22C:253 Lecture 4

Scribe: Sriram Penumatcha

9th Septemeber 2002

MINIMUM MAKESPAN (MMS):
INPUT: A set of n jobs with processing times $p_1, p_2, \ldots, p_n \in Q^+$ and $m \in Z^+$.
OUTPUT: An assignment of the given jobs to m machines with minimum makespan.

Consider an arbitrary assignment of the n jobs to the m machines. Let J_i be the set of jobs assigned to machine i, $1 \leq i \leq m$. The *completion time* of machine i, denoted T_i, is $T_i = \sum_{j \in J_i} p_j$. The *makespan* of the assignment is $\max_{1 \leq i \leq n} T_i$.

There are two easy lower bounds on the optimal makespan, OPT:

1. The largest processing time of any job.
$$LB1 = \max_{1 \leq j \leq n} p_j.$$

2. The average completion time of a machine.
$$LB2 = \frac{\sum_{1 \leq j \leq n} p_j}{m}.$$

It is clear that $LB1 \leq OPT$ and $LB2 \leq OPT$ and so if we let LB denote the combined lower bound, $LB = \max\{LB1, LB2\} \leq OPT$.
Here is a simple factor-2 approximation algorithm.

1. Order jobs arbitrarily.
2. Process jobs in order, assigning each job to machine with smallest completion time, currently.

Claim: Let T be the makespan of the assignment produced by the above algorithm, then $T \leq 2 \cdot OPT$.
Proof: Let machine i be the machine with maximum completion time. Let job j be the job which is assigned last to machine i. Suppose the completion time of machine i, just before job j was assigned to it is t. Then, every machine has processing time at least t, implying that $\sum_{1 \leq j \leq n} p_j \geq m \cdot t$. This implies that
$$LB \geq \frac{\sum_{1 \leq j \leq n} p_j}{m} \geq t.$$
Also, we know that $LB \geq LB1 \geq p_j$. Then we have $T = t + p_j \leq LB + LB = 2 \cdot LB \leq 2 \cdot OPT$.
□

PTAS for Minimum Makespan: We want to devise a factor-$(1+\epsilon)$ approximation algorithm for MMS for any $\epsilon > 0$. In order to do this, we first establish a connection between MMS and Bin Packing.

BIN PACKING
INPUT: $t \in Q^+$ and n objects of sizes $a_1, a_2, a_3, \ldots, a_n \in (0, t]$.
OUTPUT: Minimum number of size-t bins needed to pack the objects.

Example: Let the size of the bins be t = 1. Suppose $n = 5$ and let the sizes of objects be $a_1 = 0.7$, $a_2 = 0.3$, $a_3 = 0.4$, $a_4 = 0.5$, and $a_5 = 0.4$. We can then pack $\{a_1, a_2\}$ in one bin, $\{a_3, a_4\}$ in a second bin , and a_5 by itself, in a third bin, so the number of bins used by this packing is 3.

The Bin packing problem is well-known to be NP-complete. A connection between MMS and Bin packing is as follows:

> n jobs with processing times $p_1, p_2, \ldots, p_n$ can be assigned to m machines with makespan t iff n objects with sizes $p_1, p_2, \ldots, p_n$ can be packed in m size-t bins.

Let I denote the set $\{p_1, p_2, \ldots, p_n\}$. Let $BINS(I, t)$ denote the fewest size-t bins needed to pack objects of sizes I. The connection between MMS and Bin packing implies:

$$OPT = \min\{t | BINS(I, t) \leq m\}.$$

Also note that $OPT \in [LB, 2 \cdot LB]$ and so

$$OPT = \min\{t \in [LB, 2 \cdot LB] | BINS(I, t) \leq m\}.$$

Therefore, an algorithm for MMS that does not really work is:

1. Compute LB.
2. Do binary search in the range $[LB, 2 \cdot LB]$ to find
$$\min\{t \in [LB, 2 \cdot LB] | BINS(I, t) \leq m\}.$$

This algorithm does not work for two reasons:

1. The query $BINS(I, t) \leq m$ cannot be answered in polynomial-time because the Bin packing in NP-Complete.
2. The number of iterations of the binary search is not polynomial in the input size.

To get around this problem, we connect MMS to a restricted version of Bin packing. This restricted version of Bin packing, that can be solved in polynomial time is as follows, assumes that the n object have k distinct sizes for some fixed k. Such a problem can be solved by dynamic programming in $O(n^2 k)$ time.

Suppose $\epsilon > 0$ is fixed and let $t \in [LB, 2 \cdot LB]$. We know that all objects have size at most t. Partition the range $(0, t]$ into the following intervals:

$$\Big(0, t\epsilon\Big), \Big[t\epsilon, t\epsilon(1+\epsilon)\Big), \Big[t\epsilon(1+\epsilon), t\epsilon(1+\epsilon)^2\Big), \ldots, \Big[t\epsilon(1+\epsilon)^k, t\epsilon(1+\epsilon)^{k+1}\Big),$$

where

$$t\epsilon(1+\epsilon)^k \leq t < t\epsilon(1+\epsilon)^{k+1}.$$

Construst a new instance of the Bin packing problem as follows:

1. Throw away objects of size less than $t\epsilon$.

2. For each of the remaining objects, round down the size of the object to the left endpoint of the interval to which it belongs. Specifically, for an object of size p_j, find an i such that

$$p_j \in [t\epsilon(1+\epsilon)^i, t\epsilon(1+\epsilon)^{i+1})$$

and replace p_j by $t\epsilon(1+\epsilon)^i$ in the new instance of bin packing.

Given that there are k left endpoints in $(0, t]$, we have and instance of Bin packing with $k+1$ distinct sizes. Now k can be related to ϵ as follows. Given that k satisfies

$$t\epsilon(1+\epsilon)^k \leq t < t\epsilon(1+\epsilon)^{k+1},$$

we obtain

$$k \leq \log_{1+\epsilon}\left(\frac{1}{\epsilon}\right) < k+1,$$

implying that

$$k = \left\lfloor \log_{1+\epsilon}\left(\frac{1}{\epsilon}\right) \right\rfloor.$$

So, we can solve this new instance of Bin packing in time

$$O(n^{2\lfloor \log_{1+\epsilon}(\frac{1}{\epsilon}) \rfloor + 2}).$$

Now we will use the solution of the new instance get a solution to the original instance.