Разбор задачи «J. Путешествия в реальности»

Автор задачи — П. Михеев, разбора — А. Полднев

Кратко сформулируем условие задачи. Дано дерево с выделенными в нём стартовой и помеченными вершинами, каждое ребро которого имеет свой вес. Требуется найти минимально возможный вес пути, начинающегося и заканчивающегося в стартовой и проходящего через все помеченные вершины. При этом если вершина v не является корнем дерева, то её родителем является вершина P_v, а ребро (P_v, v) имеет вес $T_v - T_{P_v} > 0$.

Ответом на задачу является удвоенный суммарный вес всех рёбер, лежащих на пути от стартовой вершины до каждой помеченной (то есть вес ребра входит в ответ тогда и только тогда, когда существует помеченная вершина w такая, что это ребро лежит на пути от вершины w до стартовой вершины; такие рёбра будем называть *замечательными*). Но этот факт, естественно, нуждается в доказательстве.

Пункт 1. Докажем, что каждое замечательное ребро должно войти в ответ, причём ровно 2 раза.

Рассмотрим некоторое замечательное ребро e, лежащее на пути от стартовой вершины (обозначим её s) до некоторой помеченной w. Поскольку циклы в дереве по определению отсутствуют, ребро e будет лежать на любом пути от s до w, так же как и на любом пути от w до s. При этом по условию нужно, в частности, выйти из s, посетить w и вернуться обратно, поэтому ребро e будет посещено ровно 2 раза: при проходе от s до w и обратно — от w до s.

Пункт 2. Покажем, что ни одно ребро, не являющееся замечательным, не войдёт в ответ. Возьмём некоторое не являющееся замечательным ребро e. Как и любое другое ребро, оно делит дерево на две части (т. е. компоненты связности, образующиеся при удалении этого ребра), но при этом по определению замечательного ребра в одной из частей будут находиться вершина s и все помеченные вершины, а в другой никаких интересущих нас вершин нет. Вполне очевидно, что при поиске замкнутого пути минимального веса, содержащего стартовую и все помеченные вершины, нет никакого смысла заходить в часть дерева, не содержащую интересующих нас вершин. (Формально говоря, можно удалить ребро e вместе с «неинтересной» частью дерева и решать ту же задачу для уменьшенного дерева: так как ответ по текущему определению вычисляется однозначно, он при таком удалении части дерева не изменится.)

Пункт 3. Итак, мы доказали, что не существует обхода, вес которого был бы меньше, чем заявленный ответ (т. е. удвоенный суммарный вес всех рёбер, лежащих на пути от стартовой вершины до любой помеченной). Осталось показать, что обход с таким весом действительно существует.

Примеров таких обходов в каждом конкретном случае, конечно же, можно привести довольно много, но для доказательства достаточно привести общий принцип обхода.

Искомый обход опишем рекурсивно. Начинается он в стартовой вершине. Пусть мы сейчас находимся в некой вершине v. Возможны следующие случаи:

- *В поддереве вершины v есть ещё не посещённые помеченные вершины.* Тогда нужно спуститься к любой непосещённой помеченной вершине и рекурсивно запуститься от неё.

- *В поддереве вершины v нет непосещённых помеченных вершин.* Здесь тоже возможны два случая:
  - *В дереве ещё остались непосещённые помеченные вершины.* В этом случае нужно подниматься к корню, пока в поддереве текущей вершины не окажется непосещённых помеченных вершин, и рекурсивно запуститься от достигнутой вершины.
  - *Все помеченные вершины дерева уже посещены.* Тогда нужно просто кратчайшим путём дойти до стартовой вершины и на этом закончить обход дерева.

Несложно заметить, что вес этого обхода действительно будет соответствовать заявленному. (Над этим вопросом читателям предлагается подумать самостоятельно.)

Следует отметить, что в данном выше и затем доказанном определении ответа можно рассматривать пути, соединяющие помеченные вершины не со стартовой, а с любой вершиной, которая принадлежит описанному выше обходу. В частности, в качестве такой вершины можно взять вершину l, являющуюся наименьшим общим предком всех помеченных вершин, в том числе стартовой (в дальнейшем будем считать, что стартовая вершина также является помеченной). Таким образом, ***ответом** на задачу является удвоенный суммарный вес всех рёбер, лежащих на пути от вершины l до каждой помеченной — такие рёбра называются замечательными.*

Реализация 1. Подъём вверх. Рассмотрим первый способ нахождения множества замечательных рёбер.

Переберём все помеченные вершины. Из каждой будем подниматься вверх, отмечая посещённые рёбра, пока не дойдём либо до корня, либо до ранее посещённого ребра. Получившееся множество посещённых рёбер будет содержать множество замечательных рёбер, поскольку вершина l лежит на любом пути, соединяющем корень и какую-нибудь помеченную вершину. Значит, из полученного множества осталось лишь удалить цепочку рёбер, соединяющую вершину l с корнем дерева. Для этого достаточно найти вершину l.

В данной задаче это можно сделать так: начав в корне, будем спускаться вниз по посещённым рёбрам, пока не дойдём до вершины, продолжить спуск из которой можно не единственным способом, — эта вершина и будет искомой вершиной l. (Чтобы реализовать такой спуск из корня, достаточно было в процессе основного алгоритма подъёма вверх из помеченных вершин для каждой вершины считать количество исходящих из неё вниз посещённых рёбер.)

В данном алгоритме каждое ребро дерева рассматривается не более двух раз: при его возможном посещении в процессе подъёма и при его возможном „удалении“ в процессе спуска из корня. Из этого следует, что время работы алгоритма линейно зависит от количества рёбер в дереве, то есть равно $O(n)$.

Реализация 2. Динамическое программирование на дереве. Второй способ нахождения множества замечательных рёбер немного сложнее первого, но на олимпиаде все команды, сдавшие задачу, сдали её исключительно этим способом.

Введём следующую функцию на множестве вершин: пусть $f(v)$ — удвоенная сумма весов рёбер, лежащих на пути от вершины v до каждой помеченной вершины из поддерева v. Ясно, что в таком случае ответом будет являться $f(l)$ или, что то же самое, наименьшее $f(v)$ для тех v, в поддереве которых есть все помеченные вершины. Эту функцию мы и будем считать с помощью динамического программирования на дереве.

Пусть мы хотим посчитать $f(v)$ для некоторой вершины v, при этом уже известны значения функции f для всех детей вершины v (известны $f(i)\ \forall i \mid P_i = v$).

Итак, $f(v) = \sum\limits_{i,\, P_i=v} f(i) + 2 \cdot \sum\limits_{j,\, P_j=v} (T_j - T_v) = \sum\limits_{j,\, P_j=v} \big(f(j) + 2 \cdot (T_j - T_v)\big)$ для всех i и таких j, в поддереве которых есть помеченные вершины. (Для всех детей, в поддереве которых нет помеченных вершин, значение функции равно нулю, поэтому в записи с помощью двух сумм в первой сумме можно рассматривать только те i, для которых $f(i) \neq 0$, и таким образом свести первую запись ко второй, с одной суммой.) Иначе говоря, для всех детей вершины v, в поддереве которых есть помеченные вершины, к $f(v)$ прибавляется значение функции в этом ребёнке $\big(f(j)\big)$ и удвоенный вес ребра, соединяющего v с этим ребёнком $\big(2 \cdot (T_j - T_v)\big)$. Это означает, что во всех детей, в которые есть смысл спускаться (поскольку там есть помеченные вершины), нужно спуститься, пройдя по ребру (v, j), обойти соответствующее поддерево и вернуться обратно в v по этому же ребру.

Посчитать значения этой функции для всех вершин дерева и, следовательно, найти ответ, легко с помощью обычного поиска в глубину (в нём же можно считать количество помеченных вершин во всех поддеревьях и затем при подсчёте функции f сравнивать его с нулём). Сложность поиска в глубину, а следовательно, и самого алгоритма, равна $O(n)$.