JOI 2020/2021 春合宿 Day2
道路の建設案 (Road Construction)

解説: 木ノ下 恭範

問題概要

- N 個の町があり、町 i の座標は (X_i, Y_i)。
- 町 i と町 j をつなぐ道路は $|X_i - X_j| + |Y_i - Y_j|$ 円かかる。
- $N(N-1)/2$ 通りのうち、安い方から K 個の道路の費用を順に出力してください。

- $N \leq 250000$
- $K \leq 250000$

小課題 1：$N \leq 1000$

- $N(N-1)/2$ 通りを全て計算して、ソートして出力すればよい。

- 時間計算量 $\Theta(N^2 \log(N))$
- 5 点

小課題 2：$Y_i = 0$

- 全ての点が X 軸上に並んでいる。
- 道路の費用は $|X_i - X_j|$ となり、数直線上の距離と一致する。
- 町は X の昇順にソートされているとする。

- 費用の安い順に、町の組を列挙していくアルゴリズムが作れる。
- 全ての町について、その左隣の町との距離を考える。最も安い組は、このうちのいずれかである。その組を $(i-1, i)$ とすれば、2 番目に安い組は $(i-1, i)$ を除外して代わりに $(i-2, i)$ を入れたときの最小である。これを繰り返せばよい。

小課題 2：$Y_i = 0$

- 費用の安い順に、町の組を列挙していくアルゴリズムが作れる。
- 全ての町について、その左隣の町との距離を考える。最も安い組は、このうちのいずれかである。その組を $(i-1, i)$ とすれば、2 番目に安い組は $(i-1, i)$ を除外して代わりに $(i-2, i)$ を入れたときの最小である。これを繰り返せばよい。
- 優先度付きキュー（std::priority_queue）を用いると効率的に実装できる。

- 時間計算量 $\mathrm{O}\big((N+K)\log(N)\big)$
- 6 点

小課題 3：$K = 1$

- 町は X の昇順にソートされているとする。

- 町 i と町 j $(i < j)$ をつなぐ費用を整理する。
- X 昇順にソートしたので、$X_i \leq X_j$ 。
- $Y_i \leq Y_j$ ならば、費用は $\left(X_j - X_i\right) + \left(Y_j - Y_i\right)$ 。
- $Y_i \geq Y_j$ ならば、費用は $\left(X_j - X_i\right) + \left(Y_i - Y_j\right)$ 。
- 絶対値がなくなったので、i に関係する項と j に関係する項に分解することが出来る。
- $Y_i \leq Y_j$ ならば、費用は $(-X_i - Y_i) + \left(X_j + Y_j\right)$ 。
- $Y_i \geq Y_j$ ならば、費用は $(-X_i + Y_i) + \left(X_j - Y_j\right)$ 。

小課題 3：$K = 1$

- 町 i と町 j $(i < j)$ をつなぐ費用を整理する。
- $Y_i \leq Y_j$ ならば、費用は $(-X_i - Y_i) + (X_j + Y_j)$ 。
- $Y_i \geq Y_j$ ならば、費用は $(-X_i + Y_i) + (X_j - Y_j)$ 。

- 各 j について、最も安い相手 i $(i < j)$ を求める。
- 町を X 昇順に読み、Y 座標を添え字とする Segment Tree で $-X_i - Y_i$ と $-X_i + Y_i$ それぞれの最小値を管理する。j の手前まで読んだ段階で、$Y_i \leq Y_j$ の範囲での $-X_i - Y_i$ の最小値を取得し、$X_j + Y_j$ を加算すればよい。$Y_i \geq Y_j$ についても同様。
- Y 座標は大きいので、添え字に使うために座標圧縮しておく。

小課題 3：$K = 1$

- 各 j について、最も安い相手 i $(i < j)$ を求める。
- 町を X 昇順に読み、Y 座標を添え字とする Segment Tree で $-X_i - Y_i$ と $-X_i + Y_i$ それぞれの最小値を管理する。j の手前まで読んだ段階で、$Y_i \leq Y_j$ の範囲での $-X_i - Y_i$ の最小値を取得し、$X_j + Y_j$ を加算すればよい。$Y_i \geq Y_j$ についても同様。
- Y 座標は大きいので、添え字に使うために座標圧縮しておく。

- 時間計算量 $\Theta(N \log(N))$
- 7 点

小課題 4：$K \leq 10$

- $K = 2$ で考えることにする。

- 小課題 3 のアルゴリズムを用いて、最も安い町の組 (a, b) を求める。
- $N(N-1)/2$ 通りある町の組を、以下の 2 種類に分類する。
  - a または b を含むもの
  - a も b も含まないもの
- 2 番目に安い組は、前者に含まれるか、さもなくば後者の中で最も安い。
- 前者は $2N - 3$ 組しかないため列挙できる。
- 後者は a, b を除いた町群に小課題 3 を適用すればよい。

小課題 4：$K \leq 10$

- $K \leq 10$ も同様に解くことが出来る。

- 以下の操作を K 回繰り返す
  - 最も安い組 (a, b) を計算する。
  - a, b を含む組を全て列挙する。
  - 町の集合から a, b を取り除く。
- 列挙された組のうち、安い方から K 組を出力する。

- 時間計算量 $\Theta(KN \log(N))$
- 20 点

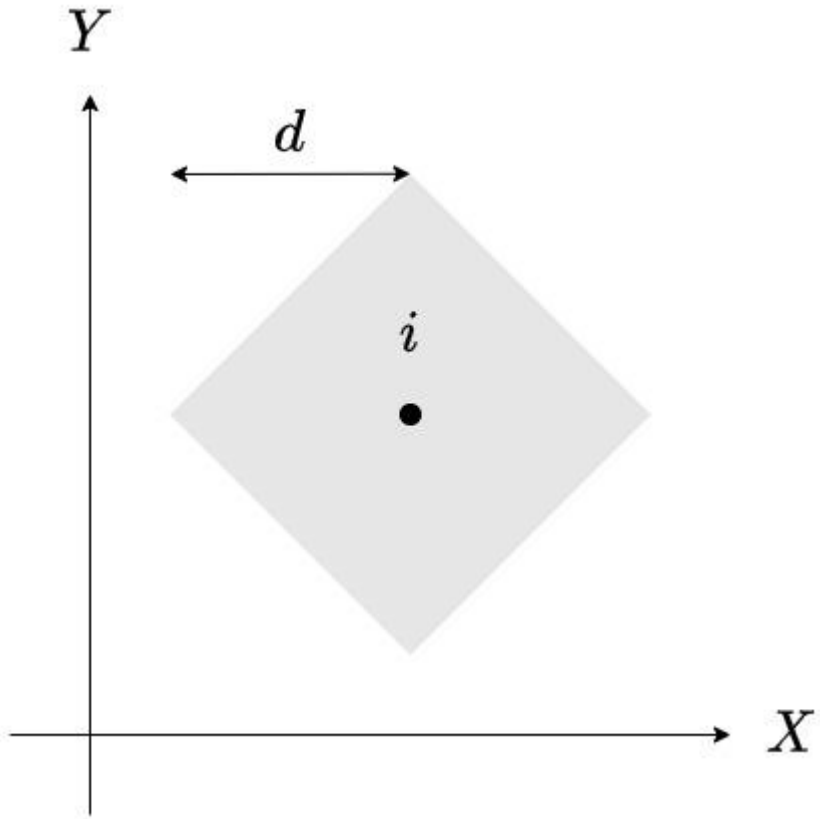
小課題 5：$N \leq 10^5$

- 費用 d 以下の町の組の数を数えることができれば、二分探索を用いることで、K 番目に安い道路の費用を計算することが出来る。その値を D とすれば、費用 $D-1$ 以下の町の組を列挙した後、D を何回か出力すればよい。

- 各町 i について、町 i とつなぐ費用が d 以下の町の個数を数えたい。また、そのような街を列挙したい。

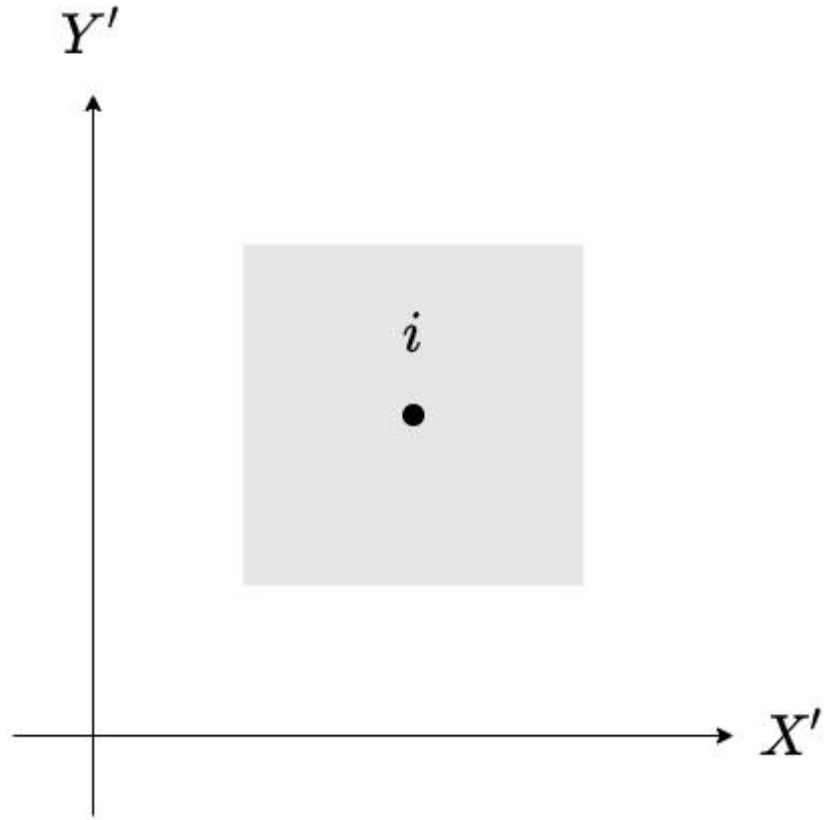
小課題 5：$N \leq 10^5$

- 町 i とつなぐ費用が d 以下の町の座標範囲を図示すると、下図のようになる。



小課題 5 : $N \leq 10^5$

- $45°$ 回転させることで正方形の領域となり、扱いやすくなる。
- $X' \coloneqq X + Y,\quad Y' \coloneqq X - Y$ とすれば、回転できる。
- 費用は $\max\{|X'_i - X'_j|, |Y'_i - Y'_j|\}$ となる。



小課題 5 : $N \leq 10^5$

- $X' \coloneqq X + Y, \quad Y' \coloneqq X - Y$
- 費用は $\max\{|X'_i - X'_j|, |Y'_i - Y'_j|\}$ となる。

- 町を X' の昇順にソートし、各 j について、費用 d 以下の範囲に入っている町 i $(i < j)$ の個数を数え、足し合わせる。
- そのような i の条件は、$X'_j - d \leq X'_i$ かつ $Y'_j - d \leq Y'_i \leq Y'_j + d$ となる。
- X' について尺取り法を用い、Y' を添え字とする Segment Tree で Y'_i が特定の範囲に入る i の個数を高速に取得できるようにすると、全体で $\Theta(N \log(N))$ で数えられる。

小課題 5 : $N \leq 10^5$

- X' について尺取り法を用い、Y' を添え字とする Segment Tree で Y_i' が特定の範囲に入る i の個数を高速に取得できるようにすると、全体で $\Theta(N \log(N))$ で数えられる。

- Segment Tree ではなく std::set を用いれば、費用 d 以下の組を全て列挙することができる。

- 時間計算量 $\Theta(N \log(N) \log(A) + K \log(K))$
- 27 点

- 速度次第では、満点が取れる。

小課題 6：満点

- 小課題 3 ($K = 1$) で用いたアルゴリズムを発展させる。
- 町を X 昇順に読み、Y 座標を添え字とする Segment Tree で $-X_i - Y_i$ と $-X_i + Y_i$ それぞれの最小値を管理する。j の手前まで読んだ段階で、$Y_i \leq Y_j$ の範囲での $-X_i - Y_i$ の最小値を取得し、$X_j + Y_j$ を加算すればよい。$Y_i \geq Y_j$ についても同様。(小課題 3)
- $-X_i - Y_i$ の最小値を $+\infty$ で更新して再び最小値を取得すれば、($Y_i \leq Y_j$ の範囲で) 2 番目に安い町を取得できる。

小課題 6：満点

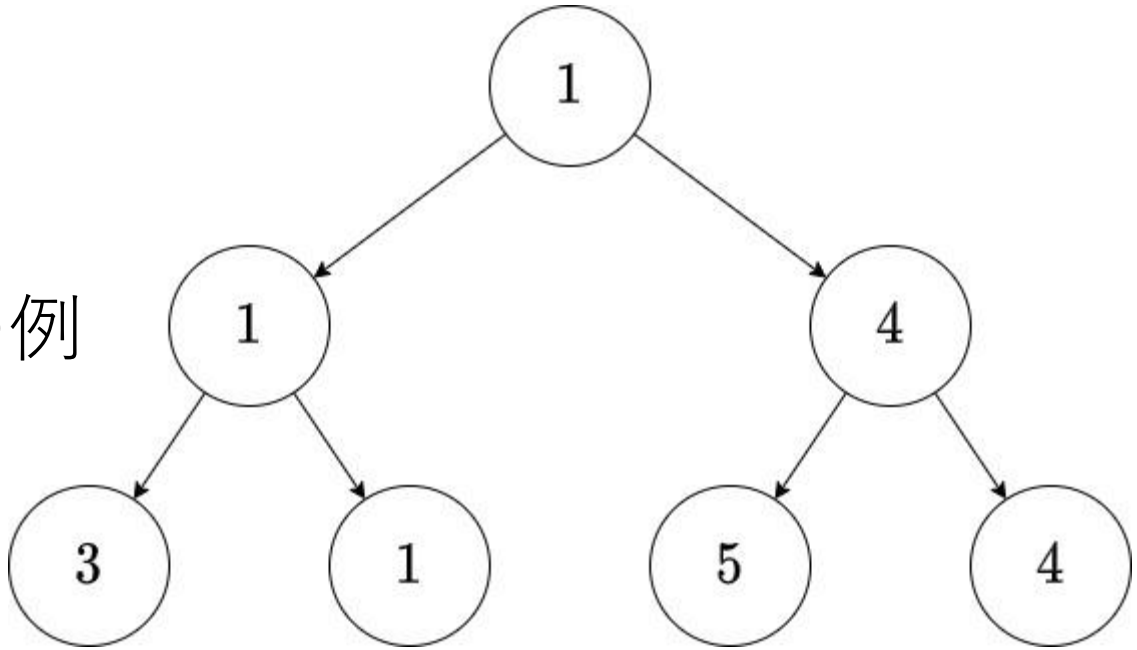
- $-X_i - Y_i$ の最小値を $+\infty$ で更新して再び最小値を取得すれば、($Y_i \leq Y_j$ の範囲で) 2 番目に安い町を取得できる。

- 小課題 2 $(Y_i = 0)$ で用いたアルゴリズムのように、最も安い組 (i, j) を出力し、(i, j) を同じ j で 2 番目に安い組 (i', j) に置き換えることを K 回繰り返す。
- これを行うためには、町を X 昇順に読む際に刻々と変化する Segment Tree の全ての時刻に後からアクセスする必要がある。($+\infty$ への更新も行う)
- 過去のデータにアクセスできるデータ構造を、永続データ構造と呼ぶ。Segment Tree を永続的にできるだろうか？

小課題 6：満点

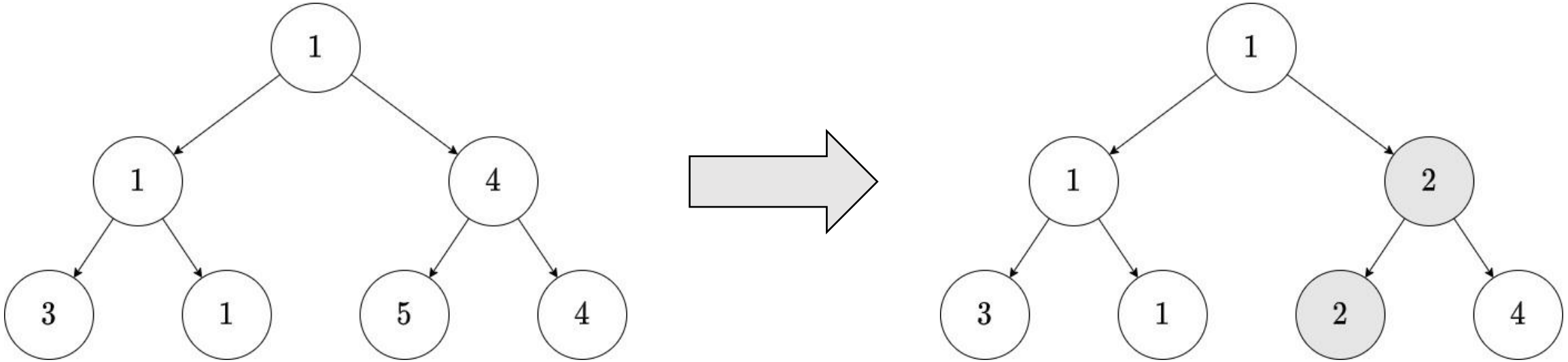
- Segment Tree を永続的にできるだろうか？
- Segment Tree の木構造を、それぞれのノードが左右の子を指すことで表現する。例えば、ノードの配列を用意して、左右の子をその添え字で表せばよい。

最小値を管理する Segment Tree の例



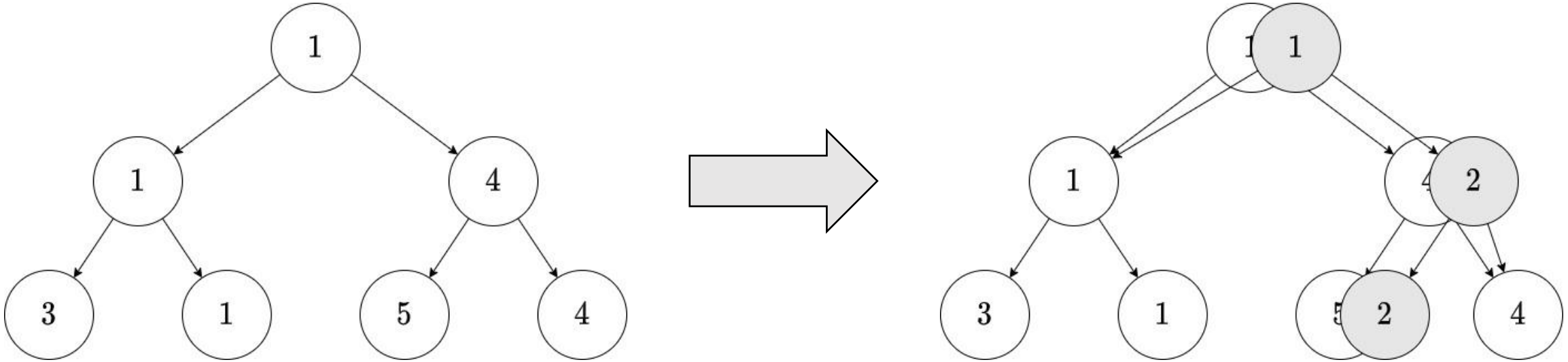
小課題 6：満点

- 5 と書かれた位置を 2 に更新することを考える。
- 通常の Segment Tree は、単にいくつかのノードの値を書き換える。(書き換わったノードを灰色で示した)



小課題 6：満点

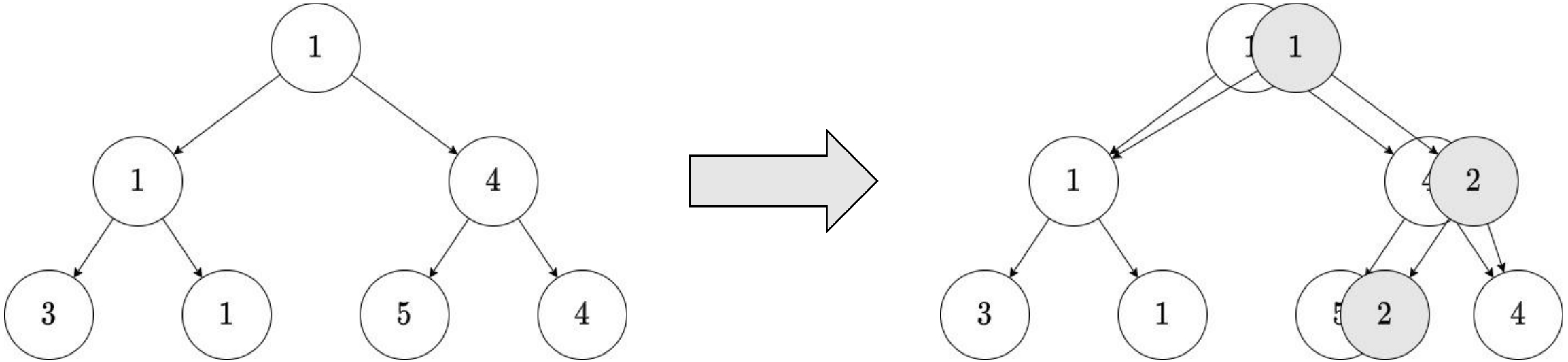
- 5 と書かれた位置を 2 に更新することを考える。
- 永続的な Segment Tree では、書き換えは行わずに新たにノードを作成する。
- 更新前と後のそれぞれの根に着目すると、その根から辺を辿って到達できるノードの集合は、通常の Segment Tree として見ることができる。



小課題 6：満点

- 1 回の更新の時間計算量は $\Theta(\log(N))$ となる。

- 解法全体の時間計算量 $\Theta\big((N+K)\log(N)\big)$
- 35 点



得点分布