1 Dominating Random Variables

Definition 1.1 A random variable $Z$ dominates another random variable $Y$ if for all real numbers $\tau$, $Pr[Y > \tau] \leq Pr[Z > \tau]$.

Remark 1.2 Observe that the random variables might or might not be independent.

In the last lecture, we saw a random variable $Y$ that is a sum of at most $T$ independent \{0,1\}-random variables, each of which has expectation at most some value $p$. We compare $Y$ with another random variable $Z$, which is a sum of exactly $T$ independent \{0,1\}-random variables, each of which has expectation exactly $p$. We last time claimed that it is more likely for $Z$ to be larger than $Y$. We prove this formally.

Claim 1.3 The random variable $Z$ dominates the random variable $Y$.

Coupling. Observe that $Y$ and $Z$ could be independent. Hence, it is incorrect to argue that the event $Y > \tau$ implies that $Z > \tau$. We use the technique of coupling: the idea is to introduce random variables $\hat{Y}$ and $\hat{Z}$ that have the same distributions as $Y$ and $Z$ respectively; however, $\hat{Y}$ and $\hat{Z}$ are correlated so that we can argue about them. In particular, they are not independent.

Suppose $Y$ is a sum of $T' \leq T$ \{0,1\}-variables such that the $i$th one has expectation $p_i \leq p$.

We define \{0,1\}-random variables $U_i, V_i$, where $i \in [T]$ in the following way. For $0 \leq i < T'$, we pick a real number $x$ uniformly at random from [0,1] independently; set $U_i := 1$ iff $x \leq p_i$ and $V_i := 1$ iff $x \leq p$. For $i \geq T'$, set just set $U_i := 0$ with probability 1, and let $V_i := 1$ with probability $p$.

Define $\hat{Y} := \sum_i U_i$ and $\hat{Z} := \sum_i V_i$. Observer that $Y$ and $\hat{Y}$ have the same distribution, and so do $Z$ and $\hat{Z}$. Moreover, since $U_i$ and $V_i$ are coupled, we always have $U_i \leq V_i$. Hence, we also have $\hat{Y} \leq \hat{Z}$ always.

Hence, we can conclude for all real numbers $\tau$ that

$$Pr[Y > \tau] = Pr[\hat{Y} > \tau] \leq Pr[\hat{Z} > \tau] = Pr[Z > \tau].$$

2 Asymptotically Optimal Job Shop Scheduling

In the last lecture, we showed an almost optimal schedule for the job shop problem. Suppose $T := \max\{C, L\}$, where $C$ is the maximum number of jobs performed by a machine, and $L$ is the maximum number of machines required by a job. We showed that there is a schedule with
makespan $2^{O(\log^* T)}T$, which almost matches the lower bound $\Omega(T)$ for any feasible schedule. In this lecture, we show it is possible to obtain a schedule with makespan $O(T)$.

The first step is the same as before. Recall we start with an infeasible schedule $S_0$, which is obtained by pretending that there is no limit on the number of jobs a machine can handle simultaneously.

We transform schedule $S_0$ into schedule $S_1$, which have the following properties.

1. For each machine, any window of size at least $T_1 := \Theta(\log T)$ has relative congestion at most $r_1 := 1$.
2. The makespan is at most $P_1 := 3T$.

Recall last time, after each transformation, we have the invariant that the relative congestion of windows of a certain size is kept at most 1. We apply a different transformation this time, which has the following invariant.

For $i \geq 1$, we apply a transformation from $S_i$ to obtain $S_{i+1}$ such that the following holds.

1. Let $T_{i+1} := \Theta(\log^c T_i)$ for some constant $c$. For each machine, any window of size between $T_{i+1}$ and $2T_{i+1}$ has relative congestion at most $r_{i+1} := r_i \cdot (1 + \frac{1}{\log T_i})$, where $c$ is some universal constant.
2. The makespan is at most $P_{i+1} := P_i \cdot (1 + \frac{1}{T})$.

The recursion continues as long as $T_{i+1} < \frac{T_i}{6 \log T_i}$. When the recursion stops, say when $i = k$, then $T_k$ is at most some constant and $k = O(\log^* T)$. Using the recursion for $r_i$ and $P_i$, we show that when the recursion terminates, both $r_k$ and $P_k$ are bounded.

**Lemma 2.1** Suppose the recursion stops for some $i = k$. Then, $T_k = O(1)$; moreover, for the schedule $S_k$, the relative congestion for every window of size at least $T_k$ is at most $r_k = O(1) \cdot r_1$ and the makespan is at most $P_k = O(1) \cdot P_1$.

**Proof:**

Observe that if $T_i$ is larger than some constant, then $T_{i+1} = \Theta(\log^c T_i) < \frac{T_i}{6 \log T_i}$. Hence, it follows that when the recursion terminates for some $i = k$, $T_k$ is at most some constant.

The result follows if we can show that both $\prod_{i<k}(1 + \frac{1}{\log T_i})$ and $\prod_{i<k}(1 + \frac{1}{T_i})$ are bounded above by some constant, where logarithm is base 2 here. Since the first term is larger, we only need to bound that.

Define $a_i := \frac{1}{\log T_i}$, for $1 \leq i < k$. We can assume that $T_{k-1} \geq 4$, otherwise we can terminate early. It follows that $a_{k-1} \leq \frac{1}{2}$. Moreover, since $\log T_{i+1} < \log T_i < 2 \log T_i$, it follows that $a_{i+1} > \frac{a_i}{2}$, for $1 \leq i < k - 1$. Hence, it follows that $\sum_{i<k} a_i \leq 1$.

Finally, $\prod_{i<k}(1 + \frac{1}{\log T_i}) = \prod_{i<k}(1 + a_i) \leq \prod_{i<k} e^{a_i} \leq e$, as required. Therefore, the makespan is at most $P_k = O(1)$ and a machine works on at most $T_k \cdot r_k = O(1)$ jobs in one time step. Increasing the time span with a further factor of $T_k r_k$ gives us a feasible schedule with makespan at most $O(P_k) = O(T)$.
3 Transforming $S_i$ into $S_{i+1}$

We will use similar techniques for the transformation. Recall that in schedule $S_i$, every window of size at least $T_i$ for every machine as relative congestion at most $r_i$.

**Scheduling by Random Delay.** We convert the schedule $S_i$ into $S_{i+1}$ in the following way. We divide the whole time span into blocks of size $B := T_i^2$. We transform each block separately and concatenate the results of all the blocks to form schedule $S_{i+1}$.

We next describe how each block is transformed. For each job $J$, we pick an integer $x_j$ uniformly at random from $\{0, 1, 2, \ldots, T_i - 1\}$ independently. Delay all operations for job $J$ in the block for $x_j$ time steps. As before, we still allow machines to work on more than 1 job at the same time. As a result, the makespan of the block can increase from $B = T_i^2$ to $T_i^2 + T_i$, i.e., increases by a factor of at most $(1 + \frac{1}{T_i})$.

We next show that with positive probability, for some $T_{i+1} = \Theta(\log^3 T_i)$ (where $c$ is a constant), all windows of size between $T_{i+1}$ and $2T_{i+1}$ for each machine have relative congestion at most $r_{i+1} := r_i \cdot (1 + \frac{1}{\log T_i})$ after the transformation.

3.1 Applying Lovasz Local Lemma

Recall that we analyze the transformation of a particular block of size $B := T_i^2$.

**Lemma 3.1** There is some $T_{i+1} = \Theta(\log^3 T_i)$ such that with positive probability, all windows of size between $T_{i+1}$ and $2T_{i+1}$ for each machine have relative congestion at most $r_{i+1} := r_i \cdot (1 + \frac{1}{\log T_i})$ after the transformation.

**Proof:** For each machine $M_i$, define $A_i$ to be the event that there is some window with size at between $T_{i+1}$ and $2T_{i+1}$ for machine $M_i$ that has relative congestion larger than $r_{i+1}$. We specify the exact value of $T_{i+1}$ later. Observe that from the recursion $r_{i+1} := r_i \cdot (1 + \frac{1}{\log T_i})$, we can deduce that $r_i \leq e < 4$.

We next form a dependency graph $H = ([n], E)$ such that $\{u, v\} \in E$ iff both machines $M_u$ and $M_v$ process the same job. Observe that $A_i$ is independent of all the $A_j$’s for which $M_i$ and $M_j$ do not process any common job.

We estimate the maximum degree of $H$. Consider machine $M_i$. Observe that it can process at most $B r_i \leq 4T_i^2$ jobs. Each of those jobs can go through at most $B \leq T_i^2$ machines. Hence, the maximum degree of $H$ is $D \leq 4T_i^4$.

We next give an upper bound on $Pr[A_i]$. Consider a fixed window $W$ of size $T_{i+1} \leq \tau \leq 2T_{i+1}$ for machine $M_i$ after the transformation. Observe that since the delay is at most $T_i$, the jobs being processed in the window $W$ could possibly come from a window $W'$ of $\tau + T_i$ time steps before the transformation. By assumption, the relative congestion of $W'$ is at most $r_i$. Hence, it follows that the maximum possible number of jobs in the window $W$ is $(\tau + T_i) \cdot r_i$. For each of those possible jobs $J_j$ that is being processed by machine $M_i$, we define $X_j$ to be the indicator random variable that takes value 1 if job $J_j$ falls into the window $W$ for machine $M_i$, and 0 otherwise.

It suffices to show how to transform the schedule from $S_i$ to $S_{i+1}$ that maintains the invariant.
Observe that $X_j$’s are independent, because the random delays are picked independently. Moreover, $E[X_j] = Pr[X_j = 1] \leq \frac{\tau}{T_i}$.

Define $Y$ to be the number of jobs that fall into the window $W$ for machine $M_i$. Then, $Y$ is the sum of $X_j$’s for the jobs $J_j$ that are performed by machine $M_i$. Note that $Y$ is a sum of at most $(\tau + T_i)r_i$ independent $\{0, 1\}$-independent random variables, each of which has expectation at most $\frac{\tau}{T_i}$.

We define $Z$ to be a sum of $(\tau + T_i)r_i$ independent $\{0, 1\}$-independent random variables, each of which has expectation exactly $\frac{\tau}{T_i}$. Recall that $Z$ dominates $Y$ and $E[Z] = r_i\tau(1 + \frac{\tau}{T_i})$.

Next we are going to use Chernoff Bound to show that with high probability $Y$ cannot be too big. Let $\epsilon := \frac{1}{3\log T_i}$. Hence, it follows that $(1 + \epsilon)E[Z] \leq r_i\tau(1 + \frac{1}{\log T_i}) = r_{i+1}\tau$. We have used the fact $T_{i+1} \leq \frac{T_i}{6\log T_i}$, which implies that $(1 + \frac{1}{3\log T_i}) \cdot (1 + \frac{\tau}{T_i}) \leq (1 + \frac{1}{\log T_i}) \cdot (1 + \frac{2T_{i+1}}{T_i}) \leq (1 + \frac{1}{\log T_i})$.

Hence, $Pr[Y > r_{i+1}\tau] \leq Pr[Z > r_{i+1}\tau] \leq Pr[Z > (1 + \epsilon)E[Z]]$. By Chernoff Bound, this is at most $\exp(-\frac{\epsilon^2E[Z]}{3}) \leq \exp(-\frac{\epsilon^2T_{i+1}}{3})$. Here, we use $E[Z] = r_i\tau(1 + \frac{\tau}{T_i}) \geq T_{i+1}$.

Note that there are trivially at most $B^2(1 + \frac{1}{T_i})^2 \leq 4T_i^4$ windows. Hence, using union bound, we have $Pr[A_i] \leq 4T_i^4 \cdot \exp(-\frac{\epsilon^2T_{i+1}}{3}) =: p$.

Hence, in order to use Lovasz Local Lemma, we need $4pD \leq 1$. Therefore, it is enough to have $4T_i^4 \cdot \exp(-\frac{\epsilon^2T_{i+1}}{3}) \cdot 4T_i^4 \leq 1$. We set $T_{i+1} := \frac{3}{\epsilon^2} \cdot \ln(16T_i^8) = \Theta(\log^3 T_i)$.

By the Lovasz Local Lemma, $Pr[\cap_i A_i] > 0$. Hence, the result follows.

4 Algorithmic Version of Lovasz Local Lemma

So far we have only used the existence version of Lovasz Local Lemma: under some limited dependency assumption, with positive probability, none of the bad events happen. However, it does not tell us how to algorithmically realize such a point in the sample space.

Beck gave a randomized algorithm in the paper “An algorithmic approach to the Lovasz Local Lemma”. However, the algorithm is involved and we would not cover that in this class.