ONLINE ALGORITHMS

Exercise Sheet 1

PD Dr. Walter Unger
Janosch Fuchs
Department of Computer Science
RWTH Aachen University

Due Date: April 24, 17:45
(Where: Lecture or Boxes in front of i1)

• Write the name, group number and enrollment number of each group member on every sheet that you hand in.
• To achieve the permission for the exam you must earn 50% of the sum of all points and present one of your solutions at least once.

Bonus Exercise 1 (1 point (bonus))
Visit https://www-i1.informatik.rwth-aachen.de/Lehre/SS18/Online.php and read the Text that explains the exercise System. Participate in the poll if you want that we consider your schedule.
If you fill out the Foodle as explained, you will receive a bonus point. It is important that you use your RWTH mail as name in the Foodle.

Exercise 2 (3 points)
The paging problem is an online minimization problem. Let there be a positive integer \( k \) which denotes the cache size. Initially the cache is filled with the first \( k \) pages \( B_0 = \{p_1, \ldots, p_k\} \). There are \( m \) different pages. An input sequence has the form \( I = \{x_1, x_2, \ldots, x_n\} \) with \( x_i \in \{p_1, p_2, \ldots, p_m\} \) as the page that arrives in time step \( i \). If a page \( x_i \) in time step \( i \) is requested and the page is not part of the current cache (\( x_i \notin B_{i-1} \)), the algorithm has to replace a page \( p_j \in B_{i-1} \) such that the new cache is of the form \( B_i = (B_{i-1} \setminus \{p_j\}) \cup \{x_i\} \). If a page is replaced in step \( i \), the output is \( y_i = 1 \). Otherwise the output is \( y_i = 0 \).
The goal is to minimize the cost of an algorithm \( A \) which is defined as \( \text{cost}(A) = \sum_{i=1}^{n} y_i \).

(a) Assume you get the paging problem as offline problem, so \( I \) is known beforehand. Describe in a few sentences how your algorithm would find the optimal solution. We denote the cost of an optimal solution as \( \text{OPT} \).

(b) You have three different types of algorithms. A LIFO \( A_1 \), a FIFO \( A_2 \) and an algorithm \( A_3 \) that replaces the page which was least frequently used. Derive for every algorithm an instance of the paging problem with \( m \geq 5 \), such that \( \text{cost}(A_i) \geq k \cdot \text{OPT} \) holds.

Exercise 3 (3 points)
A friend asks you for help. He/She wants to go on a ski trip. Both of you do not know how the weather will be. But there are only two states: good enough for skiing or not. Your friend is for \( n \) days on the trip and you can decide each day to rent or buy your ski with the knowledge of the weather for the current day. Renting costs 1 and buying \( k \). If skis are bought they will not be rented anymore.
Can you tell your friend what to do? Prove that your approach is never worse than \( m \) times the cost that would appear if you would know the weather beforehand. Minimize the \( m \) (You do not need to prove that your \( m \) is minimal).

Exercise 4 (3 points)
The vertex cover problem asks for a given Graph \( G = (V, E) \) for the smallest subset \( C \subseteq V \) such that for every edge \( \{v, u\} \in E \) at least one vertex \( v \) or \( u \) is in \( C \), i.e. the smallest set of vertices such that every edge is covered by at least one vertex.
In the online version, you receive one vertex after another together with all edges that connect the presented vertex with previously vertices presented in previous rounds.
Show that there can not exist a constant \( m \) and an online algorithm \( A \) such that the solution of of \( A \) is always at most \( m \) times the size of an optimal (offline) solution. To do so, present a strategy for an adversary that depends on the possible choices of the algorithm.

<table>
<thead>
<tr>
<th>Exercise</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Points</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>Bonus</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Score</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>