Kasper Dalgaard Larsen
20053122
Approximation Algorithms
Handin 3

1 Greedy Load Balancing Algorithm

In this section we study the greedy load balancing algorithm.

Part 1. Show that the greedy algorithm gives a 2-approximation, and give a tight example.

First we recall the simple lower bound on OPT from the lecture:

$$\text{OPT} \geq \max\{p_{\max}, \sum_i p_i/m\}$$

where $p_{\max} = \max_i p_i$. Next we show that the solution computed by the greedy algorithm has cost at most

$$2 \cdot \max\{p_{\max}, \sum_i p_i/m\}$$

Assume first that $p_{\max} > \sum_i p_i/m \Rightarrow m > \sum_i p_i/p_{\max}$ and assume for the sake of contradiction that the greedy algorithm returns a solution of cost more than $2 \cdot p_{\max}$. Let j be the index of any machine having a finishing time of at least $2 \cdot p_{\max}$. Notice that if we remove the last job from machine j, then it still has finishing time at least $2p_{\max} - p_{\max} = p_{\max}$. Now since the greedy algorithm always assigns jobs to the machine with least current load, this implies that all machines have load at least $p_{\max}$. Thus $\sum_i p_i \geq mp_{\max} \Rightarrow m \leq \sum_i p_i/p_{\max}$, a contradiction. Secondly assume that $\sum_i p_i/m \geq p_{\max}$ and that the greedy algorithm returns a solution of cost more than $2 \cdot \sum_i p_i/m$. Again let j be the index of a machine with finishing time more than $2 \cdot \sum_i p_i/m$ and remove the last job. By the same arguments as before, we know that all machines have a finishing time of more than $2 \cdot \sum_i p_i/m - p_{\max}$. Summing the processing times on all machines we get that $\sum_i p_i > m \cdot (2 \cdot \sum_i p_i/m - p_{\max}) = 2\sum_i p_i - mp_{\max} \Rightarrow \sum_i p_i/m < p_{\max}$, a contradiction.

The following example (almost) shows the tightness: We have n jobs and $m = (n-1)/a$ machines where a is a parameter to be fixed later. We have am jobs of cost 1 and 1 job of cost a. The arbitrary order chosen by the greedy algorithm is to assign all the length 1 jobs first. This distributes all the length 1 jobs evenly amongst the m machines, and finally the length a job is assigned arbitrarely to one of the machines, giving a total cost of $2a$. In the optimal solution, the length a job is assigned to a machine first, and then the remaining am jobs are assigned greedily. This gives a cost of $a + \lceil a/m \rceil$, thus the approximation ratio is

$$\frac{2a}{a + \lceil \frac{a}{m} \rceil}$$

Choosing $a = \sqrt{n-1}$ we have $m = a = \sqrt{n-1}$, and the approximation ratio becomes

$$\frac{2a}{a+1} = \frac{2m}{m+1} = \frac{1}{\frac{1}{2} + \frac{1}{2m}} = \frac{2 - \frac{2}{m}}{1 - \frac{1}{m^2}} \geq 2 - \frac{2}{m}$$

Part 2. Show that if jobs are ordered in decreasing length, then the approximation ratio is $\frac{3}{2}$.

Again, we use the lower bounds from above, giving

$$\text{OPT} \geq \max\{p_{\max}, \sum_i p_i/m\}$$

We will assume that $p_1 \geq p_2 \geq \cdots \geq p_n$, implying that jobs get assigned in this order by the modified greedy algorithm. Assume first that $p_{\max} > \sum_i p_i/m$ and that the modified greedy algorithm gives a solution of cost more than $\frac{3}{2}p_{\max}$. Let j be the index of a machine with maximum load and consider the last job assigned to it. This job has length p_k for some k. Removing this job, we know that after assigning the jobs of length $p_1 \ldots p_{k-1}$, all machines have load more than $\frac{3}{2}p_{\max} - p_k$. Since this is greater than $p_{\max}$, we must have at least 2 jobs on every machine, thus $m \leq \frac{k-1}{2}$. Summing all weights, we get that

$$\sum_i p_i > \sum_{i=k}^{n} p_i + m \cdot (\frac{3}{2}p_{\max} - p_k) \Rightarrow$$

$$\frac{3}{2}\sum_i p_i < \sum_{i=1}^{k-1} p_i + mp_k \Rightarrow$$

$$\frac{1}{2}\sum_{i=1}^{k-1} p_i + \frac{3}{2}\sum_{i=k}^{n} p_i < mp_k \leq \frac{(k-1)}{2}p_k \Rightarrow$$

$$\frac{k-1}{2}p_k + \frac{3}{2}\sum_{i=k}^{n} p_i < \frac{k-1}{2}p_k \Rightarrow$$

$$\frac{3}{2}\sum_{i=k}^{n} p_i < 0$$

which is a contradiction since all weights are positive. Secondly, assume $\sum_i p_i/m \geq p_{\max}$ and that the modified greedy algorithm gives a solution of cost more than $\frac{3}{2}\sum_i p_i/m$. Let j and k be as before and remove job k from machine j. The load on all machines after assigning the jobs of length $p_1, \ldots, p_{k-1}$ is then more than $\frac{3}{2}\sum_i p_i/m - p_k \geq \frac{3}{2}p_{\max} - p_k$. Thus $m \leq \frac{k-1}{2}$ and

$$\sum_{i=1}^{k-1} p_i > m \cdot (\frac{3}{2}\sum_i p_i/m - p_k) = \frac{3}{2}\sum_i p_i - mp_k$$

We can now repeat the calculations from above and get our contradiction.

2 Question 2

First check if there are any jobs of length $> T$, in which case we return **No**. Otherwise, define variables $M(x_1, \ldots, x_k)$ as in the hint. Notice that since $a_i < T$ for all i, we have $M(x_1, \ldots, x_k) = 1$ when $\sum x_i = 1$ and $M(0, 0, \ldots, 0) = 0$. We now compute the M variables in iterations, such that in the j'th iteration, we compute the answer for all combinations of x_i where $\sum x_i = j$. To fill out entry $M(x_1, \ldots, x_k)$ in the j'th iteration, we "fix" the jobs on the last machine. This is done by trying all combinations of values $(y_1, \ldots, y_k)$ such that $\sum y_i > 0$ and $y_i \leq x_i$ for all i. Intuitively, this corresponds to deciding how many jobs y_i of length a_i to place on the last machine. For each such combination, if $\sum y_i \cdot a_i < T$, we let

$$M(x_1, \ldots, x_k) := \min\{M(x_1 - y_1, \ldots, x_k - y_k) + 1, M(x_1, \ldots, x_k)\}.$$

where $M(x_1, \ldots, x_k) = \infty$ if it has not yet been assigned a value. Once we reach the n'th iteration, we can read off how many machines are needed to schedule the jobs given as input. If this is greater than the number of available machines, return **No**, and otherwise return **Yes**. To also obtain a valid schedule, one could store the jobs assigned to the last machine whenever overwriting the $M(x_1, \ldots, x_k)$ variables. Backtracking through the variables would give a valid schedule.

Analysis. First notice that there are $\binom{j+k-1}{k-1}$ ways of choosing k integers summing to j. Thus we have the following bound on the running time

$$\sum_{j=2}^{n} \binom{j+k-1}{k-1} \sum_{i=1}^{j} \binom{i+k-1}{k-1} = \sum_{j=2}^{n} \sum_{i=1}^{j} \binom{j+k-1}{k-1} \binom{i+k-1}{k-1}$$

where the inner binomial coefficient originates from choosing the y_i's and the outer from choosing the x_i's. This sum is bounded by

$$n^2 \binom{n+k-1}{k-1}^2 \leq n^2 \left(\frac{(n+k-1)e}{k-1} \right)^{2k-2} = n^2 \left(\frac{en}{k-1} + e \right)^{2k-2}$$

which for $k \geq 4$ and $n \geq e/(1 - e/3)$ is bounded by

$$n^2 n^{2k-2} = n^{2k}$$