Домашняя работа № 2

Автор: Минеева Екатерина

Задача 1

Описание алгоритма «MatchingApprox»:

Начнем с пустого паросочетания. В цикле по всем ребрам будем каждое из них добавлять в паросочетание, если это возможно – то есть, если в текущем ребре (u, v) обе вершины не входят в паросочетание, построенное к этому моменту.

Докажем, что такой алгоритм построит паросочетание, отличающееся от максимального не более, чем в 2 раза.

▲ (Индукция по количеству ребер в исходном графе.)

База: Если ребро одно, то максимальное паросочетание состоит из этого ребра. Такое же паросочетание найдет и $MatchingApprox$.

Шаг: Предположим, что для любого графа, в котором $< n$ ребер $MatchingApprox$ находит паросочетание отличающееся по размеру от максимального не более, чем вдвое.

Пусть в графе $G(V, E)$, $|E| = n$, $M = \{(u_1, v_1), (u_2, v_2), \dots (u_k, v_k)\}$ – паросочетание, найденное $MatchingApprox$. M^* – паросочетание максимального размера, $|M| = k^*$. Заметим, что $k = 0$ в том и только том случае, если в исходном графе нет ребер – в этом случае утверждение верно. Допустим, $k \neq 0$, рассмотрим ребро (u_k, v_k) – добавленное в паросочетание последним. Возможны 2 случая:

1 случай: $(u_k, v_k) \in M^*$. Рассмотрим граф $G'(V', E')$, где $V' = V \setminus \{u_k, v_k\}$, $E' = E \setminus (((\{u_k, v_k\} \times V) \cup (V \times \{u_k, v_k\}))$. .

Паросочетание, которое найдет $MatchingApprox$ это $M' = M \setminus \{(u_k, v_k)\}$.

Заметим, что $M'^* = M^* \setminus \{(u_k, v_k)\}$ – максимальное паросочетание для G'. Если бы это было не так, то существовало большее, чем M^* паросочетание для G. По предположению индукции (его можно применить, поскольку $|E'| < |E|$) получаем, что $2|M'| \geq |M'^*|$. При этом $|M'^*| = |M^*| - 1$, $|M'| = |M| - 1 \Rightarrow 2(|M| - 1) \geq |M^*| - 1 \Leftrightarrow 2|M| \geq |M^*| + 1 \geq |M^*|$. То есть утвердение верно и для G.

2 случай: $(u_k, v_k) \notin M^*$. Путь тогда $(u_k, v'_k), (u'_k, v_k) \in M^*$. Рассмотрим граф $G'(V', E')$, где $V' = V \setminus \{u_k, v_k, u'_k, v'_k\}$, $E' = E \setminus (((\{u_k, v_k, u'_k, v'_k\} \times V) \cup (V \times \{u_k, v_k, u'_k, v'_k\}))$.

Пусть M' – паросочетание, которое найдет $MatchingApprox$ в G'. Заметим, что $|M'| \leq |M| - 1$: исключили вершины u_k, v_k, которые входили в одно ребро в M, и вершины v'_k и u'_k, каждая из которых могла быть концом одного из ребер в M. При этом $M'^* = M^* \setminus \{(u_k, v'_k), (u'_k, v_k)\}$ – максимальное паросочетание в G'. Аналогично: предположим, что есть паросочетание большего размера в G', тогда и в G есть паросочетание большего размера. При этом $|M'^*| = |M^*| - 2$.

Поскольку $|E'| < |E|$ к графу G' можно применить предположение индукции. $2|M'| \geq |M'^*| \Rightarrow 2(|M| - 1) \geq 2|M'| \geq |M^*| - 2 \Rightarrow 2|M| \geq |M^*|$. То есть в этом случае утверждение для G тоже верно.

■

Детали реализации и оценка сложности:

Чтобы проверять, можно ли добавить очередное ребро в паросочетание, заведем булев массив $used$ длины n, в котором $used[v_i] = true$, если вершина в паросочетании и $false$ иначе. Тогда сложность работы алгоритма $\underline{O}(m)$. Памати затрачивается $\underline{O}(m + n)$