2023 International Conference on Wireless Communications and Signal Processing (WCSP) | 979-8-3503-2466-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/WCSP58612.2023.10404519

Computing Redundancy in Blocking Systems: Fast Service or No Service

Pei Peng
Nanjing University of Posts and Telecommunications
Nanjing, China
Email: pei.peng@njupt.edu.cn

Emina Soljanin
Rutgers, The State University of New Jersey
Piscataway, USA
Email: emina.soljanin@rutgers.edu

***Abstract*—Redundancy in distributed computing systems reduces job completion time. It is widely employed in practice and studied in theory for queuing systems, often in a low-traffic regime where queues remain empty. Motivated by emerging edge systems, this paper initiates a study of using redundancy in blocking systems. Edge nodes often operate in highly unpredictable environments, and replicating job execution improves the job mean execution time. However, directing more resources to some computing jobs will block the execution of others. We evaluate the system performance using two metrics: job computing time and job blocking probability. We show that the job computing time decreases with increasing replication factor, but the job blocking probability does not. Therefore, there is a tradeoff. Interestingly, some minimal replication significantly reduces computing time with almost no blocking probability change. This paper proposes the system service rate as a new combined metric to evaluate the tradeoff and a single system's performance indicator. We conclude that the optimal number replication factor that maximizes the system service rate changes with the distribution parameters and the arrival rate.**

***Index Terms*—Edge computing, redundancy management, blocking system, service rate.**

I. Introduction

Distributed computing systems implement redundancy to mitigate the adverse effect of straggling job execution [1]. There is a large body of work on computing with redundancy. It includes system performance evaluation, distributed code design, secure computing, and analysis of various tradeoffs, e.g., [2]–[4] and references therein. There is also recent literature on using redundancy in edge computing systems [5]–[7]. With rare exceptions [8], this literature assumes queuing systems, most often in a low-traffic regime where queues remain empty. Motivated predominantly by emerging edge systems, this paper initiates a study of using replication in blocking systems.

Edge computing is an inter-layer between the cloud and the end-user. It provides storage and computing infrastructure at nodes located one or two network hops from the end-user [9], [10], thus avoiding communication delay. However, edge systems operate on much smaller storage and computing resources than the cloud [11]. Therefore an edge system has to send computing requests to the cloud when all edge workers are busy, and communication time becomes a significant part of the delay. This paper addresses two edge computing goals: 1) to increase the computing speed of each job taken for execution and 2) to reduce the probability of having to send a job to the cloud for execution.

Using resources redundantly can decrease the mean computing time, our first goal. However, with the higher resource usage, and thus lower availability, the system will send more jobs to the cloud for execution, which goes against our second goal to execute more jobs locally. An edge computing system that sends jobs to the cloud once the resources become unavailable acts locally as a blocking system.

When a job arrives, the edge system replicates it to several workers. We aim to find the optimal replication factor that minimizes the job computing time and the job blocking probability. Simultaneously minimizing the job computing time and job blocking probability is generally impossible. Since both performance metrics are crucial to edge computing, we need to find the tradeoff between these two metrics. Recent literature has considered the service rate a critical performance metric, especially in distributed storage systems. For example, [12] evaluates the service rate of a distributed storage system to find an optimal storage allocation strategy, and [13] uses the service rate region as an essential consideration in the design of erasure-coded distributed systems. This paper proposes the system service rate as a new metric that combines the two classical metrics: computing time and blocking probability.

The paper's organization is as follows. In Sec. II, we describe the system model, introduce the system service rate metric, and formulate the problems. In Sec. III, we consider the exponential service time and find the optimal number of groups that minimizes the job blocking probability or maximizes the system service rate. In Sec. IV, we consider the shifted exponential service time and analyze the optimal number of groups under different shift parameters and the job arrival rate. The conclusions are given in Sec. V.

II. System Model and Problem Formulation

System Architecture and Operation: We consider a system, as shown in Fig. 1, with a single front-end controller node and multiple computing servers, which we refer to as workers. The controller node manages the entire computing cluster of nodes. Such architecture is commonly implemented in modern frameworks, such as Apache Mesos, and edge computing systems with limited storage and computing resources [14].

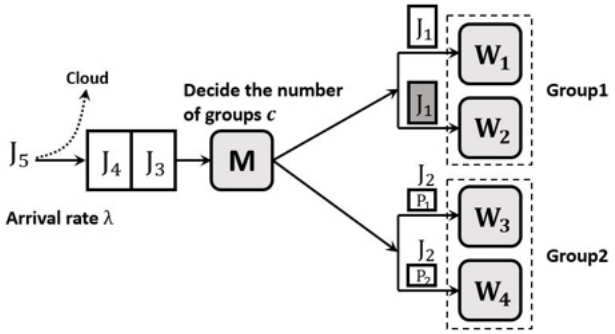

Fig. 1. **System Model:** Combination of the blocking system and the distributed computing system.

The controller node creates m copies for each arriving job and assigns each copy to a worker in a group. In Fig. 1, the controller node sends the J_1 and its copy to workers 1 and 2 ($m = 2$), and sends the J_2 and its copy to workers 3 and 4 ($m = 2$). This execution replication mitigates straggling and reduces the job's expected completion time. The larger the replication factor m, the higher the reduction of the job's average completion time.

On the other hand, when all workers are busy, the new request for job execution gets blocked. An edge computing system may send such jobs to the cloud (see Fig. 1, job J_5). There is a significant communication delay between the edge and the cloud, which may be much longer than the expected job computing time. Therefore, we want the system to serve more jobs and send fewer to the cloud.

Traditional Performance Metrics: There are two performance metrics of interest in the described system: 1) the job computing time T_{job} and 2) the job blocking probability P_b. The system's goal is to minimize both of these numbers. The design parameters for a given system size (fixed number of workers) are the replication factor m and the number of groups c. Increasing m and decreasing c reduce T_{job} (improves the first performance metric). However, the effect of m and c on P_b is not immediately apparent. Increasing m will temporarily occupy more servers per job, but it will also make the jobs stay in the system for a shorter time.

Service Rate as a Performance Metric: In general, there may not be an optimal replication factor m or the number of groups c that simultaneously minimizes both $\mathbb{E}[T_{\text{job}}]$ and P_b. We combine the above two metrics to get a new metric we refer to as the *system service rate* and define it as follows:

$$\sigma = \frac{1 - P_b}{\mathbb{E}[T_{\text{job}}]}. \tag{1}$$

We separately analyze the optimal number of groups c that maximizes σ changes with the computing time distribution parameters.

Job Arrival and Computing Time: We assume that the job arrivals follow a Poisson process with a rate λ, which allows us to model the blocking system as an M/G/c/K queue. Analyzing the blocking probability of an M/G/c/K queue that models the blocking operation of the system in Fig. 1 is a highly complex problem. To better understand it, we consider the Erlang B model, the M/M/c/c queue, where the queue length equals the number of groups.

For each worker, we first assume that the computing time follows the exponential distribution $\text{Exp}(\mu)$, which is a straightforward model and widely used in distributed computing. We will use this distribution to explain our system model. Later, we consider that the computing time follows a more general distribution, the shifted exponential distribution $\text{S-Exp}(\Delta, \mu)$, which is also widely used in distributed computing [2], [4], [15]. Here, Δ is an initial handshake time, after which the worker will complete the job in some $\text{Exp}(\mu)$ time.

Problem formulation: We consider a system in Fig. 1 with a single controller node and N workers. There are c size m groups of workers, giving $N = c \cdot m$. (We assume that c, N and m are integers.) The controller assigns a replica of each arriving job to one of the m servers in an available group of workers. The job is blocked or forwarded to the cloud if no such group is available.

Our goal is to evaluate two performance metrics, the expected job computing time $\mathbb{E}[T_{\text{job}}]$ and the job blocking probability P_b for systems with Poisson job arrivals and shifted exponential service time, as described above. We compute the replication factor m that minimizes $\mathbb{E}[T_{\text{job}}]$ and the replication factor m that minimizes P_b. We show how we can achieve the desired tradeoff between these two traditional metrics by selecting an appropriate m. We also aim to compute the proposed service rate metric that combines the two traditional performance metrics, see (1). We find the optimal number of groups c that maximizes the service rate.

Parameters and Notation:

N –	number of workers in the system
m –	replication factor, number of workers in a group
c –	number of groups, $c = N/m$
T_{job} –	job computing time in the system
P_b –	job blocking probability
σ –	system service rate
λ –	job arrival rate
Δ –	shift parameter of S-Exp distribution
μ –	rate parameter of S-Exp or Exp distribution

III. Exponential Service Time

The job computing time is defined to measure how much time the job spends in the system occupying resources. Here, we consider the worker computing time follows $\text{Exp}(\mu)$. Since the blocking system uses m-fold replication, the expected job computing time is given by

$$\mathbb{E}[T_{\text{job}}] = \frac{1}{m\mu}. \tag{2}$$

Observe that $\mathbb{E}[T_{\text{job}}]$ reaches its minimum at $m = 1$. Moreover, T_{job} follows the exponential distribution with the rate parameter $m\mu$.

A. Job Blocking Probability

Here, we consider the Erlang B model, the M/M/c/c queue, where the queue length equals the number of groups. For a

blocking system with c groups and the job arrives as a Poisson process with the rate λ, the job blocking probability is

$$P_b = \frac{(\rho)^c/c!}{\sum_{j=0}^{c}(\rho)^j/j!} \tag{3}$$

where $\rho = \lambda\,\mathbb{E}[T_{\text{job}}]$. The above expression shows that for a given c, P_b increases with ρ. From (2), we know that $\mathbb{E}[T_{\text{job}}]$ is a function of m, in which $m = \frac{N}{c}$. Then we will take $K = \frac{\lambda}{N\mu}$. Thus, P_b is a function of c and we can rewrite (3) in the following,

$$P_b(c) = \frac{(Kc)^c/c!}{\sum_{j=0}^{c}(Kc)^j/j!} \tag{4}$$

We find the optimal c that minimizes the job blocking probability P_b in Theorem 1.

Theorem 1. *For the blocking system with Poisson(λ) arrivals and* Exp($m\mu$) *computing time, the job blocking probability P_b increases with the number of groups c and reaches the minimum at $c = N$ (i.e., $m = 1$).*

Proof. Assume $c_1 > c_0$, from (4), the blocking probability is

$$\begin{aligned} P_b(c_1) &= \frac{(Kc_1)^{c_1}/c_1!}{\sum_{j=0}^{c_1}(Kc_1)^j/j!} = \frac{1}{\sum_{j=0}^{c_1}\frac{c_1!/j!}{(Kc_1)^{c_1-j}}} \\ &= \frac{1}{\sum_{j=0}^{c_1}\prod_{i=1}^{j}\frac{c_1-j+i}{Kc_1}} < \frac{1}{\sum_{j=0}^{c_1}\prod_{i=1}^{j}\frac{c_0-j+i}{Kc_0}} \\ &< \frac{1}{\sum_{j=0}^{c_0}\frac{c_0!/(c_0-j)!}{(Kc_0)^j}} = \frac{(Kc_0)^{c_0}/c_0!}{\sum_{j=0}^{c_0}(Kc_0)^j/j!} = P_b(c_0) \end{aligned}$$

Since $c \in [1, N]$, P_b incraeses with the number of groups c and reaches its minimum at $c = N$.

1) Numerical Analysis: We evaluate (2) and (4) for $\mathbb{E}[T_{\text{job}}]$ vs. c and P_b vs. c. We consider a system with $N = 24$ workers, the job arrives following the Poisson distribution with the rate $\lambda = 1$, and the computing time for each worker follows the exponential with $\mu = 0.1$. The results are plotted

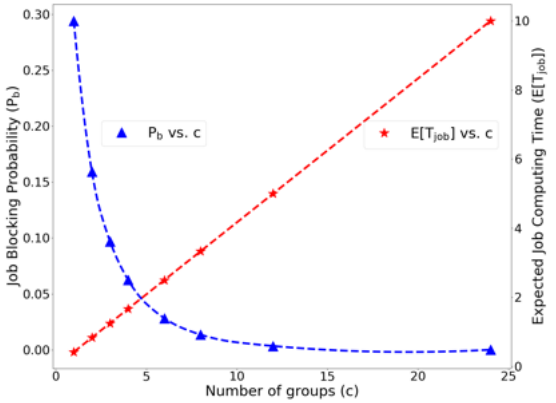

Fig. 2. Expected job computing time $\mathbb{E}[T_{\text{job}}]$ and job blocking probability P_b as a function of c.

in Fig. 2. The figure shows that $\mathbb{E}[T_{\text{job}}]$ increases linearly with c, which means that introducing more redundancy provides higher computing speed. Meanwhile, the figure also shows that P_b decreases with increasing c, which means that introducing more redundancy leads to more jobs being dropped. We also observe that P_b is close to 0 and decreases slowly when c is large; when $c \leq 6$, P_b decreases sharply. These observations are consistent with the theoretical analysis of the optimal number of groups. Considering both job computing time and job blocking probability, we find that each metric requires a very different optimal c.

B. System Service Rate

From the above analysis, we conclude that it is impossible to minimize the job computing time and blocking probability simultaneously. Interestingly, some minimal replication significantly reduces the computing time with almost no blocking probability change. Therefore, we adopt the system service rate, introduced in Sec. II, as a single performance indicator describing the tradeoff between the job computing time and the job blocking probability. Since the job completion distribution with m-fold replication is Exp($m\mu$) and $m = \frac{N}{c}$, the system service rate is

$$\sigma(c) = \frac{(1-P_b)}{\mathbb{E}[T_{\text{job}}]} = \frac{N\mu\sum_{j=0}^{c-1}(Kc)^j/j!}{c\sum_{j=0}^{c}(Kc)^j/j!} \tag{5}$$

The following theorem gives the optimal number of groups c.

Theorem 2. *For the blocking system with Poisson(λ) arrivals and* Exp($m\mu$) *computing time, the system service rate σ reaches the maximum at $c = 1$ (i.e., $m = N$).*

Proof. From (5), we know that $\sigma(1) = \frac{N\mu}{1+K}$. Assume that $c_1 \geq 2$,

$$\begin{aligned} \frac{\sigma(c_1)}{\sigma(1)} &= \frac{(1+K)\sum_{j=0}^{c_1-1}(Kc_1)^j/j!}{c_1\sum_{j=0}^{c_1}(Kc_1)^j/j!} = \frac{1+(Kc_1)^{c_1}/c_1!}{c_1\sum_{j=0}^{c_1}(Kc_1)^j/j!} \\ &+ \frac{\sum_{j=1}^{c_1-1}[(Kc_1)^j/j! + K(Kc_1)^{j-1}/(j-1)!]}{c_1\sum_{j=0}^{c_1}(Kc_1)^j/j!} \end{aligned}$$

First, we consider the left-hand side of the above formula. Since $c_1 \geq 2$ we have $1 + (Kc_1)_1^c/c_1! < c_1 + c_1(Kc_1)_1^c/c_1!$. Second, we consider the right-hand side of $\frac{\sigma(c_1)}{\sigma(1)}$. Since $j \leq c_1 - 1$, we have

$$\frac{(Kc_1)^j}{j!} + \frac{K(Kc_1)^{j-1}}{(j-1)!} < \frac{2(Kc_1)^j}{j!} \leq \frac{c_1(Kc_1)^j}{j!}.$$

Then we have,

$$\begin{aligned} \frac{\sigma(c_1)}{\sigma(1)} &< \frac{c_1 + c_1(Kc_1)^{c_1}/c_1! + \sum_{j=1}^{c_1-1}c_1(Kc_1)^j}{c_1\sum_{j=0}^{c_1}(Kc_1)^j/j!} \\ &= \frac{c_1\sum_{j=0}^{c_1}(Kc_1)^j/j!}{c_1\sum_{j=0}^{c_1}(Kc_1)^j/j!} = 1 \end{aligned}$$

Therefore, we have the result that the system service rate σ reaches the maximum at $c = 1$.

1) Numerical Analysis: We evaluate (5) for different values of $\lambda \in \{0.01, 1, 10\}$. Since it is easy to know that σ decreases with the increasing λ according to (3) and (5). We plot the normalized σ vs. c in Fig. 3 to see the changes in the system service rate. We consider a system with $N = 24$ workers, the job arrives following the Poisson distribution with rate $\lambda = 1$, and the computing time for each worker follows the exponential with $\mu = 0.1$. The figure shows that the system

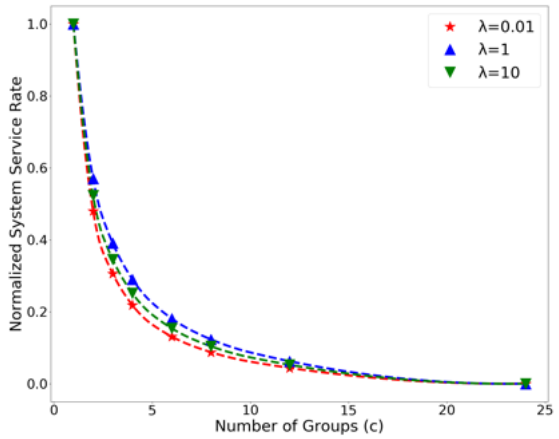

Fig. 3. Normalized system service rate $\frac{\sigma-\sigma_{\min}}{\sigma_{\max}-\sigma_{\min}}$ for exponential computing time as a function of the number of groups c.

service rate σ always decreases with the increasing c and reaches the maximum at $c = 1$. This observation is consistent with the conclusion of Theorem 2. When comparing the curves $\lambda = 0.01$ and $\lambda = 1$, we know that σ decreases relatively smoother when the arrival rate is larger. When the arrival rate is sufficiently large (e.g., $\lambda = 10$), σ decreases sharply.

IV. Shifted Exponential Service Time

In this section, we consider the shifted exponential as a more general service time distribution. Since the system uses m-fold replication and the worker computing time follows S-Exp(Δ, μ) , the expected job computing time is given by

$$\mathbb{E}[T_{\text{job}}] = \Delta + \frac{1}{m\mu}. \tag{6}$$

Observe that $\mathbb{E}[T_{\text{job}}]$ reaches its minimum at $m = 1$. Moreover, T_{job} follows the shifted exponential distribution with the shift Δ and the rate parameter $m\mu$. See, e.g., [16].

A. Job Blocking Probability

We also consider the Erlang B model. For a blocking system with c groups and the job arrives as a Poisson process with the rate λ, we adopt the expression of the job blocking probability in Eq. (3). Since $\rho = \lambda(\Delta + \frac{1}{m\mu})$ and $m = \frac{N}{c}$. Then we can rewrite (3) in the following,

$$P_b(c) = \frac{(K(\Delta N\mu + c))^c/c!}{\sum_{j=0}^{c}(K(\Delta N\mu + c))^j/j!} \tag{7}$$

where $K = \frac{\lambda}{N\mu}$ is a constant. According to Theorem 1, we have the following lemma that finds the optimal c that minimizes the job blocking probability P_b.

Lemma 1. *For the blocking system with Poisson(λ) arrivals and* S-Exp$(\Delta, m\mu)$ *computing time, the job blocking probability P_b increases with the number of groups c and reaches the minimum at $c = N$ (i.e., $m = 1$).*

Proof. Similar to the proof of Theorem 1, assume $c_1 > c_0$, from (7), the blocking probability is

$$\begin{aligned} P_b(c_1) &= \frac{1}{\sum_{j=0}^{c_1}(\prod_{i=1}^{j}\frac{c_1-j+i}{c_1+\Delta N\mu})\frac{1}{K^j}} \\ &< \frac{1}{\sum_{j=0}^{c_1}(\prod_{i=1}^{j}\frac{c_0-j+i}{c_0+\Delta N\mu})\frac{1}{K^j}} < \frac{1}{\sum_{j=0}^{c_0}\frac{c_0!/(c_0-j)!}{(K(\Delta N\mu+c_0))^j}} \\ &= \frac{(K(\Delta N\mu + c_0))_0^c/c_0!}{\sum_{j=0}^{c_0}(K(\Delta N\mu + c_0))^j/j!} = P_b(c_0) \end{aligned}$$

Since $c \in [1, N]$, P_b incraeses with the number of groups c and reaches its minimum at $c = N$.

Numerical Analysis: We evaluate (6) and (7) for $\mathbb{E}[T_{\text{job}}]$ vs. c and P_b vs. c. We consider a system with $N = 24$ workers, the job arrives following the Poisson distribution with the rate $\lambda = 1$, and the computing time for each worker follows the shifted exponential with $\Delta = 5$ and $\mu = 0.1$. The results are plotted in

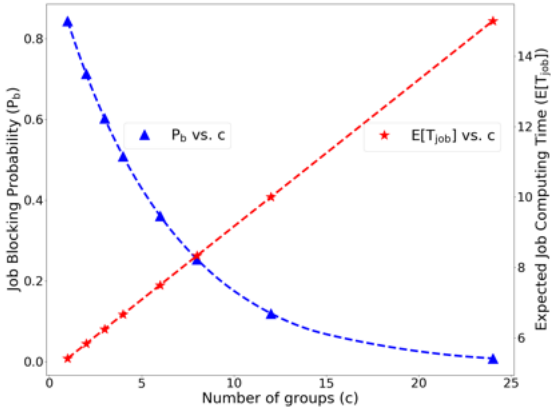

Fig. 4. Expected job computing time $\mathbb{E}[T_{\text{job}}]$ and job blocking probability P_b as a function of the number of groups c.

Fig. 4. We observe that $\mathbb{E}[T_{\text{job}}]$ increases linearly with c and P_b decreases with increasing c. We also observe that P_b is close to 0 only when c is large enough, and it does not decrease sharply compared to the results in Fig.2. The figure clearly shows that it is impossible to minimize the job computing time and the job blocking probability simultaneously. Considering both job computing time and job blocking probability, we find that each metric requires a very different optimal c. Interestingly, the job blocking probability may not always decrease with the number of groups for some heavy-tail distributions, e.g., Pareto distribution. We will explore this problem in future work.

B. System Service Rate

For the shifted exponential service time, we also need to use the system service rate to find the tradeoff between the job computing time and the job blocking probability. For a blocking system with N workers and c groups, when

the job completion distribution with m-fold replication is S-Exp$(\Delta, m\mu)$, the expression of the system service rate is

$$\sigma(c) = \frac{N\mu \sum_{j=0}^{c-1}(K(\Delta N\mu + c))^j/j!}{(N\Delta\mu + c)\sum_{j=0}^{c}(K(\Delta N\mu + c))^j/j!}. \quad (8)$$

We know that the optimal c that maximizes the system service rate σ changes with different parameters. However, considering the complexity of (8), optimizing σ under a general shifted exponential service time is difficult. In the following, we separately analyze two important system parameters: the shift parameter and the job arrival rate. From the analysis, we want to know how the optimal c changes with different values of the system parameters.

1) Shift Parameter: We analyze two special (Δ, μ) parameter regions: 1) $\Delta \ll \frac{1}{\mu}$, which makes T_{job} exponentially distributed, and 2) $\Delta \gg \frac{1}{\mu}$, which makes T_{job} equal to constant Δ. Apparently, the first case shows that the random part $\text{Exp}(\mu)$ is much larger than the constant part Δ, and the second case is on the contrary.

In the first case, the job completion distribution with m-fold replication is $\text{Exp}(m\mu)$. According to Theorem 2, the system service rate σ reaches the maximum at $c = 1$. That is, when the random part is much larger than the constant part, it is better to decrease the number of groups to achieve a larger system service time. Next, we analyze the second case where the computing time is approximately constant. Then S-Exp$(\Delta, m\mu)$ is approximated by the constant Δ. The system service rate is $\sigma(c) = \frac{1-P_b(c)}{\Delta}$. According to Theorem 1, $1 - P_b(c)$ increases with c. Therefore, σ increases with c and reaches the maximum at $c = N$.

The above analysis implies that the optimal c that maximizes the system service rate lies between 1 and N for a general value of the shift parameter Δ. When the random part of the shifted exponential service time is larger, the optimal c is smaller; when the constant part becomes large, the optimal c also becomes larger.

2) Job Arrival Rate: The arrival rate is also an important parameter that affects the system service rate. According to the system structure, we may infer that the system service rate changes as follows. When λ is small, few jobs arrive. Even with a large replication factor, the blocking system can serve almost all jobs. Thus, introducing more redundancy may improve the system performance. When λ is large, the system has to handle many jobs concurrently. It is, thus, reasonable to decrease the replication factor to process more jobs in the system. In Theorem 3, we verify the first scenario in which λ is sufficiently small.

Theorem 3. *For the blocking system with Poisson(λ) arrivals and* S-Exp$(\Delta, m\mu)$ *computing time, the system service rate σ always reaches the maximum at $c = 1$ when the arrival rate $\lambda \leq \frac{N\mu}{(\Delta N\mu+1)^2}$.*

Proof. When the replication factor $m = N$, the number of groups $c = \frac{N}{m} = 1$. From Eq.(8), we have $\sigma(1) = \frac{N\mu}{(\Delta N\mu+1)(1+K(\Delta N\mu+1))}$, where $K = \frac{\lambda}{N\mu}$. When $c = c' \geq 2$, we have $\sigma(c') = \frac{N\mu}{(\Delta N\mu+c')}(1 - P_b) \leq \frac{N\mu}{(\Delta N\mu+c')}$.

To satisfy $\sigma(c') \leq \sigma(1)$, we need

$$\frac{N\mu}{(\Delta N\mu + c')} \leq \frac{N\mu}{(\Delta N\mu + 1)(1 + K(\Delta N\mu + 1))}$$

That is,

$$K \leq \frac{c' - 1}{(\Delta N\mu + 1)^2} \Leftrightarrow \lambda \leq \frac{N\mu(c' - 1)}{(\Delta N\mu + 1)^2}$$

Since $c' \geq 2$, the right-hand side reaches the minimum at $c' = 2$. Therefore, $\sigma(c') \leq \sigma(1)$ holds for any $\lambda \leq \frac{N\mu}{(\Delta N\mu+1)^2}$. □

Next, we consider the scenario where λ is sufficiently large. Using a method similar to the proof of Theorem 3, we find the optimal c in the following lemma.

Lemma 2. *When the arrival rate $\lambda > \frac{2N\mu}{(\Delta N\mu)^2-2}$ (where $\Delta N\mu > \sqrt{2}$), the system service rate σ always reaches the maximum at $c \geq 2$.*

Proof. From Eq.(8), we have $\sigma(1) = \frac{N\mu}{(\Delta N\mu+1)(1+K(\Delta N\mu+1))}$ and $\sigma(2) = \frac{N\mu(1+\alpha)}{(\Delta N\mu+2)(1+\alpha+\alpha^2/2)}$, where $K = \frac{\lambda}{N\mu}$ and $\alpha = K(\Delta N\mu + 2)$.

For $\sigma(2)$, we have

$$\sigma(2) > \frac{2N\mu(1+\alpha)}{(\Delta N\mu + 2)(2 + 3\alpha + \alpha^2)} = \frac{2N\mu}{(\Delta N\mu + 2)(\alpha + 2)}.$$

To satisfy $\sigma(2) > \sigma(1)$, we need

$$\frac{2N\mu}{(\Delta N\mu + 2)(\alpha + 2)} \geq \frac{N\mu}{(\Delta N\mu + 1)(1 + K(\Delta N\mu + 1))}$$

Then we have,

$$2K(\Delta N\mu + 1)(\Delta N\mu + 1) - (\Delta N\mu + 2)\alpha > 2$$
$$\Leftrightarrow 2K(\Delta N\mu + 1)(\Delta N\mu + 1) - K(\Delta N\mu + 2)(\Delta N\mu + 2) > 2$$
$$\Leftrightarrow K((\mu)^2 - 2) > 2.$$

When $\Delta N\mu > \sqrt{2}$, we have $\lambda > \frac{2N\mu}{(\Delta N\mu)^2-2}$. Then it is clear that $\sigma(2) > \sigma(1)$ holds. □

Generally speaking, decreasing the number of groups and increasing the replication factor provides better performance when the arrival rate is low. When the arrival rate is high, it is better to increase the number of groups. However, when the arrival rate is sufficiently high, the number of groups and the replication factor do not affect the system service rate.

3) Numerical Analysis: We evaluate σ to see how the system service rate changes with c. We consider a system with $N = 24$ workers, the job arrives following the Poisson distribution with the rate $\lambda = 1$, and the computing time for each worker follows the shifted exponential distribution. Since the replication factor $m = \frac{N}{c}$ (Since both m and c are integers, we have $c \in \{1, 2, 3, 4, 6, 8, 12, 24\}$). We assume $\mu = 1$.

In Fig. 5, we evaluate (8) for three different values of $\Delta \in \{0.1, 1, 10\}$. We normalize σ to observe the changes. When $\Delta \ll \mu$, σ reaches the maximum at $c = 1$, which means

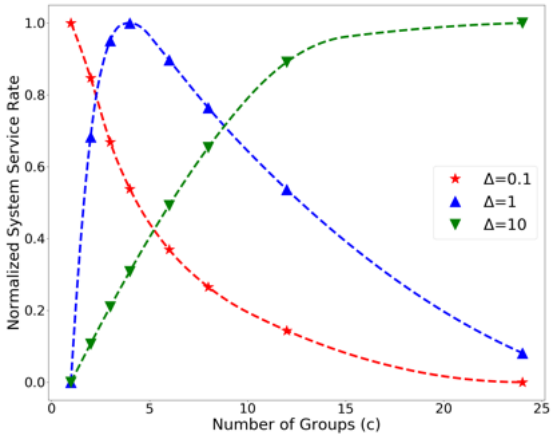

Fig. 5. Normalized system service rate $\frac{\sigma-\sigma_{\min}}{\sigma_{\max}-\sigma_{\min}}$ for shifted exponential computing time as a function of the number of groups c.

introducing more redundancy provides a higher system service rate. When $\Delta \gg \mu$, σ reaches the maximum at $c = 24$, which means that reducing proper redundancy provides a higher system service rate. Otherwise, the optimal c lies between 1 and 24 (the optimal $c = 4$ in Fig. 5).

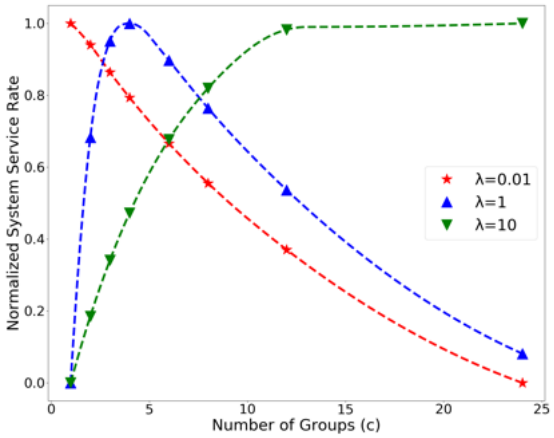

Fig. 6. Normalized system service rate $\frac{\sigma-\sigma_{\min}}{\sigma_{\max}-\sigma_{\min}}$ as a function of the number of groups c.

In Fig. 6, we evaluate (8) for three different values of $\lambda \in \{0.01, 1, 10\}$. We normalize σ to observe the changes. When λ is sufficiently small, σ reaches the maximum at $c = 1$. When λ is sufficiently large, σ reaches the maximum at $c = 24$. Otherwise, the optimal c lies between 1 and 24 (the optimal $c = 4$ in Fig. 6). Therefore, we conclude that the optimal c increases with λ.

V. Conclusions

This paper initiates a study of the value of using replication in blocking systems. We assume a typical computing architecture and computing time probability distribution and look into how the job replication factor affects the expected job completion time and blocking probability. We show that the job computing time decreases with increasing replication factor but so does the job blocking probability. Therefore, there is a trade-off between these two classical metrics. Interestingly, some minimal replication significantly reduces computing time with almost no blocking probability change. This paper also proposes and analyses the system service rate as a new combined metric and a single system's performance indicator.

Acknowledgment

This work was supported in part by the University Science Research Project of Jiangsu Province (Grant No. 23KJB120009), the Natural Science Research Start-up Foundation of Recruiting Talents of Nanjing University of Posts and Telecommunications (Grant No. NY222008) and the NSF-BSF Award FET-2120262.

References

[1] J. Dean and L. A. Barroso, "The tail at scale," *Communications of the ACM*, vol. 56, no. 2, pp. 74–80, 2013.
[2] R. Bitar, P. Parag, and S. El Rouayheb, "Minimizing latency for secure coded computing using secret sharing via staircase codes," *IEEE Trans. on Communications*, 2020.
[3] P. Peng, E. Soljanin, and P. Whiting, "Diversity/parallelism trade-off in distributed systems with redundancy," *IEEE Trans. on Information Theory*, vol. 68, no. 2, pp. 1279–1295, 2021.
[4] S. Dutta, V. Cadambe, and P. Grover, "Short-dot: Computing large linear transforms distributedly using coded short dot products," *Advances in Neural Information Processing Systems*, vol. 29, pp. 2100–2108, 2016.
[5] J. Zhang and O. Simeone, "On model coding for distributed inference and transmission in mobile edge computing systems," *IEEE Communications Letters*, vol. 23, no. 6, pp. 1065–1068, 2019.
[6] J. Wang, C. Cao, J. Wang, K. Lu, A. Jukan, and W. Zhao, "Optimal task allocation and coding design for secure edge computing with heterogeneous edge devices," *IEEE Trans. on Cloud Computing*, 2021.
[7] Y. Han, D. Niyato, C. Leung, C. Miao, and D. I. Kim, "Dynamics in coded edge computing for iot: A fractional evolutionary game approach," *IEEE Internet of Things Journal*, 2022.
[8] U. J. Ferner, M. Médard, and E. Soljanin, "Toward sustainable networking: Storage area networks with network coding," in *50th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2012, Allerton Park & Retreat Center, Monticello, IL, USA, October 1-5, 2012*, 2012, pp. 517–524.
[9] S. Yi, Z. Hao, Z. Qin, and Q. Li, "Fog computing: Platform and applications," in *2015 Third IEEE workshop on hot topics in web systems and technologies (HotWeb)*. IEEE, 2015, pp. 73–78.
[10] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *2016 IEEE Internat. Symp. on Inform. Theory (ISIT)*. IEEE, 2016, pp. 1451–1455.
[11] S. Noghabi, L. Cox, S. Agarwal, and G. Ananthanarayanan, "The emerging landscape of edge-computing," *ACM SIGMOBILE GetMobile*, March 2020. [Online]. Available: https://www.microsoft.com/en-us/research/publication/the-emerging-landscape-of-edge-computing/
[12] P. Peng, M. Noori, and E. Soljanin, "Distributed storage allocations for optimal service rates," *IEEE Trans. on Communications*, vol. 69, no. 10, pp. 6647–6660, 2021.
[13] M. Aktaş, G. Joshi, S. Kadhe, F. Kazemi, and E. Soljanin, "Service rate region: A new aspect of coded distributed system design," *IEEE Trans. on Information Theory*, vol. 67, no. 12, pp. 7940–7963, 2021.
[14] C. Mouradian, D. Naboulsi, S. Yangui, R. H. Glitho, M. J. Morrow, and P. A. Polakos, "A comprehensive survey on fog computing: State-of-the-art and research challenges," *IEEE communications surveys & tutorials*, vol. 20, no. 1, pp. 416–464, 2017.
[15] G. Joshi, E. Soljanin, and G. W. Wornell, "Efficient redundancy techniques for latency reduction in cloud systems," *TOMPECS*, vol. 2, no. 2, pp. 12:1–12:30, 2017.
[16] P. Peng, E. Soljanin, and P. Whiting, "Diversity vs. parallelism in distributed computing with redundancy," in *IEEE International Symposium on Information Theory, ISIT 2020, Los Angeles, CA, USA, June 21-26, 2020*, pp. 257–262.