Домашняя работа

Котов Артем, МОиАД2020

а чего так грустно-то в этот раз?

4 февраля 2021 г.

Содержание

Task 1

Условие: Про разбиение на циклы.

Решение.

Сделаем граф из двух долей, в каждую поместим все вершины исходного графа. Соединим доли между друг другом ребрами единичной пропускной способности так, как они были соединены в исходном графе, но с условием, что ребра выходят только из левой доли.

Найдем максимальный поток в таком графе. Если он равен $|V| = n$, то мы смогли разбить на граф на циклы, если нет, то невозможно разбить граф на циклы.

Почему это работает? Рассмотрим конечный корректный ответ. Первое наблюдение: если пустить поток в цикле из истока в исток (т.е. как бы мы вышли их вершинки и в нее же вернулись), то величина потока через каждую вершину цикла одна и та же и равна 1, если пропускные способности 1. Второе наблюдение: в цикле если поток дошел, например, из вершины 1 до вершины 2, а из нее дошел до вершины 3, из которой вернулся в вершину 1, то на нашем рассматриваемом графе это бы значило, что у нас есть как бы цепочка: рассмотрим ребро $v_i \to v_j$, посмотрим, куда идет поток из v_j в левой доли, пусть идет в v_k, опять посмотрим куда идет ребро из v_k в левой доли и т. д., для корректного цикла мы в конце обязательно вернемся в 1-ую вершину.

Теперь, когда мы одновременно из вершины сливаем приходящий поток, и пускаем новый поток, то это эквивалентно предыдущей схеме, разве что мы как бы можем посчитать количество вершин в цикле, соответственно, если суммарное количество вершин в циклах равно полному количеству вершин в графе (можно опять же посмотреть на корректное разбиение и понятно, что это должно быть), то разбить можно. Восстановить непосредственно само разбиение можно по процедуре рассматривания куда идет поток от вершины из левой доли, если мы в нее пришли в правой доли (“цепочка” описанная ранее).

Update: Забыл про оценку сложности: для построенной сети ФФ отработает за $\mathcal{O}(|f|E')$, где $|f| = V$, так как мы ожидаем именно такой поток, а E' складывается из количества ребер в исходном графе E, количества ребер, ведущие из исток в левую долю (их V штук), и ребер, ведущие из правой доли в сток (их тоже V штук), тогда сложность можно оценить как $\mathcal{O}(|f|E') = \mathcal{O}\big(V(2V + E)\big)$.

Теперь важное уточнение, исходно я считал, что наш граф ну хотя бы связный, тогда $E \geq V - 1$ (для разных компонент тоже можно очень грубо оценить чем-то типо $E \geq \frac{V}{2}$), хотя для циклов мы можем гарантировать $E \geq V$, тогда в полученной оценке сложности доминирующим окажется вклад VE, следовательно итоговая сложность $\mathcal{O}(VE)$.

■

Task 2

Условие: Про единственность минимального разреза

Решение.

a) Сначала найдем минимальный разрез $C(s,t)$, это будет стоить нам по алгоритму Ф-Ф $\mathcal{O}(E|f|)$ (да-да, он нам выдает максимальный поток, но можно и восстановить минимальный разрез, например, запустив два раза `dfs` из истока и стока по дополняющей сети, получается, так как максимальный поток равен величине минимального разреза, т.е. за $\mathcal{O}(E)$ по потоку восстановить минимальный разрез, что не хуже, чем исходный Ф-Ф). Теперь пробежим по всем ребрам найденного минимального разреза и попытаемся увеличить пропускную способность текущего ребра и запустить вновь Ф-Ф (тут вообще говоря спорный вопрос, а за полиномиальное ли время Ф-Ф находит нам поток, так как $|f| \simeq \exp(V,T)$, например). Если каждый раз найденный максимальный поток увеличивался, то исходный разрез — единственный минимальный, если хотя бы на одном шаге не увеличивался, то это значит, что мы мажорированы каким-то другим минимальный разрезом. Вся эта процедура стоила нам $\mathcal{O}(E \cdot E|f|) = \mathcal{O}(E^2|f|)$.

b) Вообще можно было бы и предыдущий пункт так же сделать: в данном потоке запустим два `dfs` из истока и стока как в предыдущем пункте, т.е. мы как бы запретим дополнительно `dfs` ходить по насыщенным ребрам (т.е. найдем какой-то минимальный разрез). В итоге мы получим две компоненты связности (с т.з. дополняющей сети), в одной будет содержаться исток, в другой — сток. Теперь проверим, что сумма мощностей множества вершин в этих двух компонентах равна мощности множества вершин исходного графа. Если это так, то разрез единственен, иначе есть еще один минимальный разрез (он или они как бы отделяет(ют) еще одну или несколько других компонент, до которых мы не добрались из s и t).

■

Task 3

Условие: Про удаление ребер для увеличение количества компонент связности.

Решение.

a) Поправляю процедуру, для начала предположим, что у нас связный граф. Будем делать следующее, выберем какую-то произвольную точку за сток и зафиксируем ее. Переберем все оставшиеся вершины в качестве стока и каждую итерацию будем запускать ФФ, обозначим его сложность за F. Тогда за VF, мы прогонимся так по всем вершинами,

найдем минимальные разрезы, возьмем минимальный из них, удалим ребра найденного минимального среди всех минимальных разреза.

Если же компонент связности много, то мы проделаем эту процедуру для каждой компоненты (почему время такое же ниже).

Почему это корректно? Рассмотрим какой-то корректный ответ, то есть у нас есть какие-то две доли, соединенные некоторым количеством ребер, которые как раз соответствуют искомым в задаче. В нашей постановке мы как бы выбираем какую-то вершину из левой доли и перебираем все вершины из правой доли, чтобы гарантированно найти эти ребра (работает в рамках того, что пропускные способности 1), то есть исток лежит в какой-то доли разреза, а сток — в другой, но тут, кстати, может возникнуть проблема, если у нас много компонент связности, ну тогда мы можем просто перебрать эти компоненты этой же процедурой, тогда сложность оценивается следующей суммой $\sum V_i F_i \leq \sum V_i \max F = V \max_i F$, где V_i — число вершин (ясно, что $\sum V_i = V$), а F_i — сложность алгоритма в i-ой компоненте.

Замечание. *Наверно, более понятно будет сказать, что глобально нам все равно кто именно был истоком, важно лишь то, в какую долю разреза она попала (здесь можно поэксплуатировать неориентированность графа), и нужно найти вершину в другой доле.*

b) В предыдущем пункте мы очень грубо оценили F_i, давайте воспользуемся тем знанием, что пропускные способности всех ребер у нас равны 1. Не умаляя общности рассмотрим только одну компоненту связности. Рассмотрим, например, алгоритм Ф-Ф: его сложность $\mathcal{O}(E|f|)$, где f — величина максимального потока. При условии, что у нас пропускные способности всех ребер 1, то $|f| \leq \deg t$, так как мы не можем выкачать в стоке больше, чем кол-во входящих ребер (а из стока ничего не выходит).

   В итоге, сложность $\mathcal{O}(\sum E \deg t_i) = \mathcal{O}(E \sum \deg t_i)$, но при этом мы знаем, что $\sum \deg t_i = 2E$, следовательно, $\mathcal{O}(E^2)$.

■

Task 5

Условие: Про инструменты

Решение.

a) Составим граф, похожий на граф из 6-ой задачи, в качестве истока будет банк, в качестве стока — наша прибыль, в первый слой поместим инструменты, в которые проведем ребра с пропускной способностью p_i (цена инструмента), второй слой — заказы,

в исток от которых проведем ребра с c_i (прибыль от заказа), между слоями проведем ребра между инструментами и заказами, для которых эти инструменты нужны, с пропускной способностью ∞ (это нужно, чтобы наш последующий разрез проходил так, чтобы не учитывать те инструменты, которые мы не купили для заказов, которые собираемся делать).

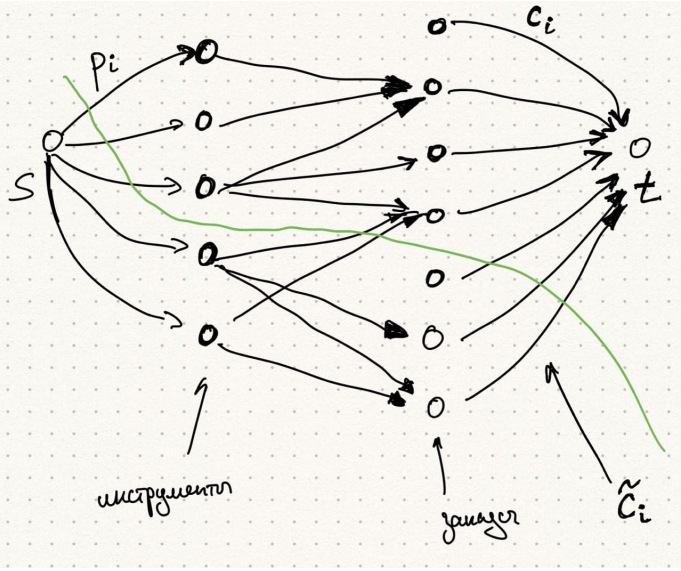

Рис. 1: Граф с разрезом (зеленая кривая). Из-за бесконечной пропускной способности ребер между слоями они не проходят через разрез (на картинке проходят, но не должны)

Рассмотрим баланс в таком графе:

$$\sum c_i - \sum p_i = C - \underbrace{(\sum \tilde{c}_i + \sum p_i)}_{C(s,t)} \Longrightarrow \sum \tilde{c}_i + \sum p_i \to \min,$$

где C — сумма вообще всех c_i, а $C(s,t)$ — величина разреза.

Теперь находим минимальный по величине разрез, тогда и прибыль максимизируется.

b) Still pending.

■