1 Sublinear-Time Approximation of Maximum Matching

Consider a graph \( G = (V, E) \). Recall that a matching is a set of edges \( M \subseteq E \) such that no two of the edges in \( M \) share an end-point. A fractional matching is the corresponding natural relaxation, where we assign to each edge \( e \in E \) a value \( x_e \in [0, 1] \) such that the summation of the edge-values in each node is at most 1, that is, for each node \( v \in V \), we have \( \sum_{e \in E(v)} x_e \leq 1 \), where \( E(v) \) denotes the set of edges incident on \( v \). We define \( y(v) = \sum_{e \in E(v)} x_e \) as the value of node \( v \) in the given fractional matching. The size of a fractional matching is defined as \( \sum_{e \in E} x_e \), and we have \( \sum_{e \in E} x_e = (\sum_{v \in V} y(v))/2 \) (why?). We call a fractional matching almost-maximal if for each edge \( e \in E \), there is one of its endpoints \( v \in e \) such that \( y(v) = \sum_{e' \in E(v)} x_{e'} \geq \frac{1}{1+\epsilon} \).

Exercise

(1a) In the class, we saw that any maximal matching has size at least \( 1/2 \) of the maximum matching. Prove that the size \( \sum_{e \in E} x_e = (\sum_{v \in V} y(v))/2 \) of any almost-maximal fractional matching is at least \( \frac{1}{2(1+\epsilon)} \) of the size of maximum matching.

Consider a maximum matching \( M^* \) and an almost-maximal fractional matching which has value \( x_e \) on each edge \( e \). We prove that \( \sum_{e \in E} x_e \geq \frac{|M^*|}{2(1+\epsilon)} \). Consider \( |M^*| \) dollars spread around, where we have put one dollar on each edge \( e \) of the maximum matching \( M^* \). By the almost-maximality of the fractional matching, each edge \( e \) has at least one endpoint \( v \in e \) such that \( y(v) = \sum_{e' \in E(v)} x_{e'} \geq \frac{1}{1+\epsilon} \). Make edge \( e \) send its one dollar to one such endpoint \( v \). This way, each node receives at most one dollar (why?). Now, make node \( v \) split its one dollar among its incident edges \( E(v) \) proportional to the values \( x_{e'} \). This way, each edge receives at most \((1+\epsilon)x_{e'}\) dollars from \( v \). More generally, each edge \( e' \) receives at most \((1+\epsilon)x_{e'}\) dollars from each of its endpoints and thus overall at most \( 2(1+\epsilon)x_{e'} \). We can conclude that \( \sum_{e' \in E} 2(1+\epsilon)x_{e'} \geq |M^*|\). In other words, \( \sum_{e \in E} x_e \geq \frac{|M^*|}{2(1+\epsilon)} \).

Thus, the above item indicates that almost-maximal fractional matchings also provide a reasonable approximation of the size of the maximum matching. But computing an almost-maximal fractional matching is much easier. We next see a LOCAL algorithm for that.

LOCAL-Algorithm for Almost-Maximal Fractional Matching: Initially, set \( x_e = 1/\Delta \) for each edge \( e \in E \). Then, for \( \log_{1+\epsilon} \Delta \) iterations, in each iteration, we do as follows:

- For each vertex \( v \) such that \( y(v) = \sum_{e \in E(v)} x_e \geq \frac{1}{1+\epsilon} \), we freeze all of its incident edges.
- For each unfrozen edge \( e \), set \( x_e \leftarrow x_e \cdot (1+\epsilon) \).

Exercise

(1b) Prove that the process always maintains a fractional matching, meaning that we always have \( \sum_{e \in E(v)} x_e \leq 1 \).

Per iteration, we freeze all edges incident on vertices \( v \) whose sum \( y(v) \) has passed \( \frac{1}{1+\epsilon} \) and then we increase unfrozen edges by a \((1+\epsilon)\) factor. Hence, the value \( y(v) \) can increase to at most \( \frac{1}{1+\epsilon} \cdot (1+\epsilon) = 1 \), but cannot pass that.
(1c) Prove that at the end, we have an almost-maximal fractional matching, meaning that for each edge $e \in E$, there is one of its endpoints $v \in e$ such that $\sum_{e \in E(v)} x_e \geq \frac{1}{1+\epsilon}$.

For each edge $e$, either during some it gets frozen because one of its endpoints $v \in e$ reaches a sum $y(v) = \sum_{e \in E(v)} x_e \geq \frac{1}{1+\epsilon}$, or the edge $e$ gets multiplies by $(1 + \epsilon)$ in each iteration. The latter means $x_e$ reaches a value of $\frac{1}{\epsilon} \cdot (1 + \epsilon)^{\log_{1+\epsilon} \Delta} = 1$. That would imply that even both of the endpoints $v \in e$ have $\sum_{e \in E(v)} x_e \geq \frac{1}{1+\epsilon}$.

Now that we have a simple LOCAL-algorithm for almost-maximal fractional matching, we use it to obtain a centralized algorithm for approximating the maximum matching. To estimate the size of maximum matching, we pick a set $S$ of $k = \frac{20\Delta \log 1/\delta}{\epsilon^2}$ nodes at random (sampled with replacement). Here, $\delta$ is some certainty parameter $\delta \in [0,0.25]$. For each sampled node $v \in S$, we run the above LOCAL-algorithm around $v$, hence allowing us to learn $y(v)$.

Exercise

(1d) Define a linear function $f : \mathbb{R} \to \mathbb{R}$ such that when applied on the sample average $\sum_{v \in S} y(v)/|S|$, the resulting value $f(\sum_{v \in S} y(v)/|S|)$ is an unbiased estimator of $\sum_{e \in E} x_e = \sum_{e \in E} x_e$. That is,

$$\mathbb{E}_S[f(\sum_{v \in S} y(v)/|S|)] = \sum_{e \in E} x_e.$$

We have $\mathbb{E}_S[\sum_{v \in S} y(v)/|S|] = \frac{2\sum_{e \in E} x_e}{n}$ (why?). Hence, it suffices to define $f(z) = nz/2$.

(1e) What is the query complexity of our sublinear-time approximation algorithm?

Per sampled node, we need to simulate the algorithm in its $(\log_{1+\epsilon} \Delta)$-hop neighborhood. The size of this neighborhood and thus also the related query complexity is at most $O(\Delta \log_{1+\epsilon} \Delta)$. Hence, the overall query complexity is $O(k \Delta \log_{1+\epsilon} \Delta) = O(\Delta^{1+\log_{1+\epsilon} \Delta} \cdot \log_{1+\epsilon} \delta)$. In terms of dependency on $\Delta$, this is much better than the $2O(\Delta)$ bound that we saw in the class.

(1f) Prove that the estimator that you defined in (1d) gives a $(2 + 3\epsilon)$-approximation of the maximum matching size, with probability at least $1 - \delta$.

By (1d), we know that the expectation of our estimator is $\sum_{e \in E} x_e$, which we know by (1a) is within a $2(1 + \epsilon)$ factor of the size of the maximum matching. We next examine how much the random value may deviate from this expectation. Define $X_i$ to be the random variable that is equal to $y(s_i)$ where $s_i$ denotes the $i^{th}$ node in our sample set $S$. Notice that $X_i \in [0,1]$ and moreover, $\mathbb{E}[X_i] = \frac{2\sum_{e \in E} x_e}{n}$. Hence, $\mu = \mathbb{E}[\sum_{i=1}^k X_i] = \sum_{i=1}^k \mathbb{E}[X_i] = k \frac{2\sum_{e \in E} x_e}{n}$. Also notice that $\sum_{e \in E} x_e \geq \frac{n}{2\Delta}$ (why?) and thus, $\mu \geq k \frac{n}{2\Delta(1 + \epsilon)} = \frac{20\Delta \log 1/\delta}{\epsilon^2} \cdot \frac{1}{2\Delta} = \frac{10\log 1/\delta}{\epsilon^2}$. Therefore, by Chernoff bound, the probability that $X = \sum_{i=1}^k X_i$ deviates by more than a $(1 + \epsilon)$ factor from its expectation $\mu$ is at most

$$2e^{-\epsilon^2 \mu/3} = 2e^{-\frac{20\log 1/\delta}{\epsilon^3}} \leq \delta.$$

Thus, with probability at least $1 - \delta$, we get an expectation within a $2(1 + \epsilon)(1 + \epsilon) \leq 2 + 3\epsilon$ factor of the maximum matching.