CSE 202 - Homework 4

4.1 Problem 2: Heaviest First

1. The greedy heuristic at each iteration picks the node with the maximum weight and deletes all its neighbours. If there is a node v that belongs to a independent set T ($v \in T$) but the node v is not in the set S picked by the greedy heuristic, it implies that a neighbour of v, $v^{'}$ has been picked by the greedy algorithm in a previous iteration and v has been deleted from the graph. This happens only when the weight of v is less than or equal to than of $v^{'}$. Therefore, for each node $v \in T$, either $v \in S$, or there is a node $v^{'} \in S$ so that $w(v) \leq w(v^{'})$ and $(v, v^{'})$ is an in the graph G.

2. Let O denote the set of nodes picked by the optimal solution. Let $S^{'}$ and $O^{'}$ denote the nodes that exclusively belong to the greedy solution set and the optimal solution set and P denote the set of nodes common to both the solution sets. Then,

$$P = S \cap O$$
$$S^{'} = S - P \text{ and } O^{'} = O - P$$

For a set A, let the notation $w(A)$ denote the cumulative weight of all the nodes in A. Since P and $S^{'}$ are disjoint and P and $O^{'}$ are disjoint, we have the weights of the sets O and S as:

$$w(O) = w(O^{'}) + w(P)$$
$$w(S) = w(S^{'}) + w(P)$$

For every node v in $O^{'}$, there exists a neighbour $v^{'}$ of v in $S^{'}$ such that $w(v) \leq w(v^{'})$ (follows from part 1 as optimal solution is an independent set). In the worst case scenario, all four neighbours of $v^{'}$ are in optimal solution $O^{'}$ while $v^{'}$ is in greedy solution $S^{'}$. In such a case, the individual weights of the 4 neighbors are less than or equal to the weight of $v^{'}$ and hence the cumulative weights of neighbors is less than or equal to 4 times the weight of $v^{'}$. Therefore it follows that $w(O^{'})$ is less than or equal to 4 times the weight $w(S^{'})$. The weight of the greedy solution set then can be expressed as:

$$w(S) = w(S^{'}) + w(P)$$

$$w(S) \geq \frac{1}{4} w(O^{'}) + w(P)$$

$$w(S) \geq \frac{1}{4} \left[w(O^{'}) + w(P) \right]$$

$$w(S) \geq \frac{1}{4} w(O)$$

Therefore the greedy algorithm produces a set of independent nodes whose weight is at least $(1/4)^{th}$ the weight of the set produces by the optimal algorithm.

4.2 Problem 3: Scheduling

1. Consider the scenario where the time requirement of the incoming jobs are 3,3,3,3,4,4,4. The given heuristic assigns the jobs with time requirements as (3,3,4,4) and (3,3,4) on the two machines. However the optimal assignment is (3,3,3,3) and (4,4,4). Hence, the given heuristic is not optimal.

2. Let t_1, t_2,...t_n denote the time requirements of the n jobs in decreasing order. Assume that the j^{th} job is the last job allocated on the machine with larger load through the greedy heuristic. Since the jobs are in decreasing order of time requirements, t_j is less than or equal to the average of job times till j. Also, since the job t_j is allocated to machine with small load till that point, the previous load on that machine (let us denote this by P) is less than or equal to average of time requirements on the two machines of the first $j-1$ jobs .

$$t_j \leq \frac{1}{j}\sum_{i=1}^{j} t_i$$

$$P \leq \frac{1}{2}\sum_{i=1}^{j-1} t_i$$

Let T_g and T_o denote the total time to complete all jobs, i.e., the maximum over the two machines of the total time of all jobs scheduled on the machine. Then T_g can be represented as the total time on machine where job j is allocated:

$$T_g = P + t_j$$

$$T_g \leq \frac{1}{2}\sum_{i=1}^{j-1} t_i + t_j$$

$$T_g \leq \frac{1}{2}\sum_{i=1}^{j} t_i + \frac{t_j}{2}$$

$$T_g \leq \frac{1}{2}\sum_{i=1}^{j} t_i + \frac{1}{2j}\sum_{i=1}^{j} t_i$$

$$T_g \leq \left(\frac{1}{2}\sum_{i=1}^{j} t_i\right)\left(1+\frac{1}{j}\right)$$

Also, the optimal assignment produces a load on the machine with larger time that is at least equal to the average of time requirements allocated on the two machines for all the n jobs i.e.

$$T_o \geq \frac{1}{2}\sum_{i=1}^{n} t_i$$

Therefore we have the ratio of time assignment on machine with larger load of greedy and optimal as:

$$\frac{T_g}{T_o} \leq \frac{\left(\frac{1}{2}\sum_{i=1}^{j} t_i\right)\left(1+\frac{1}{j}\right)}{\frac{1}{2}\sum_{i=1}^{n} t_i}$$

$$\frac{T_g}{T_o} \leq \frac{\left(\frac{1}{2}\sum_{i=1}^{n} t_i\right)\left(1+\frac{1}{j}\right)}{\frac{1}{2}\sum_{i=1}^{n} t_i}$$

$$\frac{T_g}{T_o} \leq 1 + \frac{1}{j}$$

For scenarios where the last job index allocated on the machine with larger load is at least 6 i.e $j \geq 6$, the ratio of greedy and optimal solution is less than or equal to $7/6$ from the above equation. For values of j from 1 to 5, we compute the upper bound of ratio as follows:

Case 1 ($j = 1$):
If the first job is the last job allocated to the machine with larger load, it follows that all the other jobs have been allocated to the other machine and time requirement of job 1 is greater than or equal to the cumulative time requirement of all other jobs. In such a case, this greedy assignment is the optimal assignment as any other allocation would involve pairing job 1 with other jobs resulting in an increase in time taken to complete all the jobs on both machines. Therefore $T_g = T_o$ and the upper bound of $7/6$ on ratio of T_g and T_o is satisfied.

Case 2 ($j = 2$):
The scenario where the second job is the last job on the machine with larger total time is not possible. This is because, if there were more than 2 jobs in total, the greedy heuristic would allocate job 3 to the machine where job 2 was placed as $t1 > t2$. And if there were only 2 jobs, the machine where the first job was allocated would be the machine with larger cumulative time.

Case 3 ($j = 3$):
If $j = 3$, it implies that the greedy allocated jobs 2 and 3 on one machine which is the one with larger cumulative time and the rest on the other. Any other allocation of jobs that does not increase the load on the machine with jobs 2 and 3 would involve swapping one of these with a later job on the other machine. This would result in pairing either job 2 or job 3 with job 1. In this case the total time to complete the jobs is greater than $t_2 + t_3$ as $t1 \geq t2$ and $t1 \geq t3$ which is counter-productive. Therefore the greedy solution is equivalent to the optimal in this case i.e. $T_g = T_o$ and the upper bound of $7/6$ on ratio of T_g and T_o is satisfied.

Case 4 ($j = 4$):
Let M_1 and M_2 denote the two machines. By greedy method, job 1 is allocated to M_1, job 2 is allocated to M_2. Since $t2 \leq t1$, job 3 is allocated to M_2. Then the following two cases can be encountered:

- $t_1 \geq t_2 + t_3$: In this case job 4 is allocated to M_2. Rest of the jobs are assigned to M_1 which only has job 1. Any other assignment that does not increase the time on M_2 must involve swapping of one or more jobs among the jobs 2,3 and 4 with a later job on M_1. Since t_1 is already at least equal to $t_2 + t_3$, any such swap that places jobs 2,3 or 4 and replacing one of this with a lower time job on M_2 would make the total time on M_1 exceed the greedy solution. Hence the allocation by the greedy solution is already optimal.
- $t1 < t2 + t3$: In this case job 4 is allocated to M_1. Rest of the jobs are assigned to M_2 which only has jobs 2 and 3 which implies that $t_1 + t_4$ is the maximum time to finish all jobs. Any

other assignment that does not increase the time on M_1 must involve swapping of job 1 with any job on M_2 or job 4 with a later job on M_2. If job 1 or 4 is swapped with a job later than the job 4 on M_2, maximum time to finish jobs is increased beyond $t1 + t4$ (greedy solution) since $t2 + t3$ is at least $t1$. If job 1 is swapped with job 2 or job 3, it would result in pairing job 1 with job 2 or job 3 which would produce a cumulative time on M_2 greater than $t1 + t4$ as $t2 \geq t3 \geq t4$. Hence the allocation by the greedy solution is already optimal.

Therefore the greedy solution is equivalent to the optimal in this case i.e. $T_g = T_o$ and the upper bound of $7/6$ on ratio of T_g and T_o is satisfied.

Case 5 ($j = 5$):
Consider the first five jobs, each of these five jobs has a time requirement of at least t_5. The allocation of the first five jobs to the two machines then produces a maximum time of completion that is at least equal to the average of these five jobs on the two machines. Which implies that the maximum time of completion for optimal scenario (T_o) is at least $5 * t_5/2$ i.e. $2.5t_5$. But since there are 5 jobs, any assignment results in allocation of at least 3 of these on one machine. We therefore have a tighter bound for maximum time of completion for optimal scenario (T_o) i.e $T_o \geq 3t_5$. We have the following equations from the previous derivation of optimal to greedy solution ratio:

$$T_o \geq \frac{1}{2}\sum_{i=1}^{n} t_i$$

$$T_g \leq \frac{1}{2}\sum_{i=1}^{j} t_i + \frac{t_j}{2}$$

We therefore obtain for $j = 5$:

$$\sum_{i=1}^{n} t_i \leq 2T_o$$

$$T_g \leq \frac{1}{2}\sum_{i=1}^{n} t_i + \frac{t_5}{2}$$

$$T_g \leq T_o + \frac{t_5}{2}$$

Since we have $t_5 \leq T_o/3$,

$$T_g \leq T_o + \frac{1}{2}\left(\frac{1}{3}T_o\right)$$

$$T_g \leq \frac{7}{6}T_o$$

Hence an approximation ratio of $7/6$ can be achieved through the greedy heuristic.
