COMS E6232 - Problem Set #2

Alex Wong (asw2181) - asw2181@columbia.edu

March 3, 2017

Problem 1

a. Consider an instance where we have two items x and y where $s_x = \epsilon$, $v_x = 2\epsilon$, $s_y = B$, $v_y = B$, and $\epsilon << 1$. The v_i/s_i ratio of item x is 2 and item y is 1. Thus, the Greedy algorithm will always pick item x regardless of how large B is and regardless how small ϵ is. So as B gets larger and/or ϵ gets smaller, the approximation ratio of the algorithm will always increase, thus showing that the approximation ratio of Greedy is not bounded by any constant. □

b. **Theorem 1b** The Modified Greedy algorithm achieves approximation ratio 2.

Proof: Let OPT be the optimal solution that has maximum value where $v(OPT) = \sum\limits_{i \epsilon OPT} v_i$ subject to $\sum\limits_{i \epsilon OPT} s_i \leq B$. We first assume that the items are ordered in a non-increasing fashion according to the ratio v_i/s_i. Let's call the first item that does not fit in the knapsack using the Greedy algorithm as item m. We know that item m's v_i/s_i ratio $\geq$ items $m+1, ..., n$'s v_i/s_i ratio. Thus, if we are able to fit some fraction of item m so that it fills up to the capacity of the knapsack, that solution will be $\geq$ OPT. We can define the fraction of item m that fits into the knapsack as α where $\alpha = (B - \sum\limits_{i=1}^{m-1} v_i)/s_m$. Thus, $OPT \leq (\sum\limits_{i=1}^{m-1} v_i) + \alpha v_m$. Since α is some fraction ≤ 1, we can also say that:

$$OPT \leq (\sum_{i=1}^{m-1} v_i) + \alpha v_m \leq (\sum_{i=1}^{m-1} v_i) + v_m$$

From the inequality above, $\sum\limits_{i=1}^{m-1} v_i$ or v_m must be at least $OPT/2$, showing that the Modified Greedy algorithm will always get a solution at least $OPT/2$, thus achieving an approximation ratio 2. □

Problem 2

a. The lower bound for OPT is $max(max_i(p_i), \frac{\sum_{i=1}^{n} p_i}{m})$.

For $m = 2$ machines and 5 jobs with processing times 3,3,2,2,2, the LPT algorithm will schedule 3,2,2 on m_1 and 3,2 on m_2, giving a makespan of 7. For OPT, we know that a lower bound is max $= max(3, 12/2) = 6$. We can achieve OPT by scheduling 3,3 on one machine and 2,2,2 on the other machine.

For $m = 3$ machines and 7 jobs with processing times 5,5,4,4,3,3,3, the LPT algorithm will schedule 5,3,3 on m_1, 5,3 on m_2, and 4,4 on m_3, giving a makespan of 11. For OPT, the lower bound is $max(5, 27/3) = 9$. We can achieve OPT by scheduling 5,4 on one machine, 5,4 on another machine, and 3,3,3 on the remaining machine.

b. If $p_n > OPT/3$, then $3p_n > OPT$, showing that no machine can process more than 2 jobs or else it would be > OPT, which would contradict OPT being the optimal solution. From this, we know that $n \leq 2m$, thus the largest m jobs will first get scheduled on each of the m machines and the rest of the $n - m$ jobs will be assigned to the machine that has the least load at the point of assignment, thus showing that the LPT schedule is optimal. $\square$

c. **Theorem 2c** LPT achieves an approximation ratio of 4/3.

Proof: Let's define j as the job that finishes last in the LPT schedule and t_j as the span of time from time 0 until the time that job j starts. We know that in the timespan of t_j, $m \cdot t_j$ amount of processing has been done. This amount of processing can't be more than the total amount of processing of all jobs, thus $m \cdot t_j \leq \sum_{i=1}^{n} p_i \implies t_j \leq \frac{\sum_{i=1}^{n} p_i}{m}$. As we saw earlier in part a, $\frac{\sum_{i=1}^{n} p_i}{m}$ is essentially a lower bound for OPT, thus we can say that $t_j \leq OPT$. Then by adding p_j to t_j, we get the makespan of the machine that has scheduled job j, and we can say that $t_j + p_j \leq OPT + p_j$. What the inequality tells us is that the processing time of job j indicates how well the LPT algorithm performs. Referring to part b, if $p_j > OPT/3$, the LPT schedule is optimal. But in the worst case, if $p_j = OPT/3$, then LPT will give an $OPT + p_j = OPT + OPT/3 = 4/3OPT$ makespan, thus showing that LPT achieves an approximation ratio of 4/3. $\square$

4. **Lemma 2.4** cost(T) + cost(blue edges) $\geq 2 \cdot$ cost(M).

Proof: We first see that $T \cup$ (blue edges of P^*) creates a connected Eulerian multigraph. We know this to be true because every vertex has an even degree. A Eulerian multigraph is essentially a multigraph that contains an Eulerian tour. Within a Eulerian tour, we know that we can create two disjoint perfect matchings of U and the sum of their costs $\leq$ cost of the Eulerian tour due to triangle inequality. We also know that both of the disjoint perfect matchings can not have a cost $<$ cost(M) or else M can't be a minimum-cost perfect matching on U. Thus,

$$2 \cdot cost(M) \leq cost(2\ disjoint\ perfect\ matchings) \leq cost(T) + cost(blue\ edges)$$

$$cost(T) + cost(blue\ edges) \geq 2 \cdot cost(M)$$

□

5. **Theorem 2.5** The variant of Christofides' algorithm achieves an approximation factor of 5/3 for the *s-t path metric TSP* problem.

Proof: Combining our conclusions from Lemma 2.3 and 2.4, we can say that:

$$cost(T) + cost(blue\ edges) + cost(red\ edges) \geq 3 \cdot cost(M) \tag{1}$$

We also know that $cost(blue\ edges) + cost(red\ edges) = P^* = OPT$ and from Lemma 2.1 that $cost(T) \leq OPT$. Thus, we can modify equation 1 to be:

$$cost(T) + OPT \geq 3 \cdot cost(M)$$

$$2 \cdot OPT \geq 3 \cdot cost(M)$$

$$cost(M) \leq 2/3\ OPT \tag{2}$$

Adding cost(T) to both sides of equation 2 and using Lemma 2.1 that $cost(T) \leq OPT$, we finally get:

$$cost(T) + cost(M) \leq cost(T) + 2/3\ OPT$$

$$cost(T) + cost(M) \leq OPT + 2/3\ OPT$$

$$cost(T) + cost(M) \leq 5/3\ OPT$$

Thus showing that the cost of the path computed by the variant of Christofides' algorithm is at most 5/3 times the cost of the optimal path. □

6.