Алгоритмы. HW#10

Тураев Тимур, 504 (SE)

1 *Может ли первая команда стать чемпионом*

Построим такую сеть: $< s, t, G, T, U, V, W >$, где s – это исток, а t – сток. Множество вершин G – это всевозможные игры между двумя командами (их число $\binom{n}{2}$). Множество вершин T – это все команды.

Ребра U соединяют сток с G, причем ребро есть только тогда, когда игры еще не сыграна. Пропускная способность таких ребер – 1.

Ребра V соединяют G с T: из вершины $g_{i,j}$ выходят 2 ребра в вершины t_i и t_j. Пропускная способность таких ребер – бесконечность. Ясно, что в каждую вершину t_i входит ровно $n-1$ ребро.

Ребра W соединяют вершины T с вершиной t, со стоком. Причем пропускная способность ребра (t_1, t) равна бесконечности, а пропускные способности остальных ребер (вида (t_i, t)) равны тому числу игр, которая команда i может выигрыть, ничего не испортив при этом для команды 1.

Теперь ясно, что максимальный поток в такой сети даст нужное нам расписание. И ответом на задачу будет «да» тогда, когда максимальный поток будет равен числу игр, которые осталось сыграть. Или, другими словами, все ребра выходящие из истока должны быть насыщены.

Остался вопрос как найти пропускную способность ребер W. Пусть осталось сыграть K игр (это число легко найти, считывая таблицу). Предположим, первая команда все их выигрывает, тогда максимальное число побед, которые может одержать команда i (то есть пропускная способдность ребра (t_i, t)) равно $K - 1 - M$, где M – число побед, которые уже одержала команда i.

2 *Задача про сетку*

Хочется, чтобы через каждую вершину проходил максимум один путь, для этого естественно желание повесить пропускную способность не только на ребра, но и на вершины.

Давайте так и сделаем: пусть у каждой вершины v будет какая-то пропускная способность d, тогда задача поиска максимального потока в такой сети сводится к обычной следующим образом: разобьем вершину v на 2 обычные вершины: v_1 и v_2 и проведем между ними ребро с пропускной способностью d. Дополнительно, для каждого ребра в исходном графе (u, v) с пропуской способностью c проведем ребро в новом графе вида $(u, v1)$ с тем же весом c, аналогично для всех ребер вида (v, u). Достаточно очевидно, что задача поиска максимального потока в исходном графе даст соответствующий максимальный поток в таком графе.

Сведем нашу задачу к такой: пусть сетка это граф, где вершины – узлы сетка, а ребра – отрезки сетки. Введем новую вершину s – исток – и соединим ее ребрами со всеми выделенными вершинами и вершину t – сток – и соединим ее со всеми вершинами на границе, то есть с вершинами степени меньше 4. Далее, как в предыдущем пункте, разобьем каждую вершину на две и назначим всем ребрами в графе вес 1.

Достаточно очевидно, что максимальный поток соответствуют непересекающимся путям в исходном графе, а его величина – их числу. Если величина потока меньше, чем было отмеченных вершин, то задача не имеет решения.

Размер получившейся сети, очевидно, $O(n \cdot m)$ – число вершин стало в 2 раза больше (и еще плюс две), а число ребер стало $2|E| + |V|$.

3 *Докажите, что после каждой итерации поиска блокирующего потока расстояние между стоком и истоком увеличивается хотя бы на единицу.*

Во-первых ясно, что расстояние между стоком и истоком с каждой итерацией не уменьшается. Действительно, пусть мы пришли к какому-либо кратчайшему $s - t$ пути P, длина которого равна $dist^{(i+1)}(t)$. Но из определения остаточной сети видно, что она состоит либо из ребер остаточной сети на предыдущей итерации, либо из обратных к ним. Если этот путь состоит только из ребер «предыдущей» итерации, то ясно, что его длина никак не меньше кратчайшего $s - t$ пути, которая равна $dist^{(i)}(t)$. А если там было хотя бы одно обратное ребро (u, v) (пусть это будет первое такое ребро – тогда по первой части $dist^{(i+1)}(u) \geqslant dist^{(i)}(u)$), то это значит, что оно появилось там из-за того, что по ребру (v, u) был пропущен какой-то поток, значит ребро (v, u) было в дополнительной сети, значит (по определению) это ребро вело на следующий уровень, то есть $dist^{(i)}(u) = dist^{(i)}(v) + 1$.

А наш путь P – кратчайший, значит расстояние до v на единицу больше, чем до u, которое в свою очередь не меньше чем оно же на предыдущей фазе, которое в свою очередь $dist^{(i)}(u) = dist^{(i)}(v) + 1$. Значит, $dist^{(i+1)}(v) \geqslant dist^{(i)}(v) + 2 > dist^{(i)}(v)$. *Если же обратных ребер несколько, то можно так итеративно рассматривать по очереди каждое обратное ребро: видно что оно вносит вклад 2 в длину пути.*

А теперь осталось заметить, что равенства вообще быть не может – иначе в этом случае получим, что нашли путь, состоящий только из «прямых» ненасыщенных ребер остаточной сети с предыдущей итерации, причем длина его не изменилась. А это значит, что наш алгоритм поиска блокирующего потока засбоил: нашли путь, который не затронул блокирующий поток, чего быть не может.

Таким образом, $dist^{(i+1)}(t) > dist^{(i)}(t)$