Optimal Load-Balancing for Heterogeneous Clusters

Anshul Gandhi, Naman Mittal, Xi Zhang
Stony Brook University *

1. INTRODUCTION

Online service providers, such as Amazon and Google, often dedicate thousands of nodes for their applications, which typically receive thousands of requests per second. Clustered systems today are inherently heterogeneous [1]. Consequently, scheduling the large amount of requests on such large heterogeneous clusters requires *scalable, heterogeneity-aware Load Balancers (LBs)*.

Randomized LBs split incoming requests among available back-end servers based on some probabilities [2]. While scalable, such randomized LBs are typically *not* heterogeneity-aware, and are not adaptive to changes in load. By contrast, heterogeneity-aware LBs probe the state of all servers in the cluster to determine, often via complex computations, the best candidate for each incoming request [1].

In this paper, we propose a novel heterogeneity-aware LB algorithm, HALB, that is based on randomized routing. Our optimal LB algorithm is simple, hence scalable, yet heterogeneity-aware and adaptive. We start with simple randomized LB algorithms and tune the request split, via queueing-theoretic analysis, for optimal performance in heterogeneous environments. Our analytical and (Apache-based) implementation results for HALB suggest that the natural solution of proportional splitting based on server speeds can be far from optimal.

2. HALB

Consider a distributed heterogeneous cluster with n different types of servers. We logically partition the servers into n groups of homogeneous servers. We only consider a single resource at a server, such as CPU. Let the number of type i servers be c_i, and let the speed of a type i server be μ_i req/s (assuming a request takes $1/\mu_i$ seconds to complete, on average, on a type i server). Let the incoming request rate be λ req/s. Under probabilistic load balancing, we split incoming requests among the available server groups. Letting p_i denote the probability of sending a request to server group i, our goal is to determine the optimal request split probability, p_i^* $(\forall i)$, so as to minimize average response time, T. Note that $\sum_{i=1}^{n} p_i = 1$.

In order to determine p_i^*, we consider a queueing-theoretic model. We assume that the arrival process is Poisson, with mean λ req/s. There is no restriction on the distribution of request sizes. We assume that there is only one class of CPU-bound requests, and each request can be served by any of the available servers. Servers can process multiple requests simultaneously (processor sharing). Based on this approximate model, we can compute T using a network of $M/G/1/PS$ queues as (here, T_i is the mean response time of a request in group i): $T = \sum_{i=1}^{n} p_i T_i = \sum_{i=1}^{n} \frac{p_i}{\mu_i - \lambda p_i / c_i}$.

*Supported by an AWS in Education Grant Award.

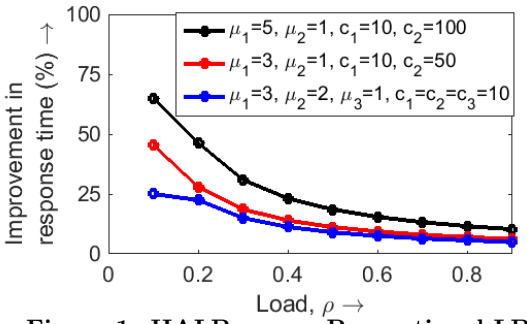

Figure 1: HALB versus Proportional LB.

The optimal probabilities, p_i^*, that minimize T can now be determined by optimizing the above expression given λ and server speeds μ_i. We omit the optimization details here, and present the final result:

$$p_i^* = \frac{c_i\mu_i \sum_{j=1}^{n}(c_j\sqrt{\mu_j}) - c_i\sqrt{\mu_i}\sum_{j=1}^{n}(c_j\mu_j) + c_i\sqrt{\mu_i}\lambda}{\lambda \sum_{j=1}^{n}(c_j\sqrt{\mu_j})} \quad (1)$$

Of course, we restrict p_i^* to be between 0 and 1. Note the dependence of p_i^* on request rate, λ.

Comparison with Proportional LB: The natural request split between different groups is $p_i = c_i\mu_i / \sum_{j=1}^{n} c_j\mu_j$. We refer to this as *Proportional LB*. Surprisingly, the optimal split, p_i^*, can be far from the proportional split, as shown by our analytical results in Fig. 1. In order to show multiple cases, we plot the percentage improvement in mean response time afforded by HALB over Proportional LB as a function of load, ρ (ratio of request rate, λ, and maximum system throughput, $\sum_{j=1}^{n} c_j\mu_j$). We see that HALB results in significantly lower T than proportional split, especially when the heterogeneity is high (difference in μ) and the load is low, with the improvement in response time being **as large as 60%** in some cases. Clearly, Proportional LB is not as good as expected. We further validate our results via implementation on a small-scale (5 node) heterogeneous cluster of dual-core Intel servers equipped with DVFS to change the CPU operating frequencies. We modify Apache's `mod_proxy` module to enable HALB. Our results (omitted due to lack of space) show that HALB provides significant improvement in response times, ranging from **39% at low loads to 19% at high loads**, when compared to Proportional LB.

3. CONCLUSION

In this paper we present a new LB that bridges the gap between scalable and heterogeneity-aware LBs. Our end-goal in this research is to build a practical LB for distributed environments; this paper represents our first step.

4. REFERENCES

[1] R. Nathuji, C. Isci, and E. Gorbatov, "Exploiting Platform Heterogeneity for Power Efficient Data Centers," in *Proceedings of ICAC 2007*.

[2] "Elastic Load Balancing Concepts." `http://docs.aws.amazon.com/ElasticLoadBalancing/latest/DeveloperGuide/TerminologyandKeyConcepts.html`.