Analysis of Balls into Bins Allocation Strategies

Randomized Algorithms Assessment

November 15, 2025

This document presents the results of a simulation comparing various randomized strategies for the Balls into Bins problem in a heavy-loaded scenario ($n = M^2$). The goal is to evaluate how limiting the decision-making process (reduced choice as 1-Choice , partial knowledge like $(1+\beta)$-Choice or Binary Queries, or delayed information such as in the b-Batched case) affects the average maximum load, defined as the **Gap** ($\bar{G}_n$). The simulation was run with $M = 100$ bins and $N_{MAX} = 10,000$ balls, with $T = 30$ repetitions for averaging the gap through different experiments.

1 Experimental Setup

The simulation parameters are defined as follows:

- **Number of Bins (M)**: 100
- **Maximum Number of Balls (N_{MAX})**: 10,000 (M^2)
- **Number of Runs (T)**: 30

The Gap ($\bar{G}_n$) is calculated as the maximum load minus the average load: $\bar{G}_n = \max(X_i) - n/M$. The final results are summarized at $n = N_{MAX} = 10,000$ and in file named "output" present in the project's main directory.

2 Results: Comparative Analysis of Strategies

The strategies are grouped into three main categories. The reference strategies, 1-Choice and 2-Choice, define the "performance" bounds.

2.1 Standard Strategies (Perfect Information)

These experiments use real-time, perfect load information (equivalent to $b = 1$). The results confirm the theoretical advantage of two choices (2-Choice) over one choice (1-Choice), which reduces the gap.

Table 1: Gap Comparison for Standard and $(1+\beta)$-Choice Strategies at $n = 10,000$

Strategy	Average Gap $\bar{G}_{N_{MAX}}$
1-Choice	26.13
2-Choice	1.83
(1+0.2)-Choice	12.07
(1+0.5)-Choice	4.40
(1+0.8)-Choice	2.43

The (1+0.8)-Choice strategy performs very close to the optimal 2-Choice (Gap ≈ 2.43 vs 1.83). Conversely, increasing the probability of a random choice, as seen in **(1 +0.2)-Choice**, causes a worst performance, approaching the 1-Choice gap.

2.2 Non updated Information (b-Batched)

This section examines how delayed load information affects the 2-Choice strategy. The batch size b is the interval after which the load vector is updated.

Table 2: Gap Comparison for b-Batched Strategies at $n = 10,000$

Strategy	Average Gap $\bar{G}_{N_{MAX}}$
2-Choice (Standard, $b = 1$)	1.83
2-Choice ($b = 100/M$)	3.30
2-Choice ($b = 1000/M$)	11.77

A small batch size ($b = M$) nearly doubles the gap (≈ 3.30). A large batch size ($b = 10M$) leads to a clear performance degradation (Gap ≈ 11.77), demonstrating that the benefit of 2-Choice is sensitive to the freshness of the load information.

2.3 Impact of Partial Information (Binary Query)

This section evaluates the 2-Choice strategy using only k binary queries to infer load information (median, quartiles).

Table 3: Gap Comparison for Binary Query Strategies at $n = 10,000$

Strategy	Average Gap $\bar{G}_{N_{MAX}}$
2-Choice (Standard, $k = \infty$)	1.83
2-Choice ($k = 1$ Query)	6.60
2-Choice ($k = 2$ Query)	3.73

The $k = 2$ Query strategy, which uses quartiles to better discriminate candidates, provides a good trade-off (Gap ≈ 3.73). While obviously being worse than perfect knowledge, it is better than the $k = 1$ Query strategy (Gap ≈ 6.60), showing that even a little more information can improve the performance.

3 Graphical Results

The following figures illustrate the evolution of the average gap ($\bar{G}_n$) for the three experimental settings, including the shaded areas, that represent the standard deviation (uncertainty) across $T = 30$ runs.

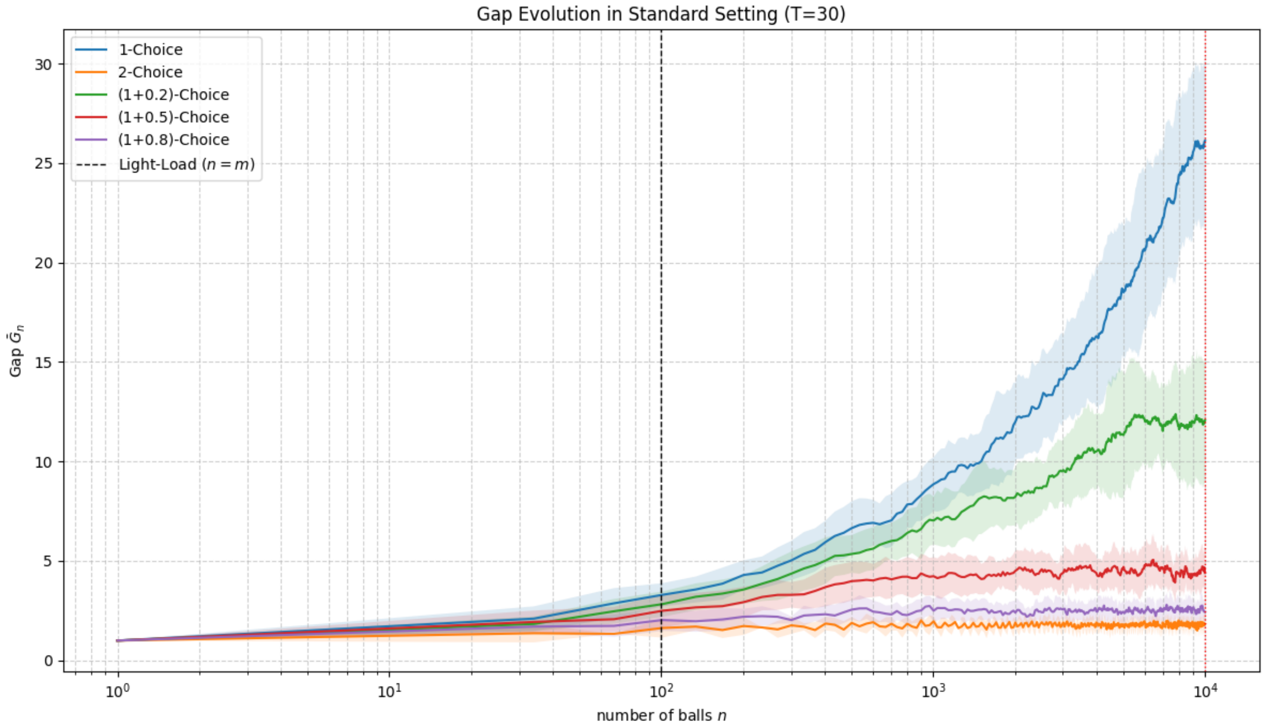

Figure 1: Gap Evolution in Standard Setting ($T = 30$). Shows that 2-Choice is optimal and performance degrades as the probability of random choice (1-Choice) increases.

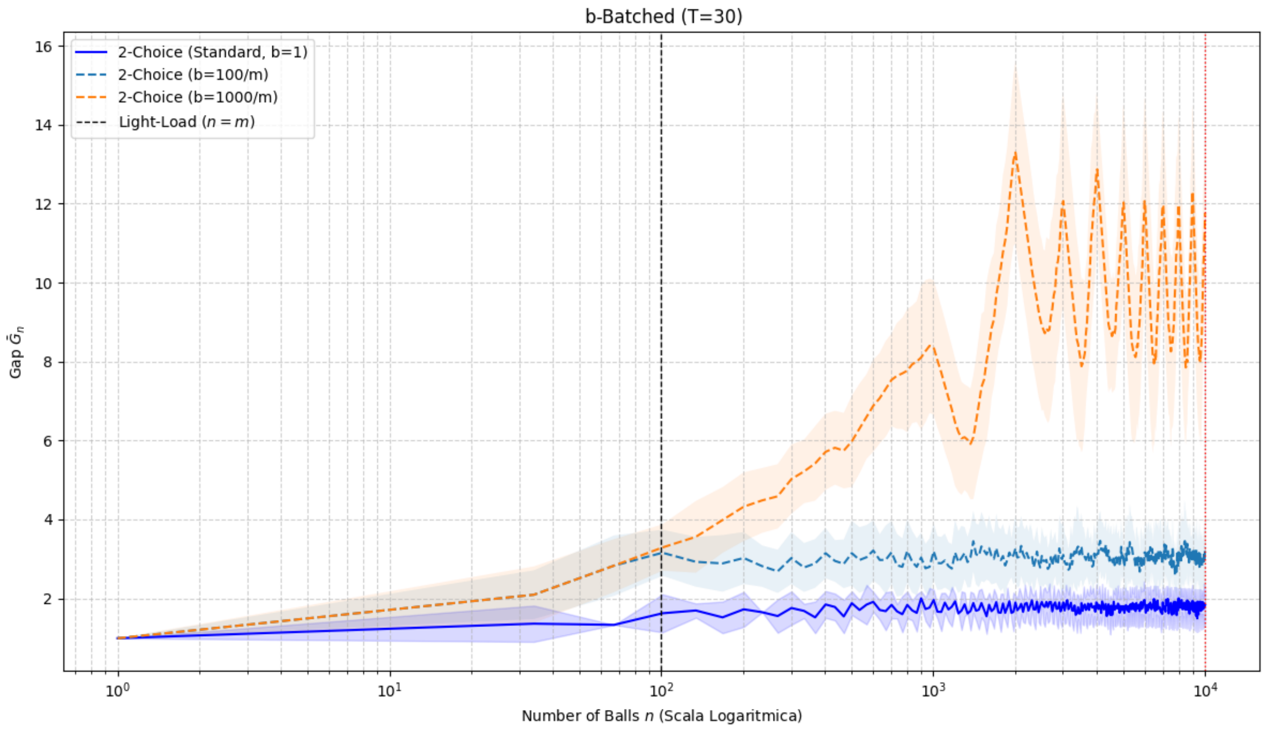

Figure 2: Gap Evolution in b-Batched Setting ($T = 30$). Shows that by increasing the batch size b and so by delaying the information update, we compromise the performance of the 2-Choice strategy.

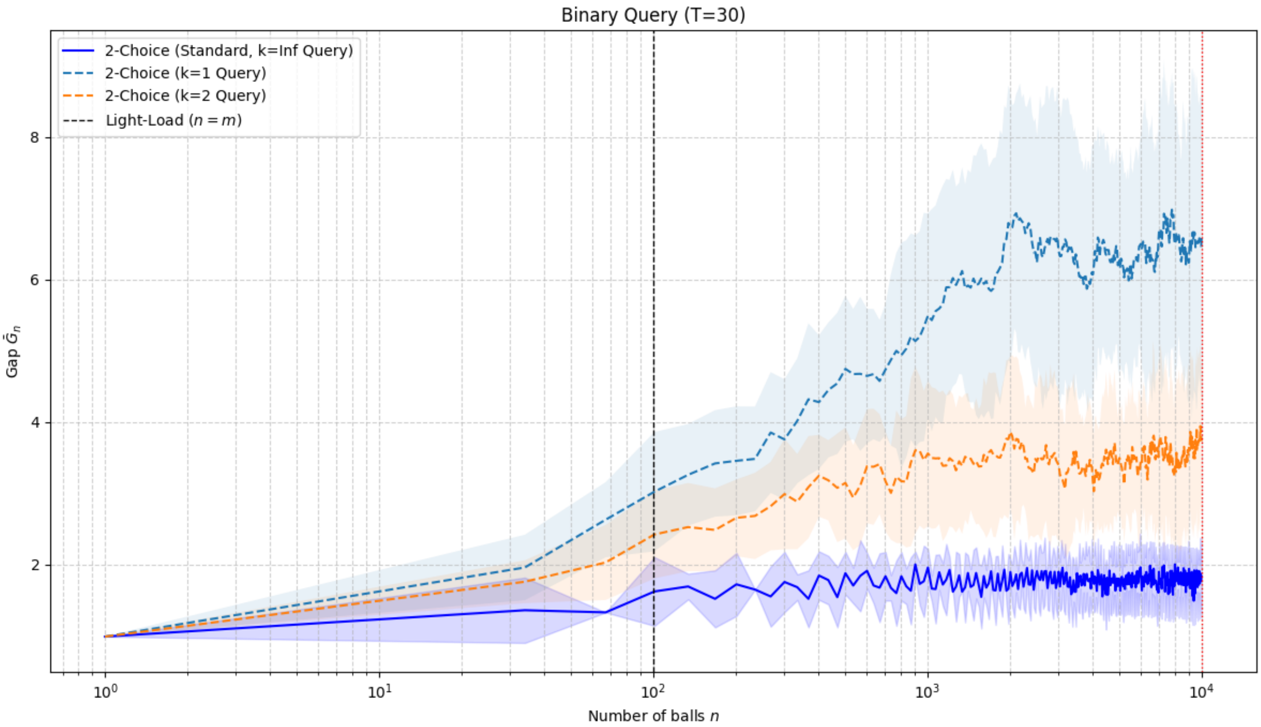

Figure 3: Gap Evolution in Binary Query Setting ($T = 30$). Shows that 2 Binary Queries provide better results than 1 Query ($k = 1$) as they approach the 2-Choice behavior.