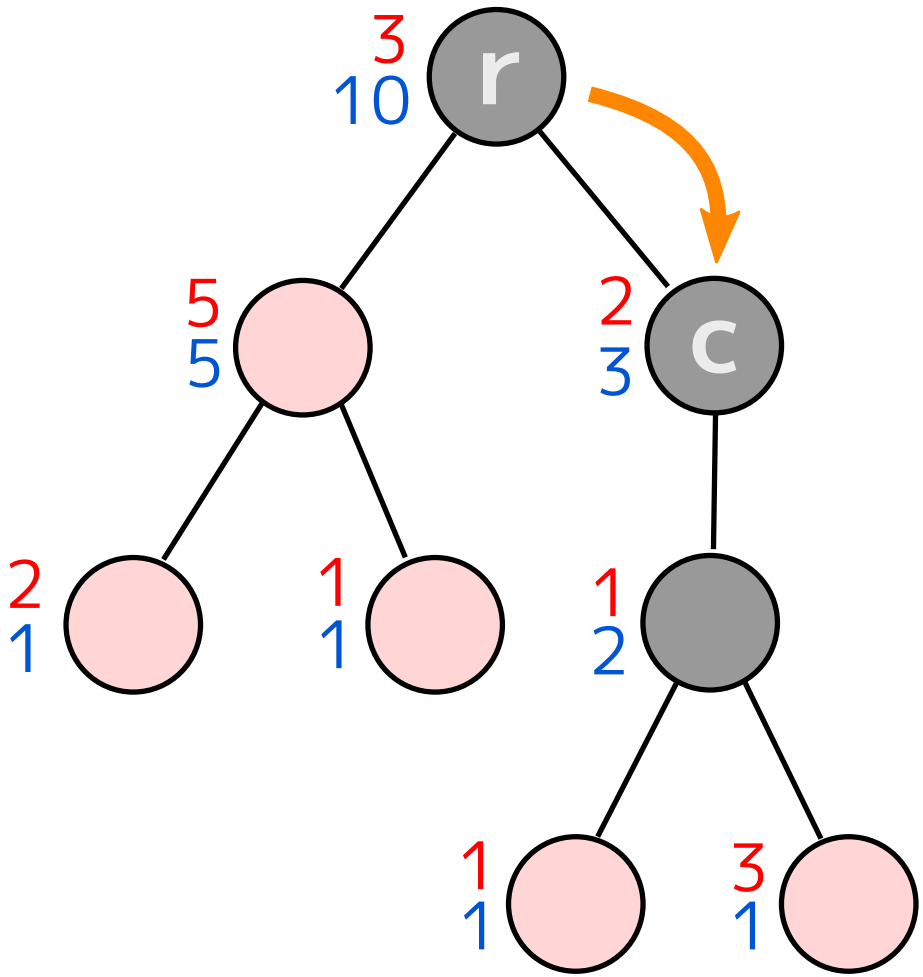
これで根が r であるときの答えは得られた
根を c に変えた時の答えはどう得られるか？



想定解法

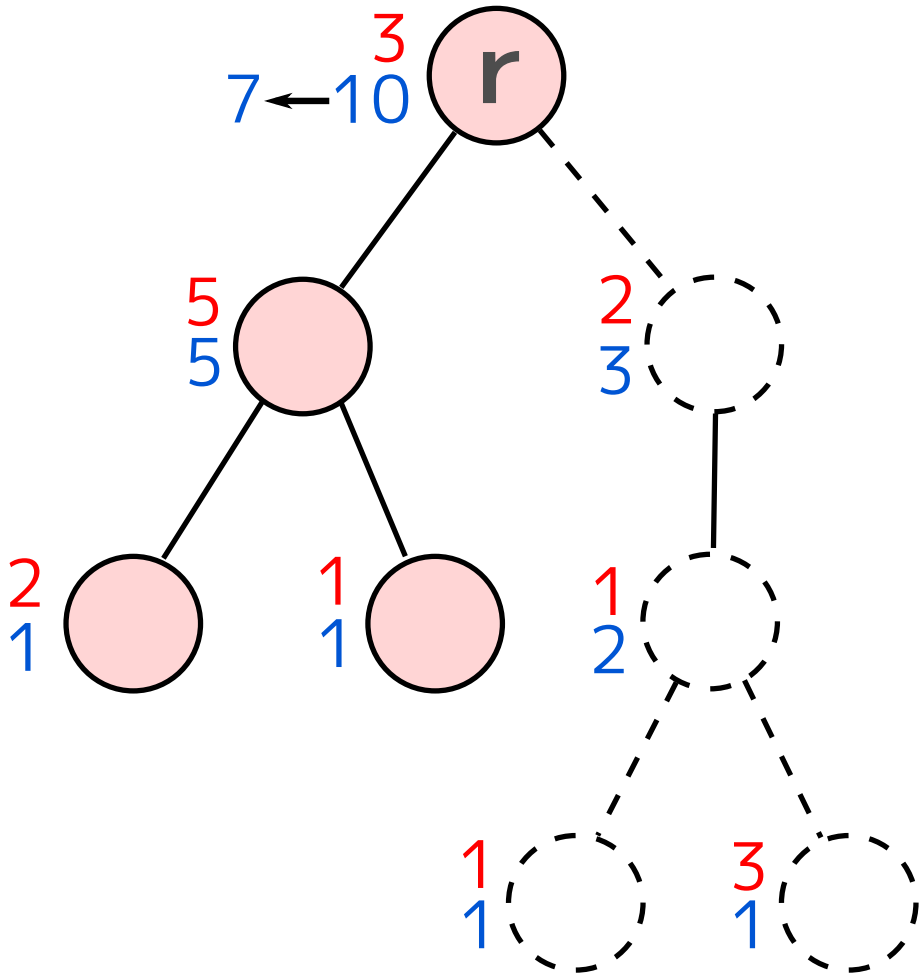
以下の順に処理！

1. c 以下の部分木を無視した状態で r について $v1, v2$ を再計算
2. c に r 以下の部分木が繋がったとして、c について $v1, v2$ を再計算



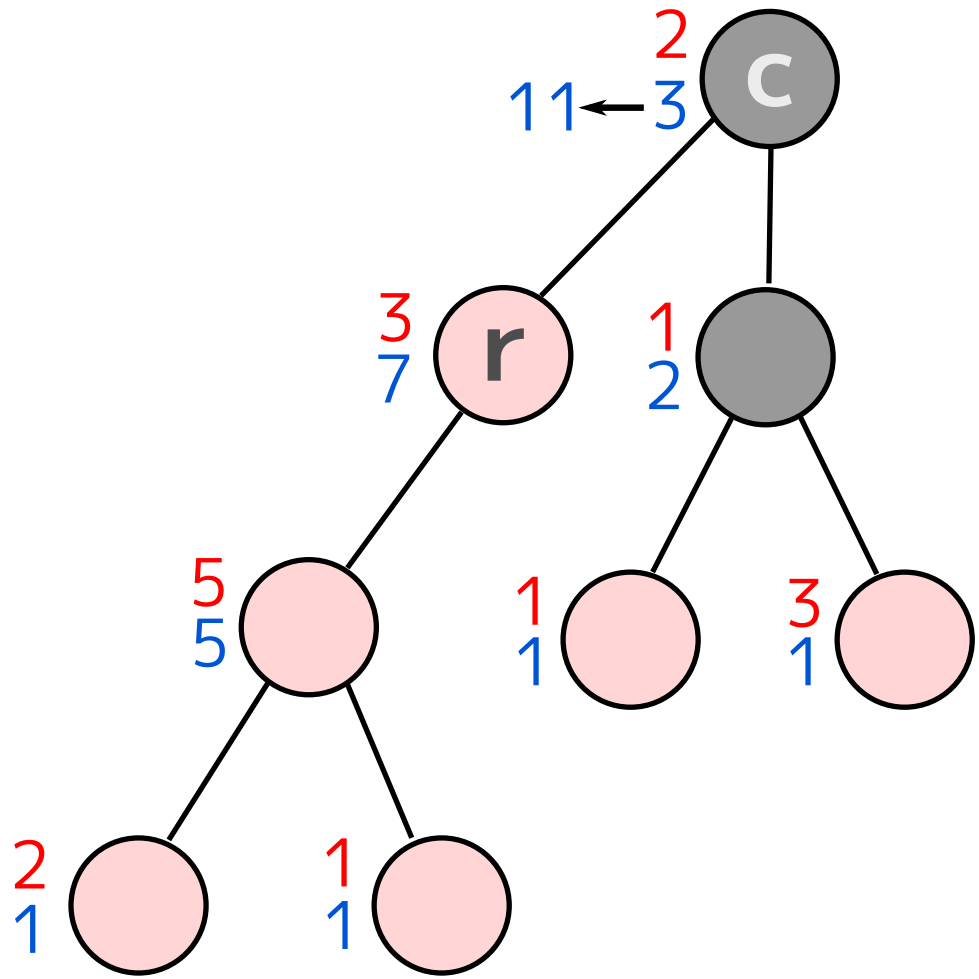
想定解法

1. の操作はこういうイメージ



想定解法

2. の操作はこういうイメージ



想定解法

- $v1, v2$ の更新は区間和が扱えるデータ構造 (SegmentTree など) を使いながらやるとできます
  - 自分と辺で直接接続している頂点の vector を、頂点番号の昇順に持つ
  - 自分の親である頂点を無視する必要があるが、$v1 = 0, v2 = \mathrm{false}$ として扱うと若干処理しやすいかも？
- それぞれの rerooting に $O(\log N)$ かかるため、全体で $O(N \log N)$ で解けます

その他

- 工夫次第だと思いますが、実装はだいぶ重いと思います
- 抽象化された全方位木 DP ライブラリは使えるのでしょうか？ 今回の問題において左・右の累積でどうにかする戦略はおそらく使えないので、きびしいだろうと思っていました

Writer 解・統計

- Tester 解
  - tsutaj (C++・241 行・8013 bytes)
  - rsk0315 (C++・376 行・10979 bytes)
- 統計
  - AC / tried: 2 / 9 (22.2 %)
  - First AC
    - On-site: ACPC_sakenichia (165 min 27 sec)
    - On-line: lyrically (123 min 1 sec)