Авиареформа

Опишем формально условия задачи. Нам дан взвешенный граф, взвешенный граф на n вершинах и m рёбрах. Для него строится граф дополнение, в котором вес ребра равен минимальному максимуму на пути между концами ребра по исходному графу. По такому же принципу считаются веса рёбер исходного графа относительно графа дополнения. Требуется вывести итоговые веса рёбер исходного графа.

Для группы где $n \leqslant 100$ подходило следующее решение: Сделаем аналог алгоритма флойда, но вместо кратчайшего пути будем искать путь с минимальным максимумом. Так за $O(n^3)$ можно будет найти веса всех рёбер в графе дополнении. Запустившись второй раз, можно найти веса рёбер исходного графа.

Теперь заметим основную идею, которая пригодится для всех оставшихся групп. Заметим, что путь с минимальным максимумом проходит по минимальному остовному дереву. Это означает, что в графе дополнении достаточно построить минимальное остовное дерево, а дальше по нему веса исходных рёбер будет восстанавливаться за счёт запроса максимума на пути (что делается за $O(m \log n)$).

Для группы где $N \leqslant 10000$ и $n \leqslant 1000$ построим остовное дерево в исходном графе за $O(n^2 \log n)$. С помощью него найдём итоговые веса рёбер в графе дополнении. На них построим остовное дерево в графе-дополнении за $O(n^2 \log n)$. За $O(m \log n)$ найдём итоговые веса рёбер в исходном графе.

Для группы когда веса не больше 2 построим компоненты связности по рёбрам веса 1 в исходном графе. Возьмём любую вершину оттуда, и найдём все достижимые вершины компоненты по рёбрам графа дополнения. Заметим, что только такие вершины в графе дополнении могут быть соединены рёбрами веса 1, между остальными будут рёбра веса 2. Тогда будем действовать следующим образом: выберем любую компоненту связности по рёбрам из 1. Далее рассмотрим только вершины и рёбра, которые в ней есть. Нам требуется найти в дополнении такого графа компоненты связности. Для этого выберем некоторую вершину A. Будем хранить текущее множество вершин, которые достижимы от A в графе дополнении (пусть это S). Добавим в очередь все рассматриваемые вершины. Будем по одной доставать вершины из очереди, и если между ними и A нет ребра, то будем добавлять в компоненту связности дополнения графа по вершине A, иначе будем заново класть в очередь. Когда вершина B в следующий раз достаётся из очереди, то проверим, что S увеличилось с момента прошлого доставания (если не увеличилось, то новые вершины точно не добавятся). Если же S увеличилось, то мы знаем, что от всех старых вершин в S не было рёбер до B в графе дополнении, но они могли бы появиться от вновь добавленных вершин. Тогда переберём их и проверим, есть ли от них рёбра в B. Когда нашли компоненту связности, то переходим к новой вершине и повторяем тоже самое. Заметим, что на каждое неудачное доставание вершины из очереди должно быть ребро между ней и какой-то вершиной в компоненте. Значит всего неудачных доставаний из очереди $O(m)$, а значит общее время работы этого алгоритма $O(m)$. После нахождения компонент связности, соединим все вершины в них произвольным остовным деревом из единичных рёбер, остальные вершины соединим рёбрами веса 2.

Для полного решения надо использовать похожие идеи. Как в алгоритме Крускала будем строить минимальный остов исходного графа. В процессе построения будут храниться компоненты связности в исходном грфае по добваленным рёбрам. Внутри них будем хранить компоненты связности графа дополнения.

При добавлении нового ребра если объединились две компоненты связности исходного графа, то посмотрим, как изменятся компоненты связности графа дополнения. Заметим, что при объединении компонент связности исходного графа S и T, если в них есть компоненты связности графа дополнения A и B, то они не могут объединится только если между каждой вершиной из A и из B есть рёбра исходного графа. Тогда переберём все такие пары вершин и проверим, есть ли между ними рёбра. Заметим, что при таком объединении между этими компонентами добавится ребро в графе дополнении с весом, равным весу текущего добавленного ребра Крускалом. Тогда храня компоненты связности в графе дополнении и объединяя их, мы сможем получить остовное дерево графа дополнения. При этом в процессе работы на каждую неудачную попытку объединание компонент будет существовать ребро исходного графа между ними, значит общее число неудачных попыток объединения будет не больше $O(m)$. Получаем, что так за $O(m \log m)$ мы сможем построить остовное

дерево в графе дополнении.