ЗАДАЧА С?. ТАКСИ

Първа подзадача: Примерни тестове

Втора подзадача:

За всяка заявка проверяваме за всеки два върха дали стойността на максималното ребро по пътя между тях е в интервала $[l; r]$. Ако е го добавяме към резултата за заявката. Реализацията става с DFS.

O($N^3 * Q$) – tolls_5p.cpp

*Забележка – съществува решение на задачата с O($N^2 * \log_2 N * Q$) с LCA, но алгоритмът е твърде сложен за точките които бих дал за него.

Трета подзадача:

От всеки връх u си пускаме DFS, който ни изчислява максималното ребро до всяко v по формулата: $dp[v] = \max\{dp[parent(v)]; edge(u, v)\}$. Отговорът е сумата от всички $dp[v]$ за всички начални върхове u.

O(N^2) – tolls_q1_5p.cpp

Четвърта подзадача:

За всяка заявка правим същото като в трета подзадача, но добавяме $dp[v]$ само ако е в интервала $[l; r]$.

O($N^2 Q$) – tolls_20p.cpp

Пета подзадача:

Ще направим следното наблюдение:

Ако имаме две дървета с размери x и y и добавим ребро, което ги свързва, броят на пътищата в новото дърво минаващи през това ребро е $x * y$.

Това, което ще направим, е да си сортираме ребрата по възходящ ред на стойността им и да ги добавяме едно по едно. Като добавяме ребро ни е гарантирано, че свързваме две дървета и че това ребро е с най-голяма стойност в новото дърво. От наблюдението ни знаем, че има $x * y$ пътища през това ребро, всяко от които допринася за отговора със стойността на реброто, тоест при всяко добавяне на ребро увеличаваме отговора с $x * y *$ стойността на реброто. Пазенето на размер и свързването на дървета може да се осъществи с DSU.

O($N * \log_2 N$) – tolls_q1_25p.cpp

Шеста подзадача:

За всяка заявка правим същото като в пета подзадача, но не разглеждаме ребра със стойност по-голяма от y, а за ребрата със стойност по-малка от x не добавяме към отговора. (Все още се свързват дърветата и пази размера с DSU.)

O($N * \log_2 N * Q$) – tolls_50p.cpp

Седма подзадача:

За тази подзадача ще разглеждаме заявките offline. Тъй като всички заявки ни започват с 1 няма да го пазим, а ще пазим само десния край на интервала. Ще си сортираме заявките по възходящ ред.

В пета подзадача, ако си запазваме отговора за дадено последно добавено ребро със стойност x можем да намерим отговора за всеки интервал $[1; x]$. Ако искаме да намерим отговора за някакво z, което не е тежест на ребро, просто намираме коя е най-голямата по-малка от z тежест на ребро и намираме отговора за нея, което ще ни е и отговора за z.

Тъй като заявките и списъка от ребра са ни сортирани, ние можем паралелно да ги обхождаме и намираме отговорите на заявките по начина от предишния абзац в приблизително линейно време. После ще си сортираме заявките обратно в началния ред и ще ги изведем отговорите им.

O($N * \log_2 N + Q \log_2 Q$) – tolls_l1_20p.cpp

Осма подзадача:

В тази подзадача ще намираме отговора за заявка $(x; y)$ като намерим отговора за заявките $(1; x - 1)$ и $(1; y)$ и ги извадим. Ако $x = 1$ приемаме, че заявка $(1; x - 1)$ е с отговор 0. Вече имаме списък от $2 * Q$ заявки като тези в седма подзадача. Оттук остава само да приложим решението от седма подзадача, като вместо да извеждаме отговора за всяка заявка извеждаме $ans_{2*k} - ans_{2*k-1}$, за да възвърнем началните отговори.

O($N * \log_2 N + Q \log_2 Q$) – tolls_100p.cpp

Автор: Даниел Гетов