2.4. Задачи комбинаторной оптимизации: о максимальном потоке, о паросочетании, о потоке минимальной стоимости.

1. Задача о максимальном потоке

Рассмотрим граф из n вершин. На его ребрах заданы пропускные способности: $c(i,j)$ — пропускная способность ребра из i в j. Также выделены две вершины: s — исток и t — сток.

Определение 1.. *Потоком называется функция $f(i,j)$, удовлетворяющая условиям:*

1. $f(i,j) \leqslant c(i,j)$
2. $f(i,j) = -f(j,i)$
3. $\sum_{k=1}^{n} f(i,k) = 0$ *для всех* $i \notin \{s,t\}$

Определение 2.. *Величиной потока называется величина* $F = \sum_{k=1}^{n} f(s,k) = \sum_{k=1}^{n} f(k,t)$

Задача: построить поток максимальной величины.

Определение 3.. *Остаточной пропускной способностью ребра называется величина* $c'(i,j) = c(i,j) - f(i,j)$.

Определение 4.. *Остаточной сетью называется граф из ребер, для которых* $c'(i,j) > 0$.

Если в остаточной сети есть путь из s в t (то есть есть $p_{1..k}$: $p_1 = s$, $p_k = t$, $f(p_i, p_{i+1}) < c(p_i, p_{i+1})$)), то можно увеличить поток вдоль этого пути на $\Delta f = \min c(p_i, p_{i+1}) - c(p_i, p_{i+1})$).

Теорема 1.. *Поток максимален* $\Leftrightarrow$ *В остаточной сети нет пути из s в t.*

Доказательство. $\Rightarrow$. Если путь есть, то можно увеличить поток вдоль ребер этого пути. $\Leftarrow$. Выделим множество вершин S, до которых есть путь из s, и множество T, до которых пути нет. Тогда $s \in S$, $t \in T$ и все ребра из S в T насыщены ($f(i,j) = c(i,j)$). Таким образом, из S в T не может течь больше, чем сейчас. □

Таким образом получаем простой алгоритм: увеличивать поток вдоль какого-то пути остаточной сети пока возможно.

Если увеличивать вдоль произвольного пути, то алгоритм может работать долго (если $c(i,j)$ целые, то зависит от них, если не целые, то может вообще не сойтись).

Модификации:

1. Увеличивать поток вдоль самого короткого пути. Работает за $O(nm^2)$.
2. (Алгоритм Диница). Сохраняем сеть обхода в ширину, чтобы быстро искать все короткие пути. Работает за $O(n^2m)$.
3. (Алгоритм Малхотры-Кумара-Махишвари). Так же, как в Динице, но быстрее находятся все короткие пути $O(n^3)$.

2. Задача о максимальном паросочетании

Определение 5.. *Паросочетание — это множество ребер, не имеющих общих концов. Величина паросочетания — количество ребер в нем.*

Задача: найти паросочетание максимальной величины.

Если граф не двудольный, то задача решается, но очень сложно, поэтому про это писать не будем.

Пусть граф двудольный: n вершин слева, n вершин справа, некоторые соединены ребрами. Приделаем слева вершину-исток, справа — вершину-сток. Ориентируем ребра слева направо и сопоставим им пропускные способности, равные 1. Тогда поиск паросочетания сводится к поиску потока (рис. 1).

Таким образом, можно находить паросочетание за $O(nm)$ (величина паросочетания не больше n, поиск пути $O(m)$).

Модификация: Ищем пути длиной не больше $\sqrt{n}$ алгоритмом Диница, остальные (можно доказать, что их останется не больше $\sqrt{n}$) обычным путем. Общее время работы $O(n\sqrt{m})$.

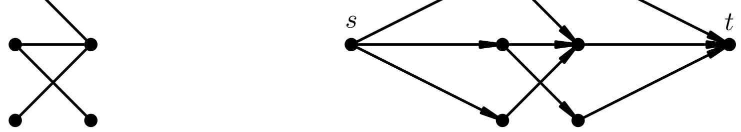

Рис. 1: Переход от паросочетания к потоку.

3. Задача о максимальном потоке минимальной стоимости

Добавим ребрам в задаче о потоке стоимости $p(i, j)$.

Определение 6.. *Стоимостью потока называется величина* $P = \sum_{i=1}^{n} \sum_{j=1}^{n} f(i, j)p(i, j)$

Задача: найти максимальный поток минимальной стоимости.

Решение: будем увеличивать поток каждый раз вдоль пути с минимальной стоимостью. Можно доказать, что тогда получится поток минимальной стоимости.