J: Averaging

原案 kawabys
解説 haji, Gacho

問題概要

N頂点の木が与えられる。初期状態として各ノードには重みx_iがある。

目的:
すべてのノードの重みの平均（小数点以下切り捨て）をtとおいた時、各ノードの重みをtまたは(t+1)にしたい。ただしノードの重みの総和は変化してはならない。このときの最小操作数。

以下の操作が何度でもできる。
任意のノードiに対して、ノードiの重みに1加算、ある一つノードiに隣接するノードの重みを1減算

制約:
$1 \le N \le 5000$, $1 \le x_i \le 1e9$

解法: 2乗の木DP

dp[ノードの数][部分木のサイズ]で状態をとることができる木DPは$O(N^2)$のDPにすることができる。

おおまかな手順:

1. 状態が N^2で遷移$O(N)$の$O(N^3)$のDPを考える。
2. 適切に遷移を削って枝狩りをする。
3. $O(N^2)$になっている。

考察

木DPを考えてみる。元の木を適当なノードを根とする根付き木として考える。

dp[i][j] :=
ノードiを根とする部分木を視点として、現在のノードiの重みがj(負の値も取りうる)のときに子孫のすべてがtか(t+1)のときの最小の操作数。

これだと配列を取れない。そこで状態のとり方を工夫する。

考察

あるノードの重みはその子孫に対して何回(t+1)を割り当てたかで復元できる。
そこでDPの状態の持ち方を工夫する。

dp2[i][j] :=
ノードiを根とする部分木を視点として、現在のノードiの子孫にちょうどj回、重み(t+1)を割り振り、子孫のすべてがtか(t+1)のときの最小操作数。

考察

状態が取れたので遷移を考える。

dp2[i] [j + (子の(t+1)の数)] =

min(dp2[i] [j + (子の(t+1)の数)],
dp2[子] [子の(t+1)の数] + abs(t × 子の部分木のサイズ + 子の(t+1)の数 - 子を根とする部分木のx_iの総和)

or(または)

min(dp2[i] [j + (子の(t+1)の数)],
dp2[子][子の(t+1)の数] + abs(t × 子の部分木のサイズ + 子の(t+1)の数 **+ 1** - 子を根とする部分木のx_iの総和)

最終的な答えは、
min(dp2[root][重みの総和 mod t] , dp2[root][(重みの総和 mod t) - 1])になる

遷移を考える

前のページの遷移だと
遷移の際に「j」と「子の(t+1)の数」を決める必要があるので、$O(N^3)$になってしまう。

そこで「j」と「子の(t+1)の数」の決める範囲を制限する

- 0 ≤ j ≤ 遷移済みの子ノードの部分木のサイズの総和
- 0 ≤ 子の(t+1)の数 ≤ 子を根とする部分木のサイズ

以上のように遷移を制限すると計算量が$O(N^2)$になる。

※なぜ$O(N^2)$になるのかは説明しません。「木DP 2乗」などで検索すると出てきます。

Judge解

haji	C++	142行	4159 byte
beet	C++	54行	1158 byte
Gacho	C++	146行	2630 byte
tubuann	C++	105行	2037 byte
uku	C++	83行	1530 byte

First Accept(onsite)

First Accept(online)

k16013wi (0:02:56)

success rate (AC / Subbmission)

7/28