Reinforcement Learning with Work Duplication for Load Balancing in Elasticsearch

Haley Li (haleyli@cs.ubc.ca), Mathias Lécuyer (mathias.lecuyer@ubc.ca)

University of British Columbia

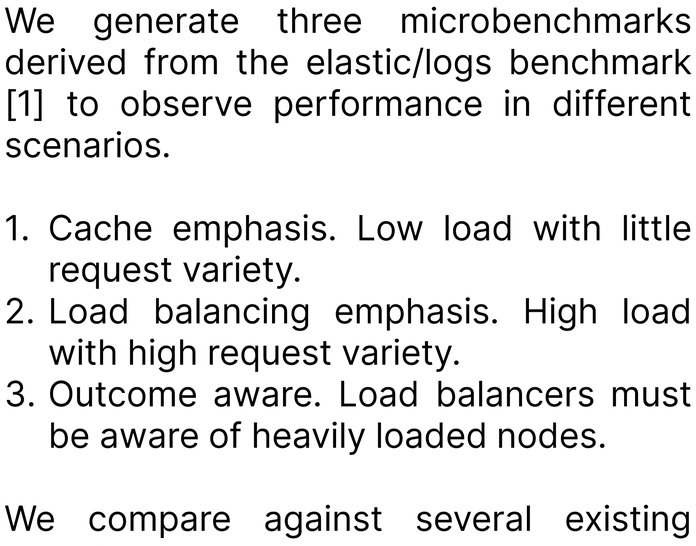
INTRODUCTION

Better decision-making in systems lead to better performance, but good decision-making is difficult due to:

- Large state spaces
- Variability in workloads
- Dependence between decisions

Reinforcement learning (RL) provides a means to learn to make these decisions. RL needs to explore to learn. Exploration steps grant learning opportunities but can be detrimental to tail latency. Balancing this is known as the explore-exploit trade-off.

We propose a method to explore for RL in systems with without the risk of poor outcomes by duplicating work so that we can explore and exploit at the same time.



METHODOLOGY

We evaluate work duplication for RL on load balancing search requests in Elasticsearch.

We generate three microbenchmarks derived from the elastic/logs benchmark [1] to observe performance in different scenarios.

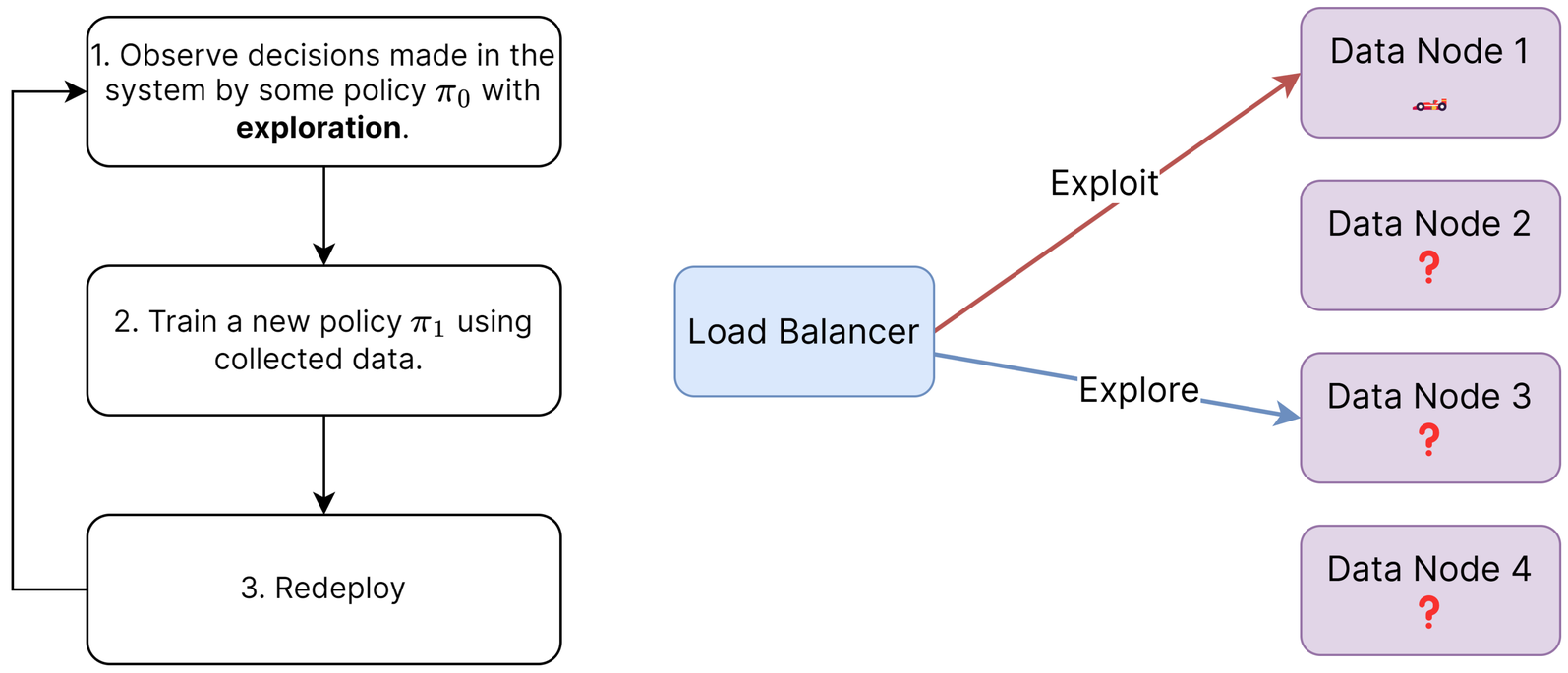
1. Cache emphasis. Low load with little request variety.
2. Load balancing emphasis. High load with high request variety.
3. Outcome aware. Load balancers must be aware of heavily loaded nodes.

We compare against several existing baseline load balancers:

1. Adaptive Replica Selection (ARS) [2, 3]
2. Random
3. Sorted
4. Round robin

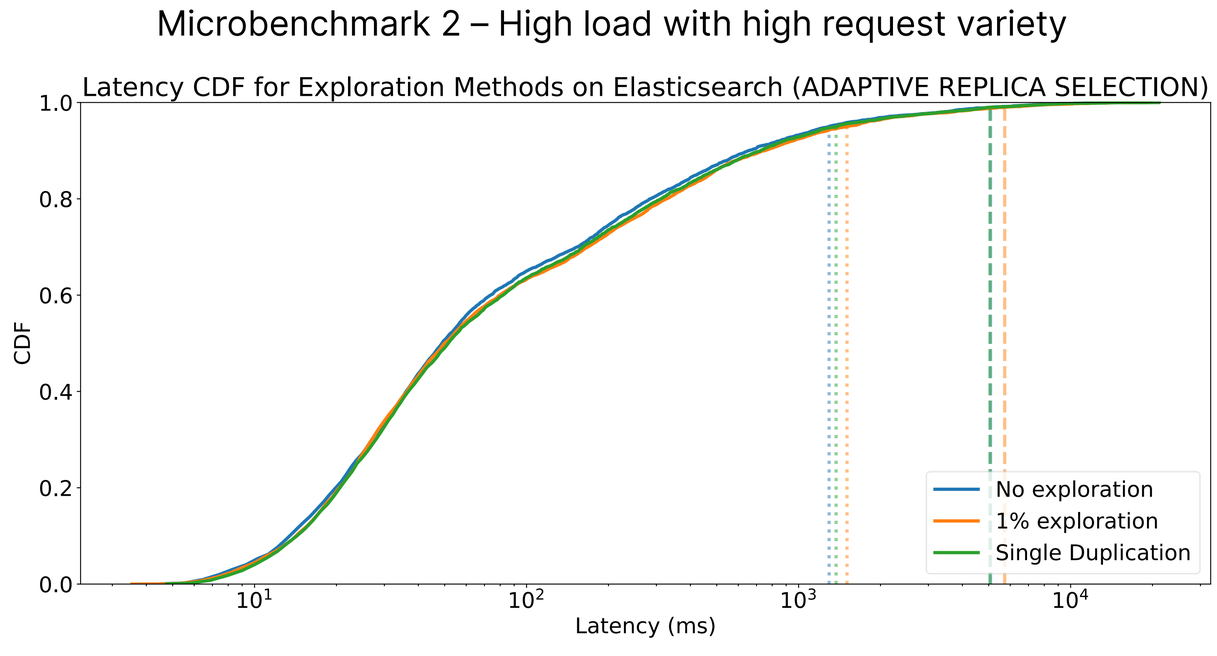
EXPLORE-RESOURCE TRADE-OFF

Work duplication lets us explore and exploit at the same time. Then we can explore without detriment to the user. The trade-off is that it costs extra resources.

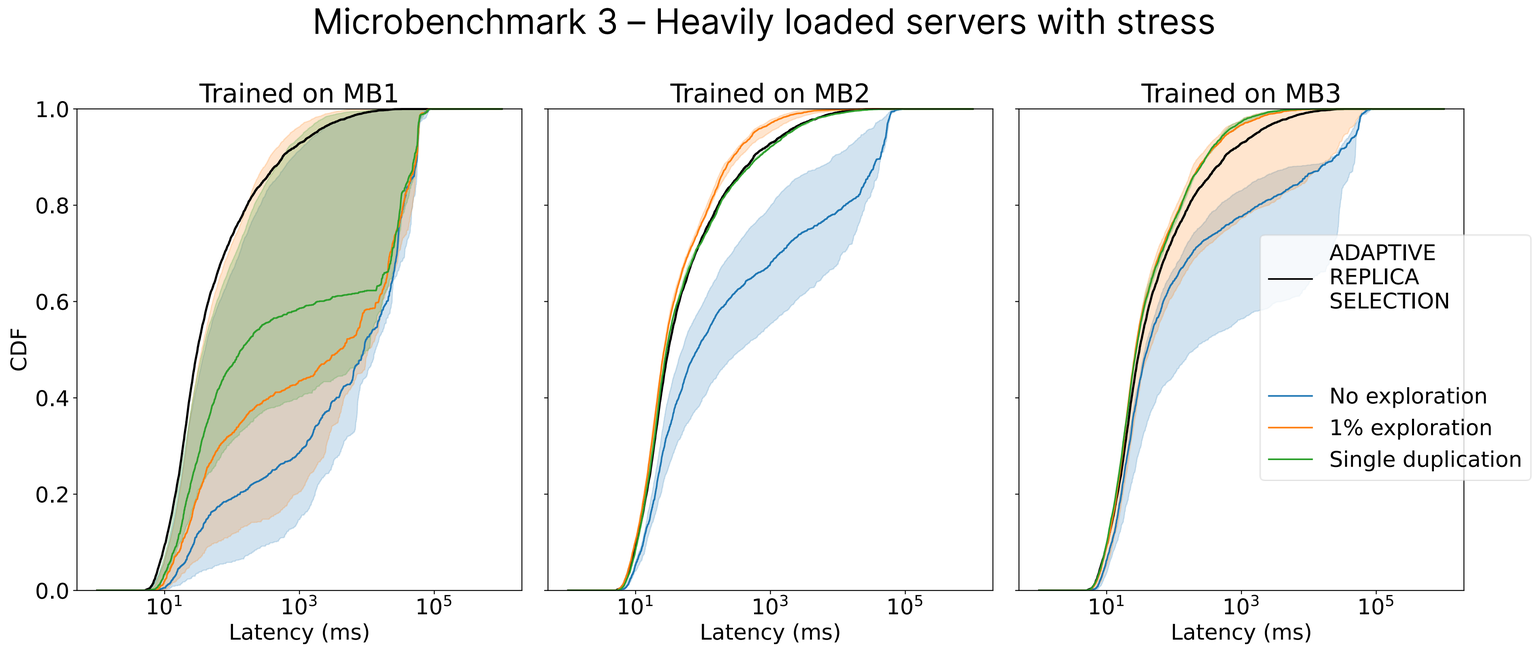


RESULTS

Cost of exploration. On average, we observe a 12% reduction in p95 latency and an 11% reduction in p99 latency when using duplication compared to exploration.



Bandit model performance. We found that bandit models could be competitive but would still benefit from true RL such as Q-learning. We found that in more complex scenarios such as microbenchmark 3, exploration was necessary to achieve better models.



REFERENCES

1. https://github.com/elastic/rally-tracks/tree/master/elastic/logs
2. https://www.elastic.co/blog/improving-response-latency-in-elasticsearch-with-adaptive-replica-selection
3. Lalith Suresh, Marco Canini, Stefan Schmid, and Anja Feldmann. 2015. C3: Cutting Tail Latency in Cloud Data Stores via Adaptive Replica Selection. NSDI 15