Homework 4

Try to be precise and to the point. Your answers should be short.

1. Suppose the selection rule for the interval scheduling problem is the following. Select a request that has the fewest possible requests overlapping it. Give an example where this rule does not provide an optimal solution.

   Solution: Not too hard. This is also in the previous homework.

   Grading scheme: Either 0 or 1.

2. Find a necessary and sufficient condition on n requests that guarantees that any optimal solution to the interval colouring problem would require n resources.

   Solution: The necessary and sufficient condition is the following: "there must be a point p such that all intervals pass over the point p in the real line.

   Grading scheme: Again, give either 0 or 1.

3. Explain an algorithm for the interval colouring problem that runs in $O(n \cdot \log(d))$ time. Here you just need to say what data structure you would use, describe your data structure, and explain why you achieve the bound. Your answer does not need to be long.

   Solution: It is implicitly assumed that the requests are ordered by their starting (or finishing) times. For any colour c, one needs to keep the finishing time of the last request with colour c. These finishing times are kept in a sorted array. So, when a request r_i is processed, the following should be done: (1) using a binary search (through the finishing times of the previous requests) assign r an appropriate colour; (2) insert the finishing time of r_i into the array. All these can be done in $O(\log(d))$ time. Hence, one can implement an algorithm for the interval colouring problem that runs in $O(n \cdot \log(d))$ time.

   Grading scheme: They need to say that one needs to keep an array of finishing times of the requests (ordered according to their starting time). They need to say that a binary search should be used to find the colour for r_i and for putting the finishing time of r_i into the array. Give 2 for proper answer. Give 0 if things make no sense. Give 1 otherwise.

   Note that some can can say $n \cdot log(n)$ algorithm is needed to sort the requests initially, and then explain what is above.

4. Solve Exercise 2 from Lecture Note 8.

   Solution: For the first rule: For r_1 the time $t_1 = 1$ and $d_1 = 11$ and for r_2 we have $t_1 = 10$ and $d_2 = 10$. Clearly, if we schedule r_2 first and then r_1 the lateness will be 0. If we schedule r_1 first and then r_2, then the lateness will be 1. So, the rule does not work.
   For the second rule: For r_1 the time $t_1 = 1$ and $d_1 = 3$ and for r_2 $t_1 = 10$ and $d_2 = 11$. Clearly, if we schedule r_2 first and then r_1 the lateness will be 8. If we schedule r_1 first and then r_2, then the lateness will be 0. So, the rule does not work.

   Grading scheme: one correct solution 1 point, and two correct solutions are 2 points. Otherwise, 0.

5. Solve Exercises 1,2, 3 from Lecture Note 9.

   Solution: Easy exercise.

   Grading scheme: For Exercise 1: give 0 or 0.5. For Exercise 2 give 0 or 0.5. For Exercise give 0 or 1.