University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
P. Bamberger, P. Schneider

Algorithm Theory
Sample Solution Exercise Sheet 13 - Bonus*

Due: Tuesday, 1st of February, 2021, 4 pm

**Bonus points can be earned normally, but do not increase the* threshold *for the "Studienleistung".*

Exercise 1: Online Vertex Cover *(10* Points)*

Let $G = (V, E)$ be an unweighted undirected graph. Consider the following online version of the *minimum vertex cover* problem. Initially, we are given the set of nodes V and an empty vertex cover $S = \emptyset$. Then, the edges appear one-by-one in an online fashion. When a new edge $\{u, v\}$ appears, the algorithm needs to guarantee that the edge is covered (i.e., if this is not already the case, at least one of the two nodes u and v needs to be added to S). Once a node is in S it cannot be removed from S.

(a) Provide a deterministic online algorithm with strict competitive ratio at most 2. *(1 Point)*

(b) Show that any deterministic online algorithm for the online vertex cover problem has strict competitive ratio at least 2. *(4 Points)*

(c) Use Yao's principle to show that any randomized online algorithm for the online vertex cover problem has a strict competitive ratio at least 3/2. *(5 Points)*

Sample Solution

(a) *We can use the same trick as in* `alg1` *on the previous exercise sheet. The proof of correctness is not necessary to obtain the point (and was already shown in the last exercise sheet).*

Online Algorithm: Start with $S = \emptyset$. When a new edge appears our online algorithm first checks whether the new edge is already covered, i.e., if it has at least one endpoint in S. If that edge is not covered the algorithm adds *both* endpoints of the new edge to S.

(b) Let us fix OPT to be an optimal offline algorithm and ALG to be any online algorithm. We (as an adversary) construct a scenario with only two incident edges where ALG has to put two vertices in S while OPT can cover both edges by only one vertex.

We send the first edge. If ALG puts both endpoints of the first edge in S then the second edge can be connected to any endpoint of the first edge. If ALG adds only one of the endpoints of the first edge to S then we connect the second edge to the endpoint of the first edge that is not in S. So the second edge has no endpoint in S and ALG has to add at least one endpoint of the second edge to S. Therefore for each choice of ALG there is an online instance where ALG returns S of size at least 2.

By contrast, OPT chooses only the middle vertex to cover both edges. Note that OPT can do this since it is offline and knows the input sequence in advance. This means that for any deterministic algorithm ALG we have a online order of edges of a graph G (given above), such that for output S of ALG and the minimum vertex cover S^* of G we have $|S| \geq 2|S^*|$. This proves that we have a *strict* competetive ratio of at least 2 for deterministic algorithms.

(c) We construct a randomized input and show that the best *deterministic* algorithm for that kind of input distribution has an expected (strict) competitive ratio of $\frac{3}{2}$. The random input is as follows. We send two adjacent edges. Let $\{u, v\}$ be the first edge being sent. Then the second edge is subject to a “coin flip”. With probability $\frac{1}{2}$ we attach it to u otherwise to v.

As before, OPT can cover both edges with just one node in S. We have to show that ALG uses at least $\frac{3}{2}$ nodes in expectation. ALG has to choose after the first edge $\{u, v\}$ arrives, which node(s) it adds to S. Since ALG is deterministic it has the option to either add both nodes, or just one (fixed) node. If ALG would add both nodes u, v to S then the competitive ratio is always 2 (no matter where the ext edges attaches to).

It is smarter for ALG to just pick just one node for the first edge, w.l.o.g. u and hope that the next edge will attach to it. Then with probability $\frac{1}{2}$ ALG is lucky and the next edge attaches to u and it needs only one node to cover both edges. However, with probability $\frac{1}{2}$ ALG is unlucky and the next edge attaches to v, thus ALG has to add a second node to S for a valid vertex cover. The expected number of nodes is $\mathbb{E}(|S|) = \frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 2 = \frac{3}{2}$.

By Yao’s principle the expectation of the best randomized algorithm on a worst case input can only be as good as that of the best deterministic algorithm on a worst case randomized input. Therefore any randomized algorithm for the online vertex cover problem has a strict competitive ratio at least $\frac{3}{2}$.

Exercise 2: Maximum Cut *(10* Points)*

Let $G = (V, E)$ be an unweighted undirected graph. A *maximum* cut of G is a cut whose size is at least the size of any other cut in G. We consider an online version of the maximum cut problem, where the nodes V of a graph $G = (V, E)$ appear in an online fashion. The algorithm should partition the nodes V into two sets A and B such that induced cut is as large as possible. Whenever a new node appears we also learn its edges to the nodes that have *already appeared before*. An incoming node has to be immediately assigned to either A or B and that decision is final.

(a) Describe a deterministic online maximum cut algorithm with *strict competitive ratio* at least 1/2.

Hint: An online algorithm for a maximization problem is said to have strict competitive ratio α if it guarantees that ALG $\geq \alpha \cdot$ OPT, *where* ALG *and* OPT *are the solutions of the online algorithm and of an optimal offline algorithm, respectively.* *(5 Points)*

(b) Show that no deterministic online algorithm for the online maximum cut problem can have a (constant) strict competitive ratio that is better (larger) than 1/2. *(5 Points)*

Sample Solution

(a) The following deterministic algorithm is strictly $\frac{1}{2}$-competetive.

Algorithm 1 Deterministic Online Maximum Cut

```
Pick arbitrary nodes $v_1, v_2 \in V$
$A \leftarrow \{v_1\}$
$B \leftarrow \{v_2\}$
for each new incoming node $v$ do
    if $deg_A(v) > deg_B(v)$ then        ▷ $deg_X(v)$ is the number of $v$'s neighbors in $X \subseteq V$.
        $B \leftarrow B \cup \{v\}$
    else
        $A \leftarrow A \cup \{v\}$
Output $A$ and $B$
```

Correctness: We distinguish between *crossing edges* with one endpoint in A and one in B and *inner edges* with either both endpoints in A or both in B. We show that the following property is a loop-invariant: The number of crossing edges is at least the number of inner edges.

This is true before entering the loop the first time (there are no inner edges as A, B each contain just one node). In the following iterations, a node is added to B iff it has more neighbors in A than it has in B and it is added to A iff it has at least as many neighbors in B as in A.

In either case, the number of *crossing* edges that are added is at least the number of *inner* edges that are added which leaves to loop invariant valid after each iteration. Due to that loop invariant, after all nodes are processed the resulting cut has size at least $|E|/2$ and $|E|$ is an upper bound for a maximum cut.

(b) Consider a graph with nodes $v_1, v_2, \ldots, v_n$. Nodes v_1 and v_2 are adjacent to *every* other node, more edges do not exist. The cut $(\{v_1, v_2\}, \{v_3, \ldots, v_n\})$ has size $2(n-2) \leq$OPT. Assume ALG is an online algorithm with competitive ratio $r > 1/2$ which computes a cut $(S, V \setminus S)$ for any graph (V, E). If ALG first receives v_1 and v_2, it must put one in S and the other in $V \setminus S$, because otherwise we had ALG$= 0$ and OPT$= 1$ in case the graph consists only of v_1 and v_2 (which an online algorithm can not know at this point). Any cut with $v_1 \in S$ and $v_2 \notin S$ (or vice versa) has size $n-1$. Therefore we have ALG/OPT$\leq \frac{n-1}{2(n-2)} \overset{n\to\infty}{\longrightarrow} 1/2$. So for n sufficiently large we have ALG/OPT$< r$, contradicting that ALG has competitive ratio r.