珍しい都市　解説

maroon

得点分布

- 小課題①の正答者　　??人
- 小課題②の正答者　　??人
- 小課題③の正答者　　??人
- 小課題④の正答者　　??人

問題概要

- N 頂点の木がある
- 頂点 x から見て、距離が unique である頂点を珍しい頂点とする
- 各頂点から見て、珍しい都市についたラベルの種類数を答える

小課題①

- N <= 2000

- 各頂点からDFSとかBFSをしてください

- O(N^2)
- 4 点

得点分布

- 小課題①の正答者　　?1人
- 小課題②の正答者　　??人
- 小課題③の正答者　　??人
- 小課題④の正答者　　??人

考察

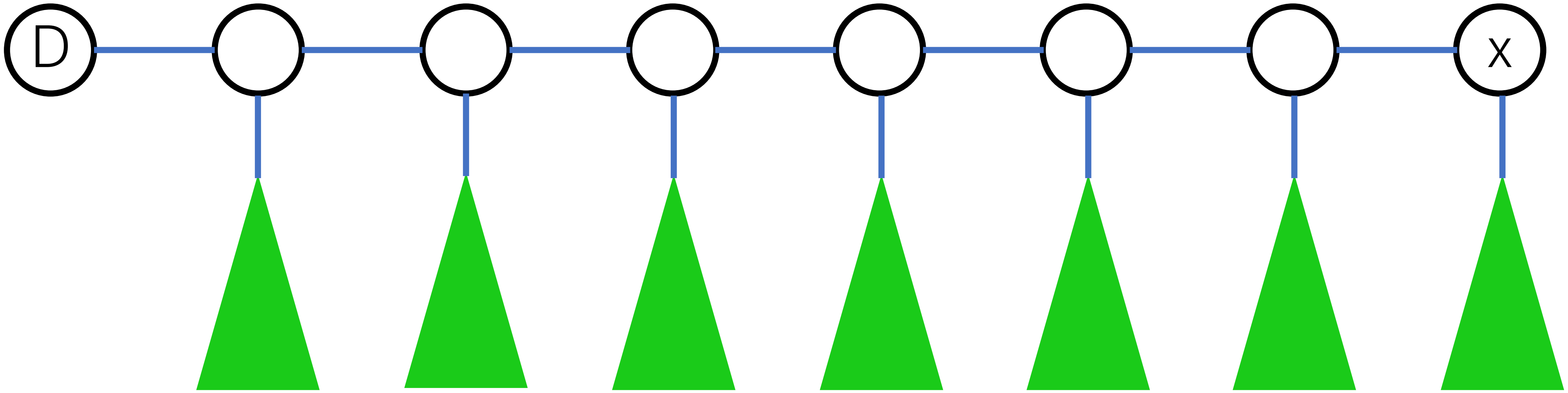
- 頂点 x から見て距離最大の頂点を一つとって D とおく
- 珍しい頂点は、存在するなら D-x パス上にある

考察

- D-x パスに注目する

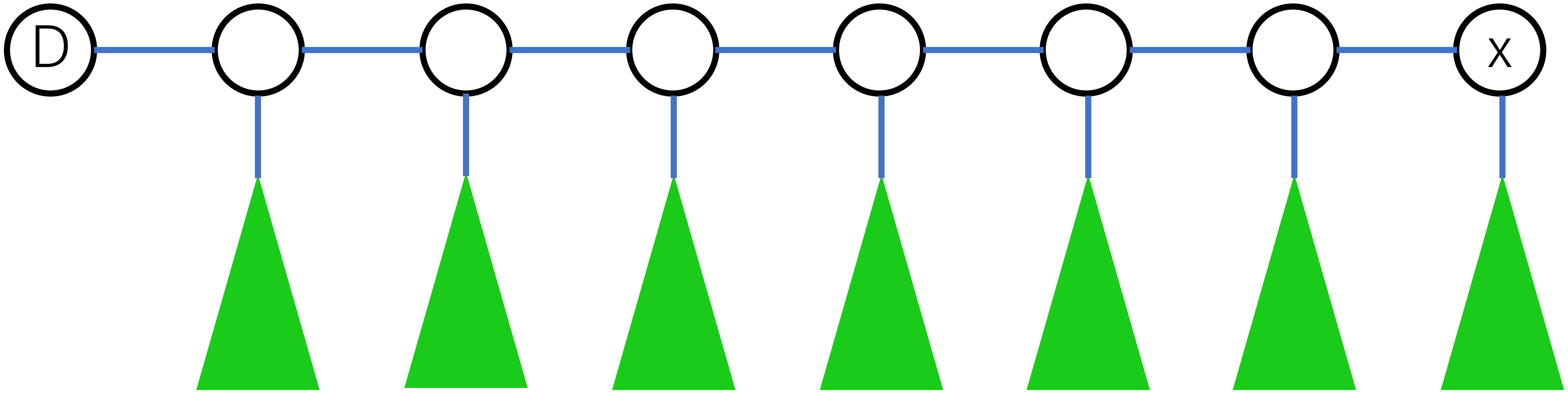
考察

- 途中で枝分かれしてた部分木を考える



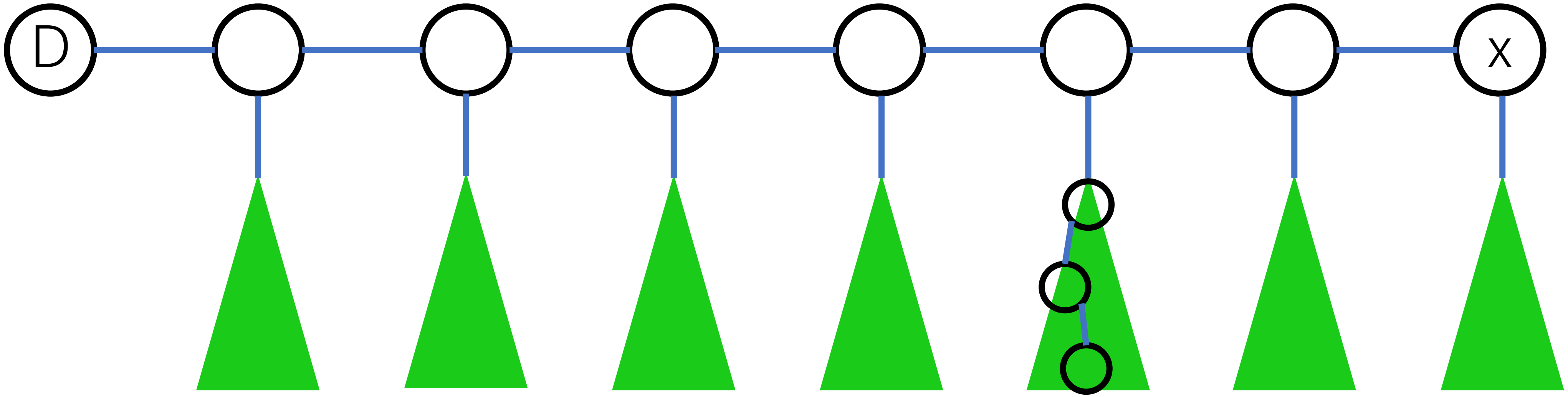
考察

- 枝分かれした部分木の最大深さの分だけ、珍しい都市の候補が消える



考察

- 枝分かれした部分木の最大深さの分だけ、珍しい都市の候補が消える

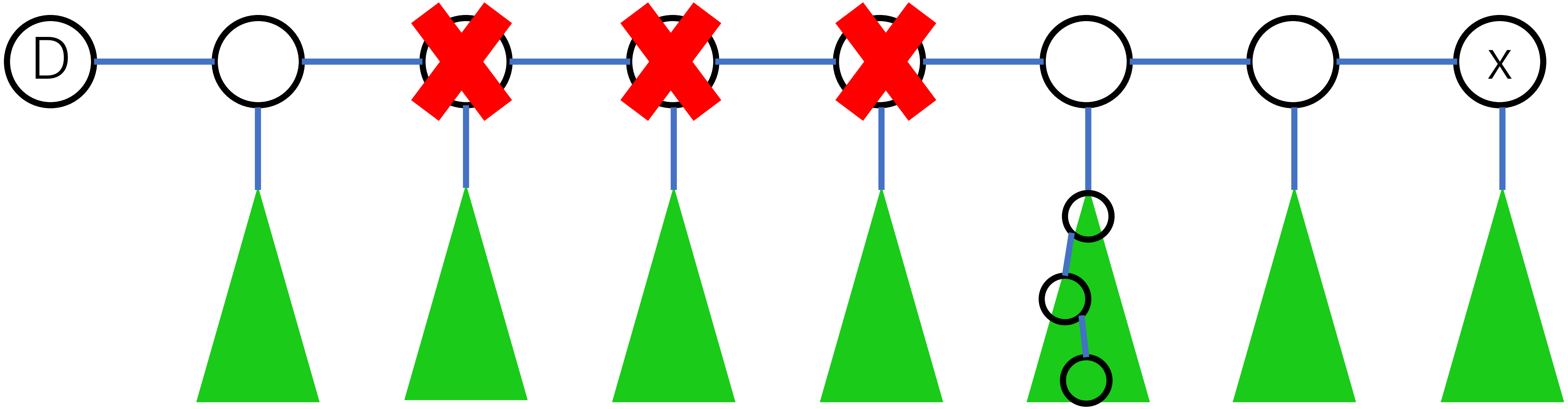


例えばここに長さ3のパスがあれば

考察

- 枝分かれした部分木の最大深さの分だけ、珍しい都市の候補が消える

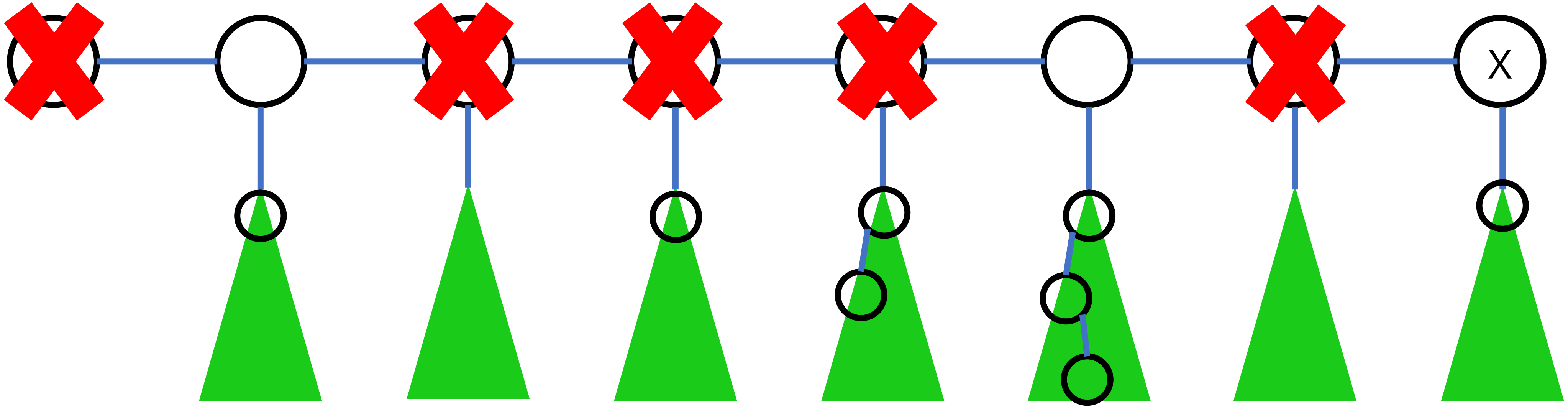
この3つの頂点は珍しい都市の候補から消える



例えばここに長さ3のパスがあれば

考察

- 最後に残った頂点は、すべて珍しい頂点



得点分布

- 小課題①の正答者　?1人
- 小課題②の正答者　??人
- 小課題③の正答者　??人
- 小課題④の正答者　0?人

考察

- ところで、木の直径の端点（最も距離の長い2点対）を任意にとって D1,D2 とおく
- 任意の頂点 x ついて、D1,D2 の少なくとも一方は、x からの距離最大の頂点

考察

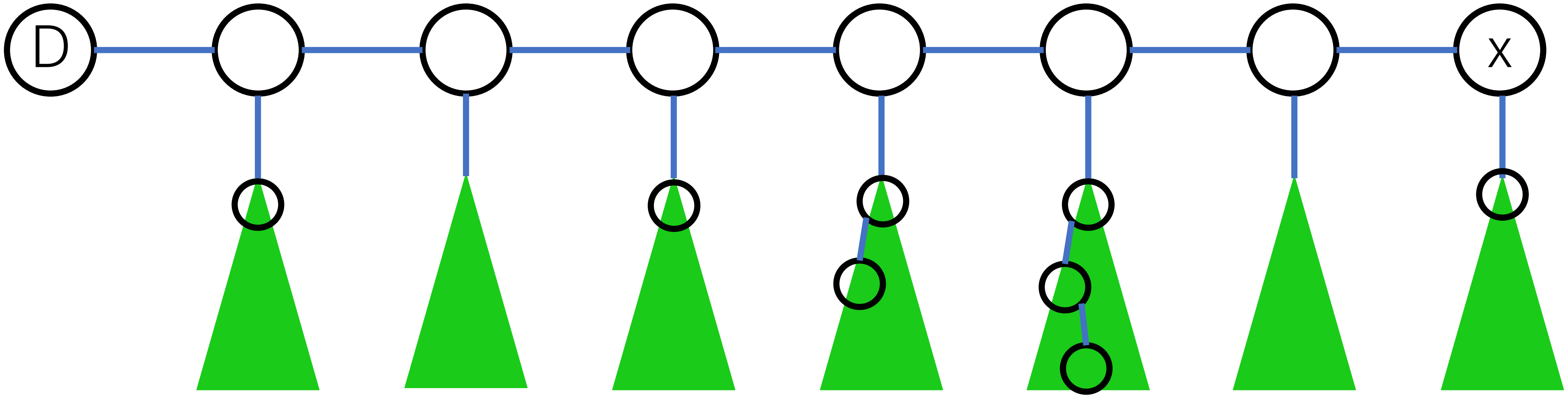
- 頂点 D1 から DFS をして、各頂点 x について、D1-x パス上に珍しい頂点が存在するかどうか判定することができればよい

- D2 についても同じ DFS をすれば、各頂点について正しい答えが求まる

考察

- D-x パス上にある珍しい頂点は、各部分木の最大深さの列からわかる

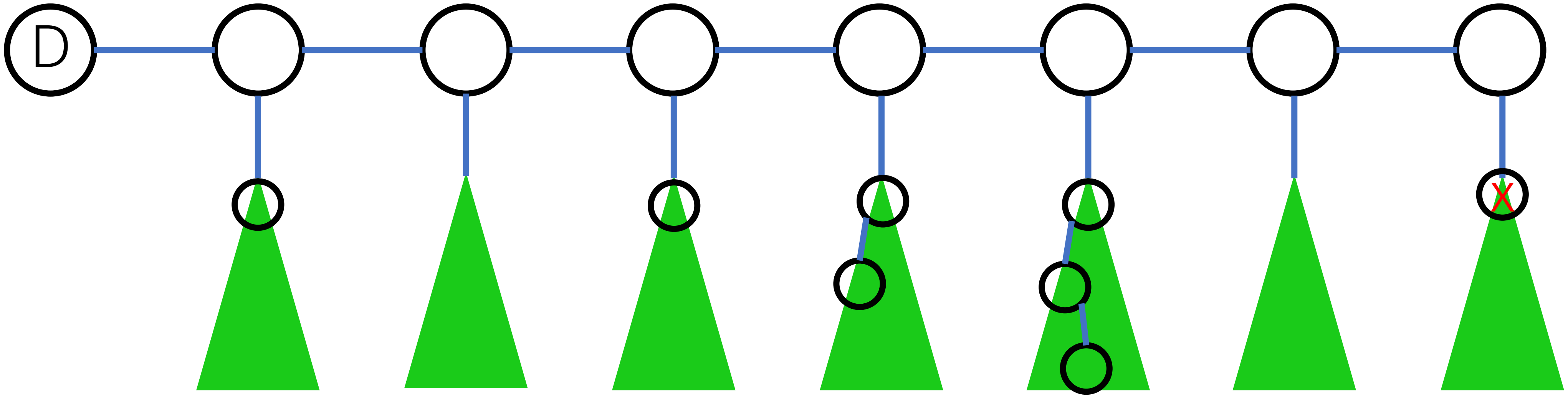
例えば以下の例では (0,1,0,1,2,3,0,1)



考察

- この列はDFSで辺を一本通るたびに、後ろの定数個の要素が変化する

例えば以下の例では (0,1,0,1,2,3,0,1)→(0,1,0,1,2,3,0,0,0)



考察

- この列を A とおくと問題は次のようになる
- A を変化させるクエリ（ただし末尾の要素のみ）がたくさん与えられる
- 各 p(1<=p<=|A|) について、p-A[p]<=j<p を満たす j を使用禁止にしたとき、禁止されない |A| 未満の j の集合を S とおく
- S の要素が珍しい頂点に対応

得点分布

- 小課題①の正答者　　?1人
- 小課題②の正答者　　??人
- 小課題③の正答者　　0?人
- 小課題④の正答者　　0?人

小課題②

- 頂点のラベルが 1 種類
- 珍しい頂点が存在するかどうか判定すればいい

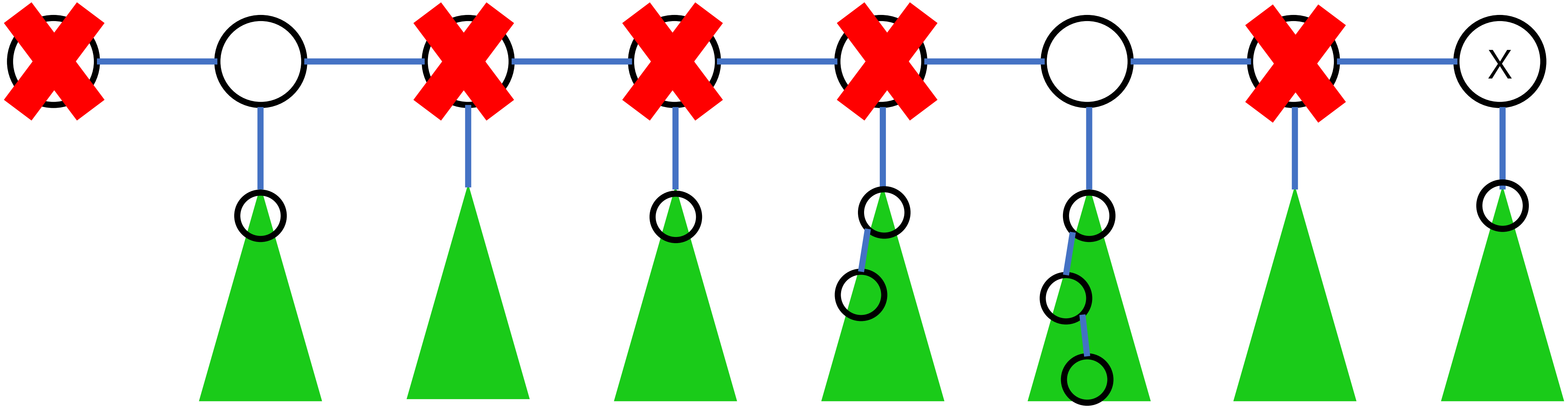
- S が空かどうか判定すればいい

小課題②

- 頂点 x に来た段階では、D-x パスのうち、x 以外から生える部分木の最大深さの列を保持しておくことにする（つまりAの末尾を削ったものを保持しておく）
- これをA’とおく
- S のminと、x から生える部分木の最大深さの比較によって、答えが求まる

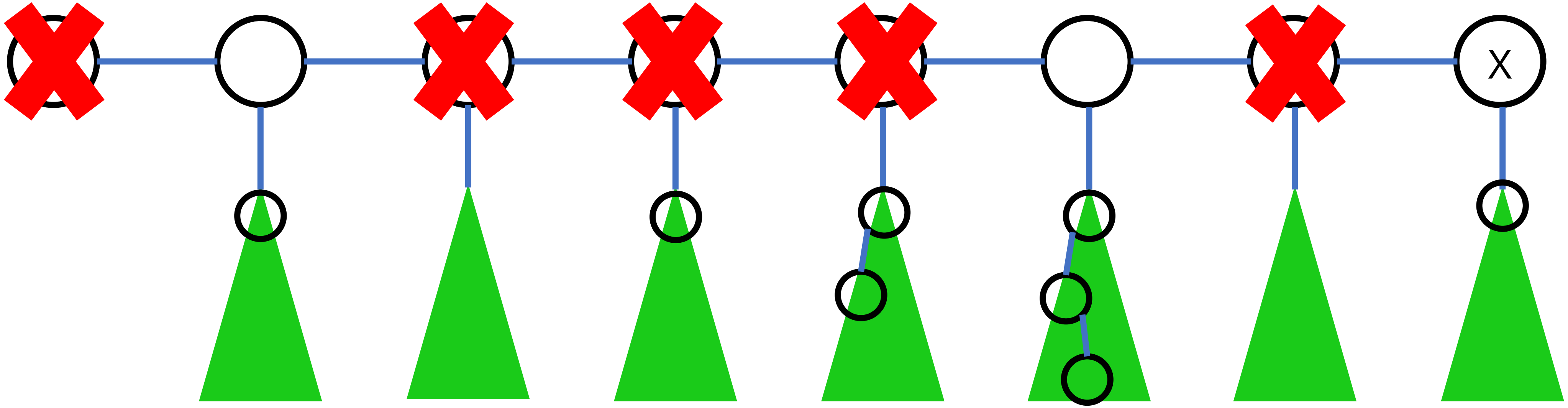
小課題②

- 例えば以下の例では A'=(0,1,0,1,2,3,0), min{S}=2



小課題②

- x から生える部分木の最大深さが6以上だとSは空になり、そうでないならminの要素が残るため空ではない



小課題②

- あらかじめ各部分木の最大深さを求めておく
- 辺を下るときにA’の末尾に追加されるのは、今下った部分木の兄弟の部分木の最大深さ
- Sのminは、そのままか、|A’|+1になるかどちらか

小課題②

- 辺を一回下る操作は O(1) でできる
- 全体でO(N)
- ここまでで 36 点

得点分布

- 小課題①の正答者　　?1人
- 小課題②の正答者　　0?人
- 小課題③の正答者　　0?人
- 小課題④の正答者　　0?人

小課題③

- ラベルがすべて異なる

- |S|のサイズがわかればいい

小課題③

- ラベルがすべて異なる

- |S|のサイズがわかればいい

小課題③

- 「各 p(1<=p<=|A|) について、p-A[p]<=j<p を満たす j を使用禁止」とあるが、補助配列Bを用意しておき、「各 p(1<=p<=|A|) について、p-A[p]<=j<p を満たす j でB[j]++」とし、B[j]=0 となる j を数えればいい

小課題③

- A の要素の変更は、B に対する区間加算クエリに対応
- B に対する区間加算と、0になる要素のカウントが高速にできればいい

小課題③

- B の要素は必ず0以上だから、min{B}=0 かどうかのチェックと、minの要素のカウントができればいい

- これはSegment Treeでできる
- 全く同じSegTreeが過去のIOIでも出題

小課題③

- Segment Tree の更新は 1 回 O(logN) でできる
- 更新は O(N) 回なので、全体で O(NlogN)
- ここまでで 68 点

得点分布

- 小課題①の正答者　　21人
- 小課題②の正答者　　0?人
- 小課題③の正答者　　0?人
- 小課題④の正答者　　0?人

小課題④

- A を変化させるクエリ（ただし末尾の要素のみ）

  をもう少し詳しく見てみる

小課題④

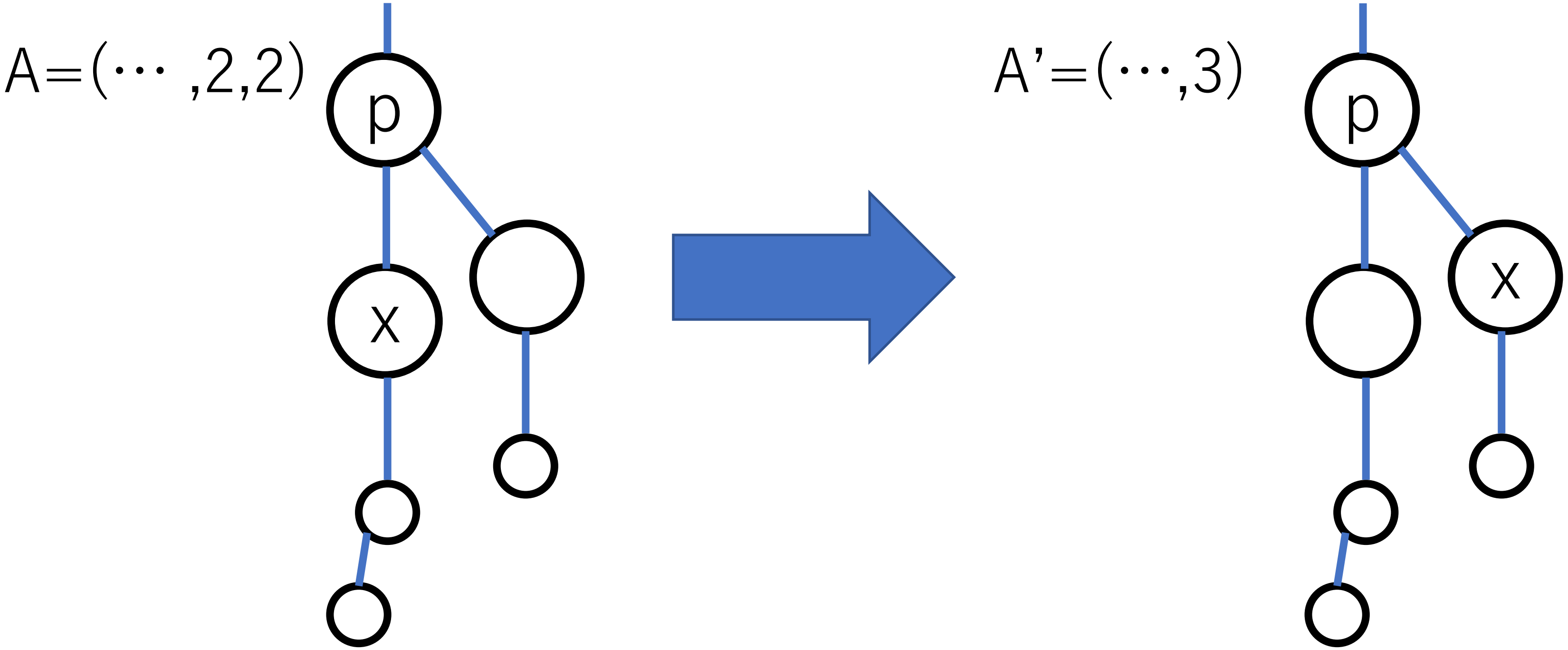
- 頂点 x に来た段階では、A’ を保持しており、頂点 x から親に戻る段階では、A を保持しているようにする

小課題④

- 「辺を下るときにA’の末尾に追加されるのは、今下った部分木の兄弟の部分木の最大深さ」とあるが、部分木の最大深さの後順に潜ることで、 A’の末尾に追加される要素は単調増加する

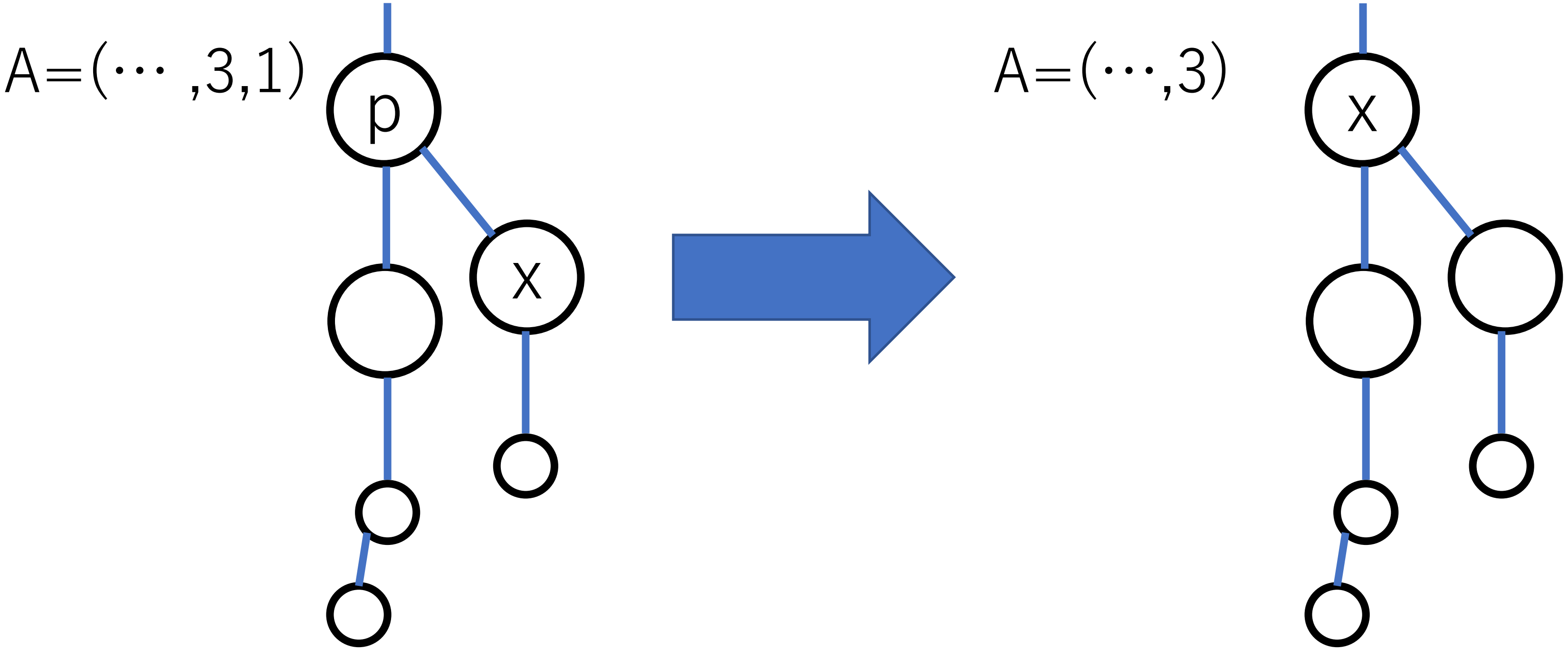
小課題④

- DFSで兄弟の部分木に移るとき、Aの末尾の要素の変化をみると、p以外の要素はSから消えることはあっても増えることはない



小課題④

- 親の頂点に戻るとき、Aの末尾の要素の変化をみると、Sの要素は消えることはあっても増えることはない



小課題④

- 以上より、Sの要素が増える回数はO(N)回、よって減る回数もO(N)で抑えられる

- Sに対する操作はstackで表現できるため、1回あたりO(1)

小課題④

- Sの要素が増減するたびに、対応する頂点のラベルの集合をいじって、ラベルの種類数を更新すればいい

- 全体でO(N)
- 100 点

得点分布

- 小課題①の正答者　　21人
- 小課題②の正答者　　01人
- 小課題③の正答者　　0?人
- 小課題④の正答者　　0?人

得点分布

- 小課題①の正答者　　21人
- 小課題②の正答者　　01人
- 小課題③の正答者　　0?人
- 小課題④の正答者　　00人

得点分布

- 小課題①の正答者　21人
- 小課題②の正答者　01人
- 小課題③の正答者　00人
- 小課題④の正答者　00人