CSC373 Summer '22
Tutorial 2
May 26, 2022

Q1 Interval Scheduling on m Machines (Clue for A1Q3!)

Let us recall the interval scheduling problem from class. We are given n jobs, where each job $I_j = [s_j, f_j)$ is an interval. Two jobs are *compatible* if their intervals have empty intersection. In class, we saw how to schedule a maximum number of mutually compatible jobs on one machine: consider the jobs one-by-one in an increasing order of their finish time (EFT), and greedily pick the job being considered if it is compatible with the ones picked so far.

Now, suppose that we have m machines available instead of just one. A feasible schedule can be thought of as a mapping $\sigma : \{1, \ldots, n\} \to \{0, 1, \ldots, m\}$, where job I_j is assigned to machine k if $\sigma(j) = k > 0$, and unassigned if $\sigma(j) = 0$. Jobs scheduled on any single machine must still be mutually compatible. Subject to that, we want to maximize the number of jobs scheduled, i.e., $|\{j \in \{1, \ldots, n\} : \sigma(j) > 0\}|$.

(a) Consider the following Earliest Start Time (EST) algorithm.

Algorithm 1: m-ISP-EST

```
Sort the jobs in an increasing order of their start time so that s_1 ≤ ... ≤ s_n
for j = 1, ..., n do
    if there is a machine k such that all jobs assigned to it are compatible with I_j then
        Assign job I_j to any such machine k
    else
        Let job I_j remain unscheduled
    end
end
```

Consider the following attempt to prove optimality of this algorithm.

1. Consider any job I_j that the algorithm does not schedule.
2. At that point, each machine must have a job scheduled that conflicts with I_j.
3. As seen in class, all m conflicting jobs must have start time earlier than s_j (due to the EST order) and finish time after s_j (since they conflict with I_j).
4. Hence, there are at least $m + 1$ jobs that all contain time s_j in their intervals.
5. Since there are only m machines, even the optimal algorithm must drop at least one of them.
6. The optimal algorithm drops a job for every job dropped by our EST algorithm. Hence, our EST algorithm schedules the maximum number of jobs.

Which step(s) of the argument above are flawed, and why?

[1,3) [2,4)

m=2 [4,5) [3,6)

(b) Next, consider the Earliest Finish Time (EFT) algorithm.

Algorithm 2: m-ISP-EFT

```
Sort the jobs in an increasing order of their finish time so that f_1 ≤ ... ≤ f_n
for j = 1, ..., n do
    if there is a machine k such that all jobs assigned to it are compatible with I_j then
        Assign job I_j to any such machine k
    else
        Let job I_j remain unscheduled
    end
end
```

Prove that this algorithm does not always yield an optimal solution by producing a counterexample.

Q2 Cops and Robbers

You are given an array $A[1, \ldots, n]$ with the following specifications:

- Each element $A[i]$ is either a cop ('c') or a robber ('r').
- A cop can catch any robber who is within a distance of K units from the cop, but each cop can catch at most one robber.

Write an algorithm to find the maximum number of robbers that can be caught.

For i=1, ..., n

if A[i] is a cop:

Find smallest index r in

[i-k, i+k] which