# Lecture 26

## Load Balancing

October 19 2016

## 1 John and the job-processor problem

Suppose John has a system in which **m jobs** arrive in a stream and need to be processed immediately. John has a collection of **n identical processors** that are capable of performing the jobs. One way of which John thinks to achieve this is to give job to each processor in a round robin fashion. But here is a catch say John's system lacks in coordination so this round robin idea don't serve his purpose. John being a student of Probability and Computing class thought of a randomized way to solve his problem. John simply assign each job to one of the processors uniformly at random. John has a very simple intuition that it will balance the jobs evenly, since each processor is equally likely to get each job. He understands that at the same time, since the assignment is completely random, he does not expect everything to end up perfectly balanced. The big question in his mind is that How well does this simple randomized approach work?

## 2 John and his claims

John being a smart student of Probability and Computing class quickly did the analysis of his problem and made few claims and also gave proof for the same.

**Claim 1 :** Consider the equation $x^x = n$. Let $x = \gamma(n)$ be the solution of this equation then $\gamma(n) = \Theta(\frac{\log n}{\log \log n})$

**Proof :**

$$x^x = n \tag{1}$$

Taking logarithm on both sides of equation 1

$$x \log x = \log n \tag{2}$$

Taking logarithm on both sides of equation 2

$$\log x + \log \log x = \log \log n \tag{3}$$

Since,

$$\log x > \log \log x \tag{4}$$

Adding $\log x$ on both sides of equation 4

$$2\log x > +\log\log x \tag{5}$$

From equation 3 and 5

$$2\log x > \log\log n \tag{6}$$

From equation 2

$$\log x = \frac{\log n}{x} \tag{7}$$

From equation 6 and 7

$$\frac{2\log n}{x} > \log\log n$$

Therefore,

$$\frac{\log n}{\log\log n} > \frac{x}{2} \tag{8}$$

From equation 3

$$\frac{\log n}{\log\log n} = \frac{\log n}{\log x + \log\log x} \tag{9}$$

Since,

$$\log x < \log x + \log\log x$$

Therefore,

$$\frac{1}{\log x + \log\log x} < \frac{1}{\log x} \tag{10}$$

Multiplying $\log n$ on both the sides of equation 10,

$$\frac{\log n}{\log x + \log\log x} < \frac{\log n}{\log x} \tag{11}$$

From equation 9 and 11,

$$\frac{\log n}{\log\log n} < \frac{\log n}{\log x} \tag{12}$$

From equation 2 and 12,

$$\frac{\log n}{\log\log n} < x \tag{13}$$

From equation 8 and 13

$$\frac{x}{2} < \frac{\log n}{\log\log n} < x$$

Since,

$$x = \gamma(n)$$

Therfore,

$$\frac{\gamma(n)}{2} < \frac{\log n}{\log\log n} < \gamma(n)$$

Hence,

$$\gamma(n) = \Theta(\frac{\log n}{\log \log n})$$

**Claim 2 :** When m = n, with probability at least $1 - n^{-1}$ no processor receives more than $e\gamma(n) = \Theta(\frac{\log n}{\log \log n})$ jobs.

**Proof :** Let $X_i$ be the random variable equal to the number of jobs assigned to processor i, for i = 1, 2, . . . , n. It is easy to determine the expected value of $X_i$: We let $Y_{ij}$ be the random variable equal to 1 if job j is assigned to processor i, and 0 otherwise; then $X_i = \sum_{i=1}^{n} Y_{ij}$ and $E[Y_{ij}] = 1/n$, So $E[X_i] = \sum_{i=1}^{n} E[Y_{ij}] = 1$. But our concern is with how far Xi can deviate above its expectation: What is the probability that Xi ¿ c? To give an upper bound on this, we can directly apply

$$Pr\bigg(X \geq (1+\delta)\mu\bigg) \leq \bigg(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\bigg)^{\mu}$$

Therefore taking $\mu = 1$ and $1 + \delta = c$,

$$Pr\bigg(X \geq c\bigg) \leq \bigg(\frac{e^{c-1}}{c^c}\bigg)$$

Therfore,

$$Pr\bigg(X \geq c\bigg) \leq \bigg(\frac{e^{c}}{c^c}\bigg)$$

Therfore,

$$Pr\bigg(X \geq c\bigg) \leq \bigg(\frac{e}{c}\bigg)^{c}$$

Taking $c = e\gamma(n)$,

$$Pr\bigg(X \geq c\bigg) \leq \bigg(\frac{1}{\gamma(n)}\bigg)^{e\gamma(n)} \tag{14}$$

For $0 < a < 1$,

$$a^e < a^2$$

Therefore from equation 14,

$$Pr\bigg(X \geq c\bigg) < \bigg(\frac{1}{\gamma(n)}\bigg)^{2\gamma(n)}$$

Therefore,

$$Pr\bigg(X \geq c\bigg) < \bigg(\frac{1}{\gamma(n)^{\gamma(n)}}\bigg)^{2} \tag{15}$$

From definition $\gamma(n)^{\gamma(n)} = n$,

$$Pr\Big(X \geq c\Big) < \left(\frac{1}{n}\right)^2$$

Therefore for all n jobs to satisfy probability becomes

$$P = nPr\Big(X \geq c\Big) < n\left(\frac{1}{n}\right)^2 = \frac{1}{n}$$

Hence,

$$P < \frac{1}{n}$$

**Claim 3 :** When there are n processors and $\Omega(n \log n)$ jobs, then with high probability every processor will have a load between half and twice the average.

**Proof :** Taking m = $tn \log n$, where $t > 1$
Therfore,

$$\mu = m/n = t \log n$$

Since,

$$Pr\Big(X < c\Big) < \left(\frac{e^{c-1}}{c^c}\right)^{\mu}$$

Where c = 2,

$$Pr\Big(X > 2\mu\Big) < \left(\frac{e}{4}\right)^{t \log n} \tag{16}$$

Since

$$Pr\Big(X < (1-\delta)\mu\Big) < e^{\frac{-1}{2}\mu\delta^2}$$

Therefore,

$$Pr\Big(X < \frac{1}{2}\mu\Big) < e^{\frac{-1}{2}t \log n(\frac{1}{2})^2} = \left(\frac{1}{n}\right)^{\frac{t}{8}} \tag{17}$$

Since John's proof is complete for all functions of form $tn \log n$. In this analysis John took t because it can be even replaced with any function of n as long as $t = T(n) > 1$ is satisfied. Hence John's proof holds good for m = $\Omega(n \log n)$ as well.