10.3. Найти самый легкий путь между двумя выделенными вершинами в графе по метрике «вес пути равен сумме двух самых тяжелых ребер в нем».

Хочется применить агоритм Дейкстры сразу к метрике «сумма двух самых тяжелых ребер», но он не работает. Есть простой контрпример.

Можно предложить следующий способ: решим два раза задачу 9.4(b) из прошлой домашней работы: сначала построим дерево кратчайший путей из вершины s, затем из вершины t (причем уже в транспонированном графе).

Таким образом, мы для каждой вершины будем знать два самых тяжелых ребра в пути **от s** и в пути **до t**.

Далее, пробежимся по всем ребрам и для каждого ребра (u-v) сделаем предположение, что это ребро входит в кратчайших путь. Посчитаем вес этого пути: это сделать просто: у нас есть два тяжелых ребра в пути (от s до u) и в пути (от v до t) плюс само это ребро. За O(1) можно корректно найти два самых тяжелых ребра и посчитать их сумму.

Проделав это для каждого ребра можно найти минимум.

Чтобы вывести путь, достаточно вместе с минимумом поддерживать ребро, на котором этот минимум нашелся, а затем восстановить этот путь по деревьям, найденным во время выполнения алгоритма Дейкстры дважды.

UPD: Алгоритм найдет самый легкий путь по той причине, что мы перебираем все ребра в графе и ищем для каждого ребра самый легкий путь, проходящий через это ребро.

Если предположить, что мы таким образом не найдем самый легкий путь, то есть наш алгоритм выдает путь1, хотя ответом является путь2, то рассмотрим этот путь2. Выделим в нем 2 самых тяжелых ребра u и v. Эти 2 ребра мы обязаны были просмотреть в алгоритме и для каждого ребра вес пути, выданный алгоритмом, был бы u+v, а раз это меньше, чем вес пути1, то получим противоречие с тем, что алгоритм выдал путь 1.

10.4. Найти кратчайший цикл.

Сделаем BFS из каждой вершины, причем будем поддерживать 2 «сущности»: длину минимального цикла и то, как этот цикл можем восстановить. После всех итераций выведем ответ.

Почему это сработает: BFS, если встретит вершину, которая уже была помечена в этой итерации (скажем нашлось ребро из вершины на глубине h в уже помеченную вершину), то значит есть цикл длиной не больше h. Затем если запустить BFS из какой-либо вершины в этом цикле, то она сможет «определить» цикл (то есть наткнуться на вершину, которая уже была помечена) на глубине не больше h.

Таким образом для каждой вершины можно найти минимальный цикл, в котором она участвует.

Осталось две задачи:

- Как поддерживать длину цикла: это просто — будем в вершины записывать из текущую глубину в BFS. Тогда если нашлось ребро из вершины u (с глубиной H1) в помеченную вершину v (глубиной H2), то значит мы нашли цикл длиной H1+H2+1

Как восстановить цикл: опять стандартная идея: массив предков, из которой мы пришли в текущую вершину. Плюс к массиву нужно поддерживать вершину, с которой начался цикл: это может быть та вершина, в которую мы пришли как в помеченную..

10.5. Найти отрицательный цикл.

На лекции мы изучили, что если алгоритм Беллмана-Форда после n итераций сделал хотя бы одну релаксацию, то это значит, что в графе есть отрицательный цикл. Иначе, такого цикла нет.

Осталось научиться выводить его.

Для этого нужно уметь выводить не только расстояние до какой-либо вершины, но и сам кратчайший путь. Это делается стандартным методом: заведем массив «предков» и когда ребро нужно прорелаксировать, для соответствующей вершины обновим «предка».

Чтобы вывести отрицательный цикл, если он есть, нужно взять вершину, расстояние до которой изменилось на последней итерации алгоритма (то есть то ребро, которое прорелаксировалось) и пойти от нее по предкам пока не войдем в цикл – он и будет искомым.

UPD: *Почему будет цикл*: Да, я забыл это объяснить. Когда мы запоминаем вершину, в которой произошла релаксация на последней фазе алгоритма, то эта вершина либо лежит на цикле отрицательного веса, либо достижима из него. Можно сделать достаточно большое число переходов по ее родителям (скажем, n — число вершин) и получить новую вершину X, которая лежит на цикле. Дальше уже от новой вершины X идти по родителям, пока не достигнем опять X — так получим цикл. Мы обязательно попадем в вершину X, так как релаксации в цикле отрицательного веса будут всегда и будут идти по «кругу» — и соответственно у каждой вершины.