

# CS486 Assignment 1

Due February 14th 10PM ET

1. These questions are about the minimal makespan problem as designed in class:

**Problem 1** (Minimal Makespan Problem).

**Input:** A collection of machines  $M_1, \dots, M_n$  are given at the start. An online sequence of jobs  $(J_1, J_2, \dots)$  are revealed over time, where each job  $J_i$  has load  $p_i(j)$  on machine  $M_j$ .

**Mechanism:** The algorithm must assign each job  $J_i$  to a machine  $M(J_i)$  upon arrival (ie, before the next job is revealed).

**Goal:** The goal is to minimize the maximal load upon any machine:

$$\max_j \sum_{J_i \text{ assigned to } M_j} p_i(j)$$

Let  $\ell_i(M_j)$  denote the load of machine  $M_j$  directly prior to the arrival of the  $i$ th job (so  $\ell_1(M_j) = 0$  for all  $j$ ). We defined the GREEDY algorithm as the algorithm which assigns  $J_i$  to the machine  $M^*$  which minimizes  $\ell_i(M_j) + p_i(j)$ ; that is, the GREEDY algorithm assigns the  $i$ th job to the machine that would have minimal load after assigning  $J_i$  to it.

- (a) In class we went through a bunch of examples regarding the natural GREEDY algorithm. The first exercise is just to tighten up a little bit of hand-waviness we did on the case of *related* machines. Recall that in the case of related machines, each machine  $M_j$  has some speed  $s_j$ , each job  $J_i$  has an associated load  $p_i$ , and  $p_i(j) = \frac{p_i}{s_j}$ .

In class, we constructed the ‘hard’ example for the GREEDY algorithm on related machines by first choosing the underlying machines to have speeds  $1, \frac{1}{2}, \frac{1}{4}, \dots, \frac{1}{2^k}$  for some constant  $k$ . The number of machines of speed  $\frac{1}{2^i}$  was given by the formula

$$n_i = \begin{cases} 1 & i = 0 \\ \sum_{j=0}^{i-1} 2^{i-j} n_j & i > 0. \end{cases}$$

Let’s call the  $n_i$  machines of speed  $\frac{1}{2^i}$  the  $\mathcal{M}_i$  machines.

To make the sequence of jobs hard, we had the jobs come in waves, where first  $n_k$  jobs arrived of load  $\frac{1}{2^k}$ , then  $n_{k-1}$  jobs of load  $\frac{1}{2^{k-1}}$ , and so on. A key realization of the ploy of the constructed instance was that each wave of  $n_i$  jobs added a load of 1 to all of the machines in  $\mathcal{M}_1, \dots, \mathcal{M}_{i-1}$ .

We did a little bit of hand-waving in this claim. This exercise is to remove the handwaving for just the first wave of jobs (the first  $n_k$  jobs). The main hint is to look back at the

formula for the choice of  $n_i$ . The claim gets a little bit harder for the later batches of jobs (because after each batch we're increasing the load on a bunch of machines, so one would have to take that into account), so we'll just leave it at the first batch.

- (b) Another common model for the underlying machines is called “restricted”, where each job  $J_i$  has some fixed load  $p_i$ , but can only be run on certain machines. In this model, we'd say

$$p_i(j) = \begin{cases} p_i & \text{if } J_i \text{ can be run on } M_j \\ \infty & \text{otherwise.} \end{cases}$$

It turns out that GREEDY is  $\Theta(\log(n))$  competitive for this problem, but it's somewhat difficult to show the upperbound. Instead, we'll show that *every* algorithm is  $\Omega(\log n)$  competitive, which (in conjunction with the claim that GREEDY is  $\Theta(\log n)$  competitive) shows that GREEDY is as good a bound as we can get for this particular case. **Hints:** Let the number of machines  $n$  be a power of 2 (so  $n = 2^k$  for some  $k$ ) and make  $p_i = 1$  for all  $i$ , for simplicity. Suppose a first batch of  $\frac{n}{2}$  jobs arrived that were possible to run on every machine. Now send in another  $\frac{n}{4}$  jobs that could only run on half the machines - which half of the machines would you want them to be runnable on to make the algorithm sad? Keep repeating this logic...

2. Tell me what you did in the last week to start thinking about what problem you'll want to work on, and with whom.