# Advanced Algorithms Course.
# Lecture Notes. Part 11

## Randomization Continued

### Chernoff Bounds

This is a very useful general tool to bound the probabilities that certain random variables deviate much from their expected values. Here we will derive one version of this bound and then apply it to a simple load balancing problem.

Let $X$ be sum of $n$ independent 0-1 valued random variables $X_i$ taking value 1 with probability $p_i$. Clearly $E[X] = \sum_i p_i$. For $\mu := E[X]$ and $\delta > 0$ we ask how likely it is that $X > (1 + \delta)\mu$, in other words, that $X$ exceeds the expected value by more than $100\delta$ percent.

Since function exp is monotone, this inequality is equivalent to $\exp(tX) > \exp(t(1+\delta)\mu)$ for any $t > 0$. Exponentiation and this free extra parameter $t$ seem to make things more complicated, but we will see very soon why they are useful.

For any random variable $Y$ and any number $\gamma > 0$ we have that $E[Y] \geq \gamma Pr(Y > \gamma)$. This is known as Markov's inequality and follows directly from the definition of $E[Y]$. For $Y := \exp(tX)$ and $\gamma = \exp(t(1+\delta)\mu)$ this yields $Pr(X > (1 + \delta)\mu) \leq \exp(-t(1+\delta)\mu)E[\exp(tX)]$.

Due to independence of the terms $X_i$ we have

$$E[\exp(tX)] = E[\exp(\sum_i tX_i)] = E[\prod_i \exp(tX_i)] = \prod_i E[\exp(tX_i)]$$

$$= \prod_i (p_i e^t + 1 - p_i) = \prod_i (1 + p_i(e^t - 1)) \leq \prod_i \exp(p_i(e^t - 1))$$

$$= \exp\left((e^t - 1)\sum_i p_i\right) \leq \exp((e^t - 1)\mu).$$

This gives us the bound $\exp(-t(1+\delta)\mu)\exp((e^t-1)\mu)$. We can arbitrarily choose $t$. With $t := \ln(1+\delta)$ our bound reads as $\left(\frac{e^\delta}{(1+\delta)^{(1+\delta)}}\right)^\mu$.

The base depending on $\delta$ looks a bit complicated, however: Using $e^\delta \approx 1+\delta$ one can see that the base is smaller than 1. For any fixed deviation $\delta$ the base is constant, and the bound decreases exponentially in $\mu$. The more independent summands $X_i$ we have in $X$, the smaller is the probability of large deviations. A direct application of the simple Markov inequality would be much weaker (therefore the detour via the exponential function).

## Load Balancing

In order to show at least one application, consider the following simple load balancing problem: $m$ jobs shall be assigned to $n$ processors, in such a way that no processor gets a high load. In contrast to the Load Balancing problem we studied earlier, no central "authority" assigns jobs to processors, but every job chooses a processor by itself. We want to install a simple rule yet obtain a well balanced allocation. (An application is distributed processing of independent tasks in networks.) To make the rule as light-weight as possible, let us choose for every job a processor randomly and independently. The jobs need not even "talk" to each other and negotiate places. How good is this policy?

We analyze only the case $m = n$. What would you guess: How many jobs end up on the same processor? To achieve clarity, consider the random variable $X_i$ defined as the number of jobs assigned to processor $i$. Clearly $E[X_i] = 1$. The quantity we are interested in is $Pr(X_i > c)$, for a given bound $c$. Since $X_i$ is a sum of independent 0-1 valued random variables (every job chooses processor $i$ or not), we can apply the Chernoff bound. With $\delta = c-1$ and $\mu = 1$ we get immediately the bound $e^{c-1}/c^c < (e/c)^c$.

But this is only the probability bound for one processor. To bound the probability that $X_i > c$ holds for *some* of the $n$ processors, we can apply the union bound and multiply the above probability with $n$. Now we ask: For which $c$ will $n(e/c)^c$ be "small"?

At least, we must choose $c$ large enough to make $c^c > n$. As an auxiliary calculation consider the equation $x^x = n$. For such $x$ we can say:
(1) $x \log x = \log n$ and
(2) $\log x + \log \log x = \log \log n$.

Here we have just taken the logarithm twice. Equation (2) easily implies

$$\log x < \log \log n < 2 \log x.$$

Division by (1) yields

$$1/x < \log \log n / \log n < 2/x.$$

In other words, $x^x = n$ holds for some $x = \Theta(\log n / \log \log n)$.

Thus, if we choose $c := ex$, our Chernoff bound for every single processor simplifies to $1/x^{ex} < 1/(x^x)^2 = 1/n^2$. This finally shows: With probability $1 - 1/n$, each processor gets $O(\log n / \log \log n)$ jobs. This answers our question: Under random assignments, the maximum load can be logarithmic, but it is unlikely to be worse.

For $m = \Theta(n \log n)$ or more jobs, the random load balancing becomes really good. Then the load is larger than twice the expected value $\Theta(\log n)$ only with probability below $1/n^2$. Calculations are similar as above.