

This and other notes are available from <http://www.cs.berkeley.edu/~yozo/cs170.fa05/>

Multiprocessor Scheduling

There are m processors and n jobs taking t_1, t_2, \dots, t_n minutes. We want to distribute the jobs to each processor so that overall running time is minimal. This is known to be NP-Complete.

- Let T^* be the length of optimal schedule. Show that $T^* \leq \frac{1}{m} \sum_j t_j$.
- Show that $T^* \geq t_j$ for all j .
- Consider the following naïve greedy algorithm:
 - 1 $T_i \leftarrow 0$ for all processors i .
 - 2 **for** $j = 1$ **to** n **do**
 - 3 Let i be the machine with smallest T_i .
 - 4 $T_i \leftarrow T_i + t_j$. (assign jobs j to machine i)
 - 5 **end**

Suppose machine k has the highest load, and suppose job j was the last job assigned to machine k .

- By the choice of k , we must have $T_k - t_j \leq T_i$ for all i . (Why?)
- Show that $T_k - t_j \leq \frac{1}{m} \sum_i t_i \leq T^*$.
- Thus it follows that $T \leq 2T^*$. (Why?)

Thus this naïve algorithm is an approximation algorithm with approximation factor $\rho = 2$.

- Now consider the new improved greedy algorithm, where it sorts the jobs in decreasing size and then runs the above naïve algorithm. So in this case we have $t_1 \geq t_2 \geq \dots \geq t_n$.
 - If $n \leq m$, the algorithm produces the optimal result (with $T = T^*$). (Why?) So we only consider the case $n \geq m + 1$.
 - Consider jobs t_1, t_2, \dots, t_{m+1} . At least one machine gets gets two of these jobs. Thus $T^* \geq 2t_{m+1}$. (Why?)
 - Again, suppose k has the highest load, and suppose job j was the last job assigned to machine k . We have $j \geq m + 1$. (Why?) Thus $t_j \leq t_{m+1} \leq \frac{1}{2}T^*$.
 - Conclude that this new algorithm computes a schedule of length T such that $T \leq \frac{3}{2}T^*$.

Traveling Salesman Problem

There are n cities, and a cost matrix c_{ij} . We want to compute the minimum-cost cycle that visits each cities exactly once. This is known to be NP-Complete.

1. Describe TRAVELING-SALESMAN as a decision problem.
2. Show that approximating general TSP to any constant factor ρ is NP-Complete. (Hint: reduce from HAMILTONIAN-CYCLE.) (This is the case if your boss insists on taking direct flights between cities!)

3. We can consider a more restricted version Δ TSP where the cost matrix satisfies the triangle inequality $c_{ij} \leq c_{ik} + c_{kj}$ for all i, j , and k . (This is the case if your boss doesn't mind stop-overs!)
4. Suppose C is an optimal tour of Δ TSP. Show that C must be smaller than the size of MST T .
5. Show how to convert a spanning tree T into a tour. What approximation ratio does this algorithm achieve?
6. Best known for Δ TSP is $\rho = \frac{3}{2}$ by Christofides (1976), which is based on the idea above. It is known that polynomial-time approximation better than $\rho = 1.01$ is not possible (unless $P = NP$).
7. Euclidean TSP is known to be ϵ -approximable.

More Questions

1. Describe MULTIPROCESSOR-SCHEDULING as a decision problem.
2. Show that MULTIPROCESSOR-SCHEDULING is NP-Complete. (Hint: reduce from KNAPSACK.)
3. Show that TRAVELING-SALESMAN is NP-Complete. (Hint: reduce from HAMILTONIAN-CYCLE).