Understanding Response Time in the Redundancy-d System

Kristen Gardner
Computer Science Dept.
Carnegie Mellon University
ksgardne@cs.cmu.edu

Samuel Zbarsky
Carnegie Mellon University
szbarsky@andrew.cmu.edu

Mark Velednitsky
IEOR Department
UC Berkeley
marvel@berkeley.edu

Mor Harchol-Balter
Computer Science Dept.
Carnegie Mellon University
harchol@cs.cmu.edu

Alan Scheller-Wolf
Tepper School of Business
Carnegie Mellon University
awolf@andrew.cmu.edu

ABSTRACT

An increasingly prevalent technique for improving response time in queueing systems is the use of redundancy. In a system with redundant requests, each job that arrives to the system is copied and dispatched to multiple servers. As soon as the first copy completes service, the job is considered complete, and all remaining copies are deleted. A great deal of empirical work has demonstrated that redundancy can significantly reduce response time in systems ranging from Google's BigTable service to kidney transplant waitlists.

We propose a theoretical model of redundancy, the Redundancy-**d** system, in which each job sends redundant copies to **d** servers chosen uniformly at random. We derive the first exact expressions for mean response time in Redundancy-**d** systems with any finite number of servers. We also find asymptotically exact expressions for the distribution of response time as the number of servers approaches infinity.

1. INTRODUCTION

Redundancy – the idea of dispatching multiple copies of the same job and waiting for the first copy to complete service – is an important strategy for reducing response times in applications ranging from Google's BigTable service to kidney transplant waitlists.

Redundancy provides significant response time improvements because it exploits two sources of variability. First, queueing times across servers can be highly variable due to load from different applications. Redundant requests wait in the queue at multiple servers, so they experience the *minimum queueing time* across these servers. Second, the *same* job might see highly variable service times at different servers. For example, in computer systems applications such as web queries, external factors such as network interference, disk seek time, and background tasks can cause a query to be slowed down unpredictably. This slowdown dominates the computation time required for the query, which is inherently quite small. This causes the query's actual service time to be very long relative to its inherent size. For example, a web query can be slowed down by up to a factor of 27 due to unpredictable background load [5]. Sending redundant requests enables a job to receive the *minimum service time* across servers.

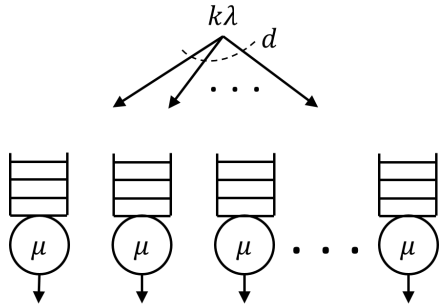

Figure 1: The Redundancy-d system consists of k servers, each providing independent exponential service times with rate μ. Jobs arrive to the system as a Poisson process with rate $k\lambda$. Each job sends copies to d servers chosen uniformly at random. A job is complete when its first copy completes service.

While it is clear that redundancy can lead to a significant reduction in response time, it is often difficult to determine how much redundancy is needed to obtain such improvements. Is sending only two copies enough to achieve most of the potential benefit? What is the additional benefit of increasing the number of copies?

We study these questions using a theoretical model called the Redundancy-**d** system (see Figure 1). The Redundancy-**d** system consists of k servers; each arriving job makes **d** copies of itself and dispatches these copies to **d** different servers chosen uniformly at random. The job is considered complete as soon as the first copy completes service.

Our *primary contribution* is providing the first exact analysis of response time in the Redundancy-**d** system. First, we derive exact closed-form expressions for mean response time as a function of the number of servers k and the number of copies per job, **d**, by modeling the system as a Markov chain. Second, we consider the system in the limit as the number of servers k approaches infinity. Under a standard asymptotic independence assumption, we derive an asymptotically exact expression for the distribution of response time. Our

exact analysis allows us to quantify the magnitude of the benefit from increasing $\mathbf{d}$.

2. ANALYSIS

Let $\rho = \frac{\lambda}{\mu}$ denote the system load. This is the total arrival rate to the system ($k\lambda$) divided by the maximum service rate of the system ($k\mu$). The system is stable as long as $\rho < 1$.

2.1 Markov Chain Approach

We begin by deriving an exact expression for mean response time in the Redundancy-$\mathbf{d}$ system as a function of k, $\mathbf{d}$, λ, and μ.

THEOREM 1. *The mean response time in the Redundancy-$\mathbf{d}$ system with k servers is*

$$\mathrm{E}[T] = \sum_{i=\mathbf{d}}^{k} \frac{1}{k\mu \frac{\binom{k-1}{\mathbf{d}-1}}{\binom{i-1}{\mathbf{d}-1}} - k\lambda}. \tag{1}$$

Our approach to proving Theorem 1 involves modeling the system as a Markov chain. We define a job's *class* as the set of $\mathbf{d}$ particular servers to which the job sends copies. There are $\binom{k}{\mathbf{d}}$ possible classes; all classes are equally likely since each job chooses its servers uniformly at random. Following [2], our system state is a list of all jobs in the system in the order in which they arrived, where we track the class of each job. We obtain the limiting distribution of the state space as an immediate consequence of Theorem 1 in [2].

THEOREM 2. *In the Redundancy-$\mathbf{d}$ system, the limiting probability of being in state $(c_m,\ c_{m-1}, \ldots,\ c_1)$ is*

$$\pi_{(c_m, c_{m-1}, \ldots, c_1)} = \mathcal{C} \prod_{j=1}^{m} \frac{\lambda_{\text{class}}}{|S_j|\mu}, \tag{2}$$

where c_j is the class of the job in position j in the queue, S_j is the set of all servers working on jobs $1, \ldots, j$ ($|S_j|$ is the number of servers in this set), and $\mathcal{C}$ is a normalizing constant.

One might think that $\mathrm{E}[T]$ follows immediately from the limiting distribution of the state space. Unfortunately, this is not the case because to find mean response time we must first find $\pi_m = \Pr\{m \text{ jobs in system}\}$. This requires summing over all $\binom{k}{\mathbf{d}}$ possible classes for each queue position j, $1 \le j \le m$. This is not straightforward because the limiting probabilities depend on the order of all jobs in the system as well as their classes.

The key observation that helps us aggregate states is that we do not actually need to keep track of the specific class in each position in the queue. Instead, it is sufficient to track the number of servers that are busy working on the first j jobs in the queue. We leverage this observation by collapsing our state space so that instead of $\binom{k}{\mathbf{d}}$ possible classes for each position in the queue, we now have at most $k - \mathbf{d}$ possible numbers of servers busy. We define $P(i, m)$ to be the limiting probability that there are m jobs in the system and i busy servers, up to a normalizing constant $\mathcal{C}$ that ensures that the limiting probabilities sum to 1. To find π_m, we need to find $P(i, m)$ for all $\mathbf{d} \le i \le k$.

At a high level, our approach takes the following steps:

1. Write recurrences for $P(i, m)$, the limiting probability that there are i busy servers and m jobs in the system.
2. Derive the normalizing constant, $\mathcal{C}$.
3. Use generating functions to obtain closed-form expressions for $\mathrm{E}[N] = \sum_{m=0}^{\infty} \pi_m$ and $\mathrm{E}[T] = \frac{\mathrm{E}[N]}{k\lambda}$.

The details of the proof can be found in [3].

2.2 Large System Limit Approach

In Section 2.1 we derived exact expressions for mean response time in the Redundancy-$\mathbf{d}$ system using a Markov chain approach. Unfortunately, this approach does not give us the distribution of response time. Even though we obtain the full distribution of the number of jobs in the system, we cannot apply Distributional Little's Law because jobs do not necessarily leave the system in the order in which they arrive. In this section we consider an alternative approach to analyzing the Redundancy-$\mathbf{d}$ system that yields closed-form expressions for the distribution of response time, which are exact under an asymptotic independence assumption as the number of servers $k \to \infty$.

THEOREM 3. *As $k \to \infty$, the response time in the Redundancy-$\mathbf{d}$ system with $\mathbf{d} > 1$ has c.c.d.f.*

$$\Pr\{T > t\} = \bar{F}_T(t) = \left(\frac{1}{\rho + (1-\rho)e^{t\mu(\mathbf{d}-1)}} \right)^{\frac{\mathbf{d}}{\mathbf{d}-1}}, \tag{3}$$

assuming queues are $\mathbf{d}$-wise asymptotically independent.

Theorem 3 relies on the assumption that the queues are asymptotically independent. To understand what we mean by asymptotic independence, first define a job's *non-redundant response time on a server i*, T_i, to be the response time that the job would experience if it arrived to the Redundancy-$\mathbf{d}$ system and sent only one copy to a randomly chosen server i. The queues are $\mathbf{d}$-wise asymptotically independent if knowing a job's non-redundant response time on servers $i_1, \ldots, i_{\mathbf{d}-1}$ does not tell us anything about the job's non-redundant response time on server $i_{\mathbf{d}}$. The analogue of this assumption has been proved in a wide range of settings [4, 6, 1]. Unfortunately, the proofs presented in the above work do not extend easily to the Redundancy-$\mathbf{d}$ system. We conjecture that the asymptotic independence assumption holds in the Redundancy-$\mathbf{d}$ system and leave the proof of this conjecture open for future work.

To prove Theorem 3, we consider a tagged arrival to the Redundancy-$\mathbf{d}$ system, which we assume without loss of generality arrived at time 0 to a system that is stationary. Our goal is to find the probability that this tagged job is still in the system at time $t > 0$. This is simply the probability that the job's non-redundant response time exceeds t on all $\mathbf{d}$ of its servers; using our asymptotic independence assumption, this is $\Pr\{T_i > t\}^{\mathbf{d}}$.

To understand the probability that a job's non-redundant response time on server i exceeds t, observe that there are two ways in which the tagged arrival could have not completed service at server i by time t. First, the tagged job could have size larger than t at server i. Second, even if the tagged job has size $S_i < t$, it will not complete at server i by time t if it does not enter service by time $t - S_i$, that is, if its non-redundant queueing time at server i, T_i^Q, exceeds $t - S_i$. This argument allows us to write an integral equation for $\bar{F}_{T_i}(t)$ in terms of $\bar{F}_{T_i^Q}(t)$:

$$\bar{F}_{T_i}(t) = e^{-\mu t} + \int_0^t \mu e^{-\mu(t-y)} \cdot \bar{F}_{T_i^Q}(y) dy.$$

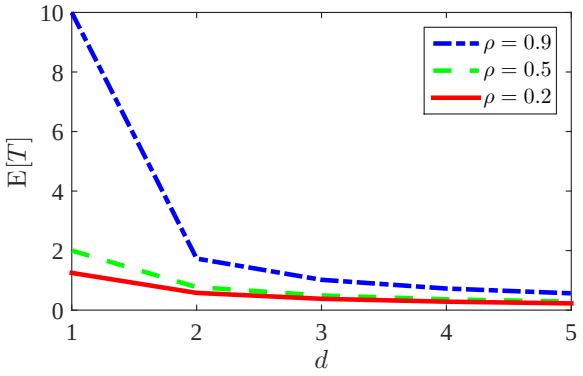

Figure 2: Mean response time $\mathrm{E}[T]$ as a function of $\mathbf{d}$ under low ($\rho = 0.2$, solid red line), medium ($\rho = 0.5$, dashed green line), and high ($\rho = 0.9$, dot-dashed blue line) load.

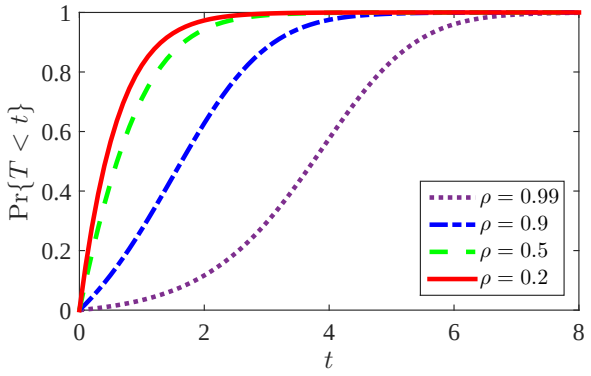

Figure 3: Probability that response time $T < t$ when $\mathbf{d} = 2$ under low ($\rho = 0.2$, solid red line), medium ($\rho = 0.5$, dashed green line), high ($\rho = 0.9$, dot-dashed blue line), and very high ($\rho = 0.99$, dotted purple line) load.

Next we need to understand $\bar{F}_{T_i^Q}(t)$, the probability that the tagged job has not entered service at server i by time t (assuming the tagged job has no other copies). To do this, we look back in time to the most recent arrival to server i before the tagged job arrived. Call this most recent arrival job A. Suppose job A arrived at time $t - Y < 0$. The tagged job will not enter service by time t if and only if either

1. Job A cannot have entered service at server i by time t because there is still some other job ahead of it. This is equivalent to saying that for job A, $T_i^Q > Y$, recalling that T_i^Q is the time that job A would spend in the queue at server i if it had no other copies.

2. Job A is in service at server i at time t. That is, job A has not departed from server i or from any of its other $\mathbf{d} - 1$ servers by time t.

This reasoning yields the following equation for $\bar{F}_{T_i^Q}(T)$:

$$\bar{F}_{T_i^Q}(t) = \int_t^\infty \lambda \mathbf{d} e^{\lambda \mathbf{d}(t-y)} \left(\bar{F}_{T_i^Q}(y) + (\bar{F}_{T_i}(y) - \bar{F}_{T_i^Q}(y)) \bar{F}_{T_i}(y)^{\mathbf{d}-1} \right) dy.$$

Solving the system of two integral equations yields the closed-form expression for the probability that response time exceeds t given in Theorem 3.

3. RESULTS

Our exact analysis allows us to quantify the response time benefit obtained from increasing $\mathbf{d}$. We assume that k is large and thus leverage our asymptotic results from Section 2.2. Throughout this section we assume the service rate at every server is $\mu = 1$.

Figure 2 shows mean response time $\mathrm{E}[T]$ in the Redundancy-$\mathbf{d}$ system as a function of $\mathbf{d}$ for low, medium, and high load. Mean response time decreases as $\mathbf{d}$ increases, with this benefit being greatest under higher loads. This is because redundancy reduces mean response time by taking advantage of two sources of system variability: queueing time variability and service time variability. When load is low, queueing times are low so the primary benefit of redundancy comes from a job receiving the minimum service time on $\mathbf{d}$ servers. When load is high, queueing times are typically higher so there is more opportunity to reduce response time by reducing queueing time as well as service time. At all loads, the most significant improvement occurs between $\mathbf{d} = 1$ and $\mathbf{d} = 2$. This improvement ranges from a factor of 2 at $\rho = 0.2$ to a factor of 6 at $\rho = 0.9$. From Theorem 3, we see that as $\mathbf{d}$ becomes large, mean response time scales in proportion to $\frac{1}{\mathbf{d}}$, indicating that there is decreasing marginal benefit from further increasing $\mathbf{d}$.

Thus far we have discussed only the mean response time; however our asymptotic analysis provides the full response time distribution. Figure 3 shows that the response time improvement is even bigger at the tail of response time than at the mean; the 95th percentile of response time decreases by up to a factor of 8 when $\rho = 0.9$.

4. REFERENCES

[1] M. Bramson, Y. Lu, and B. Prabhakar. Asymptotic independence of queues under randomized load balancing. *Queueing Systems*, 71(3):247–292, 2012.

[2] K. Gardner, S. Zbarsky, S. Doroudi, M. Harchol-Balter, E. Hyytiä, and A. Scheller-Wolf. Reducing latency via redundant requests: Exact analysis, June 2015.

[3] K. Gardner, S. Zbarsky, M. Harchol-Balter, and A. Scheller-Wolf. Analyzing response time in the Redundancy-d system. Technical report, CMU-CS-15-141R, 2016.

[4] N. Vvedenskaya, R. Dobrushin, and F. Karpelevich. Queueing system with selection of the shortest of two queues: An asymptotic approach. *Probl. Peredachi Inf.*, 32(1):20–34, 1996.

[5] Y. Xu, M. Bailey, B. Noble, and F. Jahanian. Small is better: Avoiding latency traps in virtualized data centers. In *Proceedings of the 4th annual Symposium on Cloud Computing*, page 7. ACM, 2013.

[6] L. Ying, R. Srikant, and X. Kang. The power of slightly more than one sample in randomized load balancing. In *Proc. of IEEE INFOCOM*, 2015.