CMPT 210: Probability and Computing

Lecture 24

Sharan Vaswani

April 9, 2024

Recap

Chernoff Bound: Let $T_1, T_2, \dots, T_n$ be mutually independent r.v's such that $0 \le T_i \le 1$ for all i. If $T := \sum_{i=1}^{n} T_i$, for all $c \ge 1$ and $\beta(c) := c\ln(c) - c + 1$,

$$\Pr[T \ge c\mathbb{E}[T]] \le \exp(-\beta(c)\,\mathbb{E}[T])$$

Randomized Load Balancing

Fussbook is a new social networking site oriented toward unpleasant people. Like all major web services, Fussbook has a load balancing problem: it receives lots of forum posts that computer servers have to process. If any server is assigned more work than it can complete in a given interval, then it is overloaded and system performance suffers. That would be bad because Fussbook users are not a tolerant bunch.

Randomized Load Balancing

Fussbook is a new social networking site oriented toward unpleasant people. Like all major web services, Fussbook has a load balancing problem: it receives lots of forum posts that computer servers have to process. If any server is assigned more work than it can complete in a given interval, then it is overloaded and system performance suffers. That would be bad because Fussbook users are not a tolerant bunch.

The programmers of Fussbook just randomly assigned posts to computers, and to their surprise the system has not crashed yet.

Randomized Load Balancing

Fussbook is a new social networking site oriented toward unpleasant people. Like all major web services, Fussbook has a load balancing problem: it receives lots of forum posts that computer servers have to process. If any server is assigned more work than it can complete in a given interval, then it is overloaded and system performance suffers. That would be bad because Fussbook users are not a tolerant bunch.

The programmers of Fussbook just randomly assigned posts to computers, and to their surprise the system has not crashed yet.

Fussbook receives 24000 forum posts in every 10-minute interval. Each post is assigned to one of several servers for processing, and each server works sequentially through its assigned tasks. It takes a server an average of $1/4$ second to process a post. No post takes more than 1 second.

Randomized Load Balancing

Fussbook is a new social networking site oriented toward unpleasant people. Like all major web services, Fussbook has a load balancing problem: it receives lots of forum posts that computer servers have to process. If any server is assigned more work than it can complete in a given interval, then it is overloaded and system performance suffers. That would be bad because Fussbook users are not a tolerant bunch.

The programmers of Fussbook just randomly assigned posts to computers, and to their surprise the system has not crashed yet.

Fussbook receives 24000 forum posts in every 10-minute interval. Each post is assigned to one of several servers for processing, and each server works sequentially through its assigned tasks. It takes a server an average of $1/4$ second to process a post. No post takes more than 1 second.

This implies that a server could be overloaded when it is assigned more than 600 units of work in a 10-minute interval. On average, for $24000 \times \frac{1}{4} = 6000$ units of work in a 10-minute interval, Fussbook requires at least 10 servers to ensure that no server is overloaded (with perfect load-balancing).

Randomized Load Balancing

Q: There might be random fluctuations in the load or the load-balancing is not perfect. How many servers does Fussbook need to ensure that their servers are not overloaded with high-probability?

Randomized Load Balancing

Q: There might be random fluctuations in the load or the load-balancing is not perfect. How many servers does Fussbook need to ensure that their servers are not overloaded with high-probability?

Let m be the number of servers that Fussbook needs to use. Recall that a server may be overloaded if the load it is assigned exceeds 600 units. Let us first look at server 1 and define T to be the r.v. corresponding to the number of units of work assigned to the first server.

Randomized Load Balancing

Q: There might be random fluctuations in the load or the load-balancing is not perfect. How many servers does Fussbook need to ensure that their servers are not overloaded with high-probability?

Let m be the number of servers that Fussbook needs to use. Recall that a server may be overloaded if the load it is assigned exceeds 600 units. Let us first look at server 1 and define T to be the r.v. corresponding to the number of units of work assigned to the first server.

Let T_i be the number of seconds server 1 spends on processing post i. $T_i = 0$ if the task is assigned to a different (not the first server). The maximum amount of time spent on post i is 1-second. Hence, $T_i \in [0, 1]$.

Randomized Load Balancing

Q: There might be random fluctuations in the load or the load-balancing is not perfect. How many servers does Fussbook need to ensure that their servers are not overloaded with high-probability?

Let m be the number of servers that Fussbook needs to use. Recall that a server may be overloaded if the load it is assigned exceeds 600 units. Let us first look at server 1 and define T to be the r.v. corresponding to the number of units of work assigned to the first server.

Let T_i be the number of seconds server 1 spends on processing post i. $T_i = 0$ if the task is assigned to a different (not the first server). The maximum amount of time spent on post i is 1-second. Hence, $T_i \in [0, 1]$.

Since there are $n := 24000$ posts in every 10-minute interval, the load (amount of units) assigned to the first server is equal to $T = \sum_{i=1}^{n} T_i$. Server 1 may be overloaded if $T \geq 600$, and hence we want to upper-bound the probability $\Pr[T \geq 600]$.

Randomized Load Balancing

Q: There might be random fluctuations in the load or the load-balancing is not perfect. How many servers does Fussbook need to ensure that their servers are not overloaded with high-probability?

Let m be the number of servers that Fussbook needs to use. Recall that a server may be overloaded if the load it is assigned exceeds 600 units. Let us first look at server 1 and define T to be the r.v. corresponding to the number of units of work assigned to the first server.

Let T_i be the number of seconds server 1 spends on processing post i. $T_i = 0$ if the task is assigned to a different (not the first server). The maximum amount of time spent on post i is 1-second. Hence, $T_i \in [0, 1]$.

Since there are $n := 24000$ posts in every 10-minute interval, the load (amount of units) assigned to the first server is equal to $T = \sum_{i=1}^{n} T_i$. Server 1 may be overloaded if $T \geq 600$, and hence we want to upper-bound the probability $\Pr[T \geq 600]$.

Since the assignment of a post to a server is independent of the time required to process the post, the T_i r.v's are mutually independent. Hence, we can use the Chernoff bound.

Randomized Load Balancing

We first need to estimate $\mathbb{E}[T]$.

Randomized Load Balancing

We first need to estimate $\mathbb{E}[T]$.

$$\mathbb{E}[T] = \mathbb{E}[\sum_{i=1}^{n} T_i] = \sum_{i=1}^{n} \mathbb{E}[T_i] \qquad \text{(Linearity of expectation)}$$

Randomized Load Balancing

We first need to estimate $\mathbb{E}[T]$.

$$\mathbb{E}[T] = \mathbb{E}[\sum_{i=1}^{n} T_i] = \sum_{i=1}^{n} \mathbb{E}[T_i] \qquad \text{(Linearity of expectation)}$$

$$\begin{aligned}\mathbb{E}[T_i] &= \mathbb{E}[T_i | \text{server 1 is assigned post } i] \Pr[\text{server 1 is assigned post } i] \\ &+ \mathbb{E}[T_i | \text{server 1 is not assigned post } i] \Pr[\text{server 1 is not assigned post } i]\end{aligned}$$

Randomized Load Balancing

We first need to estimate $\mathbb{E}[T]$.

$$\mathbb{E}[T] = \mathbb{E}[\sum_{i=1}^{n} T_i] = \sum_{i=1}^{n} \mathbb{E}[T_i] \qquad \text{(Linearity of expectation)}$$

$$\begin{aligned}
\mathbb{E}[T_i] &= \mathbb{E}[T_i | \text{server 1 is assigned post } i] \Pr[\text{server 1 is assigned post } i] \\
&\quad + \mathbb{E}[T_i | \text{server 1 is not assigned post } i] \Pr[\text{server 1 is not assigned post } i] \\
&= \frac{1}{4}\frac{1}{m} + (0)(1 - 1/m) = \frac{1}{4m}.
\end{aligned}$$

$$\implies \mathbb{E}[T] = \sum_{i=1}^{n} \frac{1}{4m} = \frac{n}{4m} = \frac{6000}{m}.$$

We first need to estimate $\mathbb{E}[T]$.

$$\mathbb{E}[T] = \mathbb{E}[\sum_{i=1}^{n} T_i] = \sum_{i=1}^{n} \mathbb{E}[T_i] \qquad \text{(Linearity of expectation)}$$

$$\begin{aligned}
\mathbb{E}[T_i] &= \mathbb{E}[T_i | \text{server 1 is assigned post } i] \Pr[\text{server 1 is assigned post } i] \\
&\quad + \mathbb{E}[T_i | \text{server 1 is not assigned post } i] \Pr[\text{server 1 is not assigned post } i] \\
&= \frac{1}{4}\frac{1}{m} + (0)(1 - 1/m) = \frac{1}{4m}.
\end{aligned}$$

$$\implies \mathbb{E}[T] = \sum_{i=1}^{n} \frac{1}{4m} = \frac{n}{4m} = \frac{6000}{m}.$$

Randomized Load Balancing

Recall the Chernoff Bound: $\Pr[T \geq c\mathbb{E}[T]] \leq \exp(-\beta(c)\,\mathbb{E}[T])$.

Randomized Load Balancing

Recall the Chernoff Bound: $\Pr[T \geq c\mathbb{E}[T]] \leq \exp(-\beta(c)\,\mathbb{E}[T])$. In our case, $c\mathbb{E}[T] = 600 \implies c = \frac{m}{10}$. Hence,

Randomized Load Balancing

Recall the Chernoff Bound: $\Pr[T \geq c\mathbb{E}[T]] \leq \exp(-\beta(c)\,\mathbb{E}[T])$. In our case, $c\mathbb{E}[T] = 600 \implies c = \frac{m}{10}$. Hence,

$$\Pr[T \geq 600] \leq \exp\left(-\beta\left(\frac{m}{10}\right)\frac{6000}{m}\right)$$

Randomized Load Balancing

Recall the Chernoff Bound: $\Pr[T \geq c\mathbb{E}[T]] \leq \exp(-\beta(c)\,\mathbb{E}[T])$. In our case, $c\mathbb{E}[T] = 600 \implies c = \frac{m}{10}$. Hence,

$$\Pr[T \geq 600] \leq \exp\left(-\beta\left(\frac{m}{10}\right)\frac{6000}{m}\right)$$

Hence, $\Pr[\text{first server is overloaded}] \leq \Pr[T \geq 600] \leq \exp\left(-\beta\left(\frac{m}{10}\right)\frac{6000}{m}\right)$.

Randomized Load Balancing

Recall the Chernoff Bound: $\Pr[T \geq c\mathbb{E}[T]] \leq \exp(-\beta(c)\,\mathbb{E}[T])$. In our case, $c\mathbb{E}[T] = 600 \implies c = \frac{m}{10}$. Hence,

$$\Pr[T \geq 600] \leq \exp\left(-\beta\left(\frac{m}{10}\right)\frac{6000}{m}\right)$$

Hence, $\Pr[\text{first server is overloaded}] \leq \Pr[T \geq 600] \leq \exp\left(-\beta\left(\frac{m}{10}\right)\frac{6000}{m}\right)$.

$$\Pr[\text{some server is overloaded}]$$

$$= \Pr[\text{server 1 is overloaded} \cup \text{server 2 is overloaded} \cup \ldots \cup \text{server m is overloaded}]$$

Randomized Load Balancing

Recall the Chernoff Bound: $\Pr[T \geq c\mathbb{E}[T]] \leq \exp(-\beta(c)\,\mathbb{E}[T])$. In our case, $c\mathbb{E}[T] = 600 \implies c = \frac{m}{10}$. Hence,

$$\Pr[T \geq 600] \leq \exp\left(-\beta\left(\frac{m}{10}\right)\frac{6000}{m}\right)$$

Hence, $\Pr[\text{first server is overloaded}] \leq \Pr[T \geq 600] \leq \exp\left(-\beta\left(\frac{m}{10}\right)\frac{6000}{m}\right)$.

$$\begin{aligned}
&\Pr[\text{some server is overloaded}] \\
&= \Pr[\text{server 1 is overloaded} \cup \text{server 2 is overloaded} \cup \ldots \cup \text{server m is overloaded}] \\
&\leq \sum_{j=1}^{m} \Pr[\text{server j is overloaded}] && \text{(Union Bound)}
\end{aligned}$$

Randomized Load Balancing

Recall the Chernoff Bound: $\Pr[T \geq c\mathbb{E}[T]] \leq \exp(-\beta(c)\,\mathbb{E}[T])$. In our case, $c\mathbb{E}[T] = 600 \implies c = \frac{m}{10}$. Hence,

$$\Pr[T \geq 600] \leq \exp\left(-\beta\left(\frac{m}{10}\right)\frac{6000}{m}\right)$$

Hence, $\Pr[\text{first server is overloaded}] \leq \Pr[T \geq 600] \leq \exp\left(-\beta\left(\frac{m}{10}\right)\frac{6000}{m}\right)$.

$$\begin{aligned}
&\Pr[\text{some server is overloaded}] \\
&= \Pr[\text{server 1 is overloaded} \cup \text{server 2 is overloaded} \cup \ldots \cup \text{server m is overloaded}] \\
&\leq \sum_{j=1}^{m} \Pr[\text{server j is overloaded}] && \text{(Union Bound)} \\
&= m \Pr[\text{server 1 is overloaded}] \leq m \exp\left(-\beta\left(\frac{m}{10}\right)\frac{6000}{m}\right) && \text{(All servers are equivalent)} \\
\implies &\Pr[\text{no server is overloaded}] \geq 1 - m \exp\left(-\beta\left(\frac{m}{10}\right)\frac{6000}{m}\right).
\end{aligned}$$

Randomized Load Balancing

Recall the Chernoff Bound: $\Pr[T \geq c\mathbb{E}[T]] \leq \exp(-\beta(c)\,\mathbb{E}[T])$. In our case, $c\mathbb{E}[T] = 600 \implies c = \frac{m}{10}$. Hence,

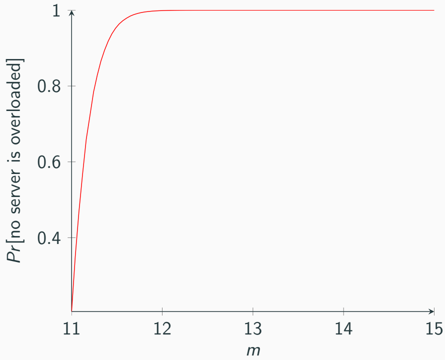
$$\Pr[T \geq 600] \leq \exp\left(-\beta\left(\frac{m}{10}\right)\frac{6000}{m}\right)$$

Hence, $\Pr[\text{first server is overloaded}] \leq \Pr[T \geq 600] \leq \exp\left(-\beta\left(\frac{m}{10}\right)\frac{6000}{m}\right)$.

$$\begin{aligned}
&\Pr[\text{some server is overloaded}] \\
&= \Pr[\text{server 1 is overloaded} \cup \text{server 2 is overloaded} \cup \ldots \cup \text{server m is overloaded}] \\
&\leq \sum_{j=1}^{m} \Pr[\text{server j is overloaded}] && \text{(Union Bound)} \\
&= m \Pr[\text{server 1 is overloaded}] \leq m \exp\left(-\beta\left(\frac{m}{10}\right)\frac{6000}{m}\right) && \text{(All servers are equivalent)} \\
\implies &\Pr[\text{no server is overloaded}] \geq 1 - m \exp\left(-\beta\left(\frac{m}{10}\right)\frac{6000}{m}\right).
\end{aligned}$$

Randomized Load Balancing

Plotting Pr[no server is overloaded] as a function of m.



Hence, as $m \geq 12$, the probability that no server gets overloaded tends to 1 and hence none of the Fussbook servers crash!

Questions?