Assignment 2 -Randomized Algorithms

Dinu Eduard-Adiel

November 2024

1 Introduction

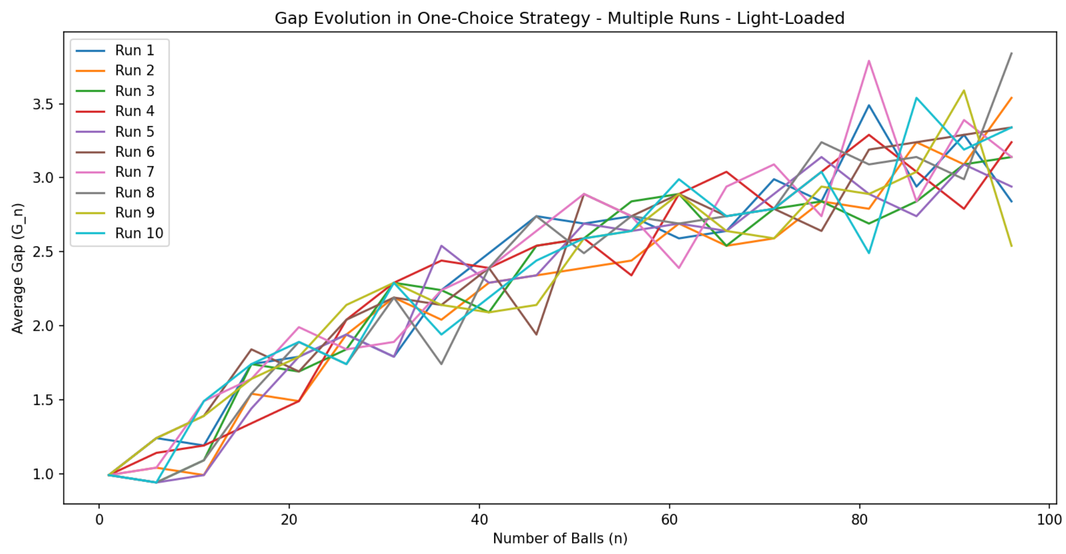
The allocation of balls into bins is a fundamental problem in computer science and mathematics, with applications ranging from load balancing in distributed systems to hash table design and randomized algorithms. In this problem, there are m bins, and n balls which must be allocated in those bins. The experiments runs as follows: for each ball we draw a number of bins $d \in \{1, 2, 3, ..., n\}$ (uniformly chosen) and then we chose out of those d bins where to put the ball (this rule can be anything, deterministic or non-deterministic). The goal of this project is to see how different strategies minimize the gap between each bin load and the average. Ideally, we will have each load equal to $\frac{n}{m}$.

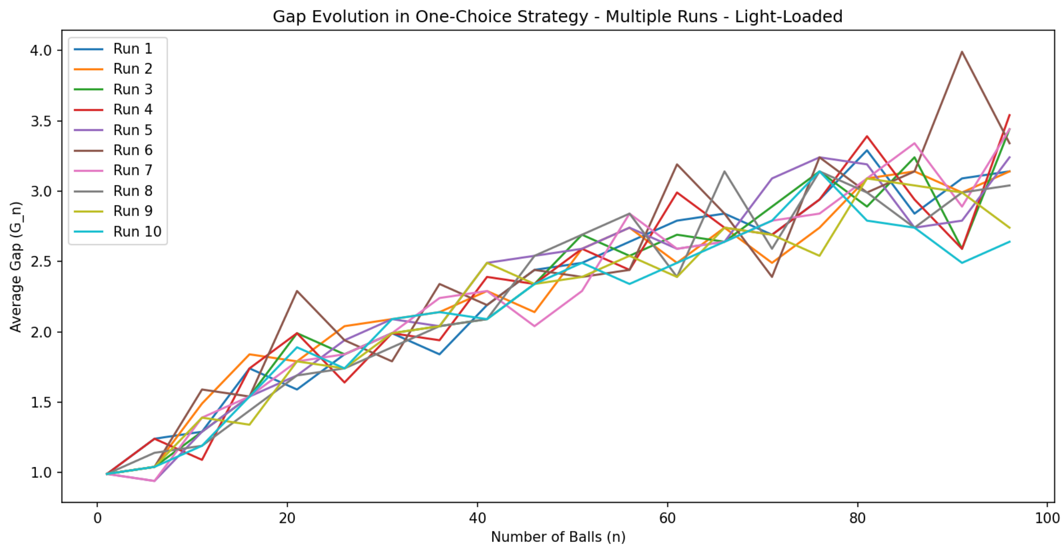
2 One Choice Strategy

In this section we will explore the one-choice strategy, that is where the first random choice is always $d = 1$, meaning that every ball has the same probability of being put inside that bin ($p(b_i \mid b_i \text{ is picked in the } d \text{ bins}) = \frac{1}{n}$).

2.1 Light-Loaded $n = m$

With $n = m$, this means that each bin has an equal opportunity to be selected. As a result, we expect that the distribution of balls across the bins will be approximately uniform, although some bins may end up with more than others due to the randomness inherent in the allocation process. The average load per bin in this case would be $\frac{n}{m} = 1$, but the maximum load can vary depending on the random choices made during the allocation process.



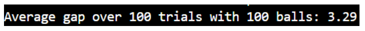


As seen from this runs, due to the randomness of the allocation, the average gap is between $2.5 - 3.9$. These experiments are characterized by: each run consists of running $T = 10$ times with the number of balls varying from $n = 1$ to $n = m = 100$ with a step increase of 5 balls.

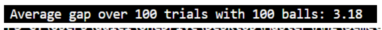
Below is a graph containing 100 trials with the average gap with $n = m$ balls. Except some outliers, the average gap is between 3 and 4 (some results shown below). If we increase the number of bins and balls to 1000 and 10000 the average increases to $4 - 5$ and $5 - 6$. The expected number of balls in each bin on average is $\frac{n=m}{m} = 1$ so the variance is $\frac{n}{m} * (1 - \frac{1}{m})$ which tends to 1 as m increases. The maximum load on average is close to

$$E[M_n] \approx 1 + \frac{\log m}{m}$$

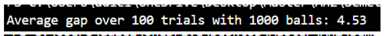
and by subtracting the average it becomes $\frac{\log m}{m}$.



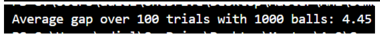
```
Average gap over 100 trials with 100 balls: 3.29
```



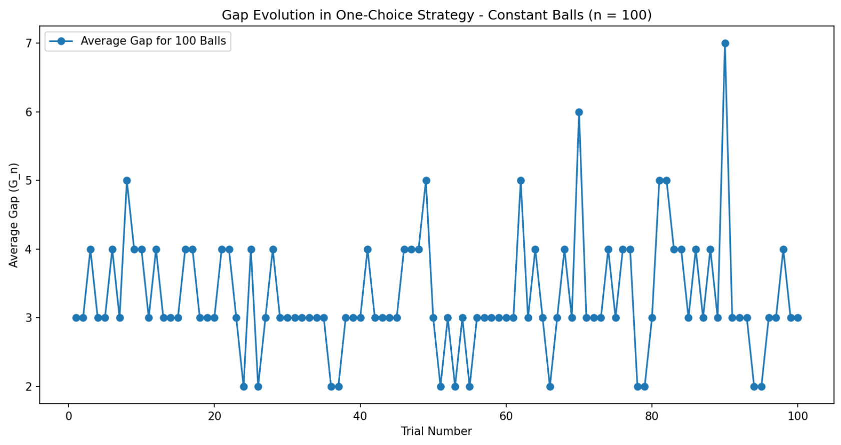
```
Average gap over 100 trials with 100 balls: 3.18
```



```
Average gap over 100 trials with 1000 balls: 4.53
```

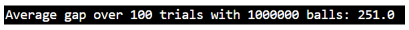


```
Average gap over 100 trials with 1000 balls: 4.45
```

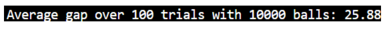


2.2 Heavy-Loaded

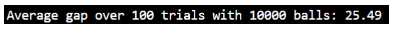
In this subsection, the strategy will have $n >> m$. In the first case, $n = m^2$ and on 100 Trials the average is around $20 - 30$ (some results below) and on the second case $n = m^3$ and the average is around $200 - 300$. The idea is that the heavier the number of balls, the higher the average gap becomes. For the $n = m^2$ case, the average gap is remarkably close to the average gap in the $n = m$ case squared. Now the average load becomes $\frac{m^2}{m} = m$. Every bin follows a binomial distribution so the variance is $m * (1 - \frac{1}{m})$.



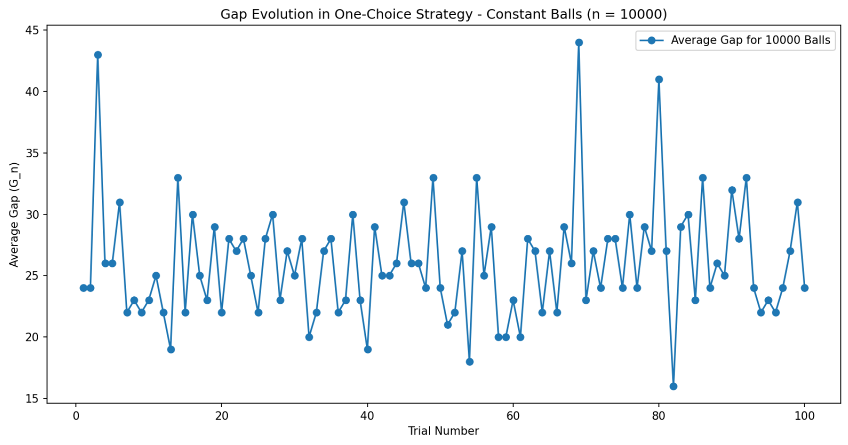
```
Average gap over 100 trials with 1000000 balls: 251.0
```

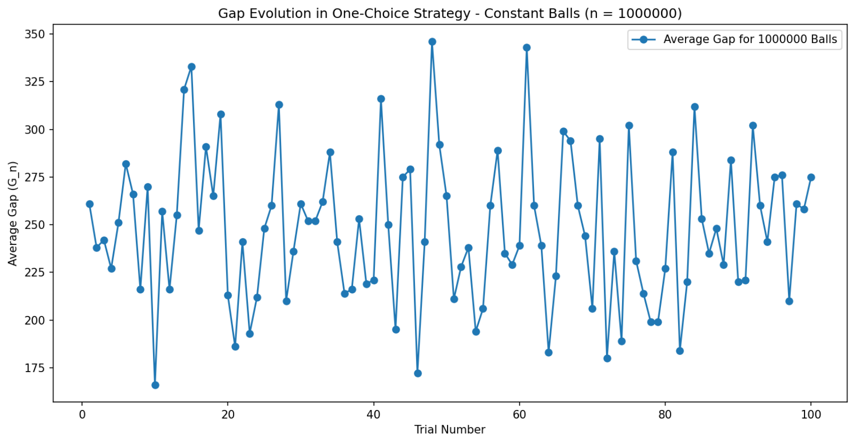


```
Average gap over 100 trials with 10000 balls: 25.88
```



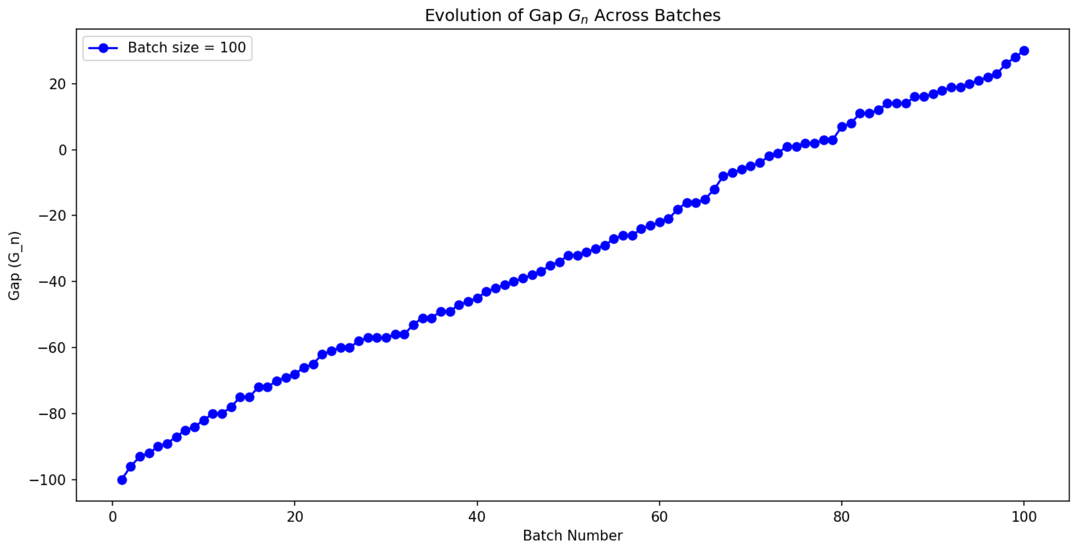
```
Average gap over 100 trials with 10000 balls: 25.49
```

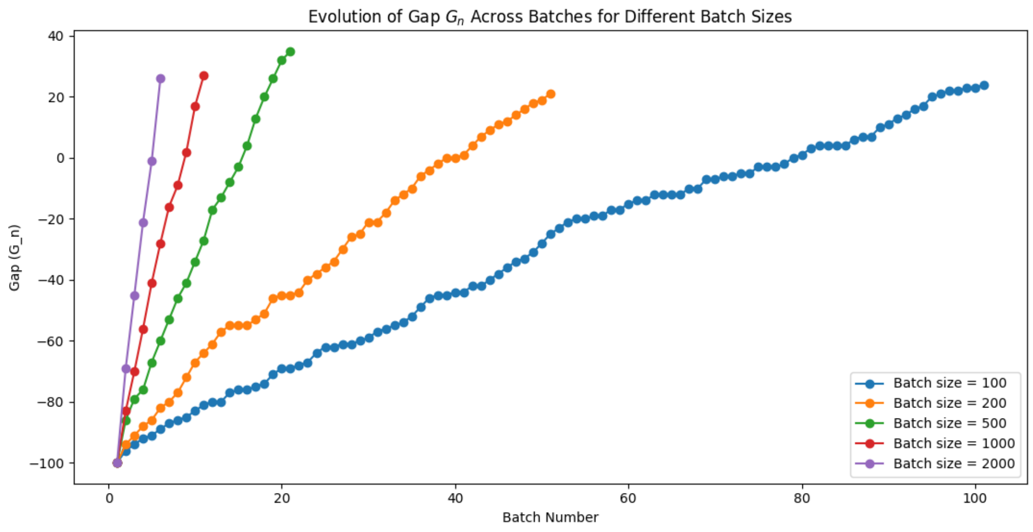




2.3 B-batched Setting

In this section, we will explore the setting where the belief of our bin loads is updated on batches. This comparison is similar to the BDI from RL (where we have a belief of how the system is going to look like, but we do not know for sure how it is actually changing). As seen in the results below there is quite a big change in results, especially for the intermediate batch sizes. The reality is that based on this strategy where we choose the bin randomly, this methods does not imply any change as even though our beliefs may change the decision will not.

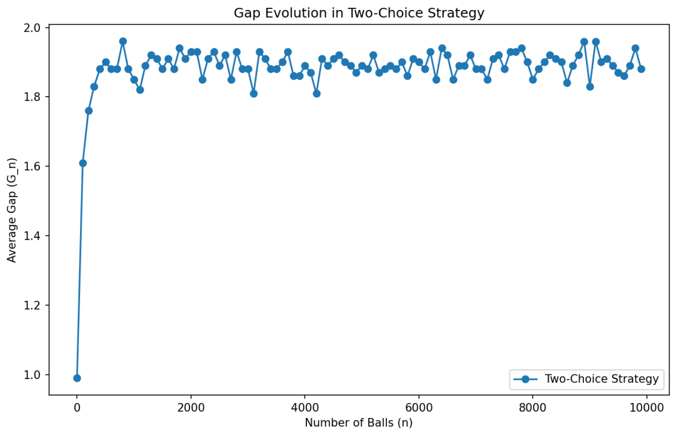




3 Two-Choice Strategy

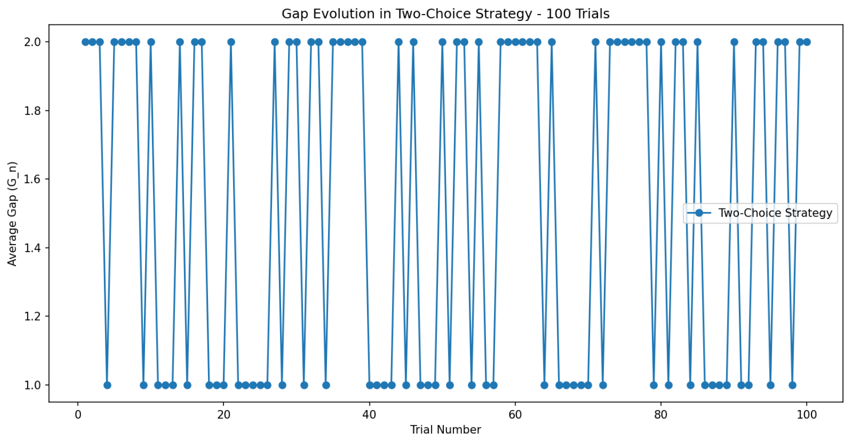
In this section, we will explore the 2-choice strategy where $d = 2$, and the bin with the smaller load is the destination out of the 2 bins. This helps reduce the average gap as n increases due to the fact that the smaller always gets filled first. The only way to fill just one bin is to always choose the same 2 bins.

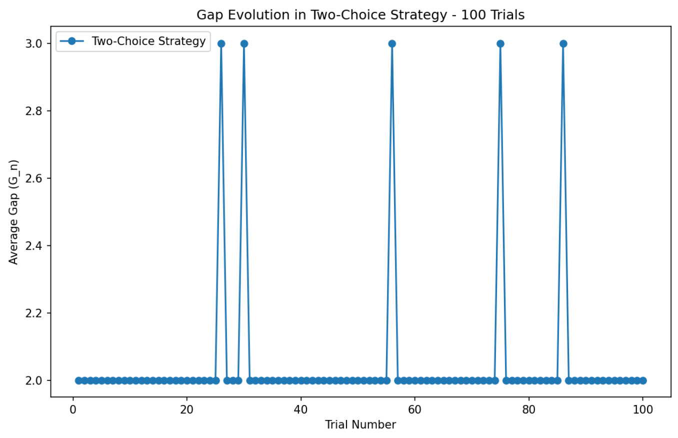
Below there are 100 trials with increasing number of balls from 100 bins from 1 to m^2 with the average gap at a 1.75 and becoming more and more stable as n increases.

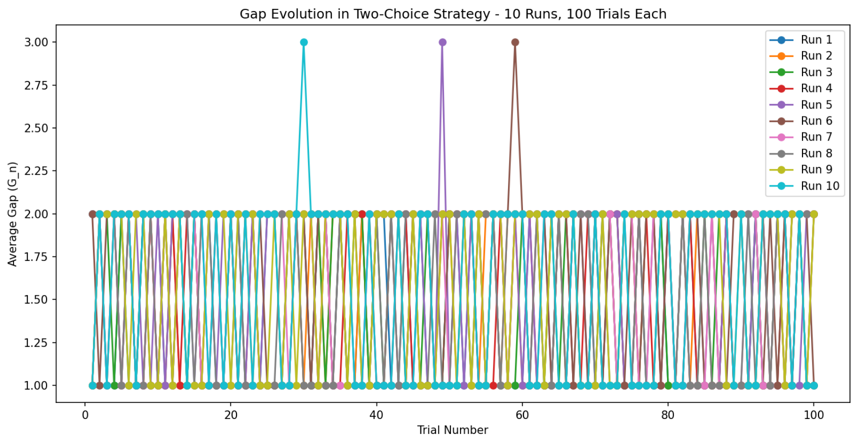


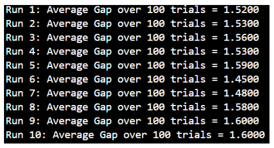
3.1 Light-loaded

In this subsection $n = m$. This the allocation becomes more stable from the beginning, due to the forced filling of the smaller bin. Unlike the previous strategy where it was more unstable and it required a lot more experiments and a larger n, here the average gap is around 1.6.





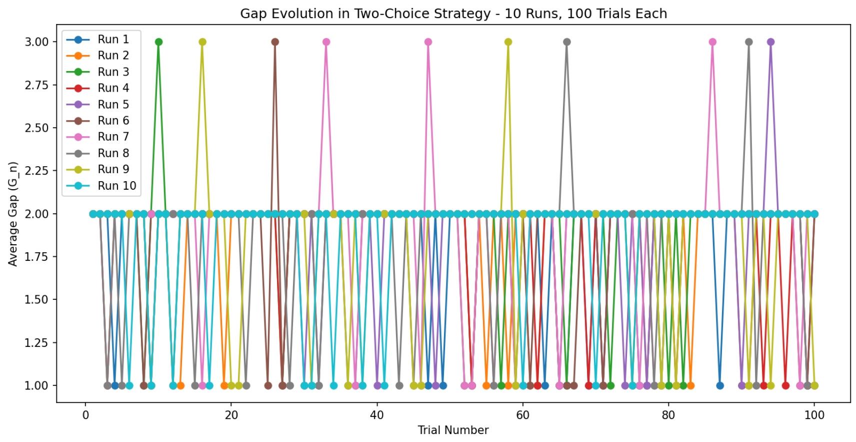


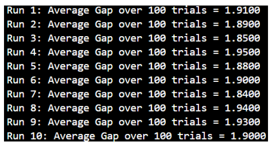


```
Run 1: Average Gap over 100 trials = 1.5200
Run 2: Average Gap over 100 trials = 1.5300
Run 3: Average Gap over 100 trials = 1.5600
Run 4: Average Gap over 100 trials = 1.5300
Run 5: Average Gap over 100 trials = 1.5900
Run 6: Average Gap over 100 trials = 1.4500
Run 7: Average Gap over 100 trials = 1.4800
Run 8: Average Gap over 100 trials = 1.5800
Run 9: Average Gap over 100 trials = 1.6000
Run 10: Average Gap over 100 trials = 1.6000
```

3.2 Heavy-loaded

When n grows the average stabilizes at around 1.9.

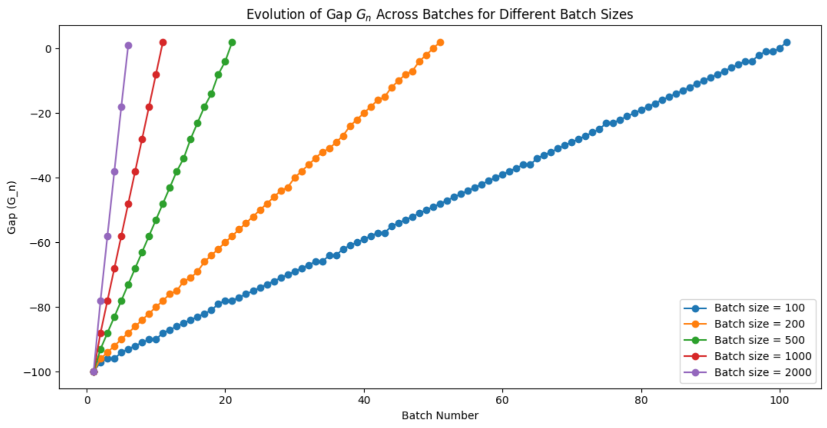


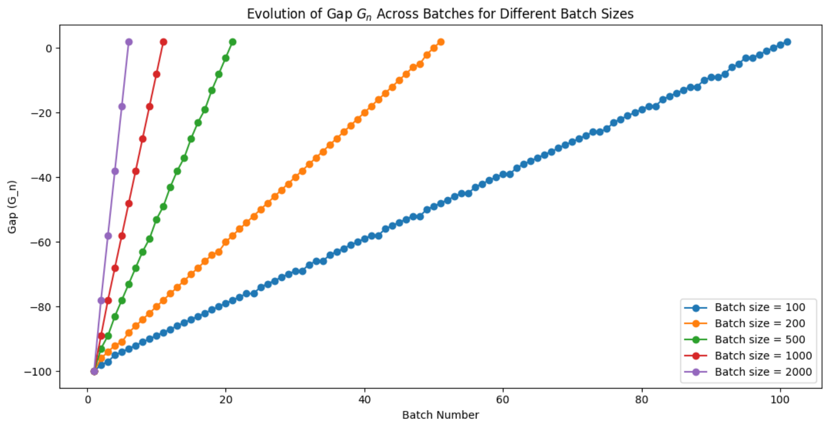


```
Run 1: Average Gap over 100 trials = 1.9100
Run 2: Average Gap over 100 trials = 1.8900
Run 3: Average Gap over 100 trials = 1.8500
Run 4: Average Gap over 100 trials = 1.9500
Run 5: Average Gap over 100 trials = 1.8800
Run 6: Average Gap over 100 trials = 1.9000
Run 7: Average Gap over 100 trials = 1.8400
Run 8: Average Gap over 100 trials = 1.9400
Run 9: Average Gap over 100 trials = 1.9300
Run 10: Average Gap over 100 trials = 1.9000
```

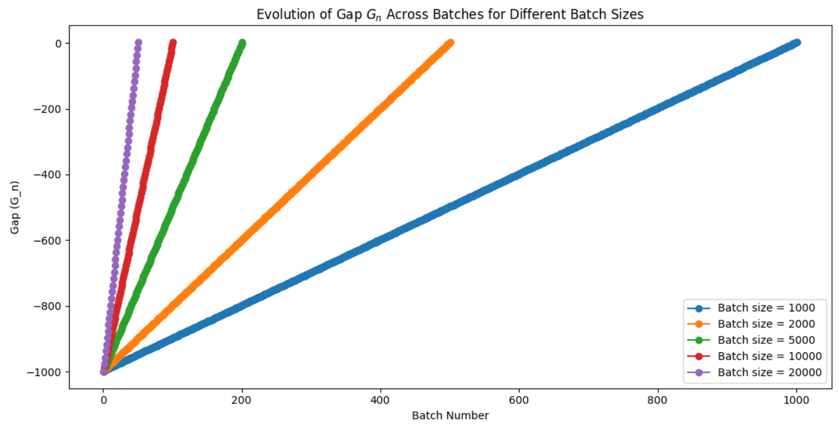
3.3 B-Batched Setting

In this subsection, we will discuss the implications of this setting. Unlike the previous strategy where no matter the belief of the system's load, the same decision will be taken, that is, a random one, here because between 2 bins we will always prefer the smaller one, this configuration will have a big impact, that is a higher "error". This is due to the fact that because during $batch_size$ ball allocations, the belief does not change. This implies that between 2 bins the smaller bin will be always preferred during the $batch_size$ ball allocations, even though every time we choose it it grows and there is a big possibility of it not remaining the smallest load bin.





It can be seen clearly that right know, the difference in the gap is not big between the different sizes, however it is bigger than the normal strategy.



4 The $(1+\beta)$-choice

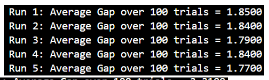
In this section, we will explore the $(1+\beta)$-choice, which is a hybrid method based on the previous two methods, that is with a certain probability (β) it will choose the one choice strategy and with $(1-\beta)$ the two-choice. Based on the previous experiments, choosing a random bin and just loading the ball in it, works like an "exploration" part, while the two-choice works in a more "greedy" manner, that is between 2 bins, the lesser filled bin is chosen. This implies that this strategy incorporates in a semi-formal way the RL technique or policy called $\epsilon - greedy$. In RL, this technique consists in a balance or trade-off between exploration (choosing a random action and going with it) and exploitation (choosing the "best" action in the present). From the introduction of the β parameter, we can already see how this methods starts to look like the RL $\epsilon - greedy$.

4.1 Light-Loaded

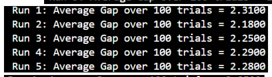
In this subsection, $n = m$.

$\beta = 0.1$

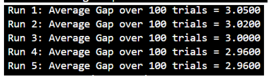
In this subsection we use the specified parameter and we increase m, n from 100 to 1000 and 10000. This results show that here the exploration factor is much lower and it impacts positively the average gap. However, on larger m and n it does not differ from the $\beta = 0.3$.



```
Run 1: Average Gap over 100 trials = 1.8500
Run 2: Average Gap over 100 trials = 1.8400
Run 3: Average Gap over 100 trials = 1.7900
Run 4: Average Gap over 100 trials = 1.8400
Run 5: Average Gap over 100 trials = 1.7700
```



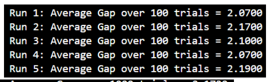
```
 Run 1: Average Gap over 100 trials = 2.3100
 Run 2: Average Gap over 100 trials = 2.1800
 Run 3: Average Gap over 100 trials = 2.2500
 Run 4: Average Gap over 100 trials = 2.2900
 Run 5: Average Gap over 100 trials = 2.2800
```



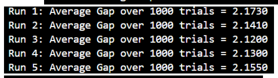
```
Run 1: Average Gap over 100 trials = 3.0500
Run 2: Average Gap over 100 trials = 3.0200
Run 3: Average Gap over 100 trials = 3.0000
Run 4: Average Gap over 100 trials = 2.9600
Run 5: Average Gap over 100 trials = 2.9600
```

$\beta = 0.3$

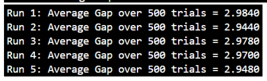
Here is a series of runs with different amounts of bins and trials. The first 2 images show 5 runs with $m = n = 100$ bins and balls while the third one show $m = n = 1000$ bins and balls. The forth images shows how the average grows from 2.1 and 2.9 to 3.7 with $m = n = 10000$. While it increases the average gap, we have to keep in mind that the number of bins and balls increases 10 times each experiment, so while the test more than doubles its size, the "error" rate is just slightly increasing. In this experiment configuration ($\beta = 0.3$) the exploration factor (the one choice percentage of being used) shows an increase in the average gap vs the 2 choice but is a drastic improvement over the simple one-choice strategy.



```
Run 1: Average Gap over 100 trials = 2.0700
Run 2: Average Gap over 100 trials = 2.1700
Run 3: Average Gap over 100 trials = 2.1000
Run 4: Average Gap over 100 trials = 2.0700
Run 5: Average Gap over 100 trials = 2.1900
```



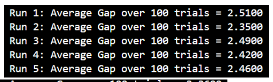
```
Run 1: Average Gap over 1000 trials = 2.1730
Run 2: Average Gap over 1000 trials = 2.1410
Run 3: Average Gap over 1000 trials = 2.1200
Run 4: Average Gap over 1000 trials = 2.1300
Run 5: Average Gap over 1000 trials = 2.1550
```



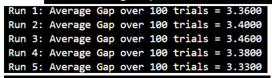
```
Run 1: Average Gap over 500 trials = 2.9840
Run 2: Average Gap over 500 trials = 2.9440
Run 3: Average Gap over 500 trials = 2.9780
Run 4: Average Gap over 500 trials = 2.9700
Run 5: Average Gap over 500 trials = 2.9480
```

$\beta = 0.5$

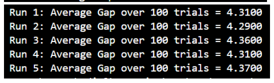
Here is a perfect balance between exploration and exploitation. As expected, from the results shown before, a bigger β increases the average gap as it involves more exploration.



```
Run 1: Average Gap over 100 trials = 2.5100
Run 2: Average Gap over 100 trials = 2.3500
Run 3: Average Gap over 100 trials = 2.4900
Run 4: Average Gap over 100 trials = 2.4200
Run 5: Average Gap over 100 trials = 2.4600
```



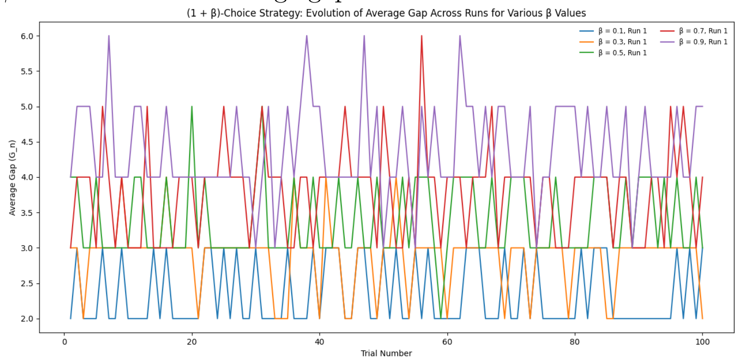
```
Run 1: Average Gap over 100 trials = 3.3600
Run 2: Average Gap over 100 trials = 3.4000
Run 3: Average Gap over 100 trials = 3.4600
Run 4: Average Gap over 100 trials = 3.3800
Run 5: Average Gap over 100 trials = 3.3300
```



```
Run 1: Average Gap over 100 trials = 4.3100
Run 2: Average Gap over 100 trials = 4.2900
Run 3: Average Gap over 100 trials = 4.3600
Run 4: Average Gap over 100 trials = 4.3100
Run 5: Average Gap over 100 trials = 4.3700
```

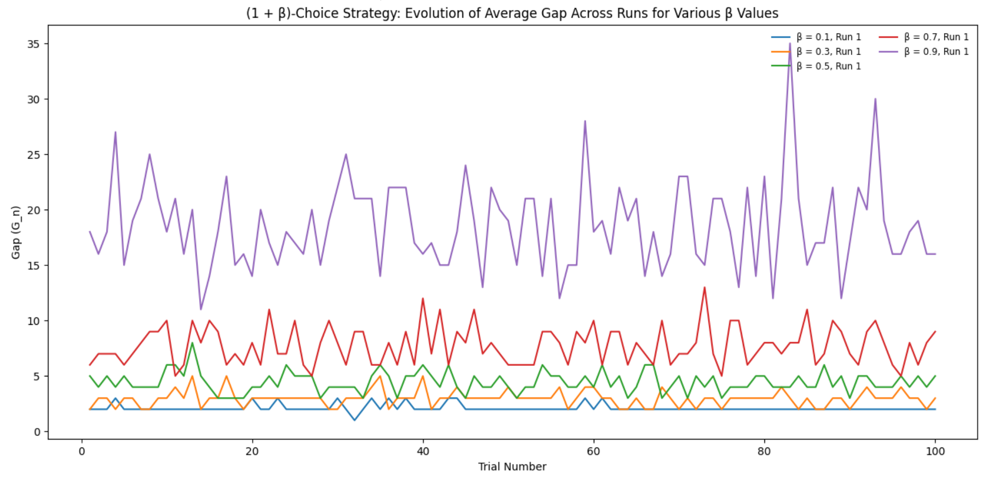
Conclusion - light-loaded

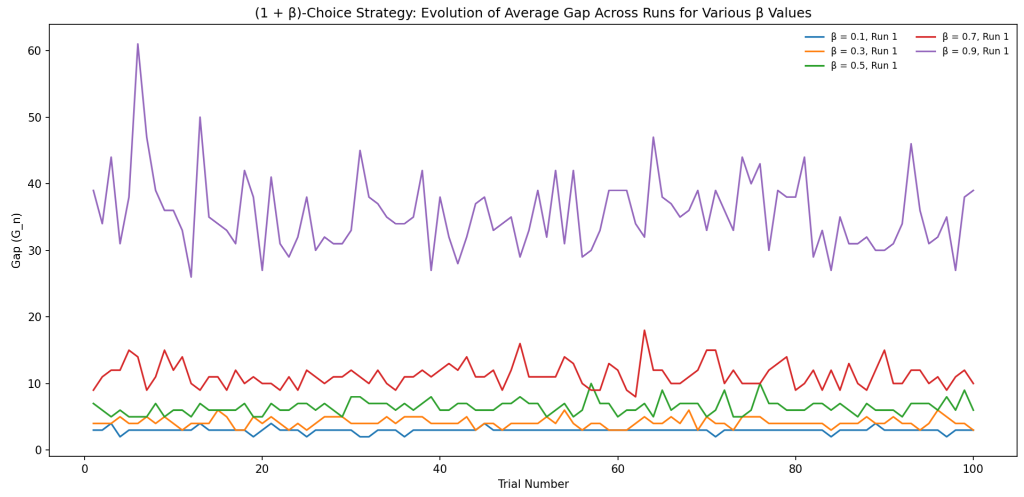
Here we can clearly see that on 1000 bins and balls, the smaller the parameter β the smaller the average gap.



4.2 Heavy-Loaded

For the heavy-load, we will use constant $n = m^2$ and the usual values for m are $[100, 1000, 10000]$.We can expect the same conclusion from the light-loaded variation, the smaller the β the better the average gap as seen below ($m = 100$ and $m = 1000$ in the second one).





5 Conclusions

Based on the previous experiments, it can be clearly seen that in the case of always having the full knowledge of the environment the greedy strategy ”two-choice” works best because it will always choose the best possible bin from those 2. It is clear that it can be upgrade by selecting a bigger d to maximize the greedy. However, when the batched setting was used, it is clearly at a big disadvantage to the normal strategy due to the greedy choice that will remain preferred during specific batches.

5.1 Repository

The link to the repository is this:
https://github.com/adieldinu/ball-allocation/tree/main