University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
P. Bamberger

Algorithm Theory
Sample Solution Exercise Sheet 7

Exercise 1: Load Balancing *(2+6+5+5+4 Points)*

Recall the load balancing problem from the lecture: Given m machines, n jobs and for each job i a processing time t_i, we want to assign each job to a machine such that the makespan (largest total processing time of any machine) is minimized. We have seen that the *modified greedy* algorithm, in which we go through the jobs by decreasing length and assign each job to the machine that currently has the smallest load, has an approximation ratio of 3/2.
In this exercise, we want to prove that the algorithm has an even better approximation ratio.

Assume we have n jobs with lengths $t_1 \geq t_2 \geq \cdots \geq t_n$. Let T be the makespan of the greedy solution and let i be a machine with load T. Further, let $\hat{n}$ be the last job that is scheduled on machine i.

(a) Shortly argue why it is sufficient to ignore jobs $\hat{n}+1, \ldots, n$ and instead prove the desired ratio between greedy and optimal for jobs $1, \ldots, \hat{n}$.

(b) Show that if an optimal solution for jobs $1, \ldots, \hat{n}$ assigns at most two jobs to each machine, the algorithm computes an optimal solution.

Hint: Think of a "canonical" way to assign at most two jobs to each machine and show that $T \leq T_{canonical} \leq T_{opt}$.

(c) Show that therefore, either $t_{\hat{n}} \leq T_{opt}/3$ or the greedy algorithm computes an optimal solution.

(d) Conclude that the algorithm has an approximation ratio of at most 4/3.

(e) Show that the 4/3 bound is tight, i.e., there is a sequence of instances for which the ratio between Greedy and OPT converges to 4/3.
Hint: Consider $2m+1$ *jobs for* m *machines, three jobs with processing time* m *and two jobs with processing times* $m+1, m+2, \ldots, 2m-1$ *each.*

Sample Solution

(a) If we run the greedy algorithm on jobs $1, \ldots, \hat{n}$, we obtain the same makespan as if we run it on jobs $1, \ldots, n$. An optimal solution for $1, \ldots, \hat{n}$ can only be smaller than one for $1, \ldots, n$. It follows that the ratio between greedy and optimal for $1, \ldots, n$ is at least as good as for $1, \ldots, \hat{n}$.

(b) If OPT assigns at most two jobs to each machine, there are at most $2m$ jobs. For convenience, if $\hat{n} < 2m$, we add $2m - \hat{n}$ empty jobs (with processing time 0) such that we have exactly $2m$ jobs. Assume we have $t_1 \geq t_2 \geq \cdots \geq t_{2m}$. The canonical algorithm pairs job j with job $2m-j+1$ and assigns it to machine j. We show that we can transform OPT to the canonical solution without increasing the makespan.

Assume OPT differs from the canonical solution and let j be the smallest integer for which job j is not paired with job $2m-j+1$ in OPT. It follows that in OPT, job j is paired with some job $k < 2m-j+1$ and job $2m-j+1$ is paired with some job $j' > j$. We have $t_{j'} \leq t_j$ and

$t_{2m-j+1} \le t_k$ and thus $t_j + t_{2m-j+1} \le t_j + t_k$ and $t_{j'} + t_k \le t_j + t_k$. Therefore, building the pairs (t_j, t_{2m-j+1}) and $(t_{j'}, t_k)$ instead of (t_j, t_k) and $(t_{j'}, t_{2m-j+1})$ does not increase the makespan. By continuing this procedure we can transform OPT to the canonical solution without increasing the makespan. It follows $T_{canonical} \le T_{opt}$.

Next we prove $T \le T_{canonical}$. The first m jobs are assigned to the same machines in both the greedy and the canonical solution. We first observe that if at some point of the Greedy execution some machine k has only one job, then also all machines $j < k$ have only one job (because otherwise Greedy would have assigned a job to a machine though there was another machine with less load). It follows that before assigning job $2m + 1 - k$ for some $1 \le k \le m$, machine k has only one job (that is job k with length t_k). If Greedy does not assign job $2m + 1 - k$ to machine k but to some machine $j > k$, machine j must have load $\le t_k$ (before the assignment) and hence afterwards load $\le t_k + t_{2m+1-k} \le T_{canonical}$. So in each step of Greedy, the load of the machine that Greedy latest assigned a job is at most $T_{canonical}$.

(c) If OPT assigns at most two jobs to each machine, then by (b) Greedy is optimal. Otherwise, there is at least one machine that OPT assigns three jobs and as $t_{\hat{n}}$ is the minimum job length, the load on this machine (and hence T_{opt}) is at least $3t_{\hat{n}}$.

(d) Recall that T is the makespan of the greedy solution, i a machine with load T and $\hat{n}$ the last job that is scheduled on machine i. It follows that before assigning job $\hat{n}$, all machines must have a load of at least $T - t_{\hat{n}}$. Therefore $T_{opt} \ge T - t_{\hat{n}}$ (the optimal makespan is at least the average load which is at least $T - t_{\hat{n}}$). By (c) we know that either Greedy is optimal or $t_{\hat{n}} \le T_{opt}/3$, so we obtain $T_{opt} \ge T - T_{opt}/3$ and hence $\frac{4}{3}T_{opt} \ge T$.

(e) Given m machines and three jobs with processing time m and two jobs with processing times $m+1, m+2, \ldots, 2m-1$ each, Greedy assigns the first $2m$ jobs to the machines such that each machine has load $3m - 1$ and puts the last job to some arbitrary machine, so the makespan is $4m-1$. An optimal solution assigns the three jobs with length m to one machine and the remaining $2m - 2$ jobs to the other machines such that each machine has load $3m$. Thus the approximation ratio is $\frac{4m-1}{3m}$ which converges to $4/3$ for $m \to \infty$.

Exercise 2: Two Knapsacks *(2+6 Points)*

Consider the following variation of the knapsack problem: Given items $1, \ldots, n$ where each item i has a positive integer *weight* $w_i \in \mathbb{N}$ and a positive *value* $v_i > 0$ and **two** knapsacks of capacities W_1 and W_2, we want to pack the items into the knapsacks such that

- for $j \in \{1, 2\}$, the *total weight* of the items in knapsack j is at most W_j.
- The *total value* of the items that are packed in either knapsack is maximized.

(a) Prove that this problem is not equivalent to the standard knapsack problem with one knapsack of capacity W_1+W_2 by showing that the total value that can be packed into one knapsack of capacity $W_1 + W_2$ can be *arbitrarily* larger than the total value that can be packed into two knapsacks of capacities W_1 and W_2.

(b) Assume that $W_1 \ge W_2$. A simple strategy would be to first compute an optimal solution for a knapsack of capacity W_1 and afterwards, with the remaining elements, an optimal solution for a knapsack of capacity W_2. Show that this algorithm always computes at least a 2-approximation for the problem.

Sample Solution

(a) Consider an instance with $W_1 = W_2 = 1$ and one item with weight 2 and arbitrarily large value.

(b) Let $v_{opt}(W_1)$ ($v_{opt}(W_2)$, resp.) be the total value of items packed into the knapsack of capacity W_1 (W_2, resp.) by an optimal algorithm and $v_{alg}(W_1)$ the total value of items packed into the knapsack of capacity W_1 by the described algorithm.

$v_{opt}(W_1)$ is a feasible (but not necessarily optimal) solution for the problem with **one** knapsack of capacity W_1. As $v_{alg}(W_1)$ is an optimal solution for this, we have $v_{alg}(W_1) \geq v_{opt}(W_1)$. Similarly, $v_{opt}(W_2)$ is a feasible solution for the problem with one knapsack of capacity W_2 and hence we have $v_{alg}(W_1) \geq v_{opt}(W_2)$ because $W_1 \geq W_2$ (making the knapsack larger can only increase the optimal solution). So we have

$$v_{alg}(W_1) + v_{alg}(W_2) \geq v_{alg}(W_1) \geq \max\{v_{opt}(W_1), v_{opt}(W_2)\} \geq \frac{v_{opt}(W_1) + v_{opt}(W_2)}{2} .$$

Exercise 3: Vertex Cover Approximation *(4 Points)*

Show that taking all nodes is a 2-approximation algorithm for the vertex cover problem in regular graphs (graphs where all nodes have the same degree)

Sample Solution

In an r-regular graph, each vertex can cover at most r edges. As the graph has $rn/2$ edges, at least $n/2$ vertices are needed to cover all edges.