The load balancing problem

Scribe: Muhan Li

Introduction

Load balancing is everywhere in our life, For example: Foreigners waiting outside the custom either automatically (each person is a load balancer) chooses the shortest waiting queue, or are guided by a specific number of guides (each guide is a load balancer) to the target queues selected by them.

In the computer world, transactions of banks made in parallel by millions of customers, web page request to Google from billions of devices, are usually served by a group of servers, either for speed or robustness. Load balancing is only needed for sharing the gigantic load among servers, but also needed for fault tolerance when one or more servers drop off line.

While there are numerous ways to balance the load, we still would like to get a mathematical analysis of the expectation, upper bounds, lower bounds etc. of different criteria, in this lecture, we are going to analyze the load balancing problem from a various dimensions.

Expectation of the max load

Definition

Suppose there are n balls and n bins, let $load(i)$ = number of balls in bin i, we would like to know the expectation of $\max\ load(i)$.

Claim:

$$\mathbb{E}\max_i\ load(i) \sim O(\frac{logn}{loglogn})$$

Proof

General Idea

This proof is kind of long, basically we are going to:

1. Prove $P\{load(i) \geq k\} \leq Q$.
2. Q is an infinitesimal when k is $\Theta(\frac{logn}{loglogn})$

Now by showing that when k is $\Theta(\frac{logn}{loglogn})$, probability becomes (approximately) 0, we know the expectation couldn't be larger than k, and therefore, the upper bound has the same order as k does.

$$
\begin{aligned}
load(i) &= \sum_{j=1}^{n} \mathbb{I}\{ball_j \to bin_i\} \\
&= B(n, \frac{1}{n}) \\
&\sim Poisson(1) \\
&(\text{By possion limit theorem})
\end{aligned}
$$

Therefore we can represent $P\{load(i) = k\}$ using the Poisson distribution $Poisson(1)$ as:

$$
P\{load(i) = k\} \approx \frac{e^{-1}}{k!}
$$

> Note that RHS of this equation is also the upper bound of $B(n, \frac{1}{n})$ as n approaches ∞:
>
> $$
> \begin{aligned}
> \lim_{n\to\infty} P(X = k) &= \lim_{n\to\infty} \binom{n}{k} p^k (1-p)^{n-k} \\
> &= \lim_{n\to\infty} \frac{n!}{(n-k)!k!} \left(\frac{\lambda}{n}\right)^k \left(1 - \frac{\lambda}{n}\right)^{n-k} \\
> &= \lim_{n\to\infty} \underbrace{\left[\frac{n!}{n^k\,(n-k)!}\right]}_{F} \left(\frac{\lambda^k}{k!}\right) \underbrace{\left(1 - \frac{\lambda}{n}\right)^{n}}_{\to \exp(-\lambda)} \underbrace{\left(1 - \frac{\lambda}{n}\right)^{-k}}_{\to 1} \\
> &= \lim_{n\to\infty} \underbrace{\left[\left(1 - \frac{1}{n}\right)\left(1 - \frac{2}{n}\right)\dots\left(1 - \frac{k-1}{n}\right)\right]}_{\to 1 \quad (*A)} \left(\frac{\lambda^k}{k!}\right) \underbrace{\left(1 - \frac{\lambda}{n}\right)^{n}}_{\to \exp(-\lambda)\ (*B)} \underbrace{\left(1 - \frac{\lambda}{n}\right)^{-k}}_{\to 1\ (*C)} \\
> &= \left(\frac{\lambda^k}{k!}\right) \exp(-\lambda)
> \end{aligned}
> $$
>
> Where stared parts $*A, *B, *C$ monotonically asymptote to their respective upper limit.

Suppose that (c could be anything larger than e^{-1} here):

$$
P\{load(i) = k\} \leq \frac{c}{k!}
$$

then (c' is a constant):

$$
P\{load(i) \geq k\} \leq \frac{c'}{k!}
$$

This is easy to prove, since you may easily prove that the geometric series: $\frac{c}{k^n}$ is convergent, and for a factorial series: $\frac{c}{k'!/k!}$ there exists a $k' > k$ and the following relation ship stands: $\frac{c}{k'!/k!} < \frac{c}{k^n}$, therefore the factorial series is also convergent by Direct Comparison Test.

> Proof:

$$
\begin{aligned}
&P\{load(i) \geq k\} \\
&= P\{ \bigcup_{|S|=k}^{|S|\in\{k...n\}} \{\text{events in } S\}\} \\
&\text{by union bound:} \\
&\leq (\text{number of sets of size k drawn from n elements}) * \frac{1}{n^k} \\
&= \binom{n}{k} * \frac{1}{n^k} \\
&\text{since } (\frac{n}{k})^k \leq \binom{n}{k} \leq (\frac{en}{k})^k \\
&\leq (\frac{en}{k})^k * \frac{1}{n^k} \\
&= (\frac{e}{k})^k
\end{aligned}
$$

With $P\{load(i) \geq k\} \leq (\frac{e}{k})^k$ we can get (Note: by union bound it is easy to prove $P\{\exists i,\ s.t.\ load(i) \geq k\} \leq c'(\frac{e}{k})^k * n$):

$$
P\{\exists i,\ s.t.\ load(i) \geq k\} \leq \frac{c'}{k!} * n
$$

For which k this prob << 1 (is extremely small)

$k!$ approximation: using Stirling's approximation: $\sqrt{2\pi k}(\frac{k}{e})^k \leq k! \leq e\sqrt{k}(\frac{k}{e})^k$, we have:

$$
k! \geq (k/e)^k
$$

$$
P\{\exists i,\ s.t.\ load(i) \geq k\} \leq \frac{c'}{k!} * n \leq c'(\frac{e}{k})^k * n
$$

let $k = \frac{c^* logn}{loglogn}$, right side becomes very small very quickly when $c^* >> 1$, in this case, almost ($prob \approx 1$) all values of $load(i) < k$, therefore, the upper bound of expectation of maximum load is $O(k)$:

$$
\mathbb{E} \max_i\ load(i) \sim O(k) = O(\frac{logn}{loglogn})
$$

Proof:

We would like $c'(\frac{e}{k})^k * n$ approximates 0, therefore:

$$
\begin{aligned}
&\text{needs } (\frac{e}{k})^k = e^{klog\frac{e}{k}} \leq \frac{1}{n} \\
&klog\frac{e}{k} \leq -\log n \\
&klog\frac{k}{e} \geq \log n \\
&(\text{let } k \geq \frac{c^* \log n}{\log\log n}) \\
&klog\frac{k}{e} \geq \frac{c^* \log n}{\log\log n} * \log\frac{c^{**} \log n}{\log\log n}
\end{aligned}
$$

since $\log\frac{c^{**}\log n}{\log\log n} = \log\log n + logc^{**} - \log\log\log n$, and $\log\log n$ is the prime factor, with some proper $c*$, we can get $klog\frac{k}{e} \geq \log n$.

> Side Proof for $c'(\frac{e}{k})^k * n$ approximates 0 when $k = \frac{c^* logn}{loglogn}$:

$$
\begin{aligned}
& c'(\frac{e}{k})^k * n \\
& = c' exp(k * log(\frac{e}{k})) * n \\
& = c' exp(k - klog(k)) * n \\
& = c' exp(\frac{c^* log(n)}{loglog(n)} - \frac{c^* log(n)}{loglog(n)} log(\frac{c^* log(n)}{loglog(n)})) * n \\
& = c' exp(\frac{-c^{**} log(n)}{loglog(n)} - c^* log(n) + \frac{c^* log(n) * logloglog(n)}{loglog(n)}) * n \\
& \text{where } c^{**} = c^* log(c^*) - c^*, \text{and } c^{**} > 0, \text{when } c^* > e \\
& \leq c' exp(-c^* log(n) * (1 - \frac{logloglog(n)}{loglog(n)})) * n \\
& \text{since the term after 1 is less than 1, and converges to 0 as n approaches } \infty, \\
& \text{for a given n, it will have an upper bound constant value.} \\
& \leq c' (n^{-1})^{c^{***}} * n
\end{aligned}
$$

value of $(\frac{e}{k})^k * n$ can be validated with the following python program:

```
def k(n):
const = 1
return const * math.log(n)/math.log(math.log(n))

def bound(n):
return (math.e/k(n)) ** k(n) * n

for i in range(3, 10):
print(bound(10**i))
```

Result:

```
# c* = 1
375.8857631133321
1731.9692710497748
7489.40050462691
30970.491239128318
123752.93253396139
480995.2720927469
1826802.3673114406

# c* = 10
1.0187311899688037e-37
8.007347680669828e-46
4.239976121434161e-54
1.9716844163751623e-62
8.822596094106527e-71
3.940753770345024e-79
1.7850935085535476e-87
```

Power of 2 choices

This algorithm is mentioned here by NGINX

Definition

Suppose there are $T,\ s.t. T = \frac{n}{e^3} \sim \frac{n}{20}$ requests, and n bins.

Pick bins i and j at random, assign the ball to the least loaded bin by comparing load(i) and load(j).

$$w.h.p. \quad \max_i load(i) \leq c * \log\log n$$

Proof

Define graph G with its nodes equal to bins.

Define T edges, connect node i and j when we compare their load and assign the request, create edge $i \to j$ if $load(j) < load(i)$, node j gets the request in this case.

Then the probability of creating a directed edge between i and j is $P\{i \to j\} = \frac{1}{n^2}$ (Note: n * n bins include $i \to j$ and $j \to i$).

We will refer to **in** degree of nodes as degree for short.

Lemma 1

> w.h.p. The size of the largest connected component $\leq 5 \log n$ (small connected component).

Lemma 2

> w.h.p The following holds:
>
> $\forall S \subseteq V$, (V: vertices set) the average degree in $G[S]$ is at most 5.

Why Lemma 1 + Lemma 2 is enough

If G satisfies Lemma 1 and Lemma 2, let component $C \subseteq G$, partition vertices in C in $O(\log\log n)$ layers: $L_1, L_2, L_3, \ldots$, where partitions are defined as:

$$\begin{gathered} L_1 = \{v \in C : \deg v < 10 \in G[C]\} \\ L_2 = \{v \in C \setminus L_1 : \deg v < 10 \in G[C \setminus L1]\} \\ \cdots \\ L_i = \{v \in C \setminus \{L_1 \cup \ldots \cup L_{i-1}\} : \deg v < 10 \in G[C \setminus \{L_1 \cup \ldots \cup L_{i-1}\}]\} \end{gathered}$$

(Note: by removing previous partitions, like L_2 will remove L_1, edges pointing to and going from vertices in L_1 are also removed, and degrees are induced degrees in the induced graph like $C \setminus L_1$ used in L_2 definition)

By Lemma 2 we get $|L_1| \geq \frac{|C|}{2}$, since degree of nodes in L_1 are defined to be double the number of max average degree in Lemma 2.

Now use Lemma 2 on $L_2 \ldots L_i$, we get $|L_2| \geq \frac{|C \setminus L1|}{2}, \ldots, |L_i| \geq \frac{|C \setminus \{L_1 \cup \ldots \cup L_{i-1}\}|}{2}$

Let $S = C \setminus \{L_1 \cup \ldots \cup L_i\}$, by Markov inequality there is:

$$P_{u \in S}\{\deg u \geq 10\} \leq \frac{\mathbb{E}_{u \in S}[\deg u]}{10} \leq 1/2$$

Therefore, we are removing at least half of the nodes in every step. Since the size of the largest connected component is $\log n$, after $\log\log n$ iterations, we will remove all nodes. And we can conclude that the number of layers will satisfy:

$$\#layers \leq O(loglogn)$$

Now, prove by induction that load of vertices in layer L_i is at most $10 * i$ (So last layer with index $k = \log\log n$, and its load is $10 * k = 10\log\log n$).

Base case:

> $i = 1$: number of requests for vertices in $L_1 \leq 10$.

With the definition of L_1, all nodes in L_1 have $\deg < 10$, even if all requests go to them, they have less than 10 requests.

Induction step:

> If the induced degree for all nodes in L_i is at most $10 * i$, stands $\forall i = 1...i < i + 1$, prove that in L_{i+1}, the induced degree is at most $10 * (i + 1)$.

We can view the assignment process continuously, wait until the load of nodes in L_{i+1} grows to $10 * i$, Then there are two types of requests to consider:

1. Backward requests $(k, i + 1), k < i$, since load in previous layers $L_1, \ldots, L_i$ are lower, they will always go to previous nodes, and and edge $k \leftarrow i + 1$ will be created.
2. Forward requests $(k, i + 1), k \geq i$, these types of requests correspond to edges in the induced graph $C \setminus \{L_1 \cup \ldots \cup L_i\}$, therefore there are at most 9 such requests according to the definition of partitions.

Therefore, the load of nodes in $L_{i+1} \leq 10 * i + 9 < 10 * (i + 1)$.

Now it is sufficient to say that:

$$\max_i load(i) \leq c * \log\log n\,, \forall i \in C$$

The same framework could be applied to all other similar components in G.

Common things used to prove Lemma 1 and Lemma 2

Lemma 1 is implied by:

> For a set $S \in G, |S| = \log n$, then number of edges in $S < |S| - 1$.

Because in this condition, it is not possible to have a connected component with size larger than $\log n$.

If we can prove:

$$P\{\exists S, s.t. |S| = m, G[s] \text{ has } k \text{ edges}\} \ll 1, s.t. m = \log n, k \geq \log n$$

then we can prove Lemma 1.

Suppose we choose a fixed set $s^* \in G, |s^*| = m$, then we can get the conditional version on s^* of the probability above:

(Note: actually we need to compute probability for all $k \geq \log n$, but it is quite easy to prove that for larger k, given that $T < \frac{n}{20}$ and $m = 5\ln n$, this probability is monotonically decreasing.)

$$
\begin{aligned}
&P\{G[S] \text{ has } k \text{ edges} | S = s^*\} \\
&(\text{union bound, total } T \text{ edges (requests), k land in } s^*, t_k \text{ is request index}) \\
&\leq \sum_{1 \leq t_1 \dots t_k \leq T} P\{\text{edge at } t_i \text{ land in } s^*\} \\
&(\text{each edge has two vertices land in } s^*, |s^*| = \text{m, out of total n vertices, total k edges}) \\
&\leq \sum_{1 \leq t_1 \dots t_k \leq T} (\frac{m}{n})^{2k} \\
&= \binom{T}{k} (\frac{m}{n})^{2k} \\
&\leq (\frac{eT}{k})^k * (\frac{m}{n})^{2k}
\end{aligned}
$$

Therefore we can get an upper bound of the the probability we need:

$$
\begin{aligned}
&P\{\exists S, s.t. |S| = m, G[s] \text{ has } k \text{ edges}\} \\
&= P\{\cup_{S \subseteq G, |S| = m} \{G[s] \text{ has } k \text{ edges}\}\} \\
&(\text{by union bound}) \\
&\leq \sum_{\forall S \subseteq G, |S| = m} P\{G[s] \text{ has } k \text{ edges}\} \\
&\leq \binom{n}{m} * (\frac{eT}{k})^k * (\frac{m}{n})^{2k} \\
&\leq (\frac{en}{m})^m * (\frac{eT}{k})^k * (\frac{m}{n})^{2k}
\end{aligned}
$$

Lemma 1 proof

let $m = \lceil 5 \ln n \rceil$

let $k = m - 1$

The the upper bound of the probability above:

$$
\begin{aligned}
&P\{\exists S, s.t. |S| = m, G[s] \text{ has } k \text{ edges}\} \\
&\leq (\frac{en}{m})^m * (\frac{eT}{k})^k * (\frac{m}{n})^{2k} \\
&(\text{see T value above, at problem definition}) \\
&= (\frac{en}{m})^m * (\frac{n}{e^2 * k})^k * (\frac{m}{n})^{2k} \\
&= e^{m-2k} * n^{m+k-2k} * m^{2k-m} * k^{-k} \\
&\leq e^{-k+1} * n * \underbrace{m^{k-1} * k^{-k}}_{(k+1)^{k-1}/k^k < (k+1)^k/k^k = e} \\
&\leq e^{-k} * n * e^2 \\
&(\text{because } k = m - 1 < 5 \ln n \ ) \\
&\leq e^2 / n^4 \\
&\ll 1
\end{aligned}
$$

Therefore w.h.p. the size of the largest connected component $\leq 5 \log n$.

Lemma 2 proof

Consider sets of size $\leq 5 \log n$ (because for set of size $> 5 \log n$, they would be disconnected, as described in Lemma 1).

We need to show that $P\{\exists S, G[S] \text{ has } > 2.5|S| \text{ edges}\} \ll 1$

(Since total degree = # edges * 2, so average degree is at most 5, as described in Lemma 2.)

Let $m \leq 5 \log n, k = 2.5m$, there is:

$$
\begin{aligned}
&P\{\exists S, s.t. |S| = m, G[s] \text{ has } k \text{ edges}\} \\
&\leq (\frac{en}{m})^m * (\frac{eT}{k})^k * (\frac{m}{n})^{2k} \\
&(\text{put in } T, k, m) \\
&= (\frac{m}{n})^{1.5m} * \underbrace{(\frac{e}{(e^2 * 2.5)^{2.5}})^m}_{<1} \\
&\ll 1
\end{aligned}
$$

Real cases

Does this bound holds for larger T? Eg: $T = n$?

We can partition total requests $T = n$ to e^3 groups (and reset bins after each partition).

The the max load for some bin $< e^3 * c * \log \log n$, and the hyper-log-log bound still holds.