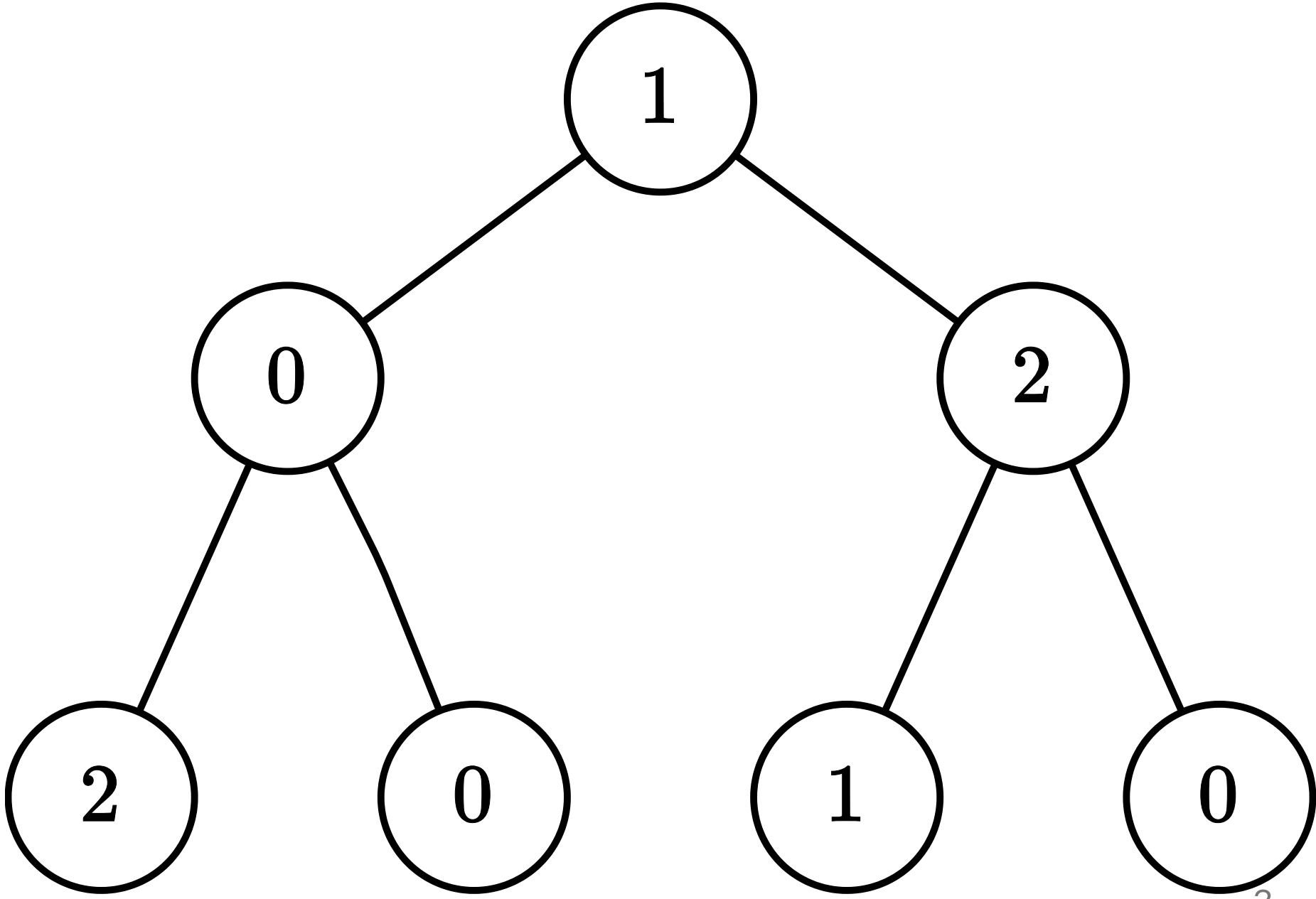
JOI 2023/2024 春季トレーニング Day 3
JOI ツアー (JOI Tour) 解説

解説担当 : tatyam

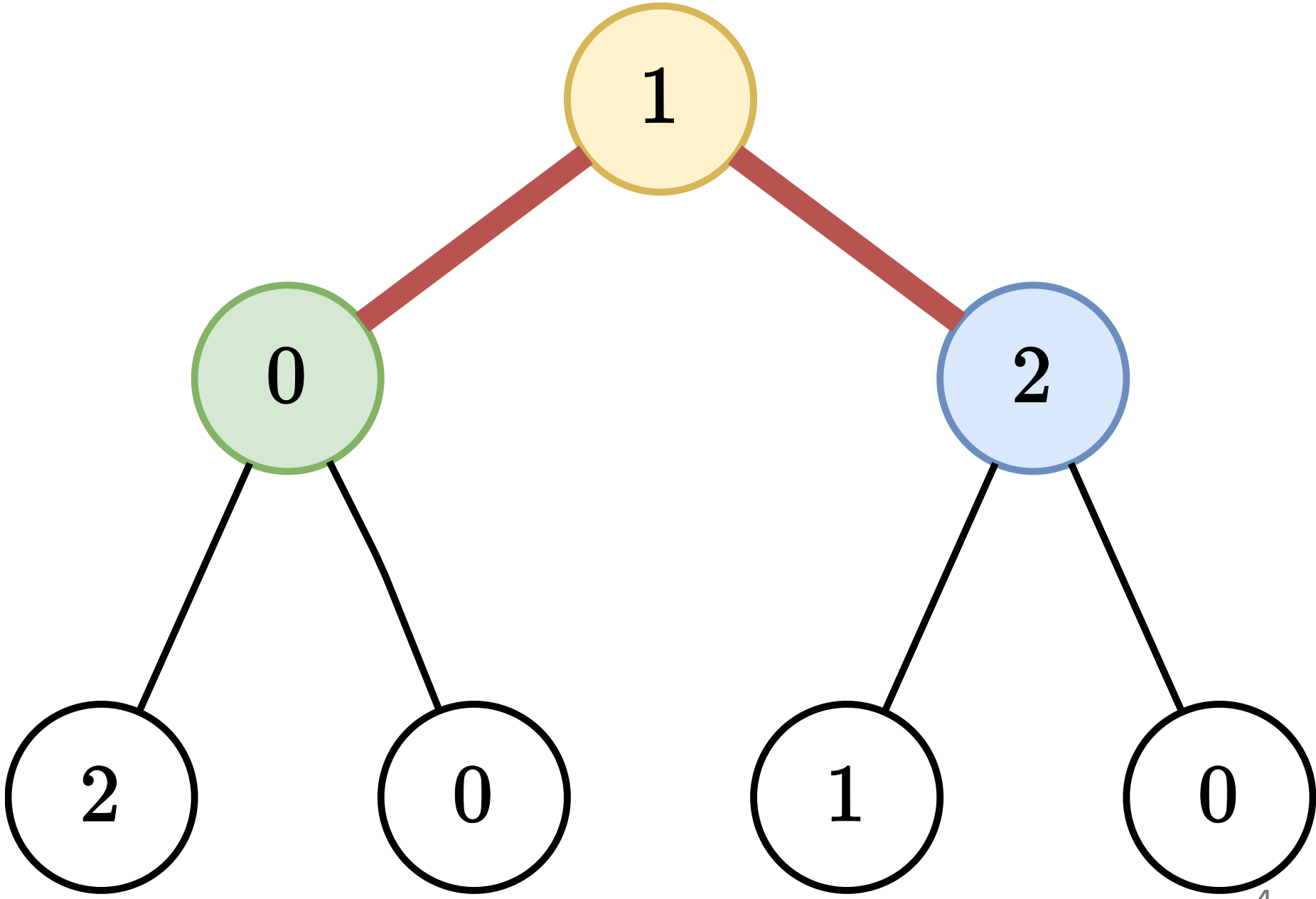
問題概要

- N 頂点の木があって，各頂点には $0, 1, 2$ いずれかの整数が書かれている.
- Q クエリを **オンラインで** 処理 :
  - 頂点 X に書かれている整数を Y に変更
  - その後，$0 - 1 - 2$ パスの個数を出力

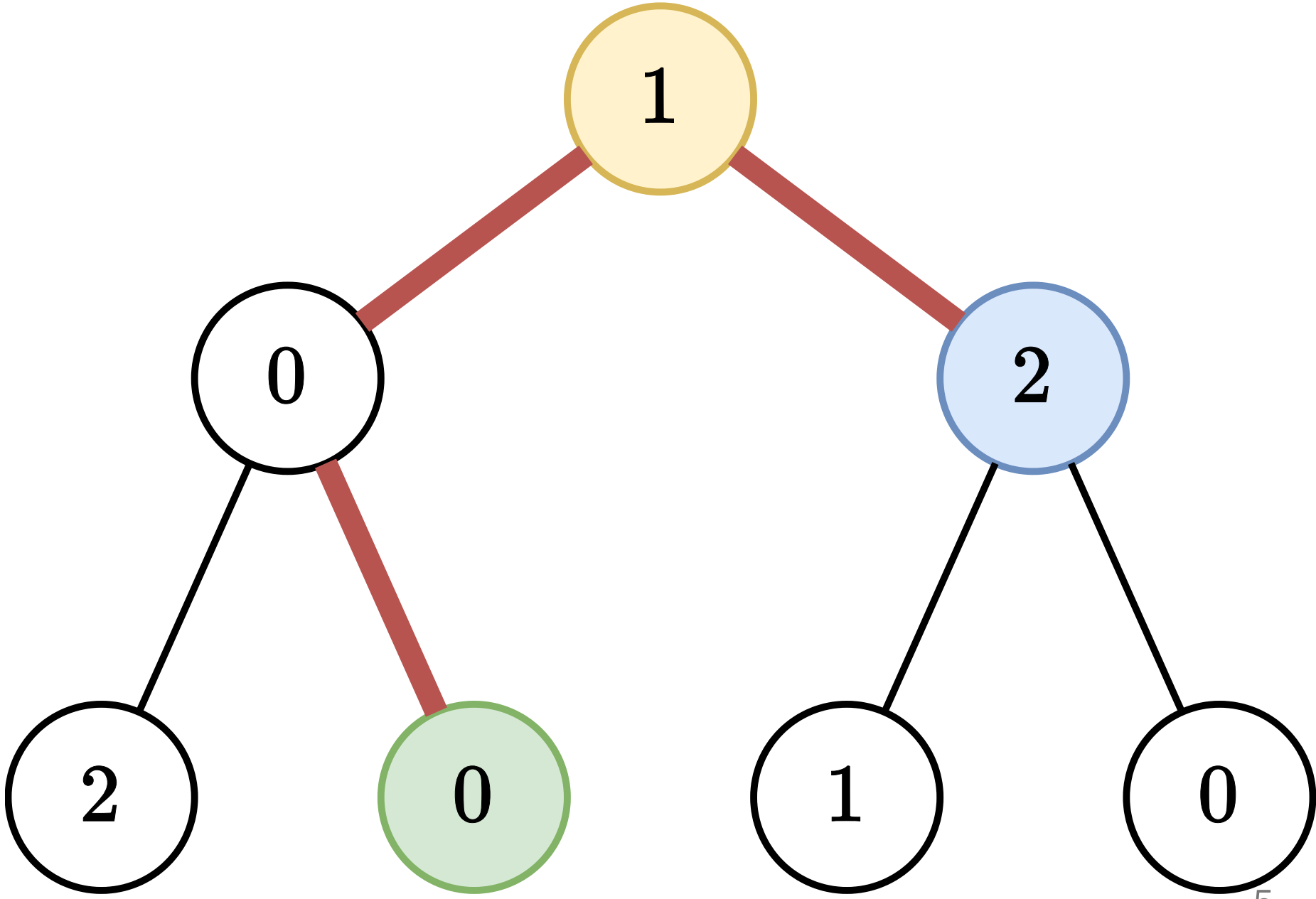
例



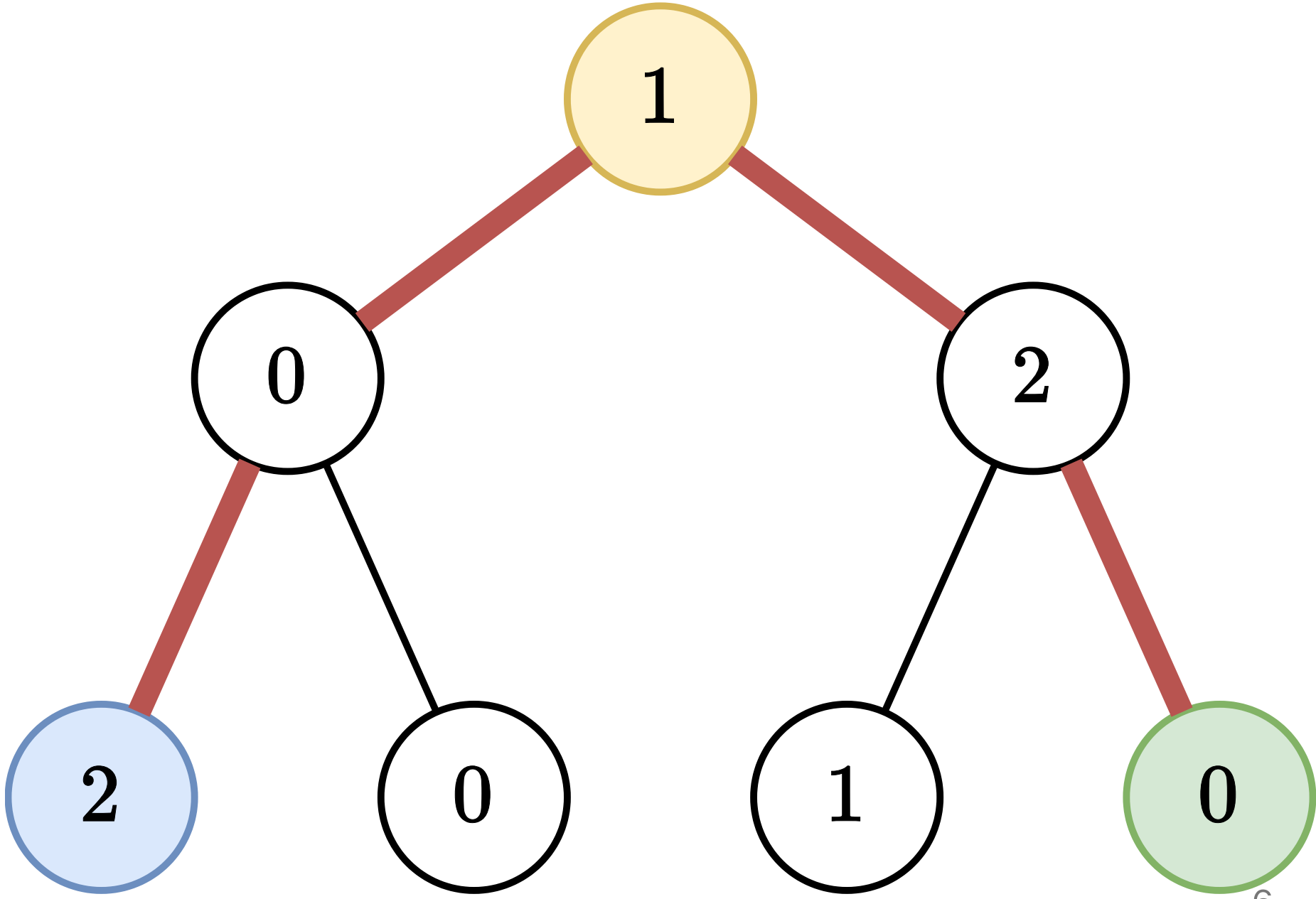
例



例



例



小課題

番号	配点	制約	計算量の例
1.	6 点	$N \leq 400,\ Q \leq 100$	$O(QN^2)$
2.	8 点	$N \leq 4000,\ Q \leq 1000$	$O(N^2 + QN)$
3.	6 点	$Q = 0$	$O(N)$
4.	16 点	パス	$O((N+Q)\log N)$
5.	16 点	完全 2 分木	$O((N+Q)\log N)$
6.	34 点	$N \leq 100000,\ Q \leq 25000$	
7.	14 点	$N \leq 200000,\ Q \leq 50000$	$O(N \log N + Q(\log N)^2)$

小課題 $1: N \leq 400, Q \leq 100$ (6 点)

- 一旦変更クエリを忘れてみる
- 木が与えられたとき，$0-1-2$ パスの個数を $O(N^2)$ 時間で数える

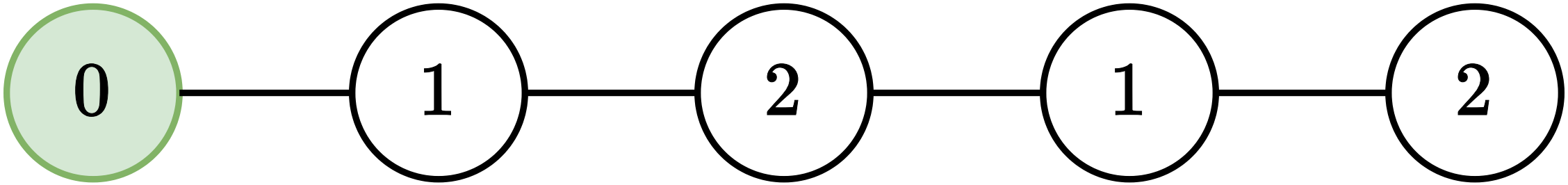
$0 - 1 - 2$ パスの個数を $O(N^2)$ 時間で数えたい

- 使う 0 を固定してみる

  → 指定された 0 から始まる $0 - 1 - 2$ パスの個数を $O(N)$ 時間で数えたい

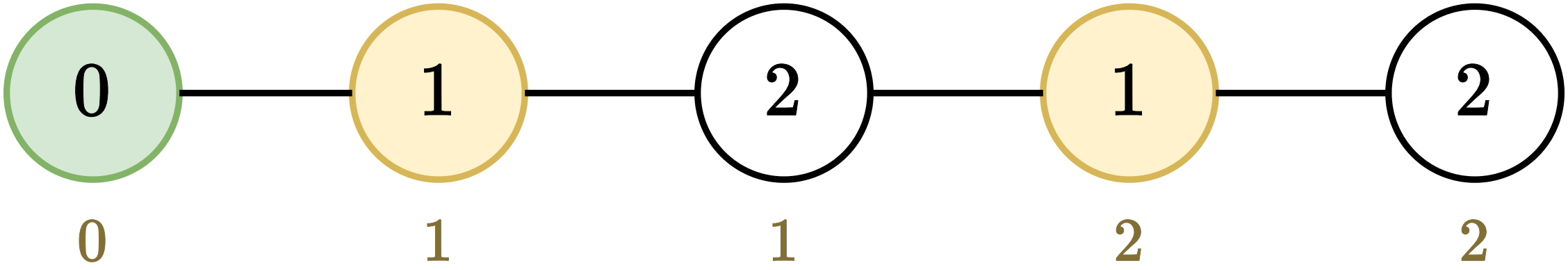
パスの場合は…？

- 使う 0 を固定してみる



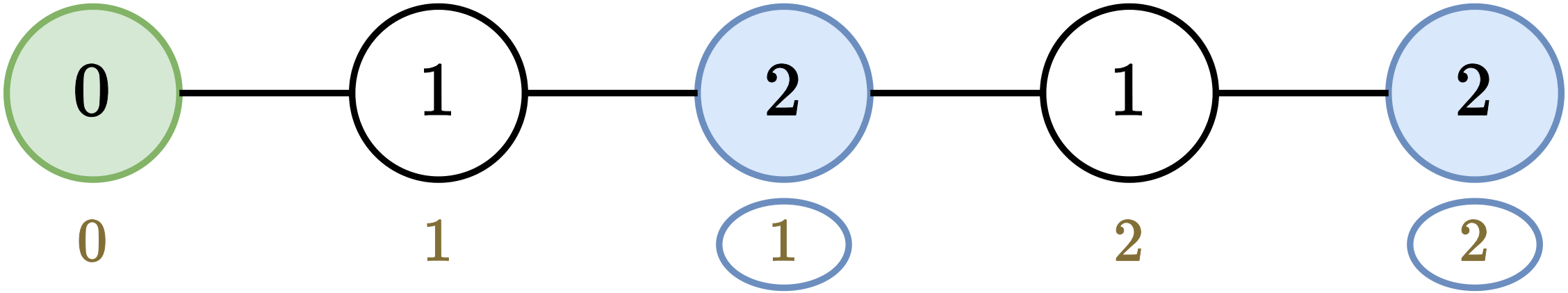
パスの場合は…？

- 使う 0 を固定してみる
- 0 から各位置までの 1 の個数を求めて…



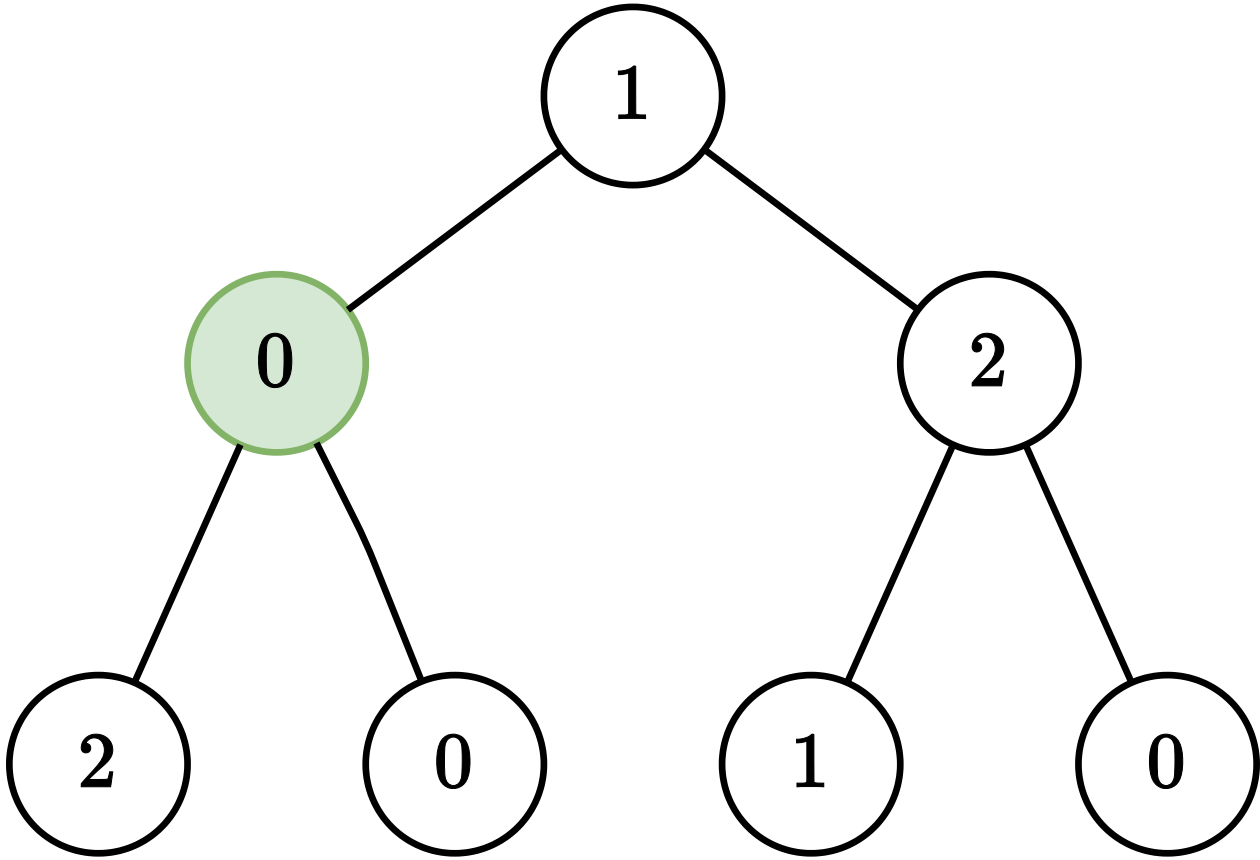
パスの場合は…？

- 使う 0 を固定してみる
- 0 から各位置までの 1 の個数を求めると,
- 使う 0 と 2 を固定した場合の $0-1-2$ パスの個数は，$0-2$ 間の 1 の個数に等しい



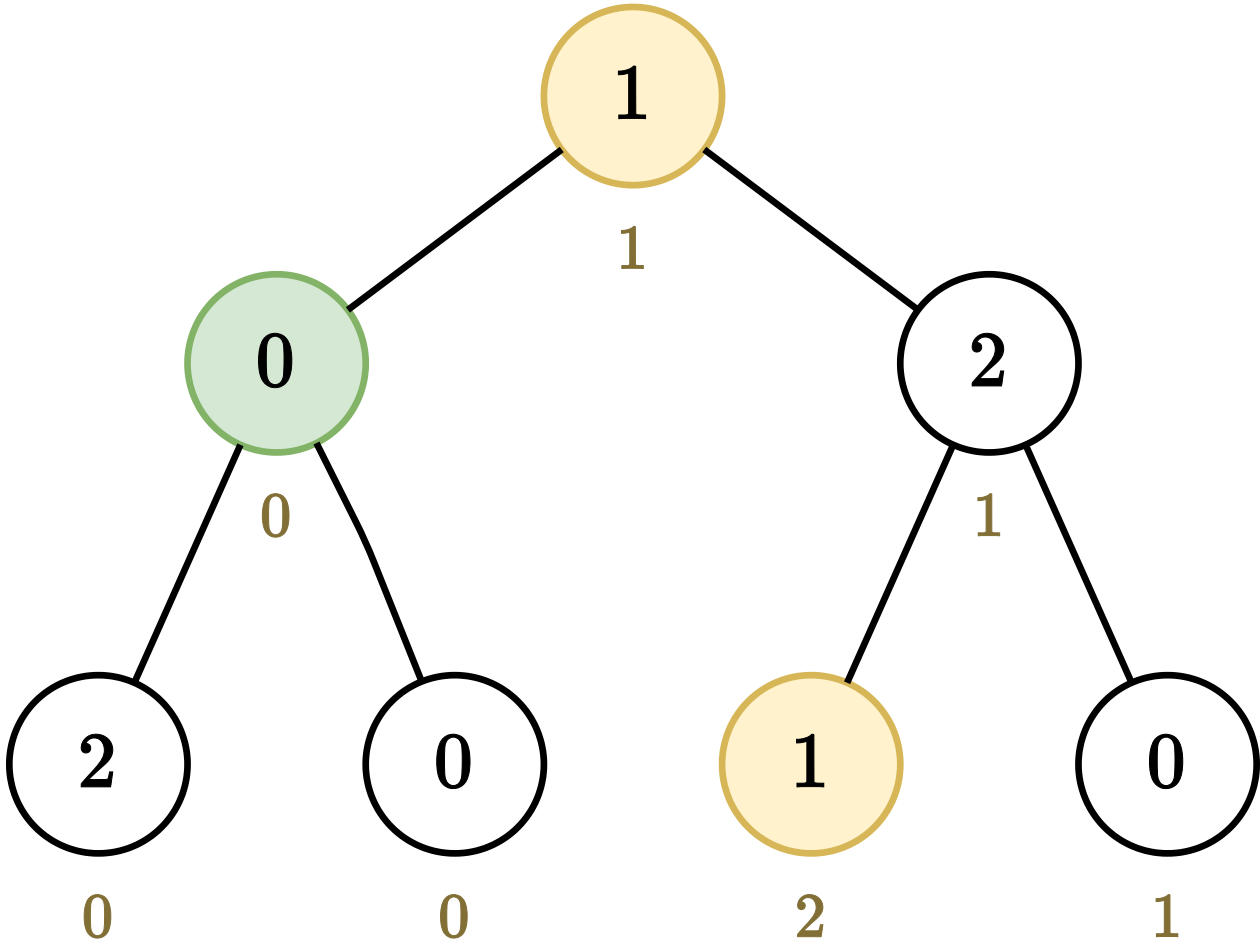
木の場合は…？

- 同じことを DFS しながら行う



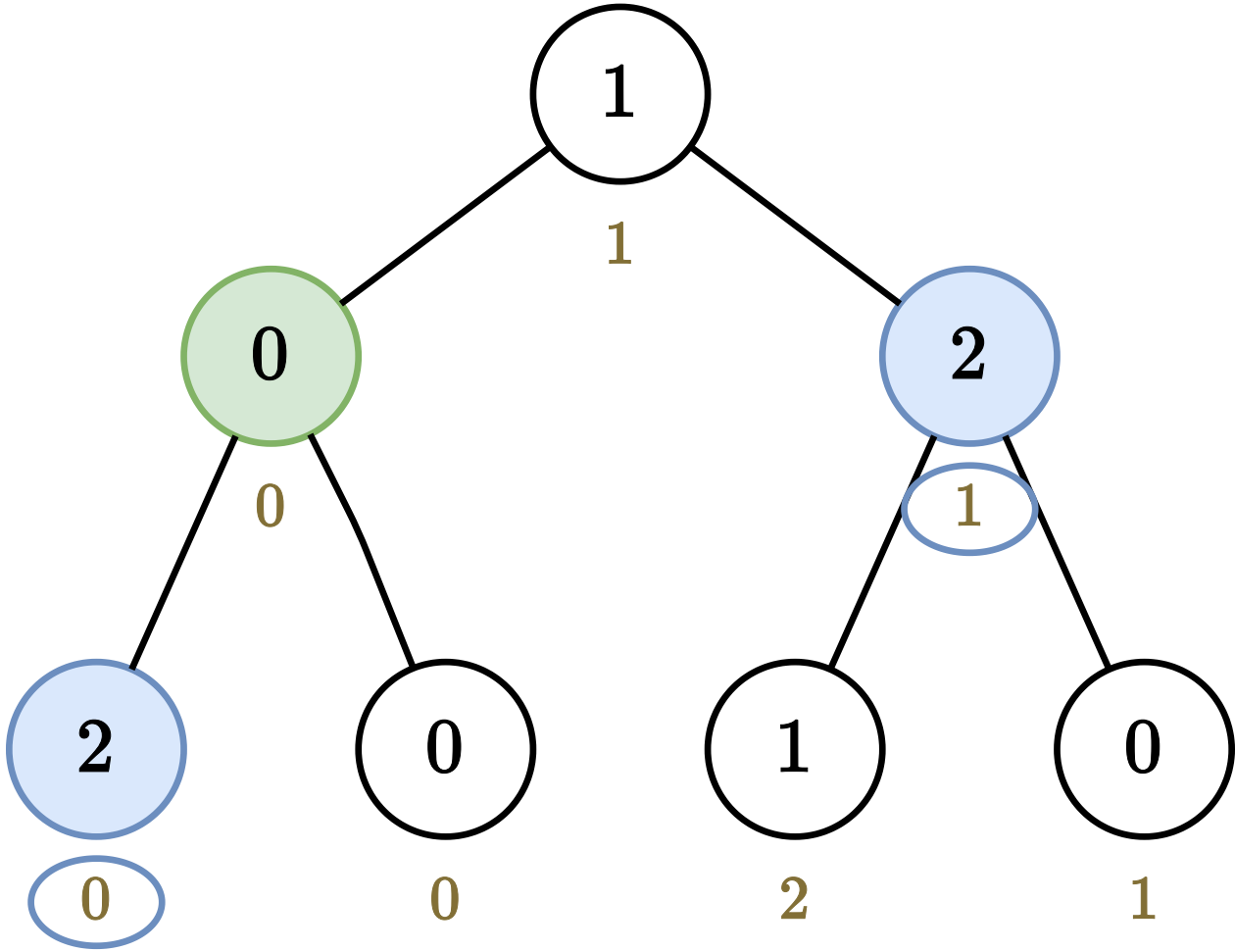
木の場合は…？

- 同じことを DFS しながら行う
- 0 から各位置までの 1 の個数を求めると,



木の場合は…？

- 同じことを DFS しながら行う
- 0 から各位置までの 1 の個数を求めると,
- 使う 0 と 2 を固定した場合の $0-1-2$ パスの個数は，$0-2$ 間の 1 の個数に等しい



小課題 $2: N \leq 4000,\ Q \leq 1000\ (8$ 点$)$

- 毎回 $\Theta(N^2)$ 時間掛けると TLE → 差分計算をしよう！

小課題 $2: N \leq 4000,\ Q \leq 1000\ (8$ 点$)$

- 毎回 $\Theta(N^2)$ 時間掛けると TLE → 差分計算をしよう！

- 毎回 $\Theta\left(N^2\right)$ 時間掛けると TLE → 差分計算をしよう！

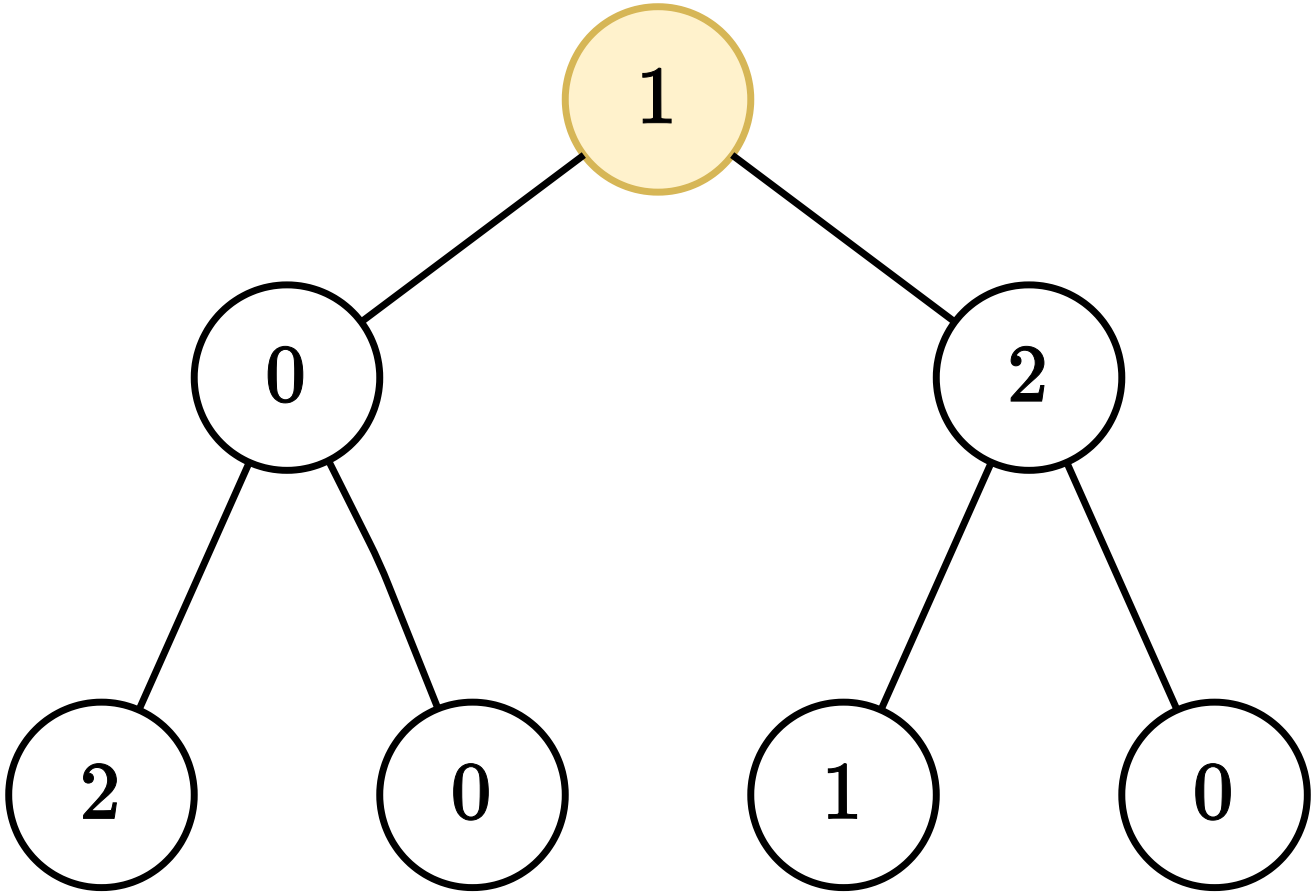
差分計算

1. 頂点 X を使う $0-1-2$ パスの個数を数えて，答えから引く
2. F_X を Y に変更する
3. 頂点 X を使う $0-1-2$ パスの個数を数えて，答えに足す

頂点 X を使う $0-1-2$ パスの個数

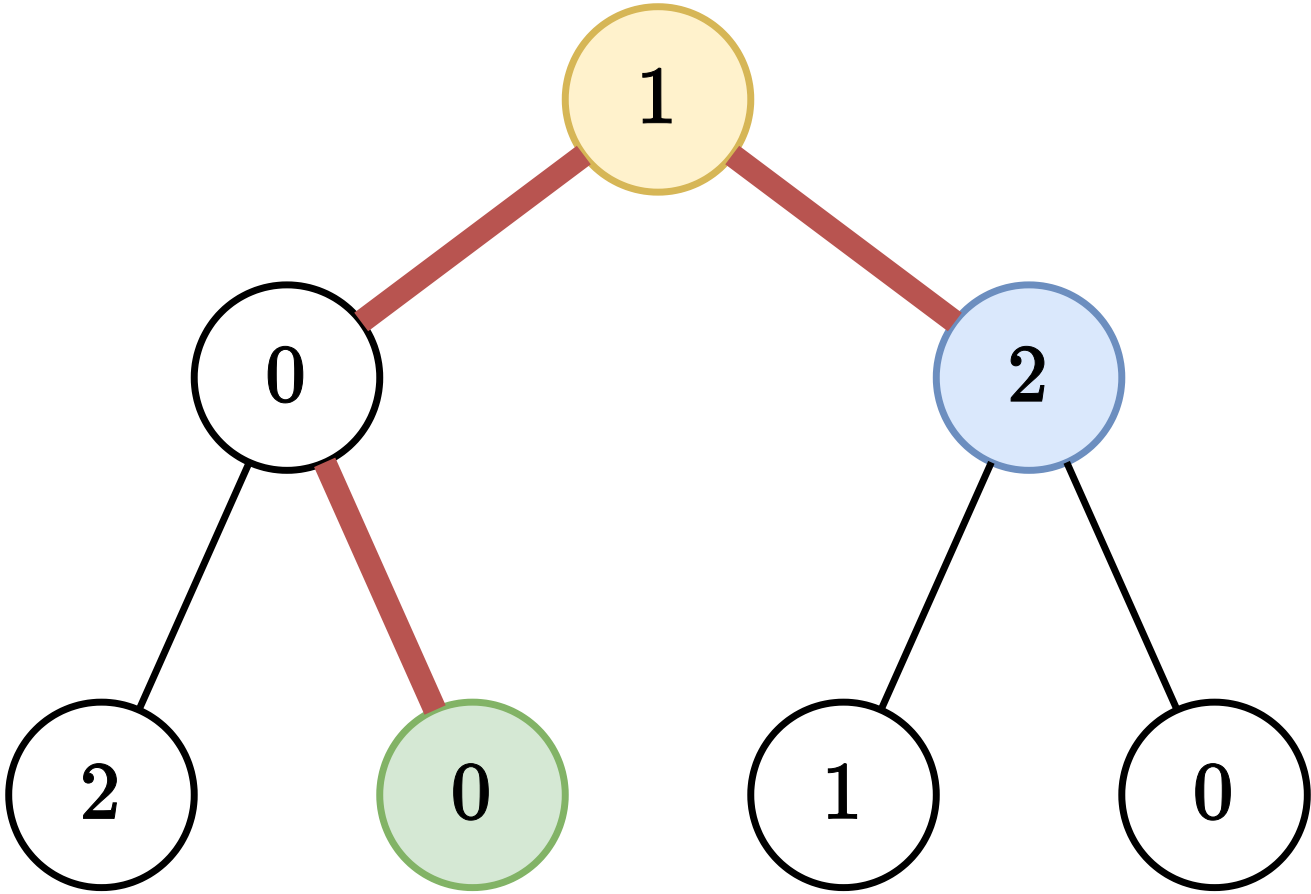
- $F_X = 0$ のとき : さっきやった
- $F_X = 2$ のとき : $0, 1, 2$ を反転させれば，さっきやった
- $F_X = 1$ のとき : ？

1 を指定されたときの
$0 - 1 - 2$ パスの個数



1 を指定されたときの $0-1-2$ パスの個数

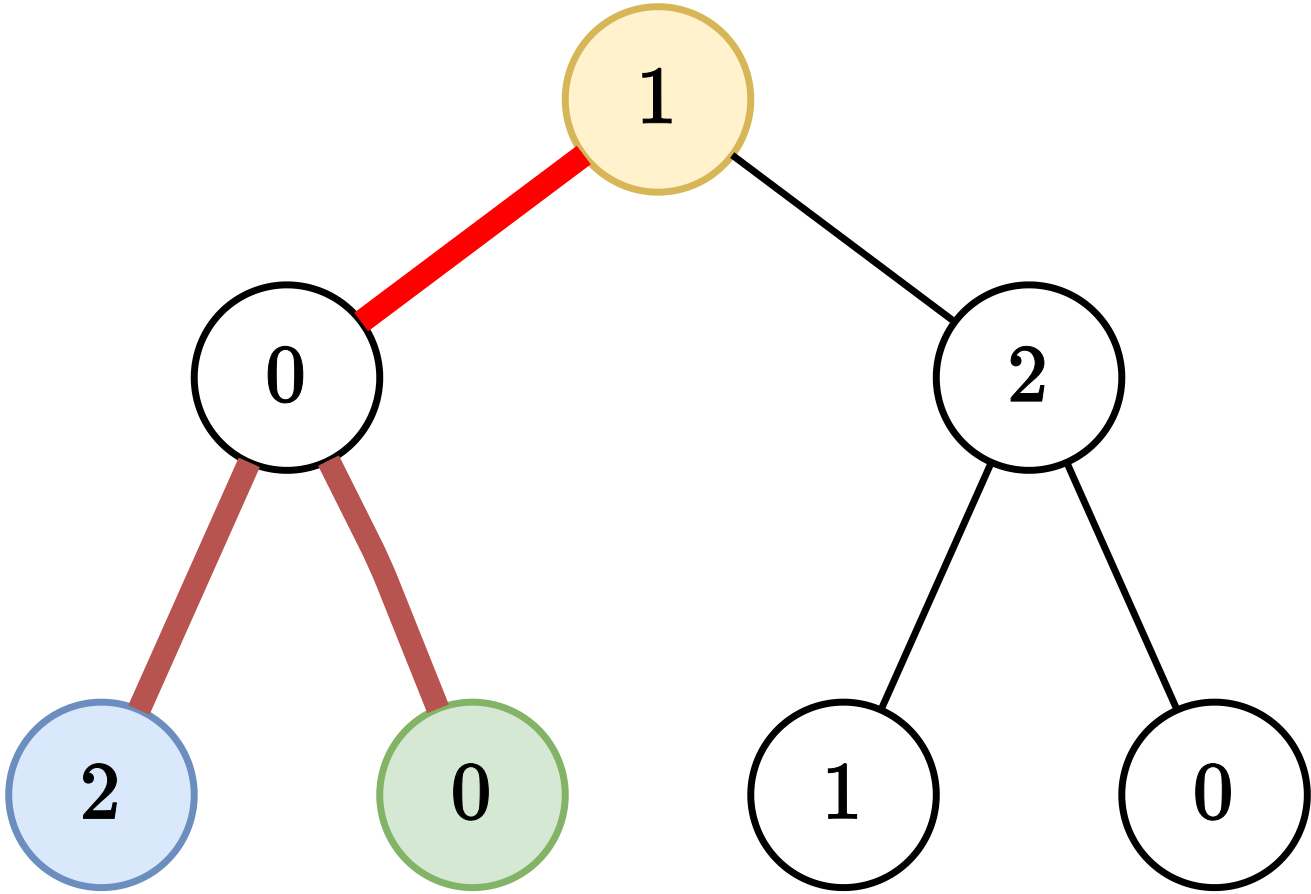
- 1 から見て 0 と 2 が別の方向にあれば良い



1 を指定されたときの $0-1-2$ パスの個数

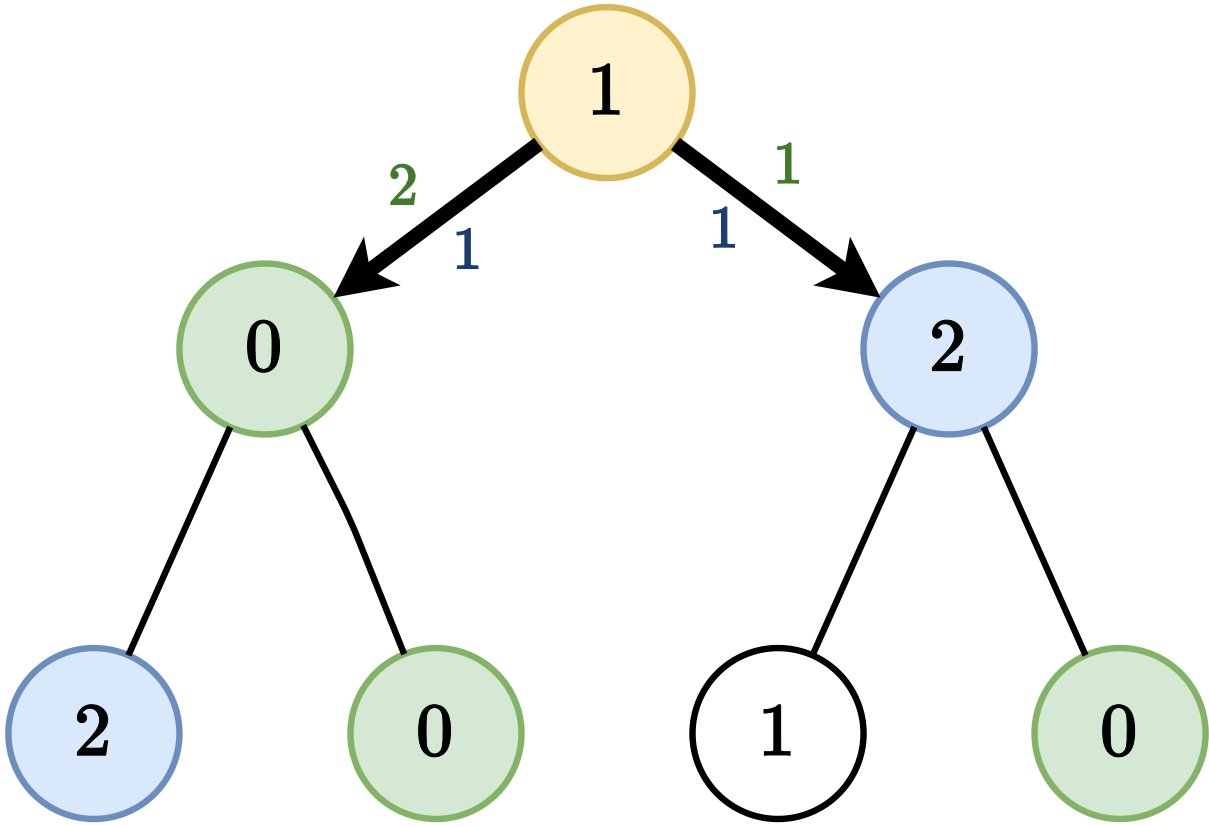
- 1 から見て 0 と 2 が別の方向にあれば良い

(同じ方向にあると，同じ辺を複数回通ってしまうため)



1 を指定されたときの $0-1-2$ パスの個数

- 1 から見て 0 と 2 が別の方向にあれば良い
- 1 から各方向への 0 の個数，2 の個数を数えて，$(0$ の個数$) \times (2$ の個数$)$ から同じ方向にあるものを引く

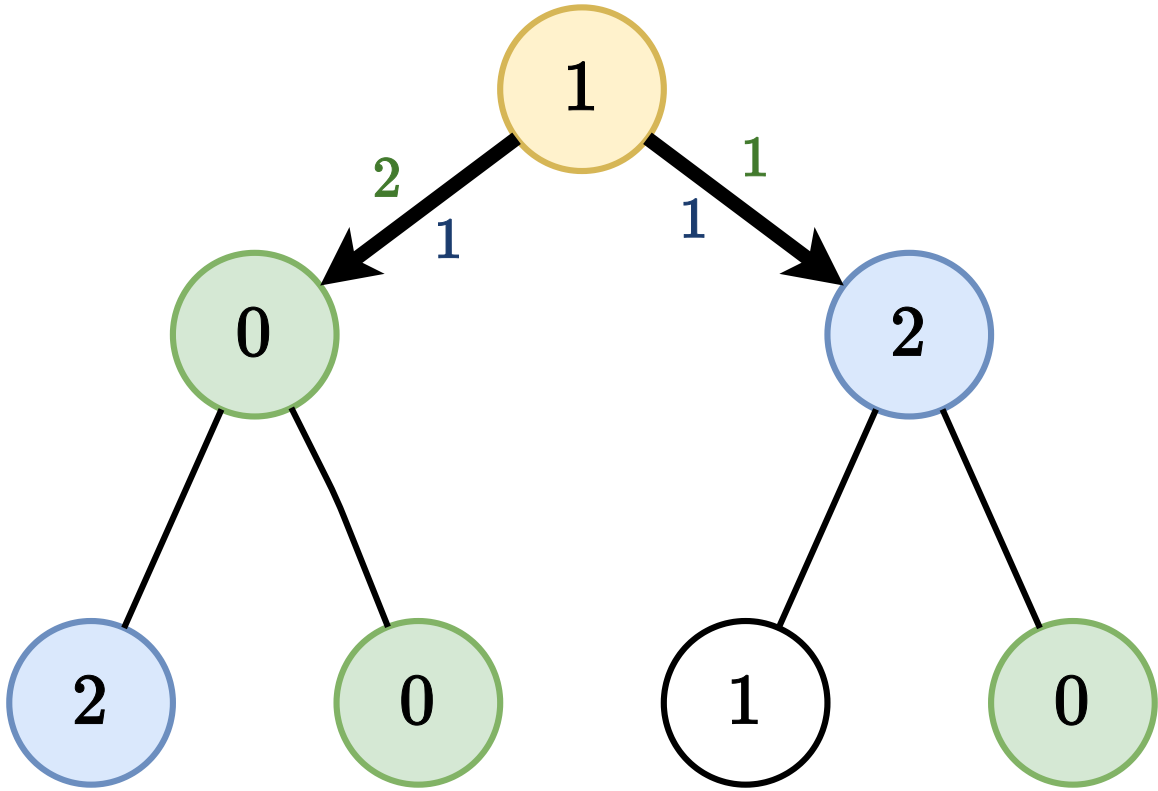


小課題 $3: Q = 0\ (6$ 点$)$

- $0 - 1 - 2$ パスの個数を $O(N)$ 時間で数えれば良い
- 先ほどの「1 を指定されたときの $0 - 1 - 2$ パスの個数」に注目！

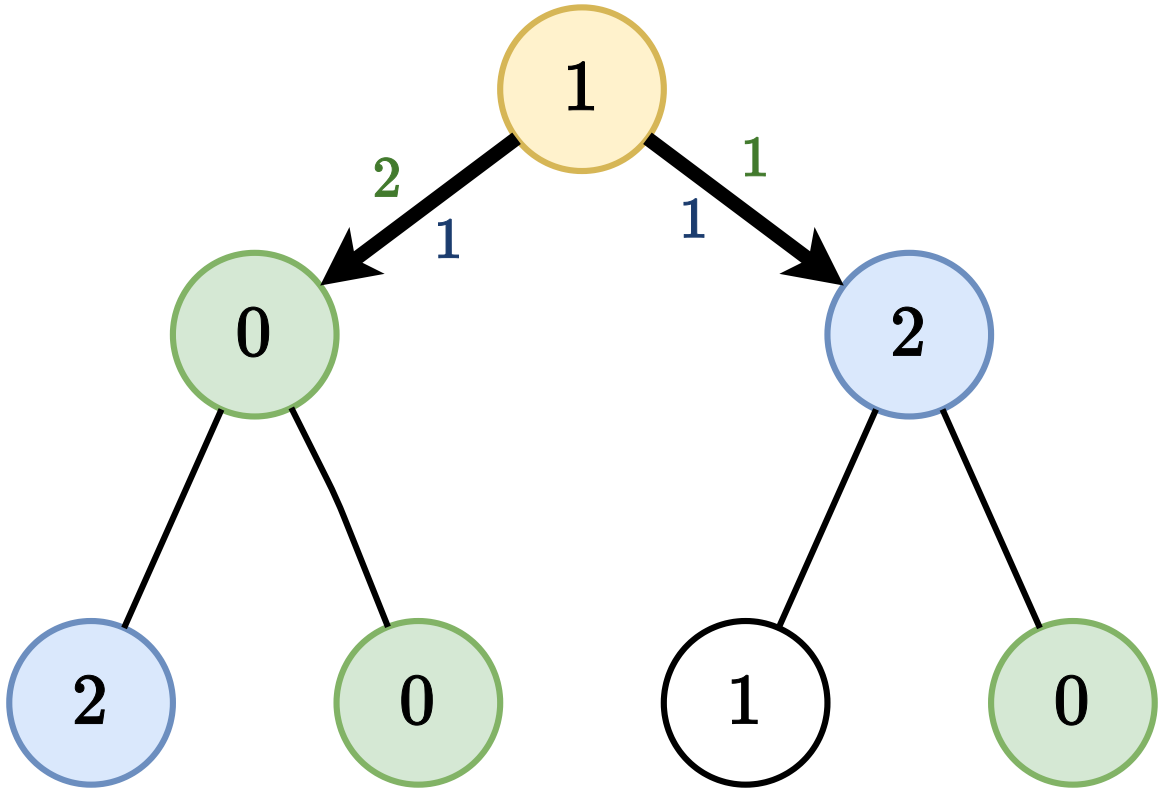
1 を指定されたときの $0 - 1 - 2$ パスの個数

- 1 から各方向への 0 の個数，2 の個数を数える



1 を指定されたときの $0 - 1 - 2$ パスの個数

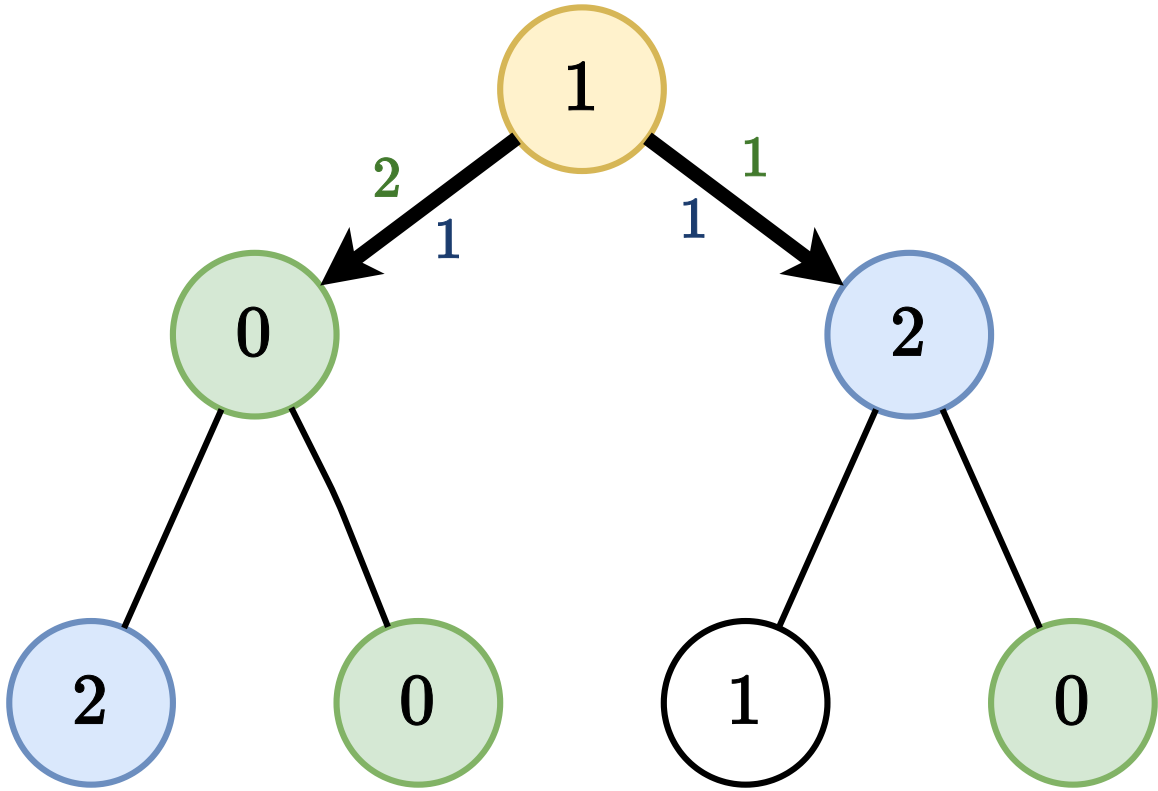
- **全頂点**から**全方向**への 0 の個数，2 の個数を数える



1 を指定されたときの $0 - 1 - 2$ パスの個数

- **全頂点**から**全方向**への 0 の個数，2 の個数を数える

→ すべての部分木について求めれば良く，$O(N)$ 時間



小課題 4 : パス (16 点)

- 変更がないとき，dp[i まで見た][j 文字一致した] の DP
- 1 点変更がある

小課題 4 : パス $(16$ 点$)$

- 変更がないとき，dp[i まで見た][j 文字一致した] の DP
- 1 点変更がある

→ **列に対する DP をセグメント木に乗せよう！**

列に対する DP をセグメント木に乗せる

- **区間**の $0 - 1 - 2$ パスの個数を求めたい

各区間は以下のデータを管理する

- $0 - 1 - 2$ の個数
- $0 - 1,\ 1 - 2,\ 1 - 0,\ 2 - 1$ の個数
- $0,\ 1,\ 2$ の個数

小課題 5 : 完全二分木 (16 点)

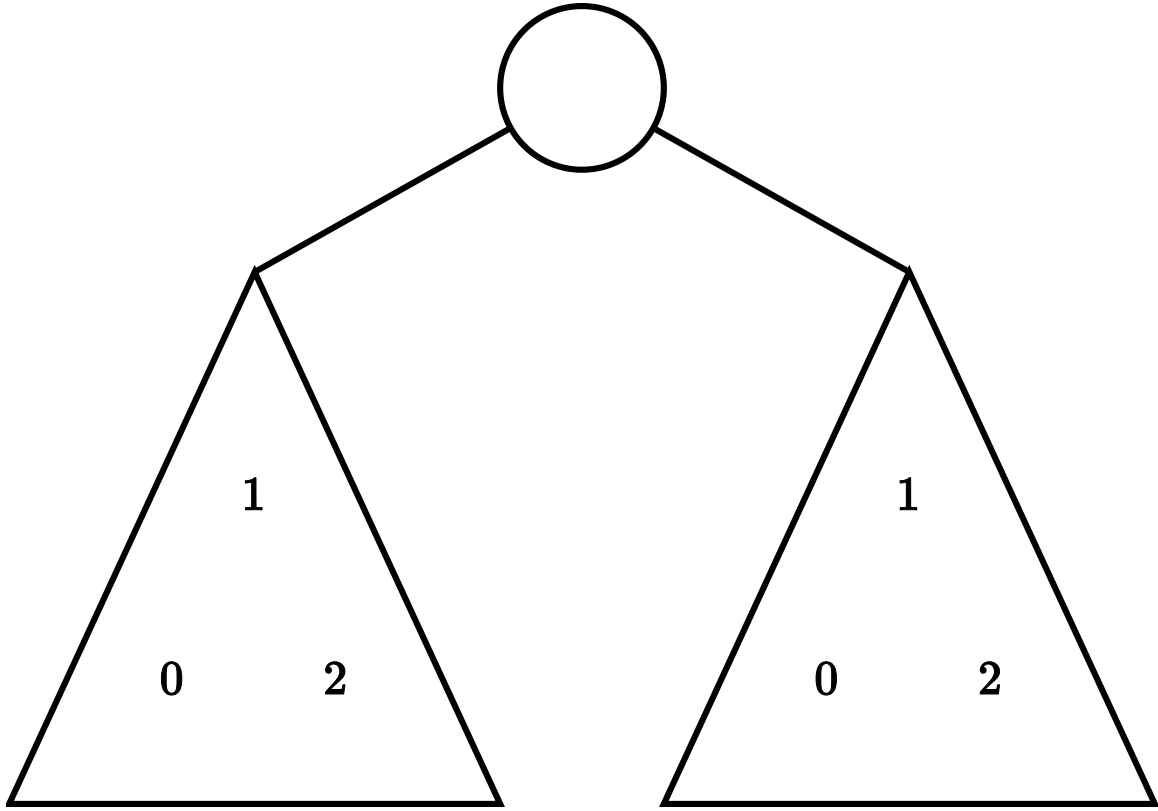
- 構造がほとんどセグメント木

→ 同様に解けないか？

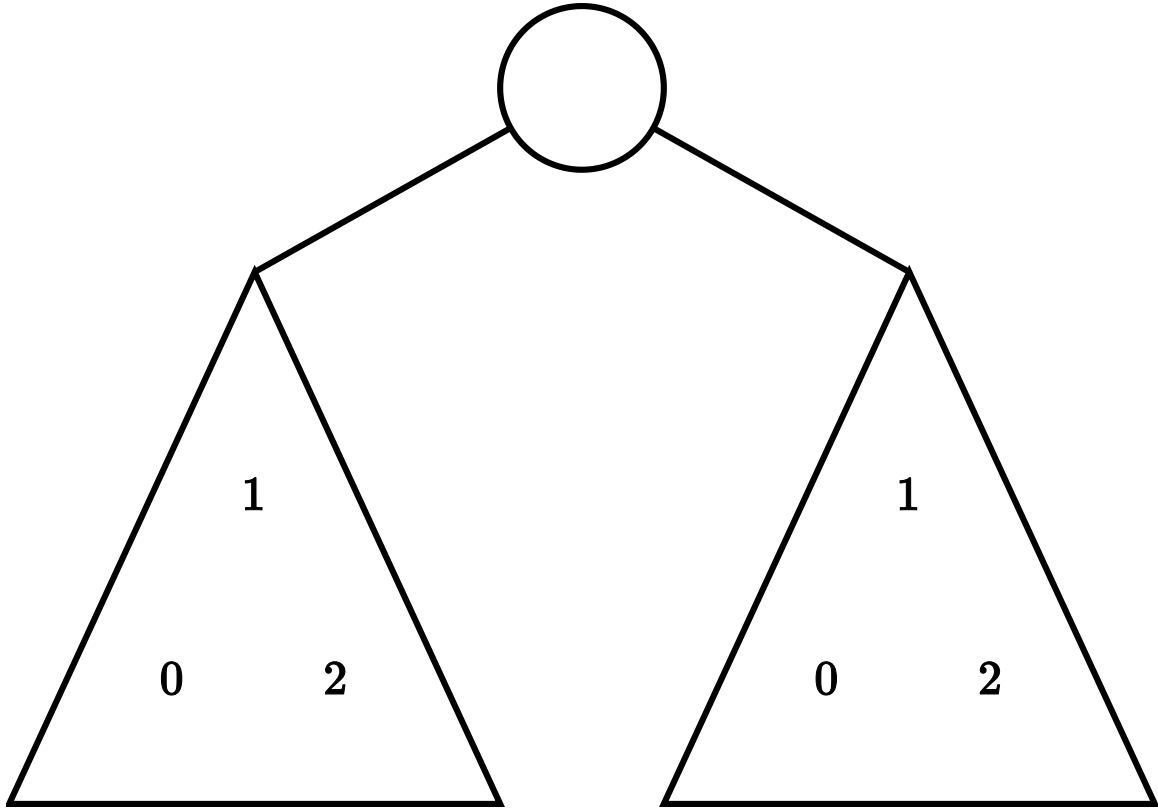
小課題 5 : 完全二分木 (16 点)

- **部分木**の $0 - 1 - 2$ パスの個数を求めたい

部分木のマージをするには…？

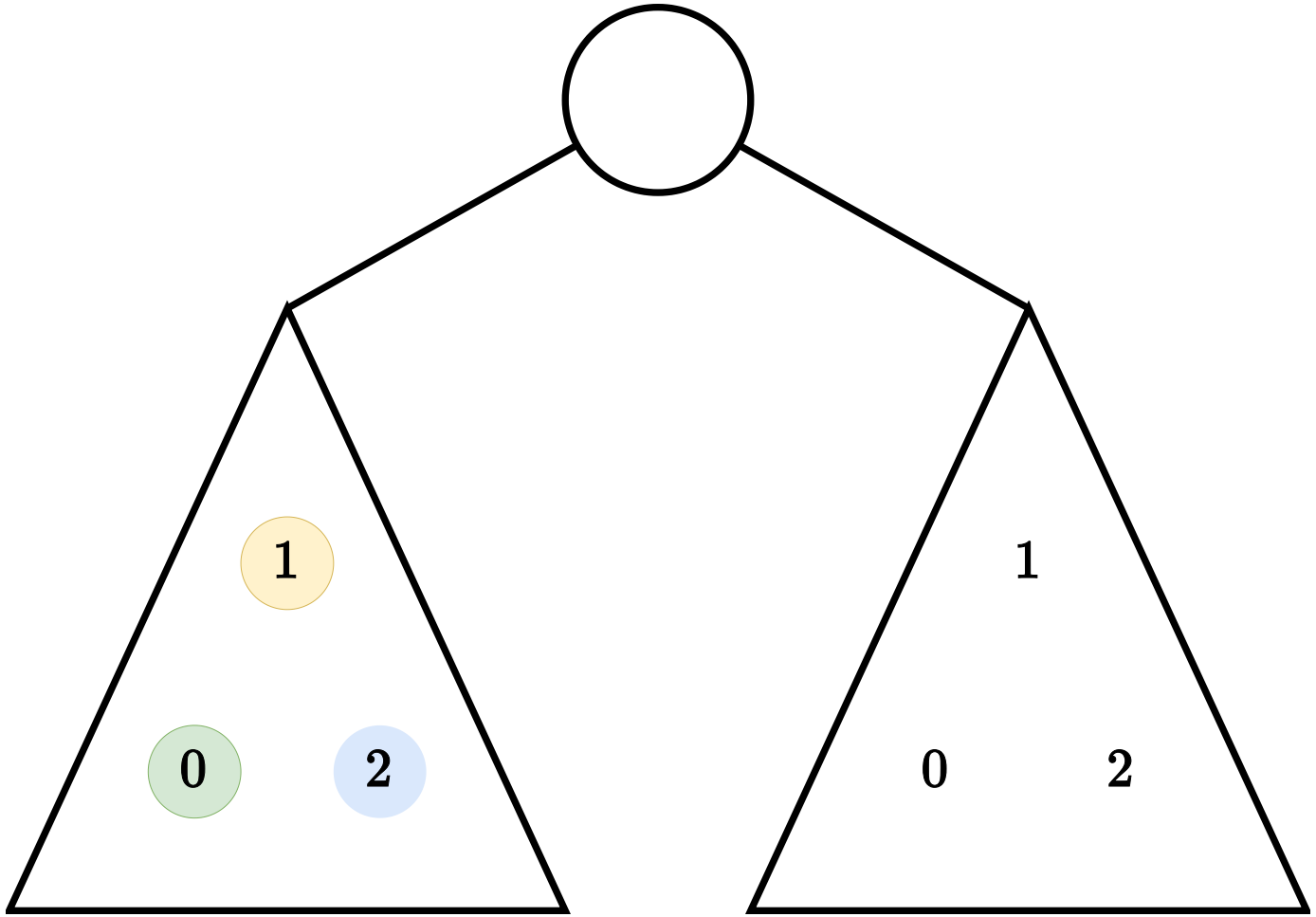


$0-1-2$ パスを
全パターン列挙しよう！



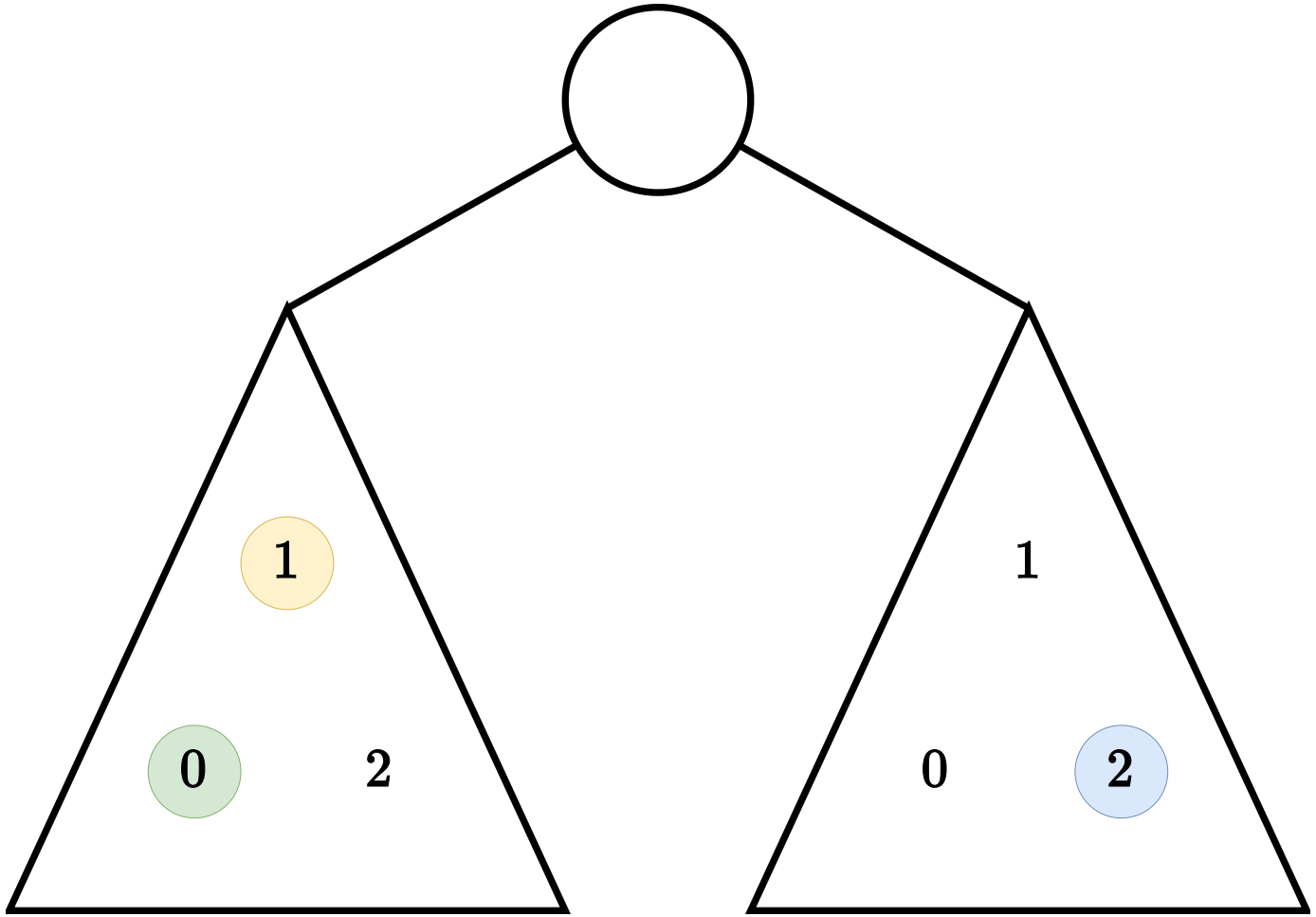
$0 - 1 - 2$ パスを全パターン列挙しよう！

- 中央の頂点を使わない場合



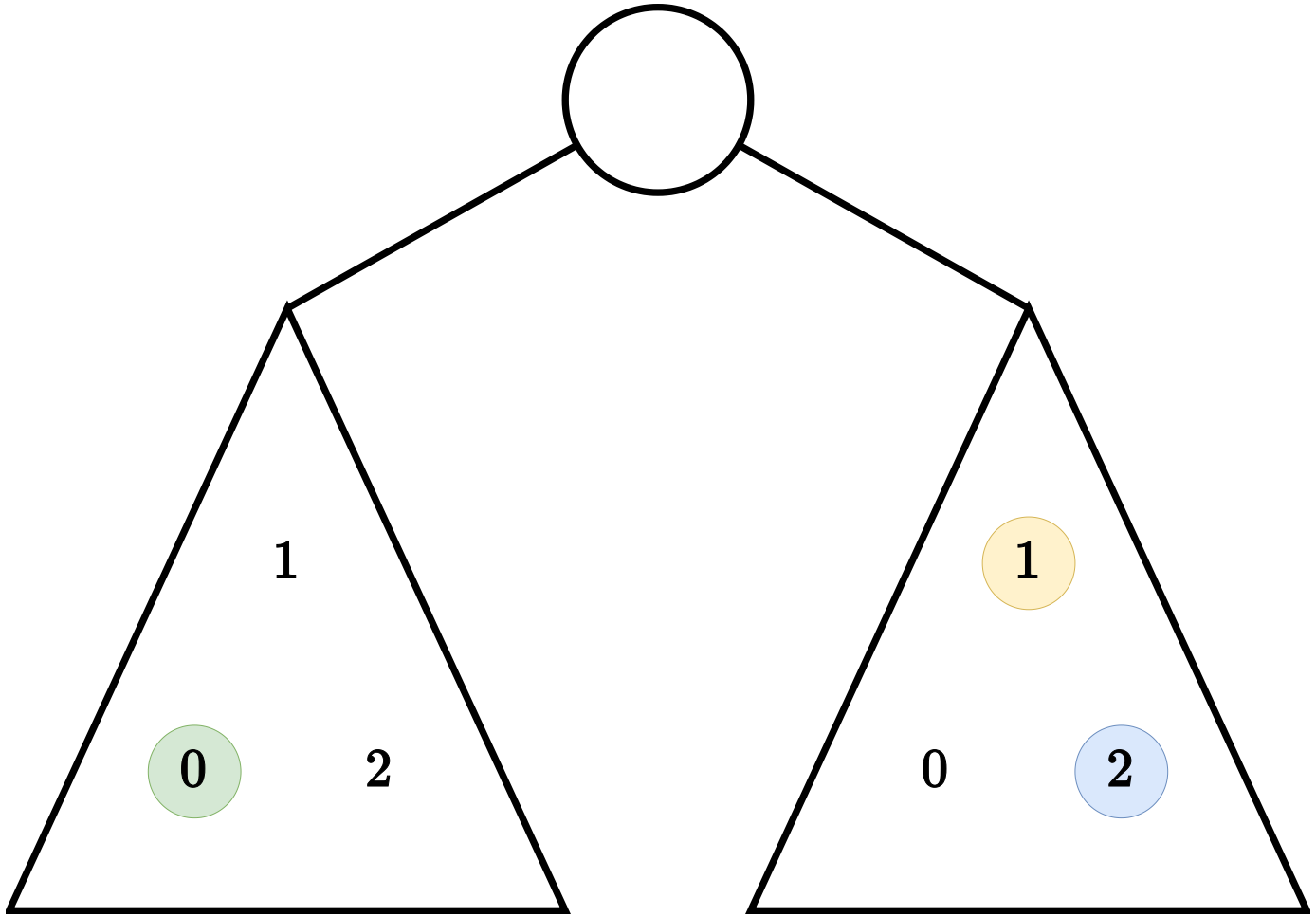
0 — 1 — 2 パスを全パターン列挙しよう！

- 中央の頂点を使わない場合



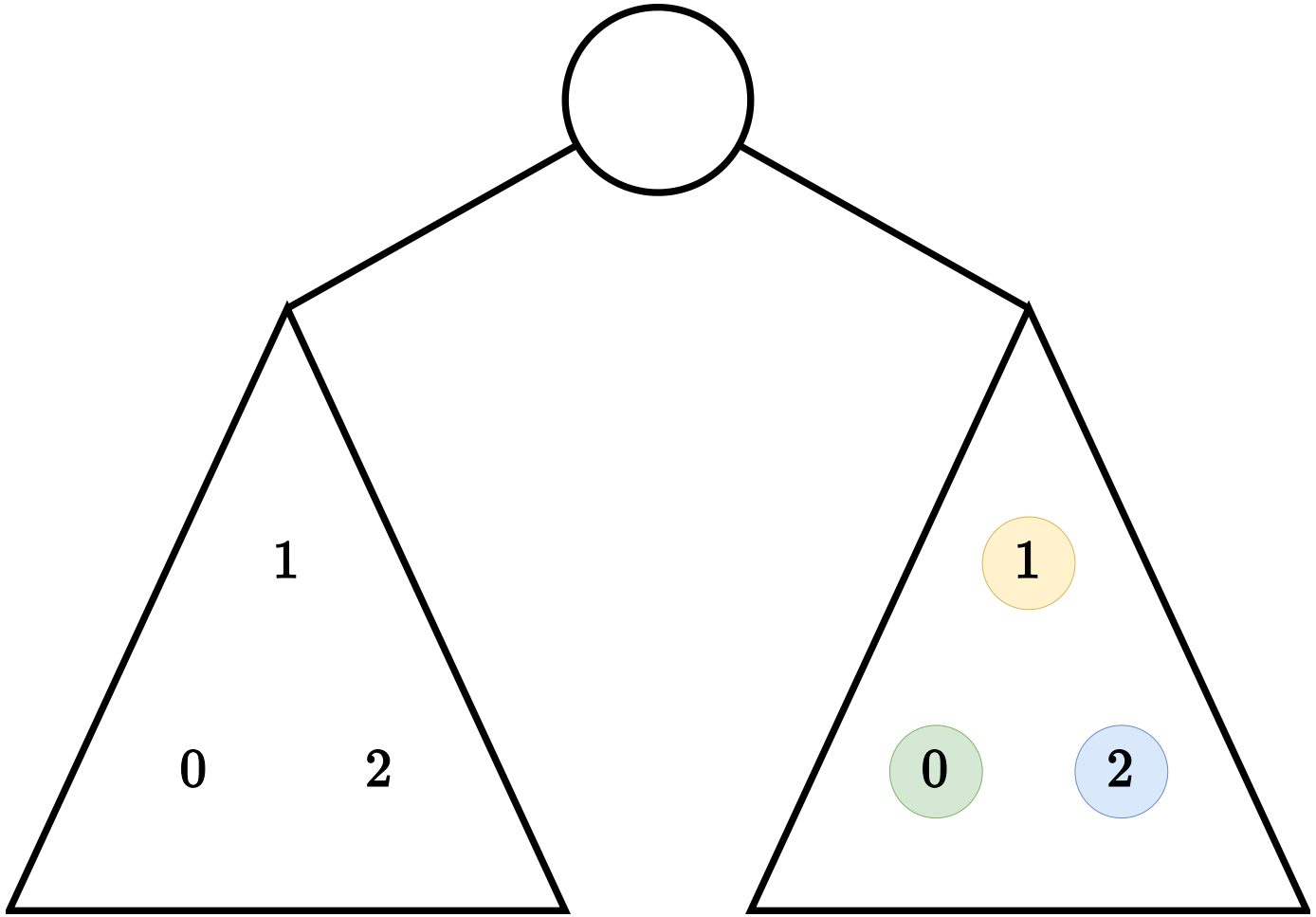
$0 - 1 - 2$ パスを全パターン列挙しよう！

- 中央の頂点を使わない場合



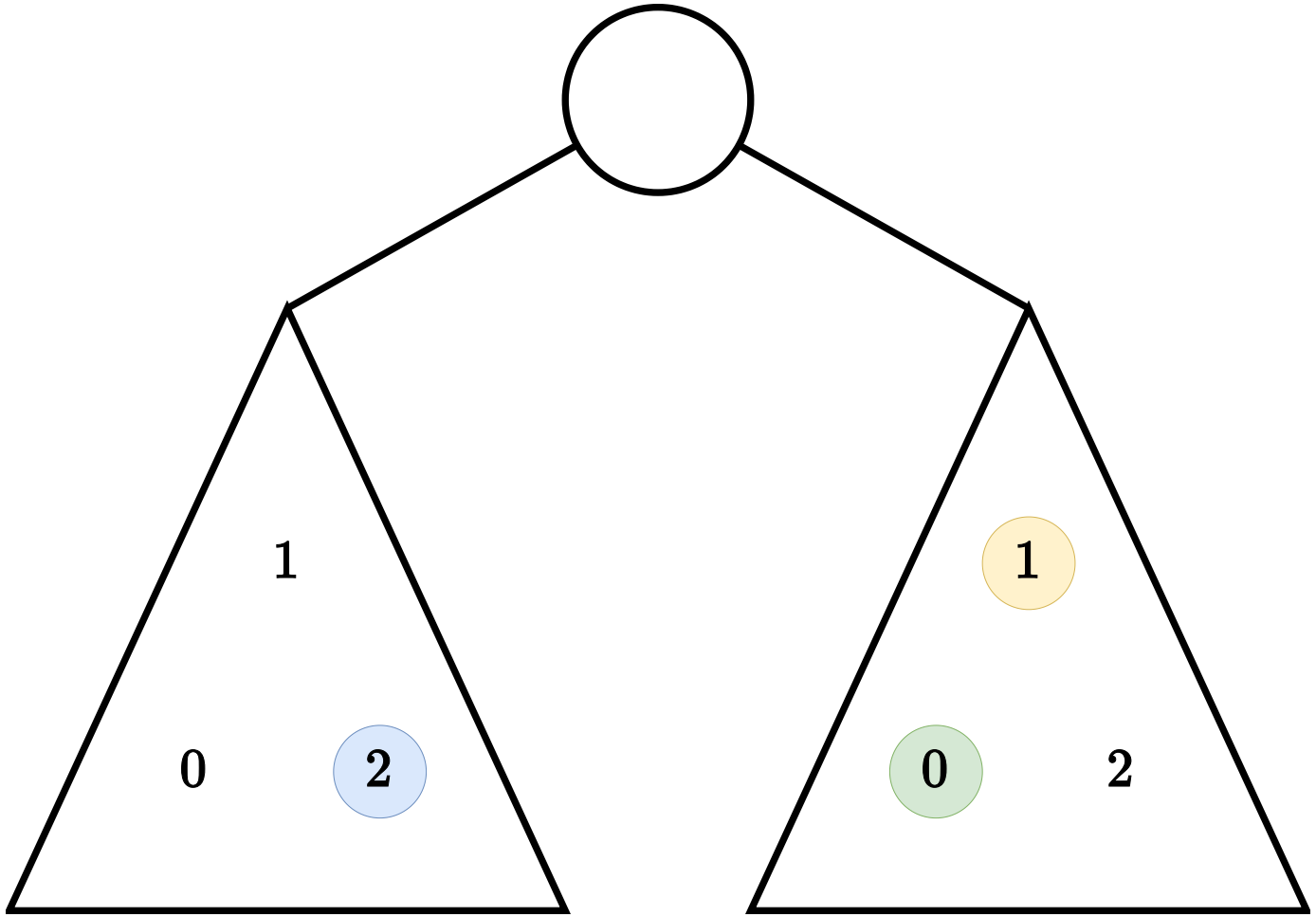
0 — 1 — 2 パスを全パターン列挙しよう！

- 中央の頂点を使わない場合



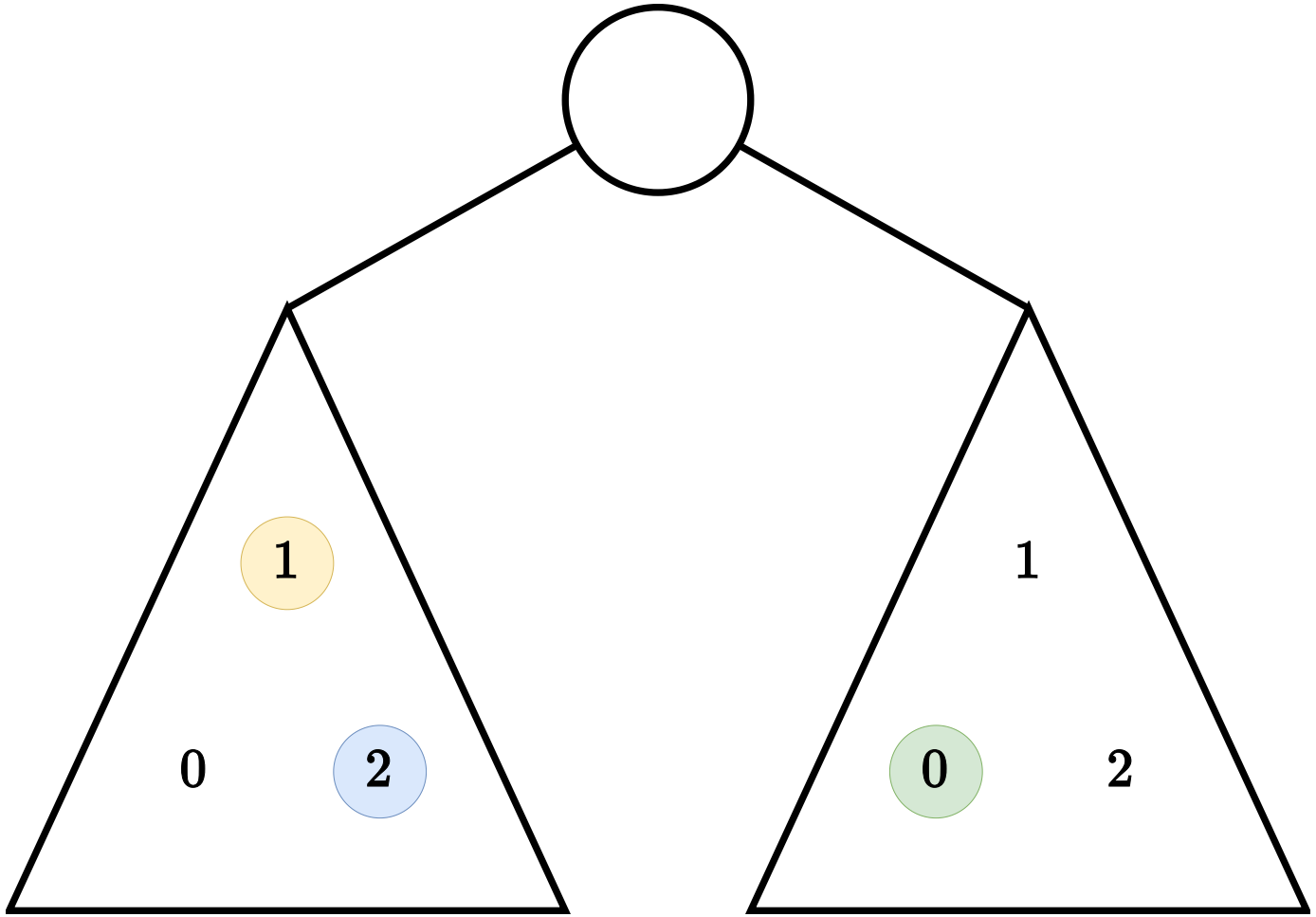
$0 - 1 - 2$ パスを全パターン列挙しよう！

- 中央の頂点を使わない場合



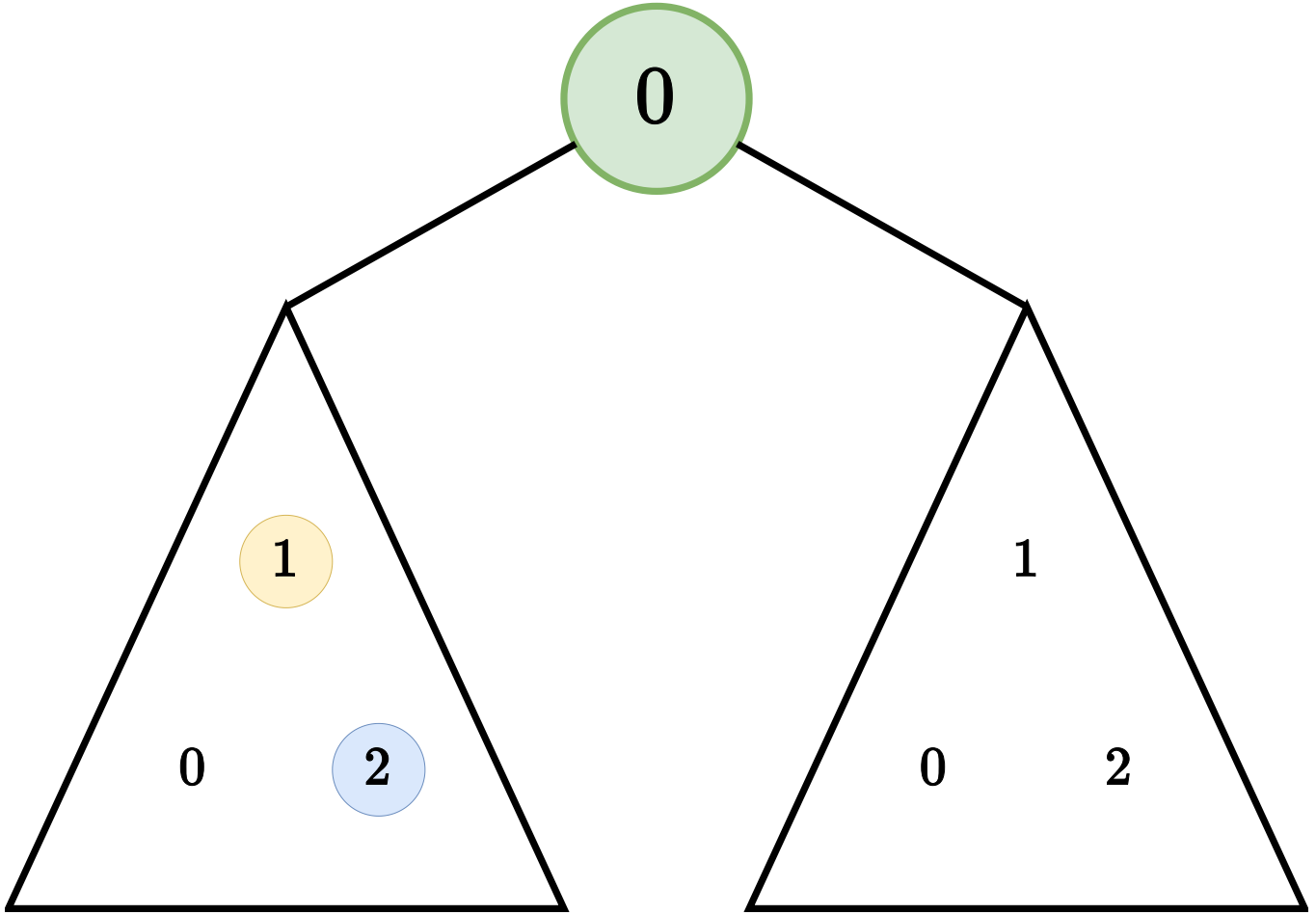
0 — 1 — 2 パスを全パターン列挙しよう！

- 中央の頂点を使わない場合



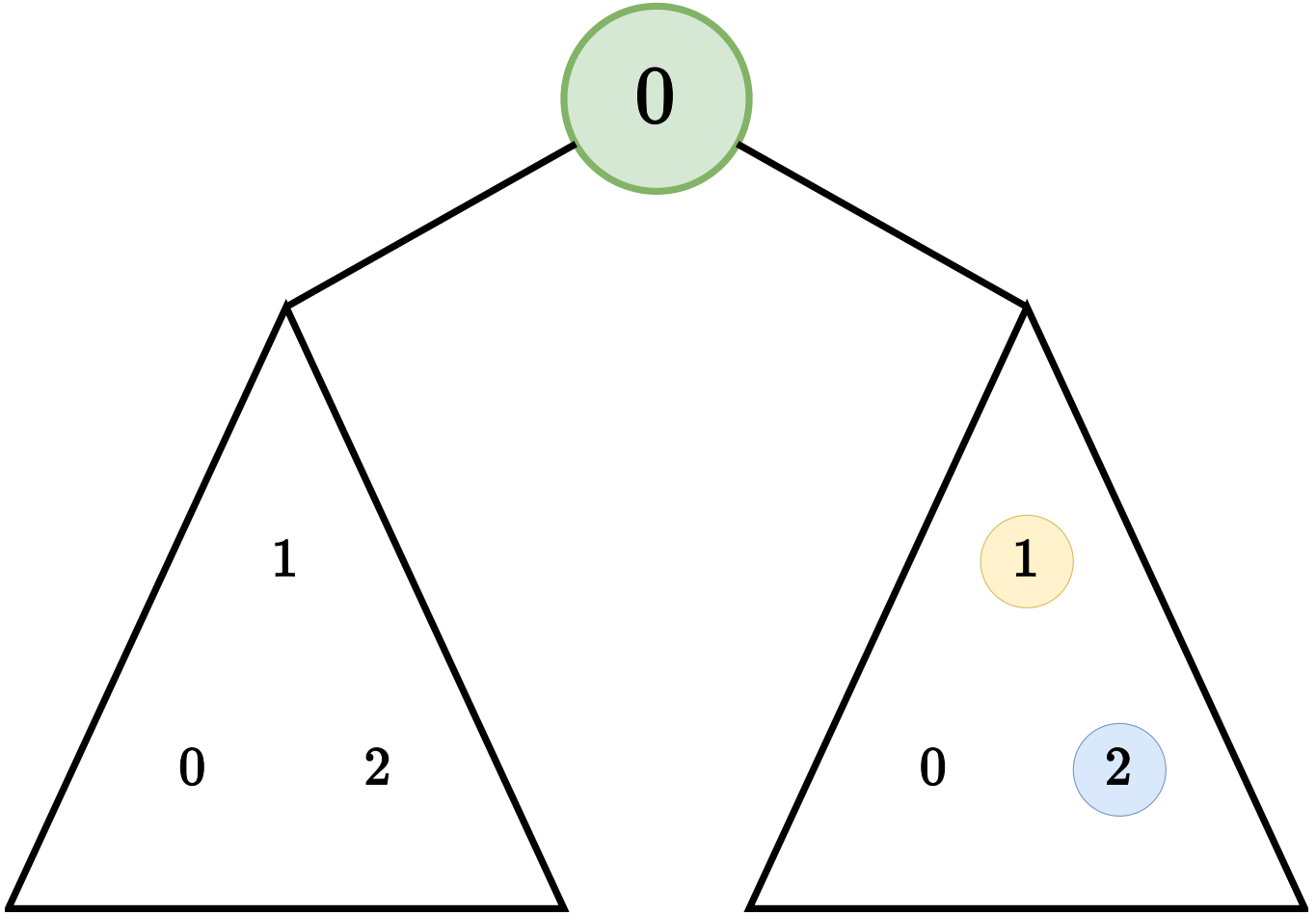
$0 - 1 - 2$ パスを全パターン列挙しよう！

- 中央の頂点を使う場合



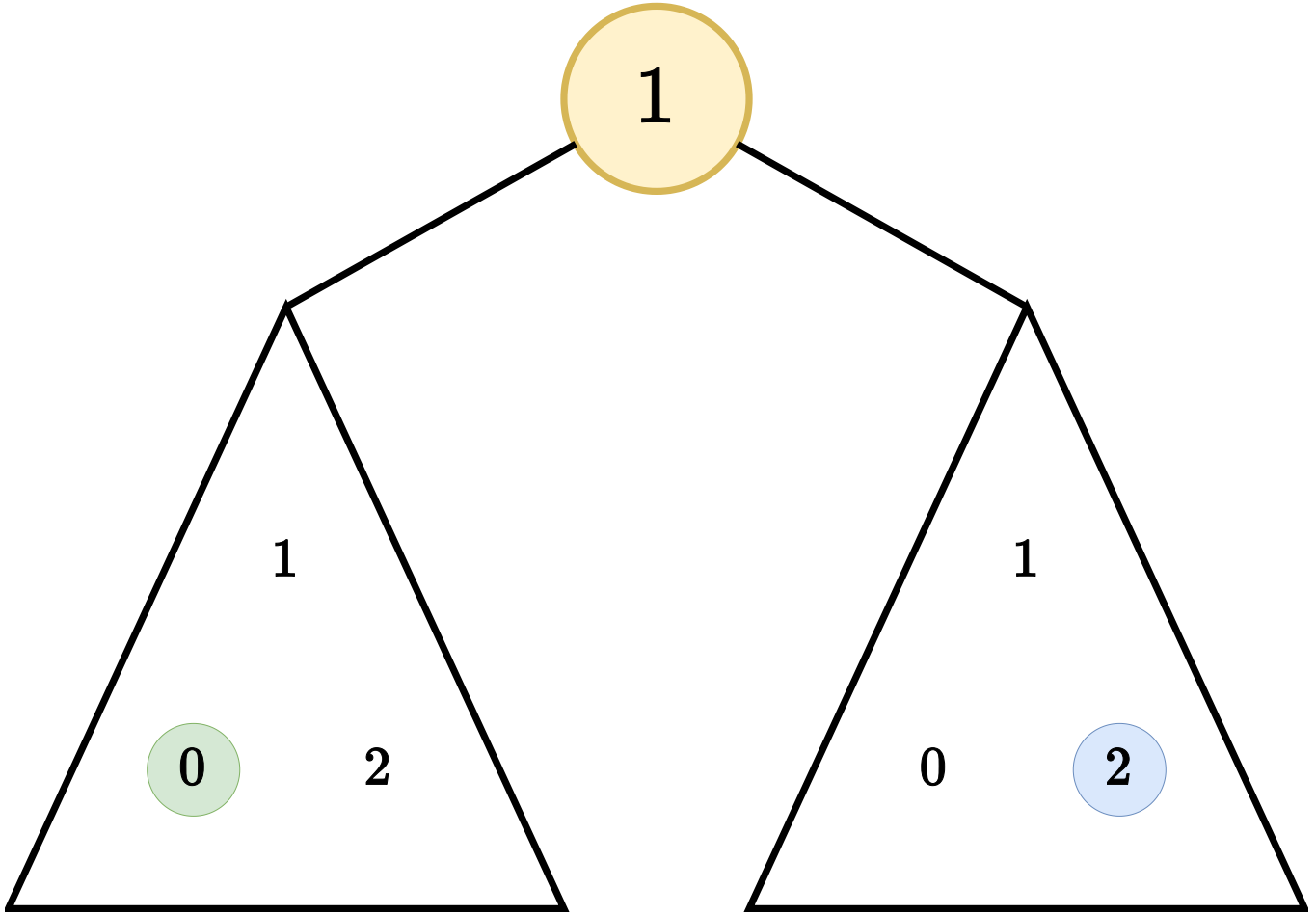
0 — 1 — 2 パスを全パターン列挙しよう！

- 中央の頂点を使う場合



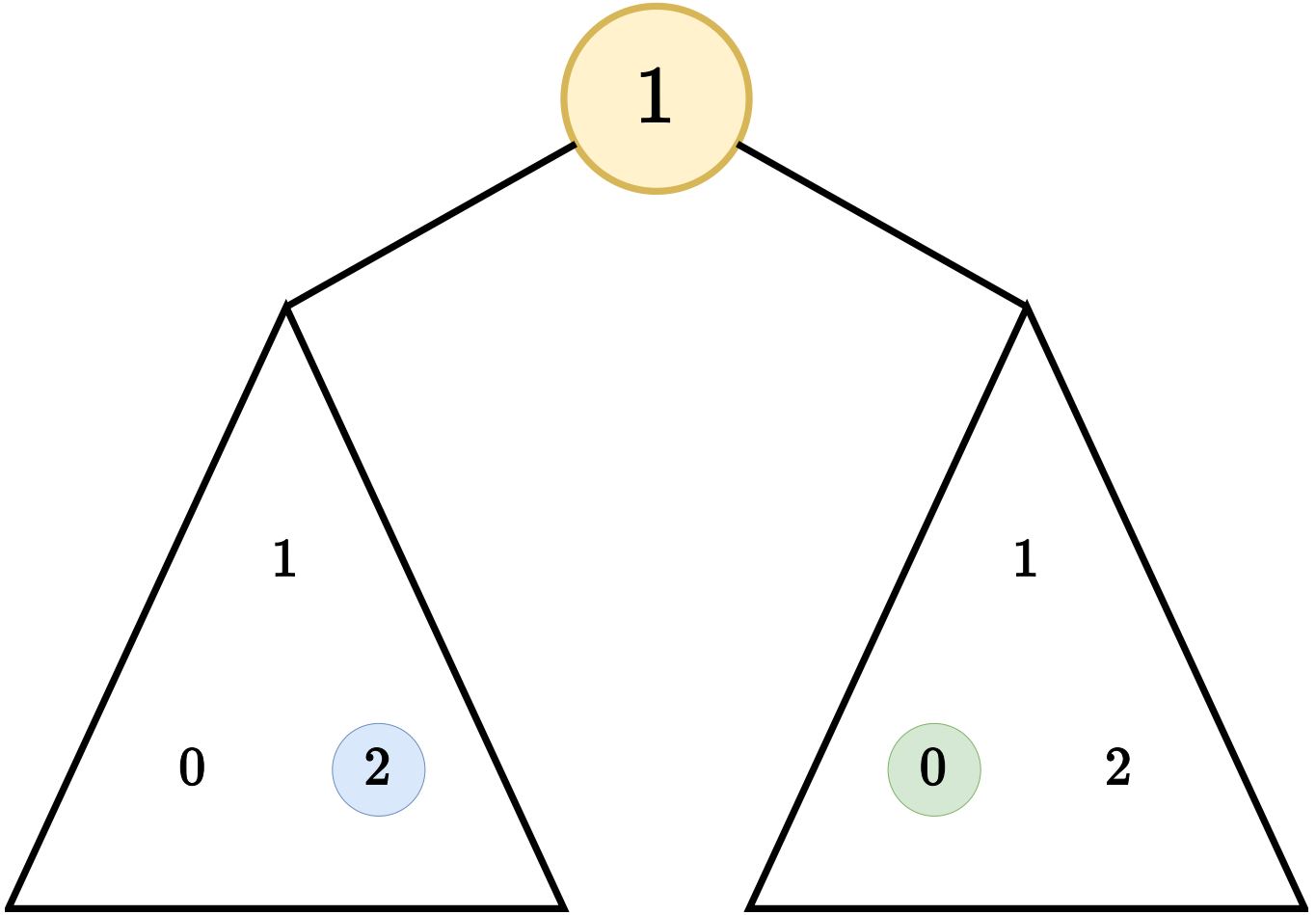
$0 - 1 - 2$ パスを全パターン列挙しよう！

- 中央の頂点を使う場合



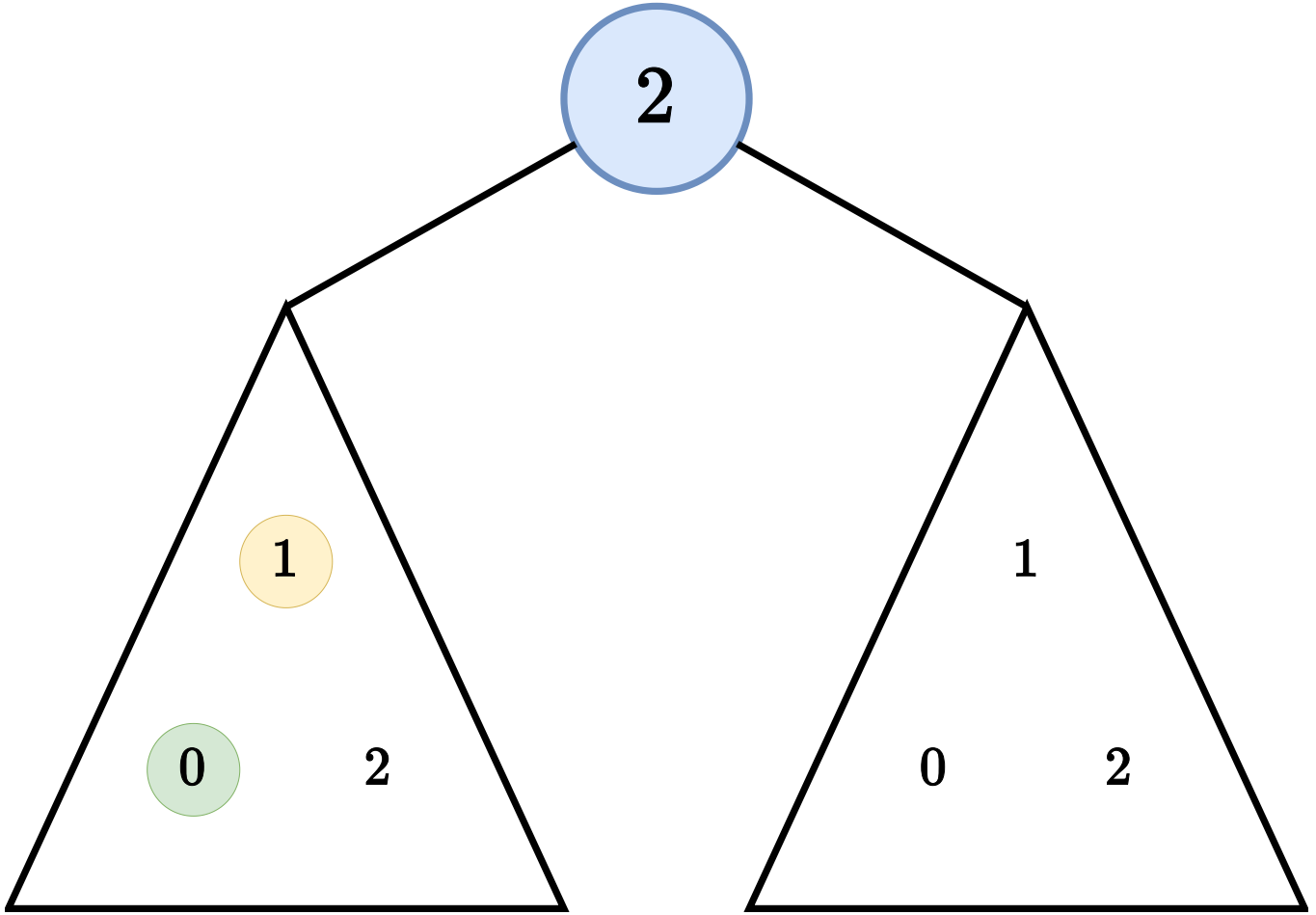
$0 - 1 - 2$ パスを全パターン列挙しよう！

- 中央の頂点を使う場合



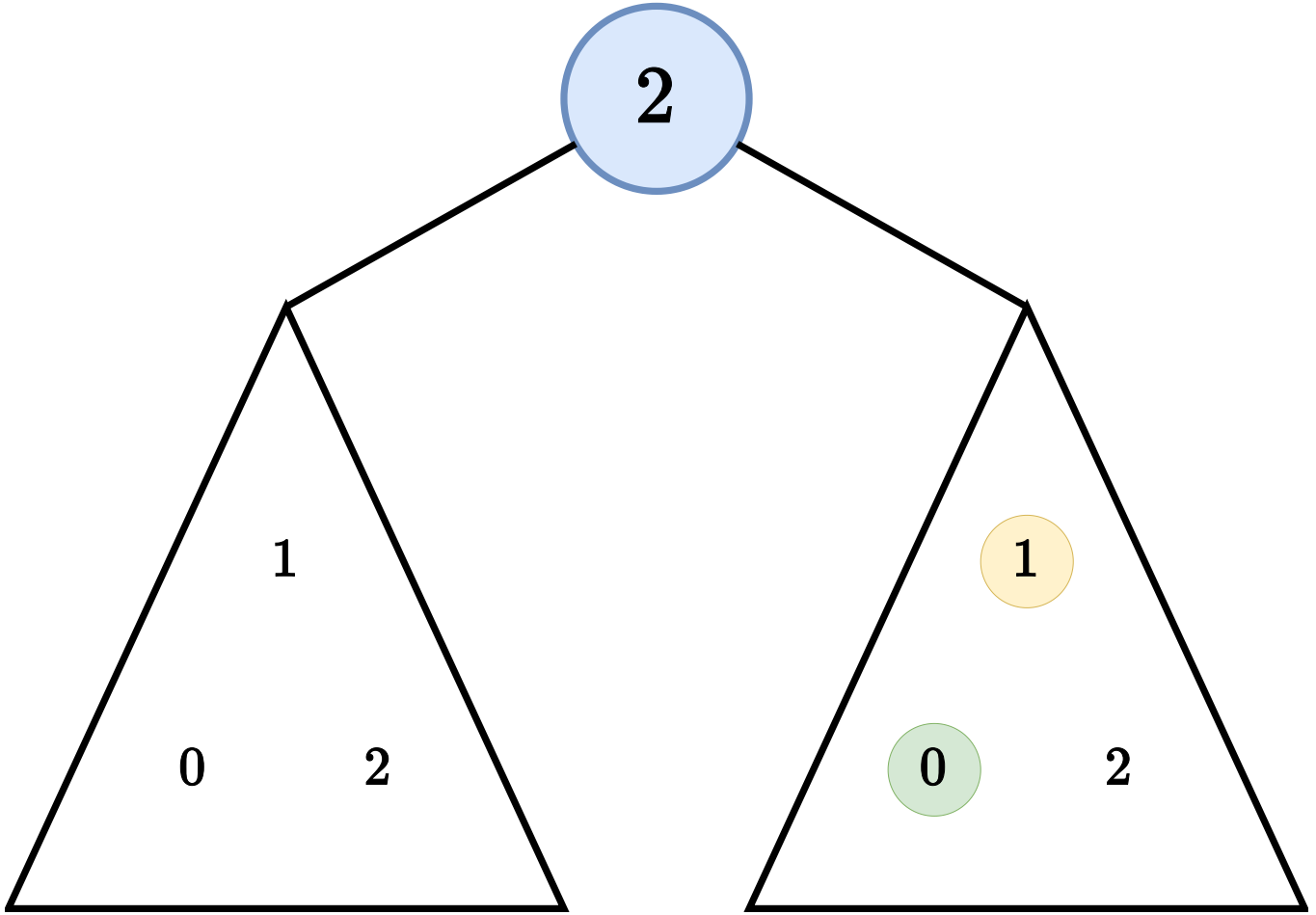
0 — 1 — 2 パスを全パターン列挙しよう！

- 中央の頂点を使う場合



0 — 1 — 2 パスを全パターン列挙しよう！

- 中央の頂点を使う場合

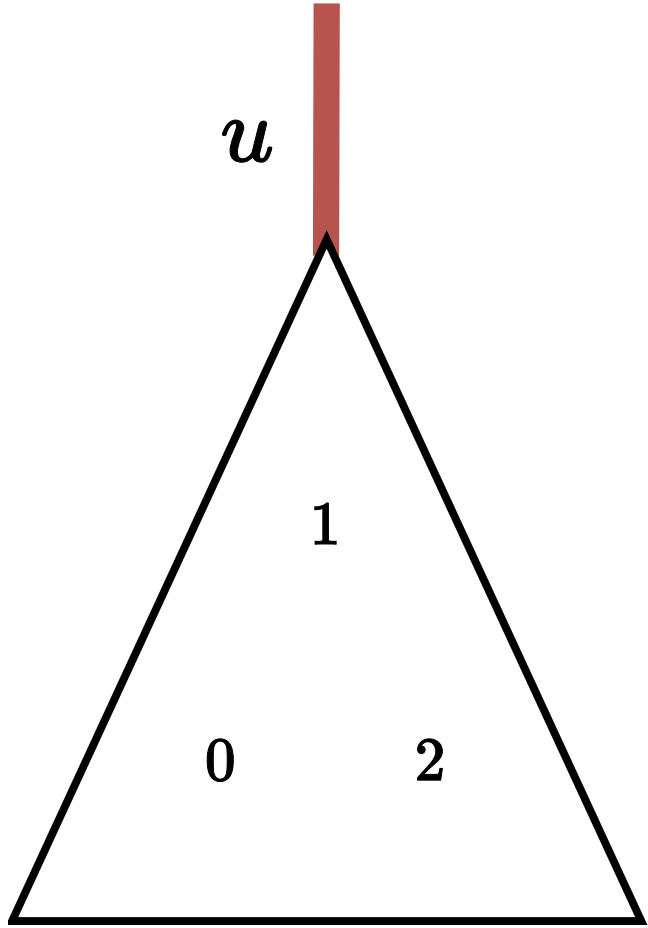


小課題 5 : 完全二分木 (16 点)

- **部分木**の $0 - 1 - 2$ パスの個数を求めたい

辺 u の部分木は以下の情報を持つ

- $0, 1, 2$ の個数
- $u - 1 - 0$, $u - 1 - 2$ パスの個数
- $0 - 1 - 2$ パスの個数



小課題 $6: N \leq 100000,\ Q \leq 25000$ (34 点)

小課題 $7: N \leq 200000,\ Q \leq 50000$ (14 点)

これを一般化したい！

小課題 $7: N \leq 200000,\ Q \leq 50000\ (14$ 点$)$

これを一般化したい！

1. 木をセグメント木のように高さの低い構造に分解
2. それぞれの構造の中で，$0, 1, 2$ の個数，$0-1,\ 1-2$ の個数などを数える

解法 A : 重心分解

- ある頂点 X を使う $0-1-2$ パスの個数は，以下が分かれば良い
  - $F_X = 0 : X-1-2$ パスの個数
  - $F_X = 1 : X$ から各方向への $0, 2$ の個数
  - $F_X = 2 : X-1-0$ パスの個数

解法 A : 重心分解

- ある頂点 X を**通過する** $0-1-2$ パスの個数は，以下が分かれば良い
  - X から各方向への $0,\ 2$ の個数
  - X から各方向への $X-1-(0/2)$ パスの個数

解法 A : 重心分解

重心分解で生じるそれぞれの木について，これらの情報を持つ

(重心を X とする)

- 全体の $0, 2$ の個数
- X から各方向への $0, 2$ の個数
- 全体の $X-1-0,\ X-1-2$ パスの個数
- X から各方向への $X-1-0,\ X-1-2$ パスの個数

解法 A : 重心分解

重心分解で生じるそれぞれの木について，これらの情報を持つ

(重心を X とする)

- 全体の $0, 2$ の個数
- X から各方向への $0, 2$ の個数
- 全体の $X-1-0,\ X-1-2$ パスの個数
- X から各方向への $X-1-0,\ X-1-2$ パスの個数

1 点変更をするとき，変更される重心分解上の木は $O(\log N)$ 個

解法 A : 重心分解

1 点変更をしたときの

- X から各方向への $X-1-0,\ X-1-2$ パスの個数

の差分を $O(\log N)$ 時間で求めたい

「X から各方向への $X-1-2$ パスの個数」の差分

- ある頂点の 2 を消したとき:
  - $X-2$ のパス上の 1 の数だけ，その方向のパスの個数が減る
- ある頂点の 1 を消したとき:
  - 1 以下の部分木内の 2 の数だけ，その方向のパスの個数が減る

「X から各方向への $X-1-2$ パスの個数」の差分

- ある頂点の 2 を消したとき:
  - $X-2$ のパス上の 1 の数だけ，その方向のパスの個数が減る

→ オイラーツアー + BIT で 1 の数を管理すれば $O(\log N)$ 時間

(行きが重み 1 , 帰りが重み -1 の辺属性オイラーツアー)

「X から各方向への $X-1-2$ パスの個数」の差分

- ある頂点の 1 を消したとき:
    - 1 以下の部分木内の 2 の数だけ，その方向のパスの個数が減る

→ オイラーツアー + BIT で 2 の数を管理すれば $O(\log N)$ 時間

解法 A : 重心分解

1. 重心分解
2. それぞれの木の各方向について，以下を構築
    - 根からのパス上の 1 の個数を管理する，
      辺属性オイラーツアー + BIT
    - 部分木内の $0, 2$ の個数を管理する，
      頂点属性オイラーツアー + BIT

ひたすらがんばると，$O(N \log N + Q(\log N)^2)$ 時間

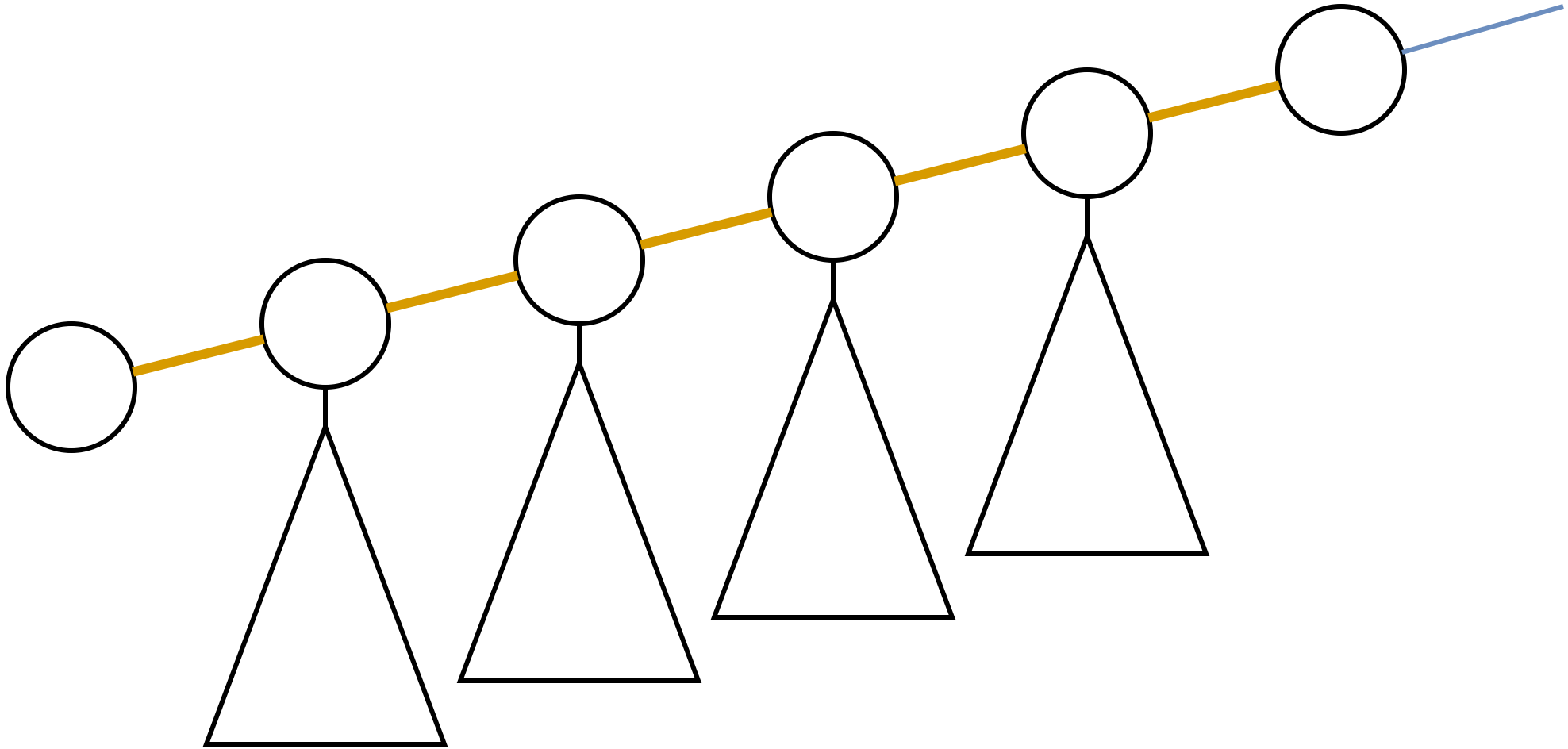
解法 B : HL 分解

- パスの場合は，セグメント木で解ける

→ HL 分解で，高さ $O(\log N)$ のパスに分解

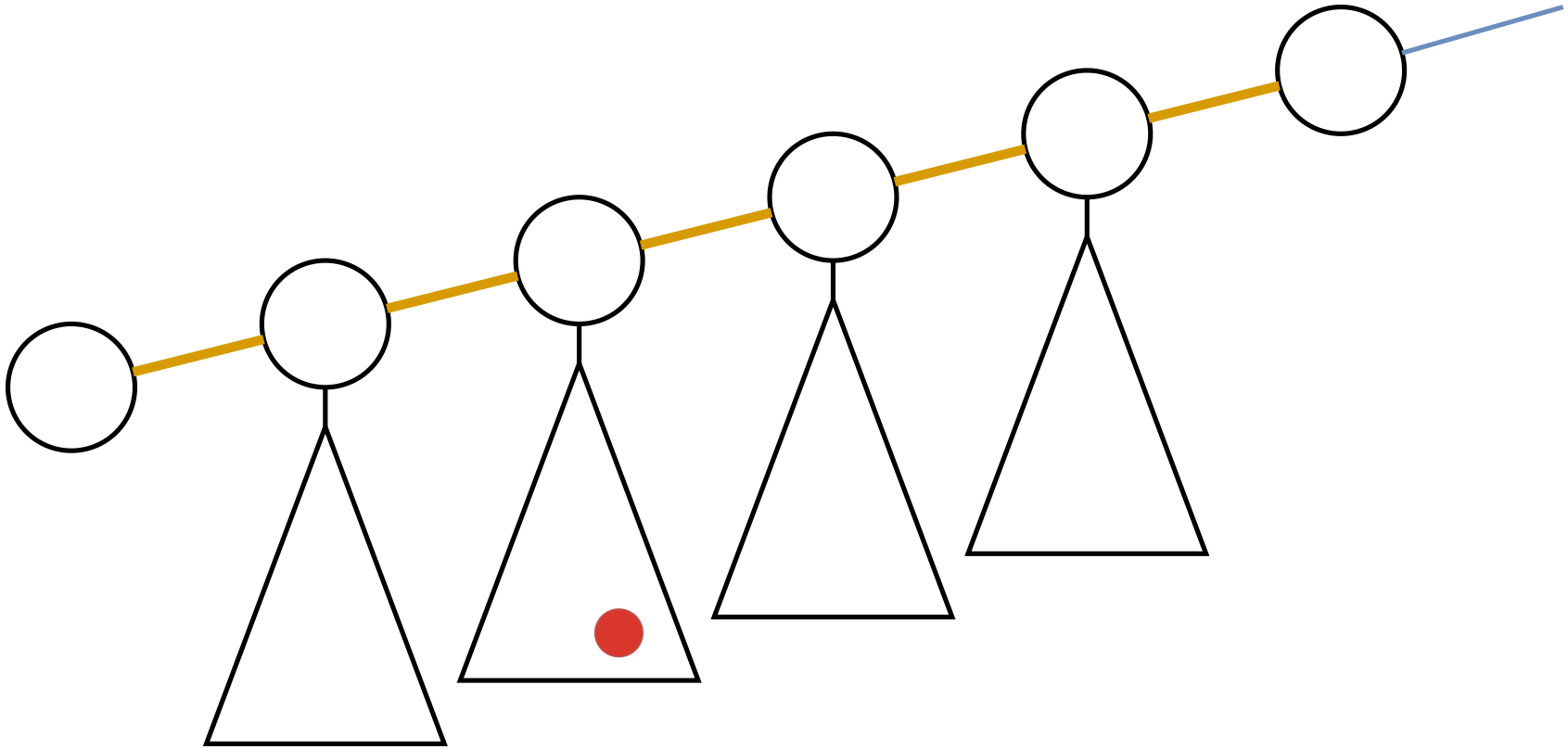
解法 B : HL 分解

- 高さ $O(\log N)$ のパスに分解，パスごとにセグ木を構築



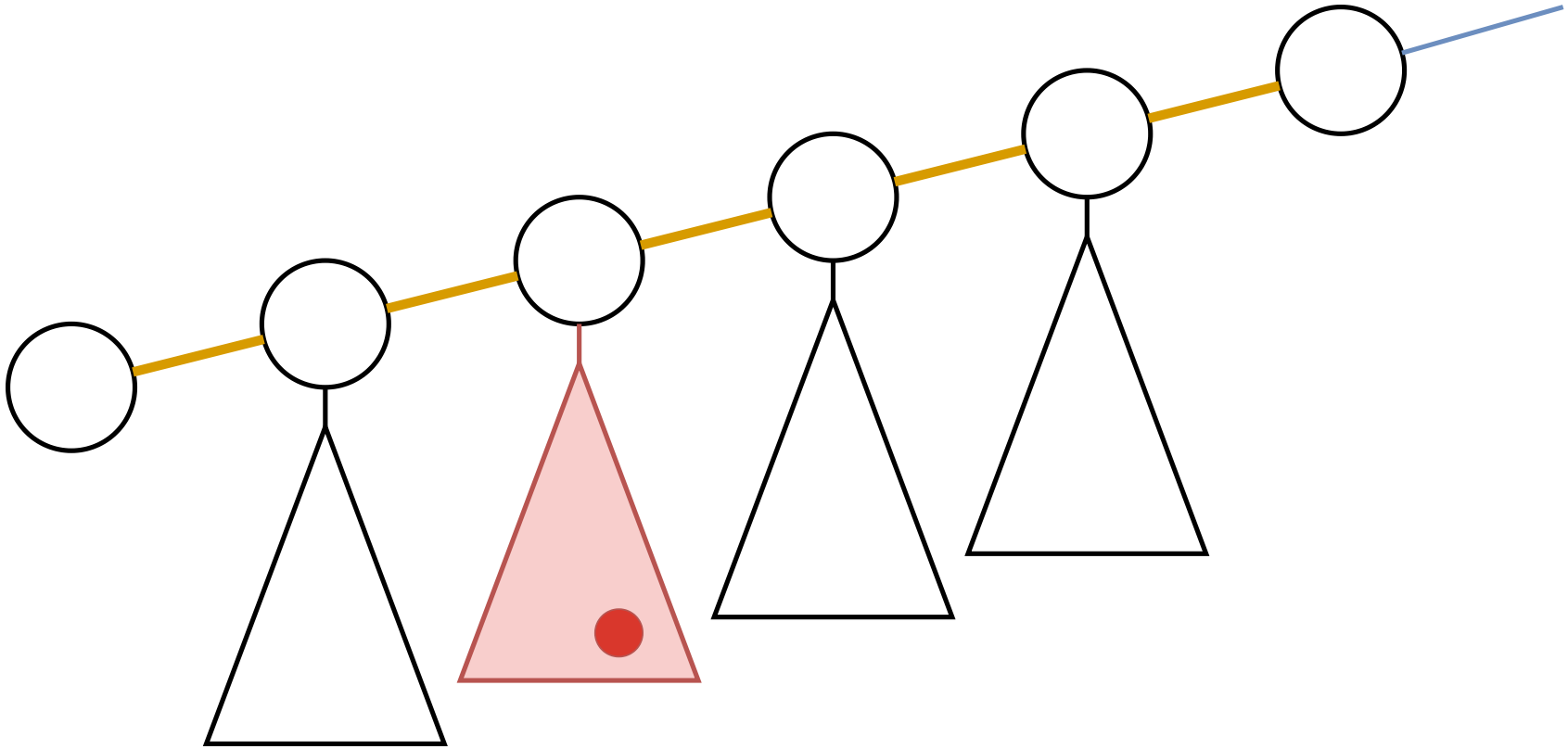
解法 B : HL 分解

- 高さ $O(\log N)$ のパスに分解，パスごとにセグ木を構築
- 1 点変更があったら，



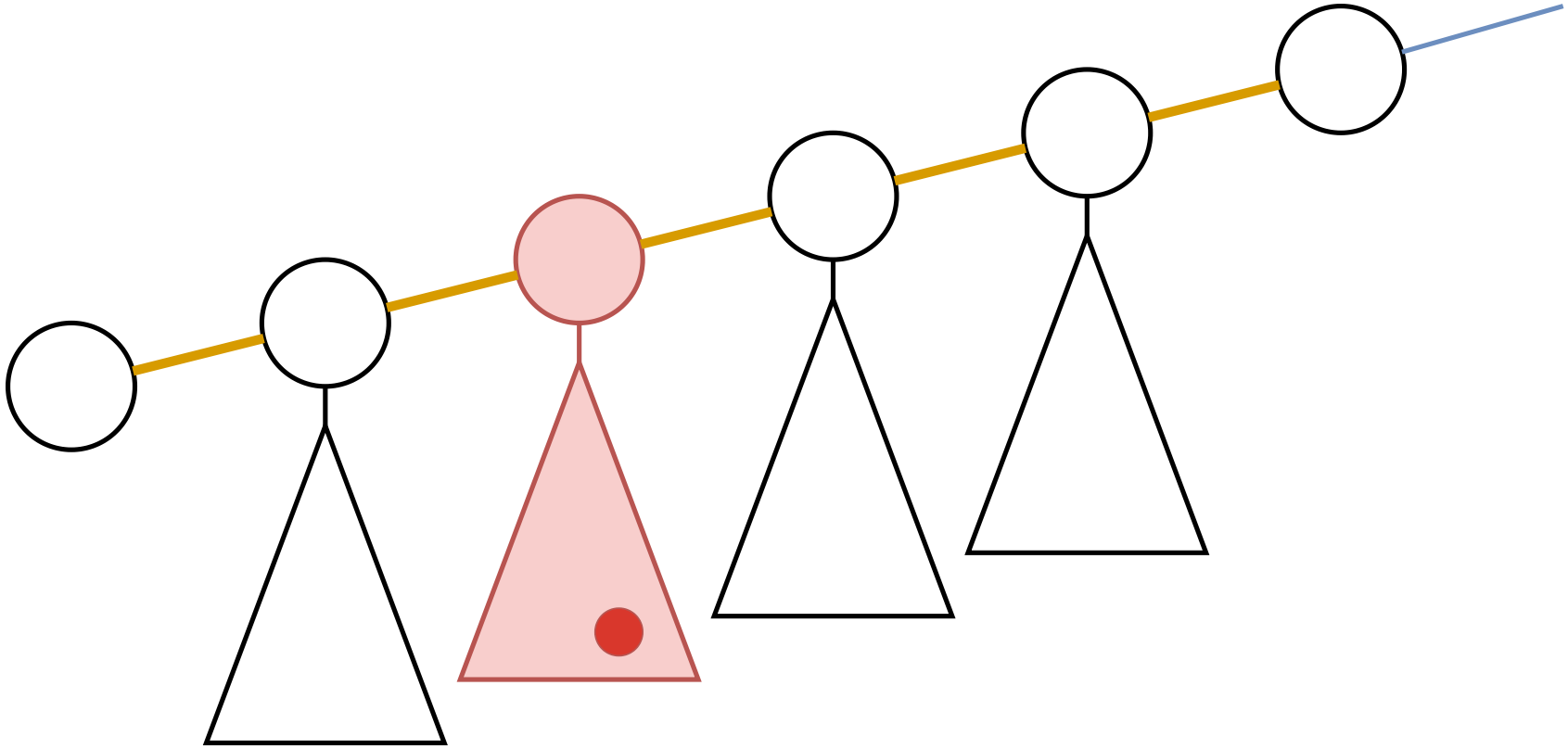
解法 B : HL 分解

- 高さ $O(\log N)$ のパスに分解，パスごとにセグ木を構築
- 1 点変更があったらその部分木を更新して，



解法 B : HL 分解

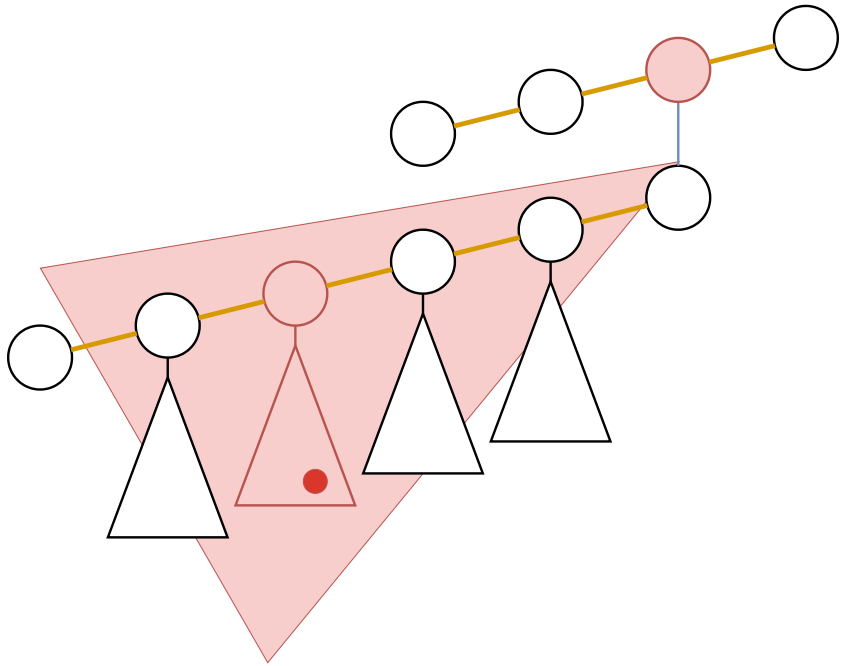
- 1 点変更があったらその部分木を更新して，部分木全体の結果でパス上のセグ木を 1 点更新



小課題 $6, 7 : N \leq 200000, Q \leq 50000 (48$ 点$)$

解法 B : HL 分解

- 1 点変更があったらその部分木を更新して，部分木全体の結果でパス上のセグ木を 1 点更新
- さらにその結果で上のパス上のセグ木を 1 点更新...



解法 B : HL 分解

セグ木の各ノードは以下の情報を持つ

- $0, 2$ の個数
- パス上の 1 の個数
- (区間の左端 / 右端) $-1-(0/2)$ パスの個数
- $0-1-2$ パスの個数

ひたすらがんばると，$O(N \log N + Q(\log N)^2)$ 時間

小課題 $6, 7 : N \leq 200000, Q \leq 50000 (48$ 点$)$

$O((N+Q)\log N)$ 時間で解けます！

解法 C : static top tree

木に対する static top tree は，列に対するセグ木を一般化したようなもの

実は実装はこの方針の方がちょっと簡単かも？

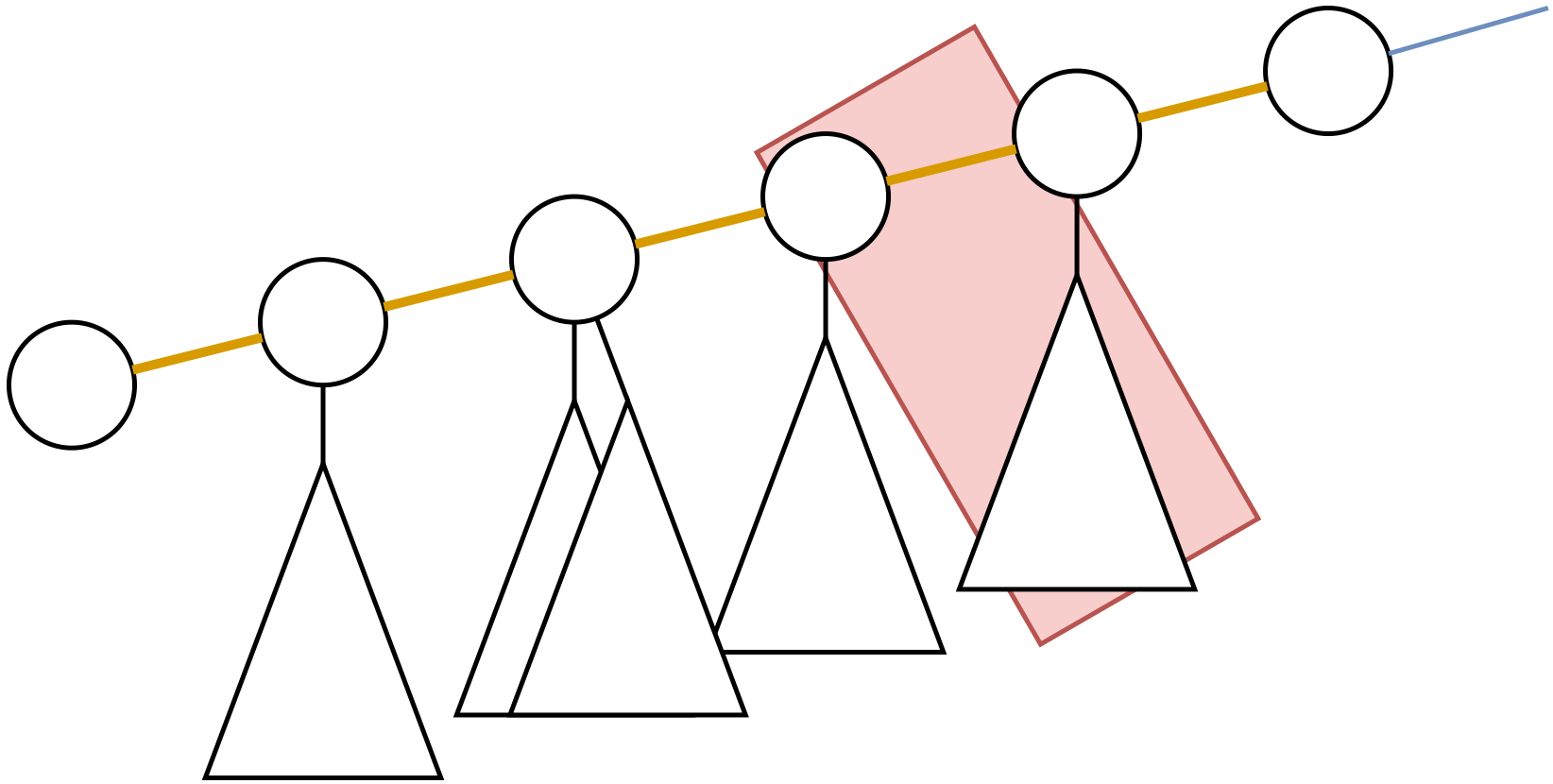
解法 C : static top tree

- top tree は辺属性の方が扱いやすい

  → 適当に根を取り，頂点の情報を親への辺に載せる

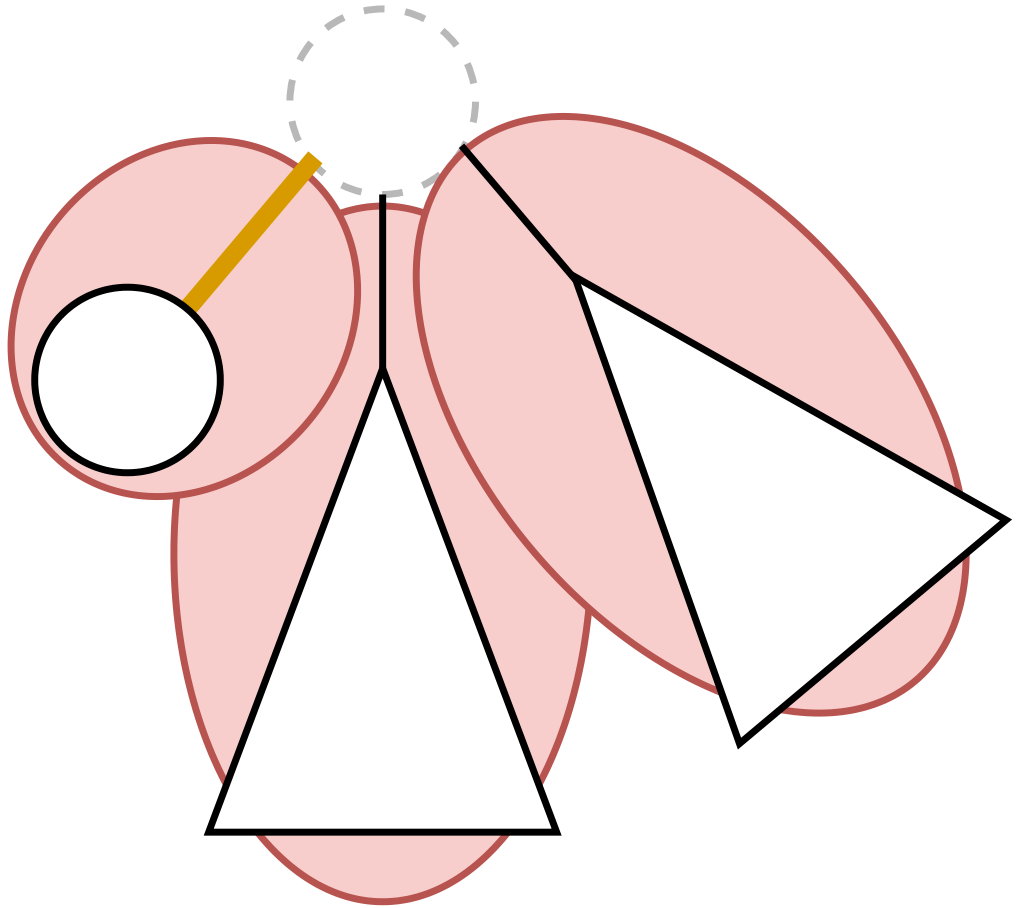
static top tree の構築

- heavy-path 上: この部分を 1 つのノードとして，高さの最大値を最小化するようにマージ



static top tree の構築

さっきの部分は，部分木ごとに
ノードを作って，マージ過程の
高さの最大値を最小化するように
マージして作る



解法 C : static top tree

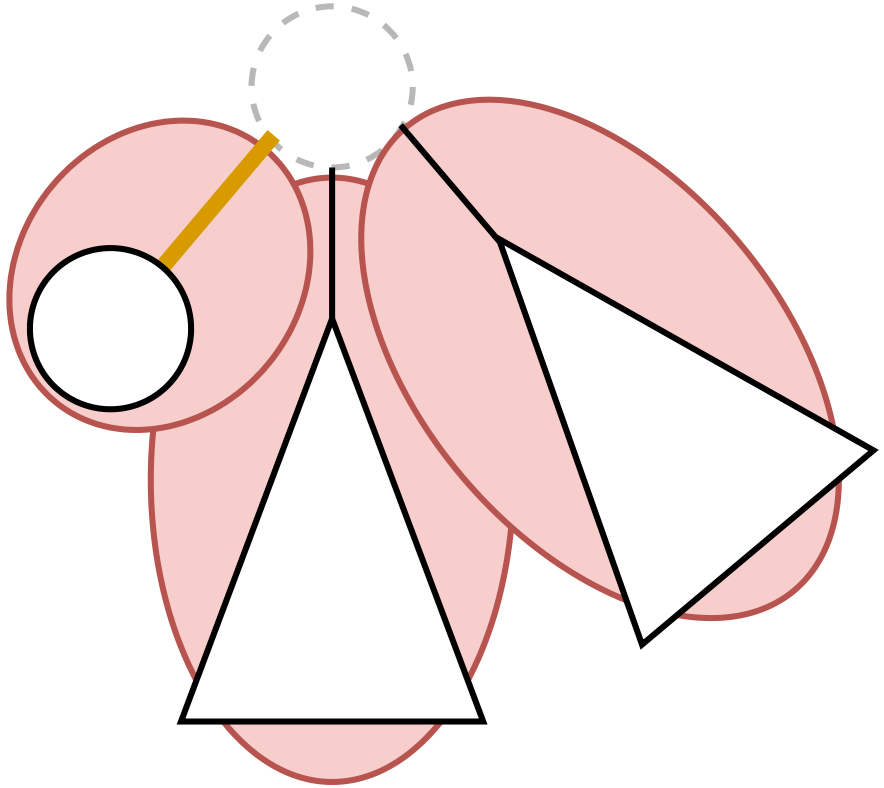
- 持つデータは HL 分解のときとだいたい同じ
- マージ過程の高さの最大値を最小化するようにマージすれば, 高さは $O(\log N)$
    - 所属する頂点数が均等になるように分割していく再帰

→ 更新が $O(\log N)$ 時間でできる！

解法 C : static top tree

マージの計算は，以下の 2 種類ができれば良い

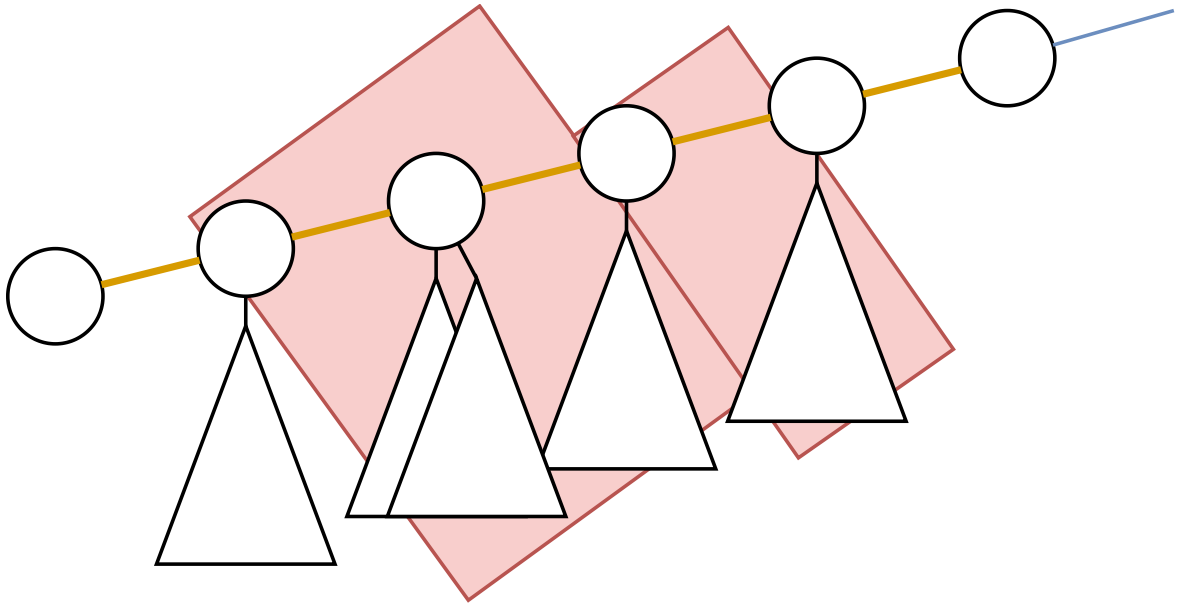
- rake
  - 根が共通である 2 つの部分木のマージ
  - 根が共通である，「heavy-path 上の区間」と「部分木」のマージ



解法 C : static top tree

マージの計算は，以下の 2 種類ができれば良い

- compress
  - heavy-path 上の連続する区間のマージ



小課題 $6, 7 : N \leq 200000, Q \leq 50000 (48$ 点$)$

解法 C : static top tree

マージの計算は，以下の 2 種類ができれば良い

- rake
  - 根が共通である 2 つの部分木のマージ
  - 根が共通である，「heavy-path 上の区間」と「部分木」のマージ
- compress
  - heavy-path 上の連続する区間のマージ

→ これができる木上のなにかの計算は $O(\log N)$ / query

これを Link-Cut 木 (を発展させたもの) で管理したものが **top tree** で，
辺の追加・削除ができる

統計情報 (JOI)

点数	1	2	3	4	5	6	7	人数	累積人数
52	o	o	o	o	o	-	-	3	3
36	o	o	o	o	-	-	-	1	-
36	o	o	o	-	o	-	-	3	7
20	o	o	o	-	-	-	-	7	14
6	o	-	-	-	-	-	-	2	16
0	-	-	-	-	-	-	-	9	25