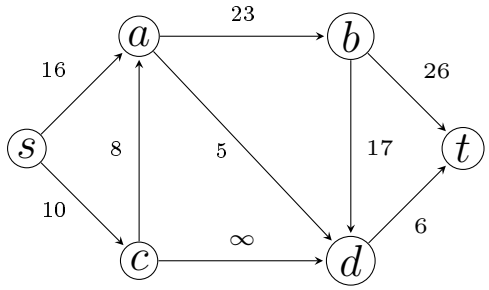
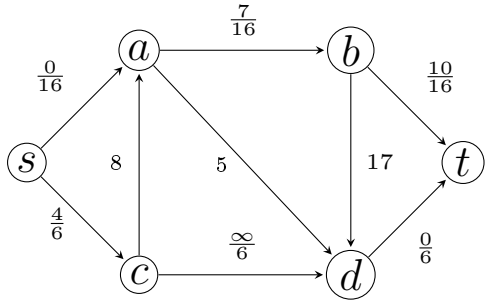
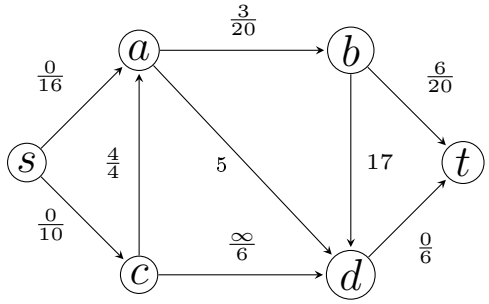
**1.** Найти максимальный поток и минимальный разрез, используя алгоритм Эдмондса-Карпа.




Первый шаг




Второй шаг




Ура мы получили максимальный поток, равный **26**.

Ну а минимальный разрез находим с помощью метода пристального взгляда на рисунок. Это набор ребер $\{(s,a),(s,c)\}$.

**2.** Пусть кто-то утверждает, что нашёл максимальный поток в некоторой сети...

Есть алгоритм, работающий за $O(1)$. Мы можем попросить этого человека доказать истинность его высказывания. В таком случае он все сделает за нас. Однако, этот алгоритм нам не подходит, т.к. у него не линейная сложность.

Вспоминаем Форда-Фалкерсона, с которым мы познакомились еще на первом семестре на АЛКТГ. По его словам, при максимальном потоке нельзя найти увеличивающие пути. Это справедливо.

Ну тогда нам просто надо проверить, можем ли мы найти какой-нибудь путь ненулевого веса из $s$ в $t$. Для этого достаточно будет пройтись по вершинам графа, начиная

с вершины $s$. Сложность обхода графа, очевидно, будет зависеть от числа вершин и числа ребер, т.е. $O(|V| + |E|)$. То есть алгоритм имеет линейную сложность, о чем нас и просят в условии.

Корректность: очевидна из алгоритма Форда-Фалкерсона.

**3** [ 7.21 ДПВ ]. Ребро в сети называется критическим, если...

1) Сначала строим рандомный кратчайший путь от $s$ до $t$ – мы это умеем делать за $O(|V| + |E|)$.

2) Далее в этом пути выбираем ребро с минимальным весом. Пусть это будет ребро $(a, b)$.

3) Строим все возможные пути из $a$ в $b$ НЕ через ребро $(a, b)$.

4) Если сумма всех путей из $a$ в $b$ (считая ребро $(a, b)$) меньше веса 2-ого по минимальности ребра, то одно из этих ребер (к примеру, $(a, b)$) является критическим.

Иначе берем второе по минимальности ребро и выполняем те же действия для него. (потом, возможно, берем 3-е по минимальности, 4-ое и т.д.)

Сложность алгоритма – $O(|V| + |E|)$, т.к. в худшем случае нам придется пройти по всем вершинам и по всем ребрам.

Оптимальность: поскольку нам в любом случае придется построить хотя бы один путь из $s$ в $t$, то быстрее чем за $O(|V| + |E|)$ мы справиться не сможем.

Корректность: действительно. По сути этот алгоритм строит небольшой подпоток исходного потока и ищем в нем насыщенное ребро. Если какое-то ребро будет насыщенным в этом подпотоке, то оно будет насыщенным и в исходном потоке.

**4** [ 7.18 ДПВ ]. Известно много вариаций задачи о максимальном потоке...

1. Имеется несколько истоков и несколько стоков, и нам нужно максимизировать общий поток из всех истоков во все стоки.

Просто склеиваем между собой все истоки и все стоки следующим образом:

1) Все $s_1, s_2 ... S_n$ становятся $s$ и все $t_1, t_2 ... t_m$ становятся $t$.

2) Если $\exists! i$ : ребро $(s_i, a)$, то появляется ребро $(s, a)$ с тем же весом.

3) Если $\exists i_1, i_2 ... i_k : \forall 1 \leqslant p, q \leqslant k \to i_p \neq i_q, p \neq q$ и ребра $(s_{i_p}, a), (s_{i_q}, a)$, то ребро $(s, a)$ будет иметь вес как сумму весов всех перечисленных ранее ребер.

4) Со стоками аналогично

2. Каждая вершина также имеет пропускную способность — максимальный поток, который может в неё входить.

1) Каждую вершину $p$ делим на $p_1, p_2$, соединяя ребром, равным весу вершины $p$. При этом в вершину $p_1$ входят все ребра, которые входили раньше в $p$, а из $p_2$ выходят все ребра, которые выходили раньше из $p$.

**Определение.** Пусть $G(V, E)$ — ориентированный ациклический граф (DAG). Множество вершинно-непересекающихся путей $P$ графа $G$ называется его *покрытием путями*, если каждая вершина множества $V$ входит в некоторый путь из $P$. Отметим, что пути могут соединять как любую пару вершин, так и состоять из одной вершины. Множество $P$ называется *минимальным покрытием путями*, если у $G$ не существует покрытия путями меньшего размера, т.е. $|P|$ минимально.

**5*.** Постройте эффективный алгоритм, который получив на вход DAG $G$ находит его (некоторое) минимальное покрытие путями.