Deterministic online matching

In the Bipartite Matching problem, the input is given by a bipartite graph $G = (U, V, E)$ such that $E \subseteq U \times V$. The goal is to compute a *matching* of G of maximum size, that is a subset of edges $M \subseteq E$ such that every vertex of G is incident to at most one edge of M.

In the online version of this problem[1], we get only vertices U on input. In particular, vertices V, its size, and edges of G are not known in advance. The goal is to build a matching M of maximum size. In each "step", we receive a vertex $v \in V$ and edges v is incident to. After receiving v, we can decide to add an edge incident to v to M. However, adding edges to M is irrevocable, that is, we can only add edges to M and M has to be a matching at every step of the algorithm. You can assume that the final graph has a perfect matching[2], which is also the optimum.

1. Design a (simple) deterministic algorithm for Online Bipartite Matching and analyze it.

2. Show that the algorithm you designed in Task 1 is optimal.

   And if it is not, then find a better one.

Randomized online matching

3. Consider the following algorithm: after receiving vertex v, we select a random unmatched neighbor $u \in U$ of v and add $\{u, v\}$ to M. Show that this algorithm does not have a better competitive ratio that the optimal deterministic algorithm.

   Hint. Creating an instance where the algorithm creates a matching with at most $\frac{n}{2}$ edges will not be easy. However, creating an instance where the algorithm produces a matching of size $\frac{n}{2} + \mathcal{O}(\log n)$ is doable.

Probabilistically checkable proofs

Definition (Class PCP)**.** A decision problem belongs to complexity class $\mathrm{PCP}_{c,s}(r(n), q(n))$ if there exists a *verifier* which

1. uses at most $r(n)$ random bits and accesses at most $q(n)$ bits of a *proof*[3],
2. if the input is a `YES` instance, then the verifier answers `YES` with probability at least c,
3. if the input is a `NO` instance, then the verifier answers `YES` with probability at most s.

Today we use $c = 1$ and $s = \frac{1}{2}$.

Theorem (PCP theorem)**.** $\mathrm{PCP}(\mathcal{O}(\log n), \mathcal{O}(1)) = \mathrm{NP}$.

4. Show that $\mathrm{PCP}(\mathcal{O}(\log n), 0) = \mathrm{P}$.

5. Show that $\mathrm{PCP}(0, \mathcal{O}(\log n)) = \mathrm{P}$.

6. Show that $\mathrm{PCP}(0, \mathrm{poly}(n)) = \mathrm{NP}$.

7. Show that Graph Non-isomorphism belongs to $\mathrm{PCP}(\mathrm{poly}(n), \mathcal{O}(1))$.

---

[1] Known as Online Bipartite Matching.
[2] That is $|U| = |V|$ and there exists a matching of size $|U|$.
[3] Also called a *certificate*.

Approximation and Online Algorithms – Tutorial no. 10 2022/05/09, 14:00

Deterministic online matching

In the Bipartite Matching problem, the input is given by a bipartite graph $G = (U, V, E)$ such that $E \subseteq U \times V$. The goal is to compute a *matching* of G of maximum size, that is a subset of edges $M \subseteq E$ such that every vertex of G is incident to at most one edge of M.

In the online version of this problem[1], we get only vertices U on input. In particular, vertices V, its size, and edges of G are not known in advance. The goal is to build a matching M of maximum size. In each "step", we receive a vertex $v \in V$ and edges v is incident to. After receiving v, we can decide to add an edge incident to v to M. However, adding edges to M is irrevocable, that is, we can only add edges to M and M has to be a matching at every step of the algorithm. You can assume that the final graph has a perfect matching[2], which is also the optimum.

1. Design a (simple) deterministic algorithm for Online Bipartite Matching and analyze it.

2. Show that the algorithm you designed in Task 1 is optimal.

   And if it is not, then find a better one.

Randomized online matching

3. Consider the following algorithm: after receiving vertex v, we select a random unmatched neighbor $u \in U$ of v and add $\{u, v\}$ to M. Show that this algorithm does not have a better competitive ratio that the optimal deterministic algorithm.

   Hint. Creating an instance where the algorithm creates a matching with at most $\frac{n}{2}$ edges will not be easy. However, creating an instance where the algorithm produces a matching of size $\frac{n}{2} + \mathcal{O}(\log n)$ is doable.

Probabilistically checkable proofs

Definition (Class PCP)**.** A decision problem belongs to complexity class $\mathrm{PCP}_{c,s}(r(n), q(n))$ if there exists a *verifier* which

1. uses at most $r(n)$ random bits and accesses at most $q(n)$ bits of a *proof*[3],
2. if the input is a `YES` instance, then the verifier answers `YES` with probability at least c,
3. if the input is a `NO` instance, then the verifier answers `YES` with probability at most s.

Today we use $c = 1$ and $s = \frac{1}{2}$.

Theorem (PCP theorem)**.** $\mathrm{PCP}(\mathcal{O}(\log n), \mathcal{O}(1)) = \mathrm{NP}$.

4. Show that $\mathrm{PCP}(\mathcal{O}(\log n), 0) = \mathrm{P}$.

5. Show that $\mathrm{PCP}(0, \mathcal{O}(\log n)) = \mathrm{P}$.

6. Show that $\mathrm{PCP}(0, \mathrm{poly}(n)) = \mathrm{NP}$.

7. Show that Graph Non-isomorphism belongs to $\mathrm{PCP}(\mathrm{poly}(n), \mathcal{O}(1))$.

---

[1] Known as Online Bipartite Matching.
[2] That is $|U| = |V|$ and there exists a matching of size $|U|$.
[3] Also called a *certificate*.