Online Algorithms: Some notes for the solution of assignment 1

1. The proof is by arguing that the new lazy algorithm can simulate the non-lazy algorithm, without making non-lazy moves. The requests are served by the same servers as the in the non-lazy algorithm, moving them from their real position. Using the triangle inequality it is clear that the total cost of the new algorithm does not increase compared to the original one.

3. The proof of the upper bound on the (expected) cost of the online algorithm is the same as in the general case and gives $m_i H_k$ for phase i. But $m_i = 1$ for every i when $N = k + 1$. Also, the optimal cost is at least 1 for each phase. (The first and last phases are different). The proof then follows.

4. The weighted caching problem can be reduced to the k-server problem on a tree, where the tree is a star, with a leaf for each page. The length of the edge from the center to each leaf equals half the cost of brining the corresponding page into fast memory. Since there is a k-competitive k-server algorithm for trees, one has a k-competitive algorithm for weighted caching.

6. The adversary is defined in the question. An upper bound on the optimal cost can be achieved by averaging over $2N - 1$ algorithms. These maintain the following invariant: during $[i, i + 1)$ 2 algorithms occupy state s_j, for any $s_j \neq s_i$, and one algorithm occupied s_i, where s_i is the state occupied by the online algorithm during $[i, i + 1)$.