Assignment 2: Max Load

Miguel Alcañiz Moya

November 11, 2024

1 Statement of the assignment

In this programming assignment, you will have to write a program to empirically study different allocation strategies of balls into bins.

Consider a collection of m bins and n balls. For each ball, we draw a certain number $d1$ of bins, uniformly at random from $\{1, ..., m\}$ and with replacement, and use some rule (deterministic or randomized) to choose in which of the d bins do we put the ball. Given bin B_i, we denote $X_i(n)$ the load of B_i, that is, the number of balls in B_i after n balls have been allocated. By definition, $\sum_{1 \leq i \leq m} X_i(n) = n$. In a balanced allocation, all $X_i(n)$ are roughly similar and close to n/m. Since we are interested in balanced allocations, we will look after strategies that minimize the maximum load $X^*(n) = max_{1 \leq i \leq m}\{Xi(n)\}$. This is equivalent to minimizing the gap

$$G_n = max_{1 \leq i \leq n} \left\{ X_i(n) - \frac{n}{m} \right\}.$$

Notice that $G_n \geq 0$ since there must always be some $X_i(n) \geq \frac{n}{m}$. In your empirical study, you will study the evolution of the gap G_n as n grows. We will divide the study of the strategies in 3 scenarios:

1. Light-loaded Scenario $n = m$
2. Mid-loaded Scenario $n = m^{\frac{3}{2}}$
3. Heavy-loaded Scenario $n = m^2$

We will take $m = 10, 10 * 2^1, ..., 10 * 2^7$ so we can see how it changes from 10 to 1280 bins. We will also see how it works for different strategies.

1. One-choice (assign every ball to a random bin)
2. Two-choices (assign every ball to the emptier bin of two random bins)
3. (1+β)-choice (do case 1. with probability β and 2. otherwise)

We will work option 3. with $\beta = 0.2, 0.5, 0.8$.
We will also see what will happen if we do a d-case option for different d's, such as $d = 3, 5, 10$.

We will also repeat the experiments for a $b - batched$ setting, where balls arrive in batches of b balls and the information on the load is the one available at the beginning of the batch.

2 GitHub

The project is delivered in the public GitHub repository with the following link:
https://github.com/miguelalcaniz02/Max-Load-Gap.git

3 One-batched option

In this section we will see how it evolves for the *normal* setting with all the information after every ball is allocated.

Recall that the experiments are represented with some graphics. In these, the x axis are in a 2-exponential proportion, that is if $j - i = k$ means that $x_j/x_i = 2^k$.

For every case we repeat it $k = 40$ times and set the mean all the max load gaps from the experiments.

3.1 Light-loaded Scenario n = m

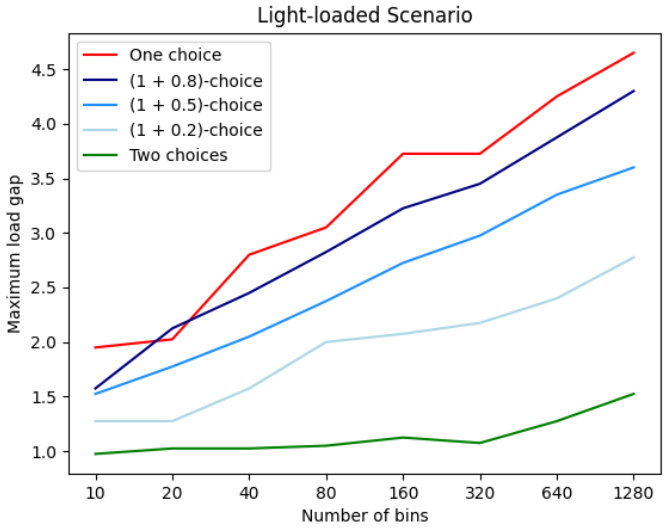

Figure 1: n = m

We can see how all the functions in the graphic grow linearly, which means that the relation between the max load gap and the number of balls grows logarithmically. We notice that the two choices option is extremely good choice, as doesn't really start growing from 1 (the least possible value) till it gets to 640 bins. Then the $(1 + \beta)$ options work as good as smaller is the β.

3.2 Mid-loaded Scenario

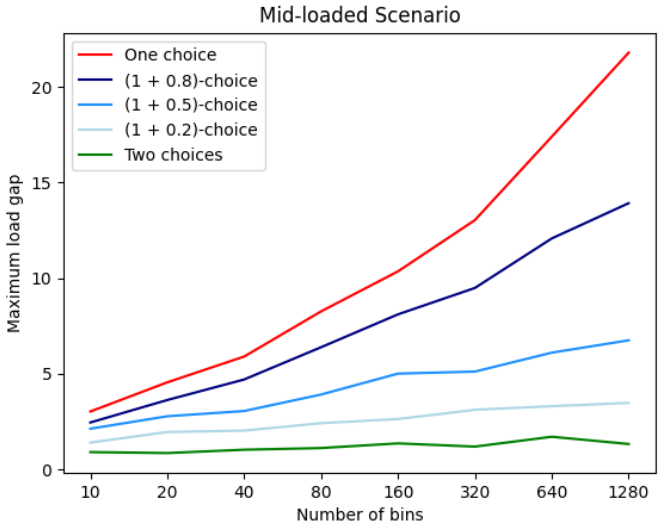

Figure 2: n = $m^{\frac{3}{2}}$

In this intermediate case we can see how the functions differ more and more as m grows. Which makes sense as we are just sampling $m^{\frac{3}{2}}$ balls. Again, the two case option seem to be really good, working really well. And the $(1+\beta)$ option is not bad for $\beta = 0.2, 0.5$.

3.3 Heavy-loaded Scenario

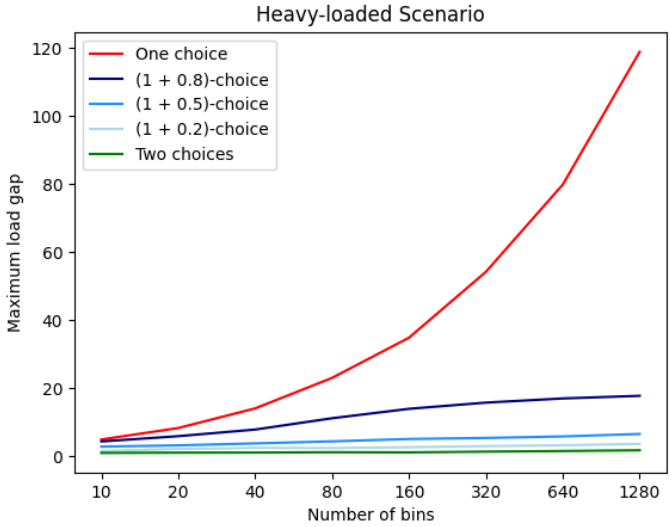

Figure 3: n = m^2

In this case we can really appreciate the help from the two choices option. Again the $(1+\beta)$ works really well for $\beta = 0.2, 0.5$. But here we can see how the max load gap really grows when m is big whereas it doesn't at all when using the two choices scenareo.

4 D-batched option

In this section we will see how it evolves for the d-bathed setting with all the information after every ball is allocated.

Recall that the experiments are represented with some graphics. In these, the x axis are in a 2-exponential proportion, that is if $j - i = k$ means that $x_j/x_i = 2^k$.

For every case we repeat it $k = 40$ times and set the mean all the max load gaps from the experiments.

4.1 Light-loaded Scenario n = m

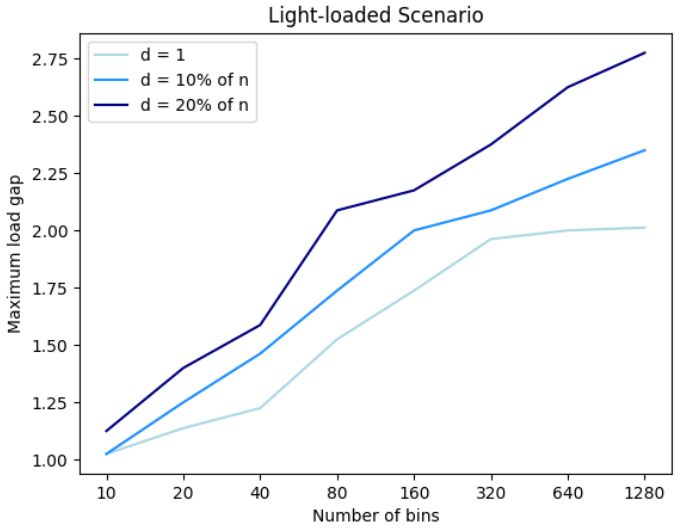

Figure 4: n = m

In this example we are seeing how it change to get the balls in batches of 1 ball, $n/10$ balls and $n/5$ balls for the 2 choices option. There is not much of a difference, specially taking $d = n/10$.

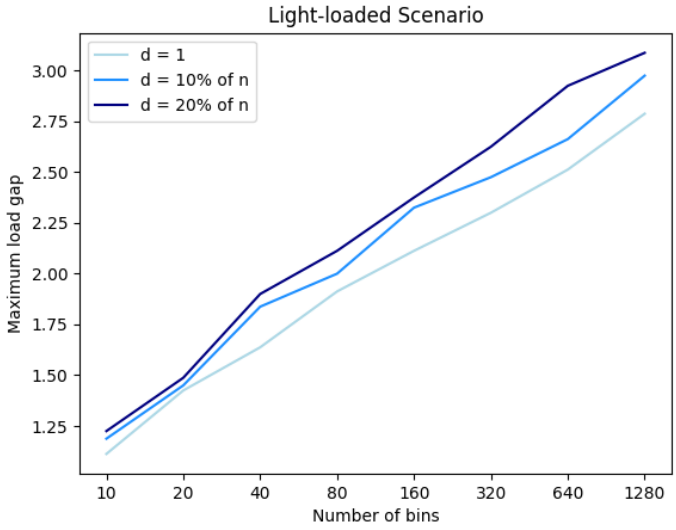

Figure 5: n = m

We can see that just adding a 0.2 probability of using the one choice instead of the two choice option we make the three lines much similar. Taking a $(1+\beta)$-choice with a higher β wouldn't make any sense because it would just be the same three lines, or at least really similar.

4.2 Mid-loaded Scenario

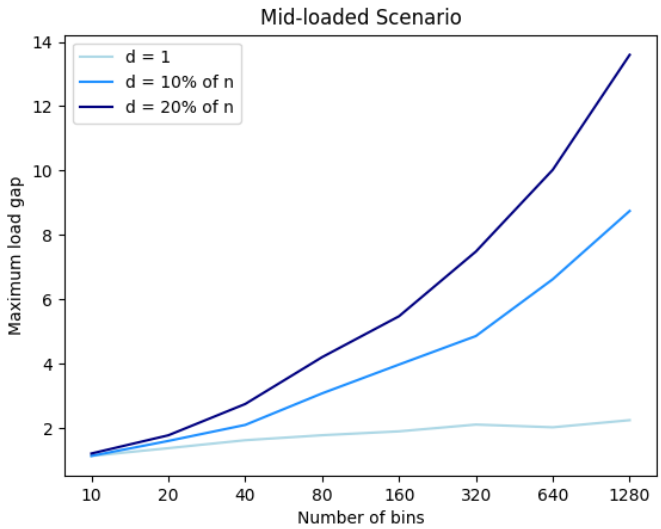

Figure 6: n = $m^{\frac{3}{2}}$

In this example we are seeing how it change to get the balls in batches of 1 ball, $n/10$ balls and $n/5$ balls for the 2 choices option. The graphic looks very similar to the graphic in 2.2 with the functions of the $(1 + 0.8)$-choice, the $(1 + 0.5)$-choice and the two choices.

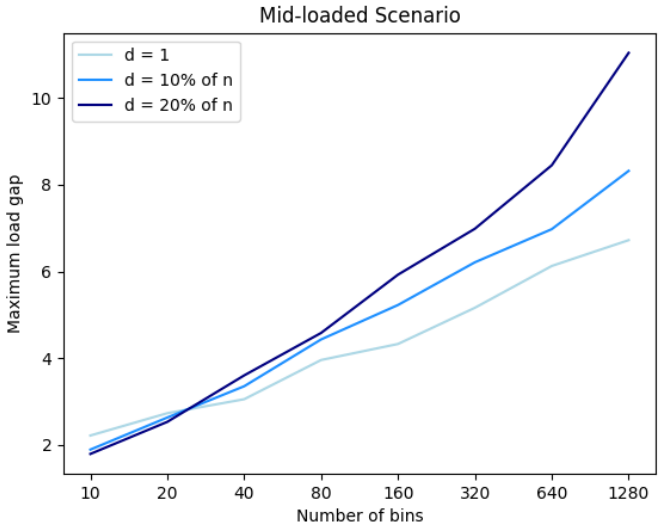

Figure 7: n = $m^{\frac{3}{2}}$

Again in the graphic above we are just watching how it changes to take different d's for the $(1+0.2)$-choice option. Again just adding a little of probability of using the one choice option makes the three calls really similar.

4.3 Heavy-loaded Scenario

Here we notice how similar the bigger d's functions are compared with d = 1, but maybe taking a d proportional to n doesn't how as much information.

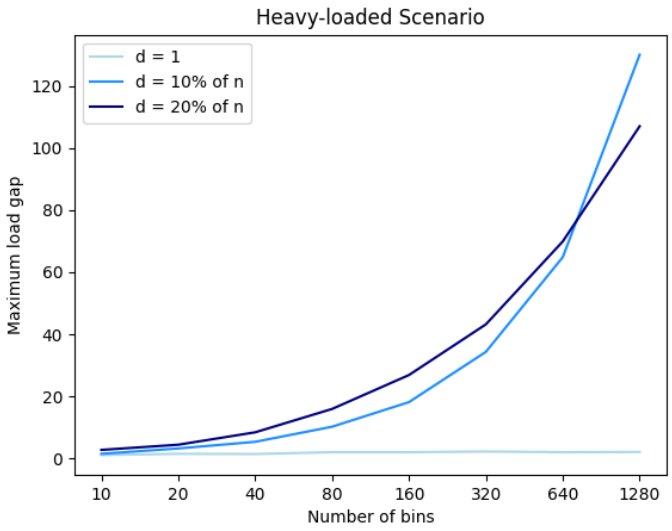

Figure 8: n = m^2

Now we taking a d proportional to the number of bins, and we notice that the maximum load gap stays really constant. This fact is quite interesting so if we make m really big and $n = m^2$ with $d = k * m$ we will expect the max load gap not to be really far from k, even for a really big m.

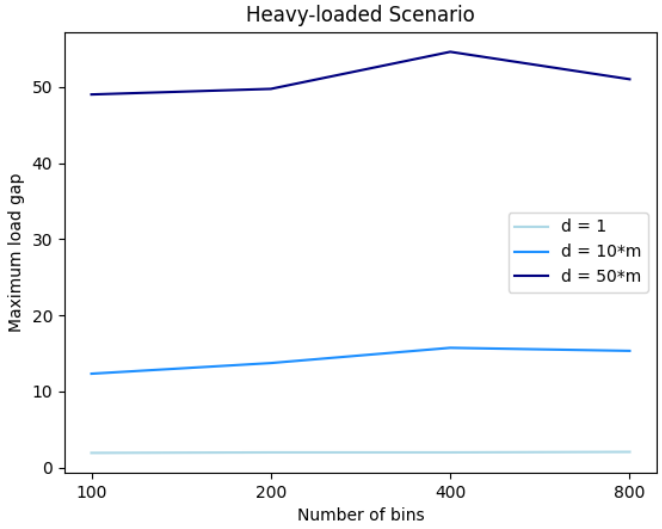

Figure 9: n = m^2