Возрастающие пути

Вначале заметим, что нам достаточно рассматривать только простые x. Из этого следует, что длина любого возрастающего пути не более 23, так как максимальный вес ребра 10^7.

Для решения подзадачи 1 нужно перебрать все пути и найти найти на них ответ. Будем вначале перебирать первую вершину пути, переберём первое ребро и один из простых делителей веса этого ребра, запустим dfs от второго конца ребра, переходя из вершин в следующие так, чтобы путь оставался простым и вес следующего ребра делился на нужную степень зафиксированного простого числа. Решение этой подзадачи работает за $O(n^2 * logMAXW)$ или $O(n^2)$ в зависимости от реализации.

В подзадаче 2 граф — цепочка из n вершин. В этой подзадаче достаточно также перебрать первую вершину, ребро выходящее из неё и простой делитель веса этого ребра. А далее запустить dfs от второго конца ребра. Заметим, что всего для одной вершины, выбранного ребра и простого будет проверено не более 24 вершин, так как длина возрастающего пути всегда не более 23. Решение этой подзадачи работает за $O(n * log^2 MAXW)$.

Для полного решения давайте разложим вес каждого ребра на простые при помощи линейного решета Эратосфена для нахождения минимального простого делителя для каждого числа от 1 до 10^7. Подвесим граф за вершину 1. После переберём все простые числа, каждое из которых является простым делителем хотя бы для веса одного из рёбер. Зафиксируем тогда какое-то простое число p и оставим только те рёбра, вес которых делится на это простое число p. Тогда у нас останутся вершины, которые соединены хотя бы с одним из оставшихся рёбер. Посчитаем для каждой такой вершины две динамики: $up[v]$ — максимальная длина возрастающего пути, приходящего из какого-то ребёнка вершины v, $down[v][k]$ — максимальная длина пути, идущего в одного из детей вершины v, такого, что вес первого ребра делится на p^k, вес второго ребра делится на p^{k+1}, $\ldots$, вес m-го ребра делится на p^{k+m-1}. Тогда научимся пересчитывать динамики для вершины v через детей: вершина u_1 и вес ребра (v, u_1) делится на p^{c_1}, вершина u_2 и вес ребра (v, u_2) делится на p^{c_2}, $\ldots$,вершина u_d и вес ребра (v, u_d) делится на p^{c_d}. Поэтому $up[v] = max(min(c_1, up[u_1] + 1), min(c_2, up[u_2] + 1), \ldots, min(c_d, up[u_d] + 1))$, а $down[v][k] = max(down[u_i][k + 1] + 1)$ для всех i таких, что $c_i \geqslant k$. Тогда ответ на задачу нужно обновить с $up[v], down[v][1]$ и перебрать все такие i и k, что $c_i >= k$, и существует j, что $u_i \neq u_j$ и $up[u_j] >= k - 1$. Тогда ответ нужно обновить с $down[u_i][k + 1] + k$. Для одного простого числа мы можем всё посчитать за $O(E(p) * logMAXW)$, где $E(p)$ — количество рёбер, вес которых делится на p и $MAXW$ — максимальный вес ребра. Суммарно для всех простых мы обработаем за $O(n * MAXD * logMAXW)$, где $MAXD$ - максимального количество простых делителей у веса ребра. Заметим, что $MAXD$ не более $logMAXW$, тогда всё решение работает за $O(n * log^2 MAXW + MAXW)$.

Для решения подзадачи 3 достаточно посчитать динамики только для простого числа 2. Решение этой подзадачи работает за $O(n * logMAXW)$.

Для решения подзадачи 4 вместо линейного решета Эратосфена достаточно для каждого ребра отдельно разложить вес на простые числа. Решение этой подзадачи работает за $O(n * \sqrt{MAXW} + n * log^2 MAXW)$.

Вместо двух динамик для каждого простого числа p можно действовать итеративно. Оставим также в графе только рёбра, вес которых кратен p. На итерации i будем хранить рёбра, на которые могут заканчиваться возрастающие пути длины i. Тогда, чтобы перейти с итерации i на $i + 1$, достаточно перебрать рёбра из вершин, которые являются концами одного из рёбер итерации i, и проверить, что вес ребра кратен p^{i+1}, а также, что это ребро не ломает простоту для какого-то пути. Заметим, что i-я итерация работает за количество рёбер, вес которых делится на p^i. Тогда суммарно решение работает за сумму степеней простых чисел в разложении всех весов рёбер. Для одного ребра сумма степеней $O(logMAXW)$. Значит для всех рёбер суммарно работает за $O(n * logMAXW + MAXW)$.