Условие задачи

- Задано дерево из n вершин
- Нужно выбрать пару чисел L и R
  - Тем самым выбрав все вершины v: $L \le v \le R$
- Требуется удовлетворить m правил:
  - Вершина v и глубина d
- Правило (v, d) требует, чтобы среди вершин $[L \dots R]$ существовала вершина u:
  - u — потомок v
  - глубина вершины u ровно на d больше глубины вершины v
- Требуется минимизировать $R - L$
  - При равенстве минимизировать L

Подзадача 1

- Ограничения: $n, m \leq 50$
- Предпосчитаем для всех пар вершин (v, u), $p[v][u]$ — является ли v предком u
- Также предпосчитаем глубину $h[v]$ всех вершин
- Можно перебрать все возможные пары (L, R)
  - Для каждого правила перебрать все вершины (L, R)
  - Проверить, что вершина u удовлетворяет правилу (v, d)
    - $p[v][u] = \text{true}$
    - $h[u] = h[v] + d$
- Всего пар $\mathcal{O}(n^2)$, правил m и каждое правило проверяется за время $\mathcal{O}(n)$
  - Итого, время работы: $\mathcal{O}(n^3 m)$

Решение быстрее

- Модифицируем правило (v, d):
  - так как глубина $h[v]$ вершины v всегда определена однозначно, то d можно заменить на $d + h[v]$
  - неперь d — требуемая глубина вершины u
- Попробуем найти для каждого L минимальный R, что (L, R) удовлетворяет всем правилам
  - Метод двух указателей
  - Первым указателем будем поддерживать L, увеличивая L
  - При увеличении L, минимальный R может только увеличиваться
  - Перебор всех таких пар работает за $\mathcal{O}(n)$
- Осталось только научиться определять, является ли (L, R) хорошей парой

Подзадача 2

- Ограничения: $n, m \leq 3000$
- Посчитаем для каждой вершины u, какие правила она может удовлетворить
  - сохранить их в список $w[u]$ за время $\mathcal{O}(nm + n^2)$
- Давайте поддерживать массив $f[i]$:
  - $f[i]$ — сколько вершин из (L, R) удовлетворяют правило (v_i, d_i)
- А также переменную r — число правил, которые удовлетворены для отрезка (L, R)
  - Если $r = m$, то отрезок хороший
- При увеличении R в отрезок добавляется одна вершина
  - нужно для всех правил i из $w[R]$ увеличить $f[i]$ на 1
    - Если $f[i]$ стало равно 1, то нужно увеличить r
- При увеличении L надо уменьшить $f[i]$ для всех i из $w[L]$
  - Если $f[i]$ стало равно 0, то нужно уменьшить r
- Увеличение L или R работает за $\mathcal{O}(m)$, итого $\mathcal{O}(nm + n^2)$

Подзадача 3

- Ограничения: $n, m \le 200\,000$, у каждой вершины один ребенок
- В этой подзадаче из правила (v_i, d_i) можно оставить только d_i
  - так как все вершины с большей глубиной являются потомками
- Каждому правилу соответствует ровно одна вершина
- Если есть несколько правил с одинаковым d_i, то можно оставить только одно такое правило
  - значит каждой вершине соответствует не более чем одно правило
- Воспользуемся массивом $f[i]$ и переменной r из решения второй подзадачи
- Список $w[v]$ имеет размер не больше чем 1
- Решение работает за $\mathcal{O}(n + m)$

Полное решение

- Ограничения: $n, m \leq 200\,000$
- Рассмотрим два правила, которые удовлетворяются общей вершиной u
  - Если бы таких не было, то размеры списков $w[v]$ равны 1
    - И задача бы решилась описанным выше алгоритмом
  - Пусть два правила: (a, d) и (b, d)
    - Если $a = b$, то одно из правил можно удалить
    - Обе вершины a и b — предки вершины u
    - Если $a \neq b$, то одна из них — предок другой
  - Пусть $s(a, d)$ — множество вершин удовлетворяющих (a, d)
  - Пусть a предок b
    - тогда $s(b, d) \subset s(a, d)$
    - то есть, (b, d) удовлетворено $\Rightarrow$ (a, d) удовлетворено
  - Удалим правило (a, d)

Полное решение, продолжение

- Запустим обход в глубину
  - будем поддерживать $z[d]$ — список правил (u_i, d_i)
    - $d_i = d$
    - Если сейчас находимся в вершине v, то u_i — предок v
    - Правила упорядочим по глубине u_i
  - Когда пришли в вершину v с глубиной $h[v]$, посмотрим на $z[h[v]]$
    - Если там записано правило, то v удовлетворяет последнее правило из $z[h[v]]$
  - Для всех правил (v, d) нужно обновить $z[d]$
  - Когда мы выходим из вершины v, нужно в обратном порядке удалить все правила (v, d)
- Тем самым мы построим массив $w[u]$ за $\mathcal{O}(n+m)$
- Общее время: $\mathcal{O}(n+m)$