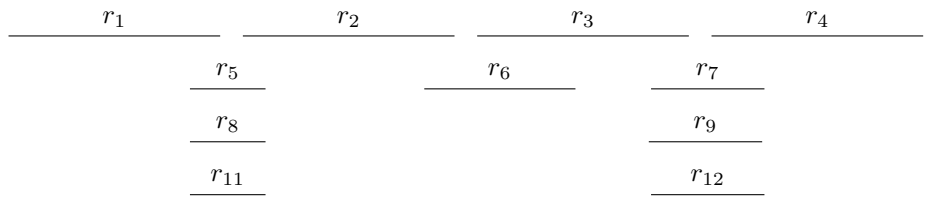
Problem 1

Question: 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:

Consider these requests at certain times:



The greedy method suggested here accepts the middle request in the second row (r_6) and thereby ensures a solution of size no greater than three.

However the optimal solution in this example is to accept the four requests in the top row.

Problem 2

Question: 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: If the requests are pairwise incompatible, then none of the requests can scheduled on the same resource, thus n requests will require n resources.

Problem 3

Question: 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 would achieve the bound. Keep your answer compact.

Solution: To ensure the runtime of our algorithm is $O(n \cdot \log(d))$ we can use a heap to store our resources.

The key of the resources is the finishing time of the last request assigned to the resource. The minimum finishing time is given the highest priority.

There are n requests, so n iterations, one for each request, and in each iteration, the highest priority resource is picked, this takes constant time.

Through the algorithm:

1. The request is compatible with the resource, it is assigned to the resource. The request now has a later/larger finishing time, so we run heapify-down on the resource, this takes $O(\log d)$ time.
2. The request is incompatible with the resource, and thus all the other resources. So, a new resource is created, and the request is assigned to the new resource. The resource is inserted into the heap, which takes $O(\log d)$ time.

Problem 3 continued on next page...

Therefore since there are n iterations, and each iteration has a running time of $\log(d)$ we have that the overall run time is $O(n \cdot \log(d))$

Problem 4

Question: Solve Exercise 2 from Lecture Note 8.

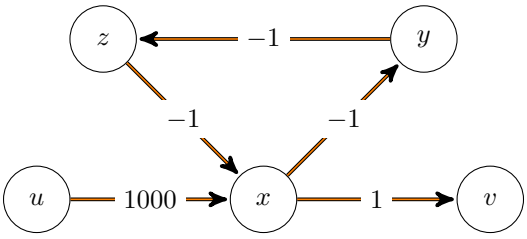
Answer:

- Schedule those requests that occupy the resources the shortest amount of time.
  - Consider the case when we have two requests (in the form $R = [t_i, d_i]$) $R_1 = [1, 15]$, $R_2 = [10, 10]$, This algorithm schedules R_1 first then R_2 and thus results in a maximum lateness of 1, while scheduling R_2 first then R_1 gives a maximum lateness of 0 (which is optimal in this case).
- Schedule those requests that have the least slack time.
  - Consider the case when we have two requests (in the form $R = [t_i, d_i, \text{slack}_i]$) $R_1 = [1, 2, 1]$, $R_2 = [10, 10, 0]$. This algorithm schedules R_2 first as it has 0 slack then R_1 which means that our maximum lateness is 9. However the optimal solution would schedule R_1 first then R_2 which has a maximum lateness of 1.

Problem 5

Question: Solve Exercises 1,2,3 from Lecture Note 9.

- Exercise 1:
  - From vertex e to x the shortest path is $e \to w \to y \to q \to z \to x$ with a weight of 13
  - From vertex a to y the shortest path is $a \to b \to f \to u \to w \to y$ with a weight of 9
- Exercise 2:
  - If there is one path from u to v then the weight of path is minimal so we are done since it is the shortest. Suppose there are k paths from u to v then we pick the one with minimal weight and that is the shortest one so we are done.
- Exercise 3:
  - If we consider this image



Then we can take the path $u \to x \to v$ which has cost 1001, or we can take the path $u \to x \to y \to z \to x \to v$ which has cost 998, or we can take the path $u \to x \to y \to z \to x \to y \to z \to x \to v$ which has cost 995 hence even though there is a path from u to v we cannot determine if the shortest path exists from u to v since we can keep going through the cycle $x \to y \to z$ to keep reducing the path cost.