Robust Load Balancing with Machine Learned Advice

SARA AHMADIAN, Google Research, United States
HOSSEIN ESFANDIARI, Google Research, United States
VAHAB MIRROKNI, Google Research, United States
BINGHUI PENG, Comlumbia University, United States

Motivated by the exploding growth of web-based services and the importance of efficiently managing the computational resources of such systems, we introduce and study a theoretical model for load balancing of very large databases such as commercial search engines. Our model is a more realistic version of the well-received balls-into-bins model with an additional constraint that limits the number of servers that carry each piece of the data. This additional constraint is necessary when, on one hand, the data is so large that we can not copy the whole data on each server. On the other hand, the query response time is so limited that we can not ignore the fact that the number of queries for each piece of the data changes over time, and hence we can not simply split the data over different machines.

In this paper, we develop an almost optimal load balancing algorithm that works given an estimate of the load of each piece of the data. Our algorithm is almost perfectly robust to wrong estimates, to the extent that even when all of the loads are adversarially chosen the performance of our algorithm is $1 - 1/e$, which is provably optimal. Along the way, we develop various techniques for analyzing the balls-into-bins process under certain correlations and build a novel connection with the multiplicative weights update scheme.

ACM Reference Format:
Sara Ahmadian, Hossein Esfandiari, Vahab Mirrokni, and Binghui Peng. 2023. Robust Load Balancing with Machine Learned Advice. 1, 1 (April 2023), 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The *balls-into-bins* paradigm is the most fundamental model for load balancing in real-time distributed systems. In the classical balls-into-bins problem, we model the real-time requests as balls and the servers as bins. In each round, a memory-less allocation algorithm places the incoming balls into one of the bins. The goal is to balance the number of balls assigned to the bins, commonly referred as loads. Constrained by the latency requirement, the algorithm is only allowed to look at the loads of a few bins. The most well-established result in the context of balls-into-bins is the *power of two choices*, or more generally, the *power of d choices* algorithm which looks at d bins uniformly at random in each round and assigns the ball to the bin with the minimum load. The maximum load of all bins is then bounded by $\frac{T}{n} + O(\frac{\log\log n}{\log d})$ [3], where n is the number of bins and T is the number of balls.

The classical formulation of balls-into-bins is extensively studied in the literature, as it captures several applications including hashing, share memory emulations and jobs allocations. However, in modern distributed service systems such as web search engines, there is a significant gap between the theory and practice. In these modern applications, since the size of the whole data is very large, each piece of the data (e.g. one dataset) is only replicated across a few machines, and the replication

Authors' addresses: Sara Ahmadian, Google Research, United States; Hossein Esfandiari, Google Research, United States; Vahab Mirrokni, Google Research, United States; Binghui Peng, Comlumbia University, United States.

XXXX-XXXX/2023/4-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

is *fixed* throughout the time. As a consequence, a query can only be addressed on servers that hold a copy of its corresponding dataset. This challenges the common assumption that one can choose an arbitrary machine to process a request. Hence, two fundamental questions arise:

Q1. How do we replicate the data across servers?
Q2. When a real-time query appears, how should it be assigned?

Given the emerging popularity of web based services, we believe it is crucial to address these questions and mitigate the gap of the classical theory.

In this work, we introduce a new balls-into-bins model that captures the practical issues arising in such distributed systems and develop a near optimal algorithm with machine learned advice.

1.1 Our Results

We present a near optimal algorithm for the load balancing problem. The performance of our algorithm is measured w.r.t. the *offline optimal solution*, which is defined as the solution that minimizes the maximum load of the servers on the actual load distribution.

Theorem 1.1. *Suppose $\epsilon > 0$, $d = \Omega(\log n/\epsilon^2 + 1/\epsilon^4)$, and the estimation error is bounded by $\lambda \in [0, 1]$. There is a randomized network construction and online load balancing algorithm such that with probability at least $1 - 1/\mathrm{poly}(n)$,*

- *The maximum load on servers is always $O(1)$ approximate to the offline optimal.*
- *When $T = \Omega(\frac{n^2 \log n}{\epsilon^2})$, the maximum load on servers is $(\frac{1}{1-\lambda \exp(-\lambda)} + \epsilon)$ approximate to the offline optimal. Moreover, no algorithm is capable of achieving $(\frac{1}{1-\lambda \exp(-\lambda)} - \epsilon)$ approximation.*

Our algorithm consists of two subroutines. First, we come up with a network construction algorithm that receives an estimation of future loads as the input, outputs a bipartite graph of clients and servers with near optimal approximation guarantee.

Theorem 1.2. *For any $\epsilon > 0$, suppose $d \geq \Omega\left(\log n/\epsilon^2 + 1/\epsilon^4\right)$ and the estimation error is bounded by $\lambda \in [0, 1]$. There is a randomized network construction algorithm, such that with probability $1 - 1/\mathrm{poly}(n)$, it achieves $(\frac{1}{1-\lambda \exp(-\lambda)} + \epsilon)$ approximation to the offline optimal. The approximation is tight as no algorithm can achieve $(\frac{1}{1-\lambda \exp(-\lambda)} - \epsilon)$ approximation.*

The second subroutine is online load balancing, where we use the power of d choice algorithm. We prove the d-choice algorithm is constant approximation in the small load regime and $(1 + o(1))$ approximation asymptotically.

Theorem 1.3. *Let $d = \Omega(\log n)$. Suppose the bipartite graph is constructed by the algorithm of Theorem 1.2, and we assign online requests using the d-choice algorithm, then with probability at least $1 - 1/\mathrm{poly}(n)$,*

- *The maximum load on servers is always constant approximate to the offline optimal.*
- *When the number of requests $T = \Omega(\frac{n^2 \log n}{\epsilon^2})$, the maximum load on servers is $(1+\epsilon)$ approximate to the optimal allocation on the graph.*

The journal version of this paper has been published at [2], and the preliminary conference version of this paper is at [1].

REFERENCES

[1] Sara Ahmadian, Hossein Esfandiari, Vahab Mirrokni, and Binghui Peng. 2022. Robust Load Balancing with Machine Learned Advice. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, 20–34.
[2] Sara Ahmadian, Hossein Esfandiari, Vahab Mirrokni, and Binghui Peng. 2023. Robust Load Balancing with Machine Learned Advice. *Journal of Machine Learning Research* 24, 44 (2023), 1–46.
[3] Petra Berenbrink, Artur Czumaj, Angelika Steger, and Berthold Vöcking. 2006. Balanced allocations: The heavily loaded case. *SIAM J. Comput.* 35, 6 (2006), 1350–1385.