Redundancy scheduling with scaled Bernoulli service requirements

Queueing Systems (2019) 93:67–82
https://doi.org/10.1007/s11134-019-09621-2

Redundancy scheduling with scaled Bernoulli service requirements

Youri Raaijmakers[1] · **Sem Borst**[1,2] · **Onno Boxma**[1]

Received: 15 November 2018 / Revised: 3 June 2019 / Published online: 22 June 2019

Abstract

Redundancy scheduling has emerged as a powerful strategy for improving response times in parallel-server systems. The key feature in redundancy scheduling is replication of a job upon arrival by dispatching replicas to different servers. Redundant copies are abandoned as soon as the first of these replicas finishes service. By creating multiple service opportunities, redundancy scheduling increases the chance of a fast response from a server that is quick to provide service and mitigates the risk of a long delay incurred when a single selected server turns out to be slow. The diversity enabled by redundant requests has been found to strongly improve the response time performance, especially in the case of highly variable service requirements. Analytical results for redundancy scheduling are unfortunately scarce however, and even the stability condition has largely remained elusive so far, except for exponentially distributed service requirements. In order to gain further insight in the role of the service requirement distribution, we explore the behavior of redundancy scheduling for scaled Bernoulli service requirements. We establish a sufficient stability condition for generally distributed service requirements, and we show that, for scaled Bernoulli service requirements, this condition is also asymptotically nearly necessary. This stability condition differs drastically from the exponential case, indicating that the stability condition depends on the service requirements in a sensitive and intricate manner.

Keywords Queueing · Redundancy · Parallel-server systems · Dispatching · Scaled Bernoulli service requirements · Stability condition

Mathematics Subject Classification 68M20 · 60K25 · 90B22

This research was partly funded by the NWO Gravitation Programme NETWORKS, Grant Number 024.002.003.

✉ Youri Raaijmakers
y.raaijmakers@tue.nl

[1] Department of Mathematics and Computer Science, Eindhoven University of Technology, P.O. Box 513, 5600, MB Eindhoven, The Netherlands

[2] Nokia Bell Labs, Murray Hill, NJ 07974, USA

1 Introduction

Redundancy scheduling has recently attracted strong interest as a strategy for significantly reducing response times in parallel-server systems [1–3,5–10,13–18]. The key feature in redundancy scheduling is replication of a job upon arrival, allowing replicas to be assigned to, say, d different servers, chosen uniformly at random (without replacement). Redundant replicas are abandoned as soon as the first of these replicas either starts service ('cancel-on-start' or c.o.s.) or completes service ('cancel-on-completion' or c.o.c.). By creating multiple service opportunities, redundancy scheduling boosts the chance of a fast response from a server that is swift to provide service and alleviates the risk of a long delay incurred when a job is assigned to a single server that may be slow. Note that the c.o.c. and c.o.s. policies both ensure that the first replica starts service at the server with the smallest *real* workload, i.e., the amount of work a server needs to complete to become idle in the absence of any arrivals, among the d selected servers. The possibly concurrent service of multiple replicas under the c.o.c. policy provides a further hedge against potentially slow execution of the first replica in the case where replicas are independent (although it may also result in wastage of service effort).

The diversity offered by redundant requests has been shown to strongly improve the response time performance, especially in the case of highly variable service requirements. Analytical results for redundancy scheduling are unfortunately scarce, however, and have largely remained limited to exponentially distributed service requirements. Specifically, Gardner et al. [8] extensively analyzed the c.o.c. redundancy policy with exponentially distributed service requirements. They established the stability condition and showed that it does *not* depend on the number of replicas d, and thus coincides with the nominal condition without any redundancy. This may be explained by the fact that even with concurrent service the expected aggregate amount of time invested in the service of a job remains equal to the mean service requirement of a single instance due to the memoryless property of the exponential distribution. In [3], the stability condition is analyzed for redundancy scheduling with exponentially distributed service requirements, but non-FCFS service disciplines such as processor sharing and random order of service, both for identical and i.i.d. replicas.

Gardner et al. [8] also derived an explicit expression for the expected latency and proved that the latency is decreasing in the number of replicas d. Another approach to derive these expressions, which also can be applied to other models, such as the $M/M/K$ queue with heterogeneous service rates or the MSCCC queue, is given in [5]. Simulation experiments additionally demonstrated greater improvements in the latency in the case of highly variable service requirements, particularly heavy-tailed distributions.

We are not aware of any analytical results for the c.o.c. redundancy policy with independent replicas and nonexponential service requirements. Hellemans & Van Houdt [12] consider the c.o.c. policy with *identical* replicas and derive a differential equation for the marginal workload distribution at each of the servers in a limiting regime where the number of servers grows large. While the differential equation implicitly captures the stability condition, it does not yield any analytical expression, and the derivations for identical replicas rely on highly specific arguments that do not extend to independent replicas. It is also worth observing that the c.o.s. redundancy policy is equivalent

to a power-of-d version of the Join-the-Smallest-Workload (JSW) policy; see [6]. In this policy polling just two servers suffices to achieve most of the performance gain. Moreover, in [6] an exact analysis is given for c.o.s. redundancy in the case of exponential service requirements. While the workload and waiting-time distributions for these policies for general service requirements do not appear analytically tractable, the stability condition is simple and coincides with the nominal condition without any redundancy, since no concurrent service takes place.

In order to gain further insight in the role of the service requirement distribution, we focus in the present paper on the behavior of the c.o.c. redundancy policy for scaled Bernoulli service requirements. While this is admittedly a rather special case, it provides a typical instance of highly variable service requirements for which redundancy scheduling is particularly relevant, and is also of intrinsic merit given the paucity of analytical results for general service requirement distributions.

First of all, we establish a simple sufficient stability condition in terms of a lower bound for the system capacity, i.e., the maximum aggregate load that can be supported. The lower bound is obtained from a stochastic coupling between the maximum workload across all the servers and the workload in a related single-server queue with the same arrival process and a service requirement that corresponds to the minimum service requirement across d replicas. The lower bound for the system capacity grows without bound with (a) the 'scale' of the service requirement and (b) the number of replicas d, but remarkably enough (c) does *not* depend on the number of servers at all (assuming that number to be at least equal to the number of replicas d). The 'scale' of the service requirement here refers to its nonzero value relative to its mean and provides a proxy for the degree of variability. The growth in the system capacity with (a) and (b) reflects the huge benefits provided by redundancy scheduling for highly variable service requirements.

In view of (c), the lower bound may at first sight seem loose for a larger number of servers, but we will use a further stochastic comparison argument to prove that it is in fact asymptotically tight when the scale of the service requirement grows suitably large. This implies that increasing the number of replicas significantly increases the system capacity, while adding servers does *not* asymptotically. Or, stated differently, given the number of replicas d, redundancy scheduling ensures that asymptotically just d servers suffice to achieve the capacity achievable with any number of servers, which further highlights the great gains provided by redundancy scheduling for highly variable service requirements.

The remainder of the paper is organized as follows: In Sect. 2, we present a detailed model description and state a sufficient stability condition for generally distributed service requirements. In Sect. 3, we prove that this condition is also asymptotically nearly necessary for scaled Bernoulli service requirements. An upper bound for the expected waiting time is derived in Sect. 4 and in Sect. 5 we provide a conclusion.

2 Workload model and sufficient stability condition

We consider a system with N parallel servers. Jobs arrive according to a Poisson process with rate λ. Each arriving job is replicated and immediately allocated to

d servers chosen uniformly at random (without replacement). The replicas at each server are served in order of arrival (FCFS), and the job is completed as soon as the first replica finishes service, whereafter the other $d-1$ replicas are instantaneously abandoned. The service requirements of the d replicas are assumed to be independent and identically distributed (i.i.d.) copies of some random variable B. Note that this model corresponds to the independent runtime (IR) model described in [8].

Let $\boldsymbol{\omega} = (\omega_1, \ldots, \omega_N)$ denote the workload of the system, where ω_i is the workload at server i, for $i = 1, \ldots, N$. Here we define workload as the *real* amount of work, i.e., the amount of work a server needs to complete to become idle in the absence of any arrivals. This may be smaller than the sum of the service requirements of all the replicas at the server since some replicas may be partly or entirely abandoned; see Example 1. Let s_j and b_j denote the sampled server and the realized service requirement of the jth replica, respectively, for $j = 1, \ldots, d$. The first replica will finish service on server s_{j^*}, where $j^* = \arg\min_{j\in\{1,\ldots,d\}}(\omega_{s_j} + b_j)$. The workload of server s_j is then $\max\{\omega_{s_{j^*}} + b_{j^*}, \omega_{s_j}\}$, for $j = 1, \ldots, d$.

Example 1 Consider a system with $N = 4$, $d = 2$ and workload state $\boldsymbol{\omega} = (4.1, 4.1, 3.5, 2.3)$. Then, after an arrival with service requirements $(2.2, 1.5)$ on servers 2 and 4, the new workload state is $\boldsymbol{\omega}_{\text{new}} = (4.1, 4.1, 3.5, 3.8)$.

Let $\boldsymbol{\omega}_{(\cdot)}$ denote the workloads arranged in descending order, thus $\boldsymbol{\omega}_{(\cdot)} = \{\boldsymbol{\omega} \in \mathbb{R}_+^N : \omega_{(1)} \geq \omega_{(2)} \geq \cdots \geq \omega_{(N)}\}$. Throughout this paper, we refer to *synchronicity* as the situation in which all workloads are equal, i.e., $\omega_1 = \cdots = \omega_N$. Moreover, let $\mathcal{S}_{\text{trun}}$ denote the truncated state space of the ordered workload vectors with $\mathcal{S}_{\text{trun}} = \{\boldsymbol{\omega} \in \mathbb{R}_+^N : \omega_{(1)} = \cdots = \omega_{(d)} \geq \omega_{(d+1)} \geq \cdots \geq \omega_{(N)}\}$.

The next property states that the d largest workloads will always be equal from some point onward. We will later see that under certain conditions the system will in fact be in full synchronicity nearly all the time.

Property 1 If $\boldsymbol{\omega} \in \mathcal{S}_{\text{trun}}$, then $\boldsymbol{\omega}_{\text{new}} \in \mathcal{S}_{\text{trun}}$, where $\boldsymbol{\omega}_{\text{new}}$ is any future workload. In other words, once the largest d workloads are equal, they will always remain equal.

Proof Consider the two options, either (i) $\min_{j\in\{1,\ldots,d\}}(\omega_{s_j}+b_j) \leq \omega_{(1)} = \cdots = \omega_{(d)}$, in which case we have $\omega_{\text{new},s_l} = \max\{\min_{j\in\{1,\ldots,d\}}(\omega_{s_j} + b_j), \omega_{s_l}\} \leq \omega_{(1)}$, for $l = 1, \ldots, d$, therefore $\omega_{(1)} = \cdots = \omega_{(d)} = \omega_{\text{new},(1)} = \cdots = \omega_{\text{new},(d)}$, or ii) $\min_{j\in\{1,\ldots,d\}}(\omega_{s_j} + b_j) > \omega_{(1)} = \cdots = \omega_{(d)}$, in which case $\omega_{\text{new},s_l} = \max\{\min_{j\in\{1,\ldots,d\}}(\omega_{s_j} + b_j), \omega_{s_l}\} = \min_{j\in\{1,\ldots,d\}}(\omega_{s_j} + b_j)$, for $l = 1, \ldots, d$, therefore $\omega_{\text{new},s_1} = \cdots = \omega_{\text{new},s_d} = \omega_{\text{new},(1)} = \cdots = \omega_{\text{new},(d)}$. In both cases $\boldsymbol{\omega}_{\text{new}} \in \mathcal{S}_{\text{trun}}$, thus by a simple induction argument it follows that there are always d servers with the same maximum workload. □

Before stating and proving a sufficient stability condition, we prove the following lemma for generally distributed service requirements.

Lemma 1 *The sequence of maximum workloads $\omega_{(1)}$ at arbitrary epochs is stochastically upper bounded by the sequence of workloads $\omega_{M/G/1}$ in a corresponding $M/G/1$ queue with arrival rate $\lambda_{M/G/1} = \lambda$ and generic service requirement $B_{M/G/1} = \min\{B_1, \ldots, B_d\}$, provided that the initial maximum workload $\omega_{(1)}$ is smaller than the initial workload in the $M/G/1$ queue.*

Proof The proof follows by induction. Note that for the initial state the statement is satisfied. Assume that $\omega_{(1)} \leq \omega_{M/G/1}$ after the kth arrival. Then, after the $(k+1)$th arrival, the new workload is $\omega_{\text{new},s_l} = \max\{\min_{j\in\{1,\ldots,d\}}(\omega_{s_j} + b_j), \omega_{s_l}\} \leq \max\{\min_{j\in\{1,\ldots,d\}}(\omega_{(1)} + b_j), \omega_{(1)}\} = \omega_{(1)} + \min_{j\in\{1,\ldots,d\}} b_j$, for $l = 1, \ldots, d$, since $\omega_i \leq \omega_{(1)}$ for all $i = 1, \ldots, N$. Thus the increase in maximum workload is bounded by $\min_{j\in\{1,\ldots,d\}} b_j$, which is exactly the increase in workload in the corresponding $M/G/1$ queue. □

Remark 1 Observe that in synchronicity, in which all servers have the maximum workload, the bound $\min_{j\in\{1,\ldots,d\}} b_j$ is tight, since here every arrival adds exactly $\min_{j\in\{1,\ldots,d\}} b_j$ work to each of the d sampled servers.

Proposition 1 *A sufficient stability condition is*

$$\lambda\mathbb{E}[\min\{B_1, \ldots, B_d\}] < 1. \tag{1}$$

Proof By Lemma 1, we know that the maximum workload in the system is bounded by the workload in a corresponding $M/G/1$ queue with arrival rate $\lambda_{M/G/1} = \lambda$ and generic service requirement $B_{M/G/1} = \min\{B_1, \ldots, B_d\}$. The (necessary and sufficient) stability condition for the latter $M/G/1$ queue is given by

$$\rho = \lambda_{M/G/1}\mathbb{E}[B_{M/G/1}] = \lambda\mathbb{E}[\min\{B_1, \ldots, B_d\}] < 1.$$

□

Remark 2 Note that Property 1, Lemma 1 and Proposition 1 have an equivalent version in the case of identical replicas with $\mathbb{E}[\min\{B_1, \ldots, B_d\}] = \mathbb{E}[B]$.

In the case $N = d$, the above condition is not only sufficient but in fact also necessary, since the system behaves exactly as the corresponding $M/G/1$ queue, as also becomes apparent from [15]. In the case $N > d$ the above condition is no longer strictly necessary; see also the stability condition for exponential service requirements in [8]. However, we will show that, surprisingly, it is asymptotically nearly necessary for independent scaled Bernoulli service requirements, which are defined as

$$B = \begin{cases} X \cdot K, & \text{w.p. } 1-p, \\ 0, & \text{w.p. } p, \end{cases}$$

where K is a fixed positive real number, and X is a general strictly positive random variable with $\mathbb{E}[X] = 1$. Moreover, we assume that $\mathbb{E}[B] = 1$, which implies that $p = 1 - 1/K$.

For notational convenience, we label jobs for which none of the d replicas have service requirement 0 as type-A jobs. For a type-A job $(X_1K, \ldots, X_dK)$ are the service requirements of the replicas at the d sampled servers, where the random variables $X_1, \ldots, X_d$ are i.i.d. copies of X. Jobs for which at least one replica but at most $d-1$ replicas have service requirement equal to 0 are called type-B jobs, and jobs for which all d replicas have service requirement equal to 0 are called type-C jobs.

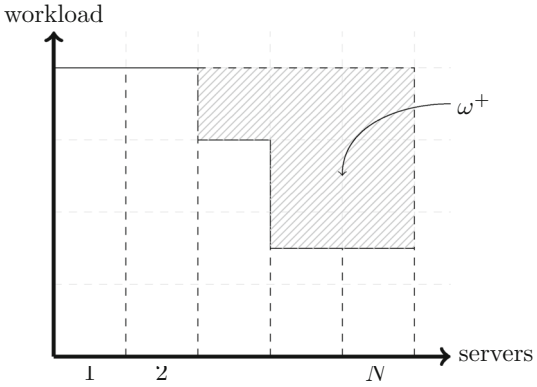

Fig. 1 Visual representation workload surplus

From Proposition 1, it follows that for independent scaled Bernoulli service requirements the sufficient stability condition reduces to

$$(1-p)^d \lambda \mathbb{E}[\min\{X_1 K, \ldots, X_d K\}] = \frac{\lambda \mathbb{E}[\min\{X_1, \ldots, X_d\}]}{K^{d-1}} < 1, \tag{2}$$

since all jobs, other than type-A jobs, which have arrival rate $(1-p)^d\lambda$ and service requirement $\min\{X_1 K, \ldots, X_d K\}$, have service requirements for which $\min\{B_1, \ldots, B_d\} = 0$.

3 Asymptotically necessary stability condition

In this section, we shall prove that the sufficient stability condition (2) is in fact also asymptotically nearly necessary. The proof relies on the property that the system is most of the time in synchronicity as K grows large.

In preparation for the proof let us first define a measure for synchronicity. Let the surplus workload, denoted by ω^+, be the sum of the (element-wise) differences between the maximum workload and the workload at server i for $i = 1, \ldots, N$, i.e., $\omega^+ = \sum_{i=1}^{N} \left(\omega_{(1)} - \omega_i\right)$; see Fig. 1 for a visual representation. Note that $\omega^+ = 0$ if and only if the system is in synchronicity.

In order to prove that the system is in synchronicity nearly all the time, we introduce an auxiliary system which is the same as our system except for three differences. In the auxiliary system (i) the workload at each server only decreases over time when in synchronicity, (ii) all type-A jobs are allocated to the first d ordered servers and (iii) only specific type-B jobs, so-called type-B_1 jobs, are considered and the other type-B jobs are omitted. We define type-B_1 jobs as ones for which $d-1$ replicas, with at least one replica with service requirement equal to 0, are allocated to the first $d-1$ ordered servers and one replica with service requirement $X_d K$ to the Nth ordered server, i.e., the server with the lowest current workload.

Below we comment on the properties of the surplus workload $\tilde{\omega}^+$ in the auxiliary system.

Property 2 The surplus workload in the auxiliary system, $\tilde{\omega}^+$, experiences downward jumps at the instants of a Poisson process of rate $\frac{(N-d)!}{N!}(1-p)p^{d-1}\lambda$, which is exactly the arrival rate of type-B_1 jobs. The sizes of the downward jumps are equal to $\min\{\tilde{\omega}_{(1)} - \tilde{\omega}_{(N)}, X_d K\}$.

Note that the surplus workload in the original system, ω^+, experiences downward jumps at a higher rate than $\tilde{\omega}^+$, since not only type-B_1 jobs decrease the surplus workload. Moreover, the sizes of the downward jumps in the surplus workload and in the surplus workload in the auxiliary system can differ, since these depend on the workloads in both systems (which are not necessarily equal).

Property 3 The surplus workload in the auxiliary system, $\tilde{\omega}^+$, experiences upward jumps of size exactly $(N-d)\min\{X_1, \ldots, X_d\}K$ as a Poisson process of rate $(1-p)^d\lambda$, which is the arrival rate of type-A jobs.

Note that the surplus workload in the original system, ω^+, experiences upward jumps of smaller or equal size, since type-A jobs add at most $\min\{X_1, \ldots, X_d\}K$ work to the current maximum workload; see Remark 1.

The number of jumps, denoted by Z, to reach synchronicity in the auxiliary system when only considering downward jumps is equal to the total number of type-B_1 jobs that are needed at each server to bridge the difference between the maximum workload and the workload at this server. Thus, the expectation of the number of jumps to reach synchronicity, when only considering type-B_1 jobs and starting in the initial workload state $\tilde{\boldsymbol{\omega}}$, where $\tilde{\boldsymbol{\omega}} \in \mathcal{S}_{\text{trun}}$, is

$$\begin{aligned}
\mathbb{E}[Z] &= \sum_{i=d+1}^{N} \mathbb{E}[\min\{n : X_1K + \cdots + X_nK \geq \tilde{\omega}_1 - \tilde{\omega}_i\}] \\
&\leq \sum_{i=d+1}^{N} \mathbb{E}[\min\{n : X_1K + \cdots + X_nK \geq \tilde{\omega}^+\}] \\
&\leq (N-d)\left(\mathbb{E}\left[\max\left\{n : S_n \leq \frac{\tilde{\omega}^+}{K}\right\}\right] + 1\right) \\
&= (N-d)\left(m\left(\frac{\tilde{\omega}^+}{K}\right) + 1\right), \qquad (3)
\end{aligned}$$

where $S_n = \sum_{j=1}^{n} X_j$ and the renewal function m (cf. [11, Definition 10.1.6]) is given by $m(t) = \mathbb{E}[N(t)]$, with $N(t) = \max\{n : S_n \leq t\}$. Note that the third line holds with equality if $\frac{\tilde{\omega}^+}{K} \notin \mathbb{N}$.

For proving an asymptotically necessary stability condition, we first need to prove the following two lemmas. Lemma 2 states that the surplus workload in the auxiliary system stochastically dominates the surplus workload in the original system, and Lemma 3 states that the surplus workload in the auxiliary system is, a high fraction of the time, equal to 0 in the long term as K grows large. Together Lemmas 2 and 3 imply that the original system will also be in synchronicity a high fraction of the time in the long term as K grows large. This in turn implies that almost every arriving

job will add $B_{M/G/1} = \min\{B_1, \ldots, B_d\}$ to the maximum workload. Observe that this is exactly the upper bound, see Lemma 1, which resulted in the sufficient stability condition.

Let $\{\boldsymbol{\omega}(t), \omega^+(t)\}_{t\geq 0}$ denote the stochastic process that describes the evolution of the workload vector $\boldsymbol{\omega} = (\omega_1, \ldots, \omega_N)$ and the surplus workload ω^+ over time. We further introduce a stochastic process $\{\tilde{\boldsymbol{\omega}}(t), \tilde{\omega}^+(t)\}_{t\geq 0}$ that describes the evolution of the workload vector $\tilde{\boldsymbol{\omega}} = (\tilde{\omega}_1, \ldots, \tilde{\omega}_N)$ and the surplus workload $\tilde{\omega}^+$ of the auxiliary system over time, with $\tilde{\omega}^+(t) = \sum_{i=1}^{N} \big(\tilde{\omega}_{(1)}(t) - \tilde{\omega}_i(t)\big)$.

Lemma 2 *The workload vectors of the auxiliary system and the original system satisfy the inequality* $\tilde{\omega}_{(1)}(t) - \tilde{\omega}_{(i)}(t) \geq \omega_{(1)}(t) - \omega_{(i)}(t)$ *for all* $t \geq 0$ *and* $i = 1, \ldots N$ *when both systems experience the same arrivals and the same generic service requirements, and start in the same initial workload state* $\tilde{\boldsymbol{\omega}} \in \mathcal{S}_{trun}$, *i.e.,* $\tilde{\boldsymbol{\omega}}(0) = \boldsymbol{\omega}(0) = \tilde{\boldsymbol{\omega}}$ *and* $\tilde{\omega}_{(1)} = \cdots = \tilde{\omega}_{(d)}$.

Proof Since both systems start in the same initial workload state it follows that $\tilde{\omega}_{(1)}(0) - \tilde{\omega}_{(i)}(0) = \omega_{(1)}(0) - \omega_{(i)}(0)$, for $i = 1, \ldots, N$. Moreover, by Property 1, it follows that $\tilde{\omega}_{(1)}(t) - \tilde{\omega}_{(i)}(t) = \omega_{(1)}(t) - \omega_{(i)}(t) = 0$ for $t \geq 0$ and $i = 1, \ldots, d$. We prove the statement for $i = d+1, \ldots, N$ by induction in time. Assume that $\tilde{\omega}_{(1)}(t_1) - \tilde{\omega}_{(i)}(t_1) \geq \omega_{(1)}(t_1) - \omega_{(i)}(t_1)$, then it should hold that $\tilde{\omega}_{(1)}(t_2) - \tilde{\omega}_{(i)}(t_2) \geq \omega_{(1)}(t_2) - \omega_{(i)}(t_2)$ for $t_2 > t_1$, when considering all the events that can occur between times t_1 and t_2:

- When no arrivals occur only the value of $\omega^+(t)$ can decrease over time, since the workload at each server in the auxiliary system only decreases over time in synchronicity. Thus, it follows that $\omega_{(1)}(t_1) - \omega_{(i)}(t_1) \geq \omega_{(1)}(t_2) - \omega_{(i)}(t_2)$ (which is a strict inequality in the case $\omega_{(i)}(t_2) = 0$), whereas $\tilde{\omega}_{(1)}(t_1) - \tilde{\omega}_{(i)}(t_1) = \tilde{\omega}_{(1)}(t_2) - \tilde{\omega}_{(i)}(t_2)$.
- In the case of an arrival of a type-A job, the value of $\tilde{\omega}^+(t)$ increases by exactly $(N-d)\min\{X_1, \ldots, X_d\}K$, whereas the value of $\omega^+(t)$ increases by at most $(N-d)\min\{X_1, \ldots, X_d\}K$; see the proof of Lemma 1 and Property 3. Also, note that a type-A job in the auxiliary system is always allocated to the first d ordered servers, instead of d servers sampled uniformly at random. Thus, it follows that $\min\{X_1, \ldots, X_d\}K = \tilde{\omega}_{(1)}(t_2) - \tilde{\omega}_{(1)}(t_1) \geq \omega_{(1)}(t_2) - \omega_{(1)}(t_1)$ and $0 = \tilde{\omega}_{(i)}(t_2) - \tilde{\omega}_{(i)}(t_1) \leq \omega_{(i)}(t_2) - \omega_{(i)}(t_1)$. Combining the latter two inequalities yields $\tilde{\omega}_{(1)}(t_2) - \tilde{\omega}_{(i)}(t_2) \geq \omega_{(1)}(t_2) - \omega_{(i)}(t_2)$.
- In the case of an arrival of a type-B job, excluding a type-B_1 job, only the value of $\omega^+(t)$ can decrease. Thus, it follows that $\omega_{(1)}(t_1) - \omega_{(i)}(t_1) \geq \omega_{(1)}(t_2) - \omega_{(i)}(t_2)$ (which is a strict inequality in the case of a type-B job that adds workload to server i), whereas $\tilde{\omega}_{(1)}(t_1) - \tilde{\omega}_{(i)}(t_1) = \tilde{\omega}_{(1)}(t_2) - \tilde{\omega}_{(i)}(t_2)$.
- In the case of an arrival of a type-B_1 job, the value of $\tilde{\omega}^+(t)$ decreases by $\min\{\tilde{\omega}_{(1)}(t_1) - \tilde{\omega}_{(N)}(t_1), X_d K\}$, whereas the value of $\omega^+(t)$ decreases by $\min\{\omega_{(1)}(t_1) - \omega_{(N)}(t_1), X_d K\}$. Observe that the decrement in the value of $\tilde{\omega}^+(t)$ can be greater than the decrement in the value of $\omega^+(t)$, see Property 2, but only if $\omega_{(1)}(t_2) - \omega_{N*}(t_2) = 0$, where $N*$ is the server at time t_2 that had the minimum workload at time t_1 (which is not necessarily the server with minimum workload at time t_2). Therefore, it follows that $\tilde{\omega}_{(1)}(t_2) - \tilde{\omega}_{(i)}(t_2) \geq \omega_{(1)}(t_2) - \omega_{(i)}(t_2)$.

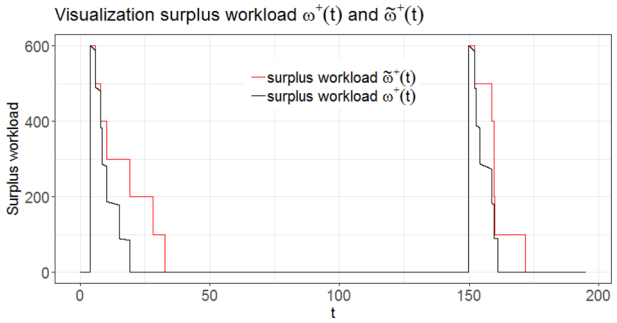

Fig. 2 Visual representation of surplus processes, where $\lambda = 50$, $N = 8$, $d = 2$ and $K = 100$

We conclude that in all scenarios it still holds that $\tilde{\omega}_{(1)}(t_2) - \tilde{\omega}_{(i)}(t_2) \geq \omega_{(1)}(t_2) - \omega_{(i)}(t_2)$. □

Lemma 2 implies that the surplus workload in the auxiliary system, $\tilde{\omega}^+(t)$, stochastically dominates the surplus workload $\omega^+(t)$ when starting in the same initial workload state; see Fig. 2. Now we prove that the surplus workload in the auxiliary system, $\tilde{\omega}^+(t)$, is a high fraction of the time equal to 0 in the long term as K grows large.

Lemma 3 *For every $\epsilon > 0$ there exists $K_\epsilon(d, N)$ such that, for all $K > K_\epsilon(d, N)$, the value of $\tilde{\omega}^+(t)$ is at least a fraction $(1 - \epsilon)$ of the time equal to 0 in the long term.*

Proof First denote $\tau_1 := \inf\{t \geq 0 | \tilde{\omega}^+(t) > 0\}$ as the time that the value of $\tilde{\omega}^+(t)$ remains equal to 0, when starting in synchronicity. Note that τ_1 is the time until the next upward jump; see Property 3. Therefore the expectation of τ_1 is given by

$$\mathbb{E}[\tau_1] = \frac{1}{(1-p)^d \lambda} = \frac{K^d}{\lambda}.$$

Denote the time that the workload in the auxiliary system remains in non-synchronicity, i.e., the time that $\tilde{\omega}^+(t) > 0$ when starting in initial workload state $\tilde{\boldsymbol{\omega}}(0) = \tilde{\boldsymbol{\omega}}$, where $\tilde{\boldsymbol{\omega}} \in \mathcal{S}_{\text{trun}}$, by $\tau_2 := \inf\{t \geq 0 | \tilde{\omega}^+(t) = 0\}$. Moreover, let $\{Y | \tilde{\boldsymbol{\omega}}(0) = \tilde{\boldsymbol{\omega}}\}$ denote the number of increments in the value of $\tilde{\omega}^+(t)$ before reaching synchronicity when starting in $\tilde{\boldsymbol{\omega}} \in \mathcal{S}_{\text{trun}}$. Then the expectation of τ_2 is

$$\begin{aligned}
\mathbb{E}[\tau_2] &= \sum_{n=0}^{\infty} \mathbb{E}[\tau_2 | Y = n, \tilde{\boldsymbol{\omega}}(0) = \tilde{\boldsymbol{\omega}}] \cdot \mathbb{P}(Y = n | \tilde{\boldsymbol{\omega}}(0) = \tilde{\boldsymbol{\omega}}) \\
&= \sum_{n=0}^{\infty} \frac{\mathbb{E}[Z | Y = n, \tilde{\boldsymbol{\omega}}(0) = \tilde{\boldsymbol{\omega}}]}{\frac{(N-d)!}{N!}(1-p)p^{d-1}\lambda} \cdot \mathbb{P}(Y = n | \tilde{\boldsymbol{\omega}}(0) = \tilde{\boldsymbol{\omega}}) \\
&\leq \frac{1}{\frac{(N-d)!}{N!}(1-p)p^{d-1}\lambda} \Bigg[(N-d) \left(m \left(\frac{\tilde{\omega}^+}{K} \right) + 1 \right) \\
&\quad + \sum_{n=1}^{\infty} \Big(n(N-d) \big(m(\mathbb{E}[\min\{X_1, \ldots, X_d\}]) + 1 \big) \cdot \mathbb{P}(Y = n | \tilde{\boldsymbol{\omega}}(0) = \tilde{\boldsymbol{\omega}}) \Big) \Bigg]
\end{aligned}$$

$$= \frac{1}{\frac{(N-d)!}{N!}(1-p)p^{d-1}\lambda}\Big[(N-d)\left(m\left(\frac{\tilde{\omega}^+}{K}\right)+1\right) + (N-d)\big(m(\mathbb{E}[\min\{X_1,\ldots,X_d\}])+1\big)\mathbb{E}[Y|\tilde{\boldsymbol{\omega}}(0)=\tilde{\boldsymbol{\omega}}]\Big],$$

with $\mathbb{E}[\min\{X_1,\ldots,X_d\}] \leq \mathbb{E}[X] = 1$. The second equality results from Wald's equation, i.e., the equality between the expected time to reach synchronicity (given the number of upward jumps) and the expected number of downward jumps (given the number of upward jumps) multiplied with the expected time between such downward jumps. The inequality in the next step results from the proof of Lemma 1, which implies that the surplus workload increases by at most $(N-d)\min\{X_1,\ldots,X_d\}K$ per upward jump, and using the bound on the expected number of downward jumps (given the number of upward jumps), i.e., Eq. (3).

Together with Wald's equation

$$\mathbb{E}[Y|\tilde{\boldsymbol{\omega}}(0)=\tilde{\boldsymbol{\omega}}] = \mathbb{E}[\tau_2](1-p)^d\lambda,$$

we can bound the expected time in non-synchronicity, namely

$$\begin{aligned}
&\mathbb{E}[\tau_2]\frac{(N-d)!}{N!}(1-p)p^{d-1}\lambda \\
&\quad\leq (N-d)\left(m\left(\frac{\tilde{\omega}^+}{K}\right)+1\right) + (N-d)\big(m(\mathbb{E}[\min\{X_1,\ldots,X_d\}])+1\big)\mathbb{E}[\tau_2](1-p)^d\lambda \\
&\quad\Leftrightarrow \mathbb{E}[\tau_2]\left(\frac{(N-d)!}{N!}(1-p)p^{d-1}\lambda - (N-d)\big(m(\mathbb{E}[\min\{X_1,\ldots,X_d\}])+1\big)(1-p)^d\lambda\right) \\
&\quad\leq (N-d)\left(m\left(\frac{\tilde{\omega}^+}{K}\right)+1\right) \\
&\quad\Leftrightarrow \mathbb{E}[\tau_2] \leq \frac{(N-d)\big(m(\frac{\tilde{\omega}^+}{K})+1\big)}{\frac{(N-d)!}{N!}\frac{\lambda}{K}(1-\frac{1}{K})^{d-1} - (N-d)\big(m(\mathbb{E}[\min\{X_1,\ldots,X_d\}])+1\big)\frac{\lambda}{K^d}} \\
&\quad= \frac{1}{\lambda}\mathcal{O}(K),
\end{aligned}$$

under the assumption that $\frac{(N-d-1)!}{N!}(1-\frac{1}{K})^{d-1} > \frac{m(\mathbb{E}[\min\{X_1,\ldots,X_d\}])+1}{K^{d-1}}$. Moreover, $m(\frac{\tilde{\omega}^+}{K}) \downarrow 0$ as K grows large, and by renewal theory (cf. [11]) we know that $m(\mathbb{E}[\min\{X_1,\ldots,X_d\}]) < \infty$ since $\mathbb{E}[\min\{X_1,\ldots,X_d\}] \leq 1$.

Now if we choose $K_\epsilon(d,N)$ such that for $K = K_\epsilon(d,N)$ one has

$$\frac{\mathbb{E}[\tau_2]}{\mathbb{E}[\tau_1]+\mathbb{E}[\tau_2]} \leq \epsilon,$$

then for all $K > K_\epsilon(d,N)$ it follows that

$$\frac{\mathbb{E}[\tau_1]}{\mathbb{E}[\tau_1]+\mathbb{E}[\tau_2]} > 1-\epsilon = 1-\mathcal{O}\left(\frac{1}{K^{d-1}}\right).$$

This completes the proof that the auxiliary surplus workload is at least a fraction $(1-\epsilon)$ of the time equal to 0 in the long term. □

Remark 3 So far it has been assumed that in the initial state the first d ordered workloads are equal, but this assumption is not necessary. One can show via an approach analogous to Lemma 3, but with the bound $\mathbb{E}[Z] < N\big(m(\frac{\tilde{\omega}^+}{K})+1\big)$, that the expected time to reach synchronicity when starting in an arbitrary initial workload state is still finite. Note that after reaching synchronicity the assumption is valid and that directly after synchronicity $\tilde{\omega}^+(t) = \omega^+(t) = (N-d)\min\{X_1,\ldots,X_d\}K$.

Now we are ready to prove the main theorem of the paper.

Theorem 1 *For every $\epsilon > 0$ there exists $K_\epsilon(d,N)$ such that, for all $K > K_\epsilon(d,N)$, a necessary stability condition for independent scaled Bernoulli service requirements is*

$$(1-\epsilon)\frac{\lambda\mathbb{E}[\min\{X_1,\ldots,X_d\}]}{K^{d-1}} < 1. \tag{4}$$

Proof From Lemma 2, we know that $\tilde{\omega}^+(t)$ stochastically dominates $\omega^+(t)$, and Lemma 3 states that for every $\epsilon > 0$ there exists $K_\epsilon(d,N)$ such that for all $K > K_\epsilon(d,N)$ the value of $\tilde{\omega}^+(t)$ is at least a fraction $(1-\epsilon)$ of the time equal to 0 in the long term. Hence, this latter statement also holds for the value of $\omega^+(t)$. Moreover, by definition, if $\omega^+(t) = 0$, then the system is in synchronicity. In synchronicity, type-A jobs add exactly $\min\{X_1,\ldots,X_d\}K$ work to the sampled servers. We conclude that, independent of the behavior in non-synchronicity, in the long term at least a fraction $(1-\epsilon)$ of the type-A jobs adds exactly $\min\{X_1,\ldots,X_d\}K$ work to the current maximum workload. Thus, for the system to be stable it should at least be able to handle these latter type-A jobs. □

Remark 4 The expected time in non-synchronicity depends on the renewal function $m(t)$; see Lemma 3. This function in turn depends on the distribution of the X component in the service requirement distribution B. For some distributions an explicit expression for $m(t)$ is known (cf. [11]):

- $X \equiv 1$: $m(t) = \lfloor t \rfloor$,
- $X \sim \text{Exp}(1)$: $m(t) = t$,
- $X \sim \text{Unif}[0,2]$: $m(t)+1 = \sum_{i=0}^{\lfloor t/2 \rfloor}(-1)^i \frac{(t/2-i)^i}{i!}e^{t/2-i}$.

4 Numerical results

In Sect. 3 it is proven that the system, for scaled Bernoulli distributed service requirements and K large enough, is a high fraction of the time in synchronicity in the long term. In this section, we will use simulation to quantify this statement for various values of N and K, where $d = 2$ is fixed.

In Fig. 3, the long-term fraction of time in synchronicity is depicted as a function of K for various values of N. The fraction $\frac{\lambda}{K}$ is kept constant to ensure that the workload

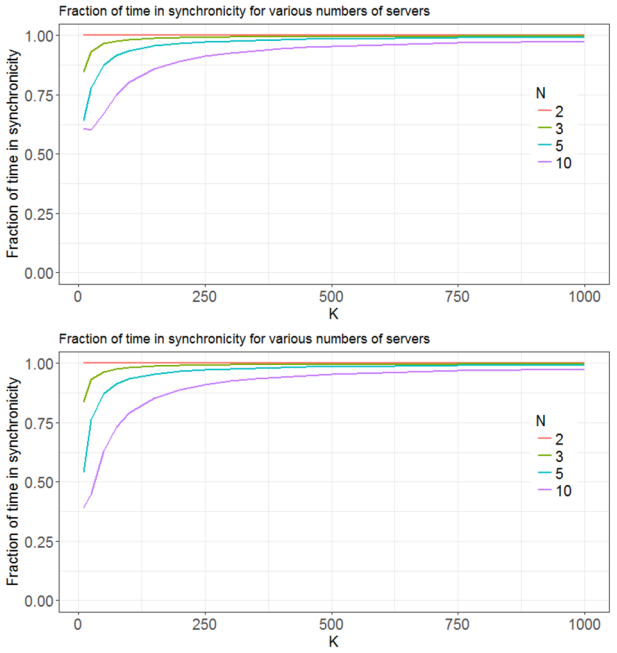

Fig. 3 Long-term fraction of time in synchronicity (obtained by simulation) for the setting $d = 2$, $X \equiv 1$ and $\frac{\lambda}{K} = 0.5$ (top) and $\frac{\lambda}{K} = 0.9$ (bottom)

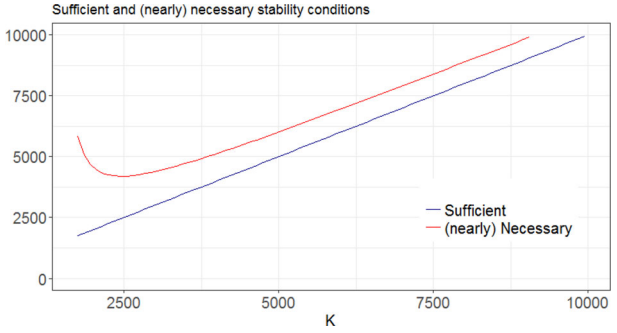

Fig. 4 Stability conditions for the setting $N = 10$, $d = 2$ and $X \equiv 1$

in the system is approximately equal for all K. It can be seen that the system with $N = d$ is always in synchronicity, which follows from Property 1. Moreover, the long-term fraction of time in synchronicity is higher for lower values of $\frac{\lambda}{K}$. The reason is that the empty state is included in the definition of synchronicity. Another observation is that for fixed λ and K, increasing N decreases the long-term fraction of time in synchronicity. This is related to the fact that $K_\epsilon(d, N)$ defined in Lemma 3 depends on N.

Figure 4 shows both the sufficient and nearly necessary stability conditions as functions of K in the setting $N = 10$ and $d = 2$. Note that the lines in the figure are

not parallel; for $K = 5000$ the difference is 1000, and for $K = 10{,}000$ the difference is 850. More specifically, for K increasing the difference vanishes. Another observation obtained by simulation (not depicted in the figure) is that the stability condition of this system is (almost) equal to the sufficient stability condition. This implies that the nearly necessary stability condition tends to be a loose bound for finite values of K.

In Lemma 1 we proved that the maximum workload is bounded by the workload in a corresponding $M/G/1$ queue. From this lemma it follows that for independent scaled Bernoulli service requirements, the maximum workload is bounded by the workload in a corresponding $M/G/1$ queue with arrival rate $\lambda_{M/G/1}(K) = (1-p)^d\lambda$ and service requirement $B_{M/G/1}(K) = \min\{X_1, \ldots, X_d\}K$ since all arrivals, other than the arrivals of type-A jobs, have service requirements for which $\min\{B_1, \ldots, B_d\} = 0$. This bound can be used to find an upper bound on the expected waiting time since an arriving job needs to wait at most for the current maximum workload, which is bounded by the workload $V_{M/G/1}$ in the corresponding $M/G/1$ queue. From $M/G/1$ theory (cf. [4, Section X.3]) we get

$$\begin{aligned}\mathbb{E}[W] &\leq \mathbb{E}[V_{M/G/1}] \\ &= \frac{\lambda_{M/G/1}(K)\mathbb{E}[B^2_{M/G/1}(K)]}{2(1-\lambda_{M/G/1}(K)\mathbb{E}[B_{M/G/1}(K)])} \\ &= \frac{(1-p)^d\lambda\mathbb{E}[\min\{X_1, \ldots, X_d\}^2]K^2}{2(1-(1-p)^d\lambda\mathbb{E}[\min\{X_1, \ldots, X_d\}]K)}. \end{aligned} \tag{5}$$

Note that this bound is tight for $N = d$ since the system behaves exactly as the corresponding $M/G/1$ queue, and is asymptotically tight in K for $N > d$. For $X \equiv 1$ constant and $d = 2$ we get

$$\begin{aligned}\mathbb{E}[W] &\leq \frac{(1-p)^2\lambda K^2}{2(1-(1-p)^2\lambda K)} \\ &= \frac{\lambda}{2(1-\frac{\lambda}{K})},\end{aligned}$$

which is linear in K if we assume that $\frac{\lambda}{K}$ is fixed. Notice that this upper bound does not depend on the number of servers.

Figure 5 shows the expected waiting time as a function of K for various values of N; again we kept the fraction $\frac{\lambda}{K}$ constant to ensure that the workload is approximately equal for all K. When comparing both figures we can conclude that for finite K the number of servers N influences the expected waiting time more than the value of the fraction $\frac{\lambda}{K}$. Moreover, for N large, a larger K is needed for the upper bound to be accurate.

Observe that with the upper bound for the expected waiting time we also have an upper bound for the expected latency, since

$$\mathbb{E}[T] \leq \mathbb{E}[W] + \mathbb{E}[\min\{B_1, \ldots, B_d\}],$$

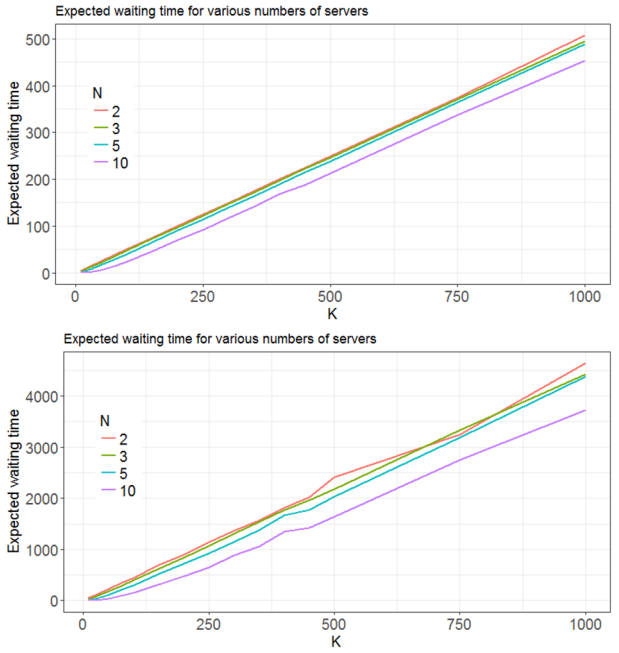

Fig. 5 Expected waiting time (obtained by simulation) for the setting $d = 2$, $X \equiv 1$ and $\frac{\lambda}{K} = 0.5$ (top) and $\frac{\lambda}{K} = 0.9$ (bottom). Note that $N = 2$ corresponds to the upper bound given in (5)

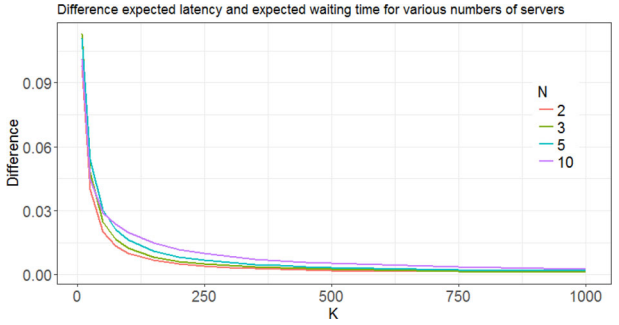

Fig. 6 Difference between the expected latency and the expected waiting time (obtained by simulation) for the setting $d = 2$, $X \equiv 1$ and $\frac{\lambda}{K} = 0.5$

where $\mathbb{E}[\min\{B_1, \ldots, B_d\}] \leq 1$ since, by assumption, $\mathbb{E}[B_i] = 1$, for $i = 1, \ldots, d$.

Note that the upper bound for the service requirements, i.e., $\mathbb{E}[\min\{B_1, \ldots, B_d\}] \leq 1$, is not (asymptotically) tight in K, since $\mathbb{E}[\min\{B_1, \ldots, B_d\}] \downarrow 0$ as K grows large. In Fig. 6 the difference between the expected latency and the expected waiting time is depicted as function of K for various values of N. Indeed, it can be seen that this difference vanishes as K grows large.

5 Conclusion

In this paper, we have proven that the maximum workload in a parallel-server system with c.o.c. redundancy is upper bounded by the workload in a related $M/G/1$ queue. This directly yields a sufficient stability condition. Moreover, we proved that in the case of independent scaled Bernoulli service requirements the system is asymptotically a high fraction of the time in so-called synchronicity in the long term. In synchronicity the upper bound of the related $M/G/1$ queue is in fact tight, and this resulted in an asymptotically necessary stability condition. Interestingly, both the sufficient and asymptotically nearly necessary conditions are independent of the number of servers, but do depend on the number of replicas d. In contrast, in the case of exponentially distributed service requirements the stability condition depends linearly on the number of servers and not on the number of replicas. This indicates that the stability condition in a c.o.c. redundancy system with i.i.d. service requirements is highly sensitive to the distribution of these service requirements.

The bound on the maximum workload also resulted in an upper bound for the expected waiting time, which is again (asymptotically) tight (as the scale of the service requirement grows large). This bound directly resulted in an upper bound for the expected latency.

We assumed that jobs arrive according to a Poisson process, but it might be possible to relax this assumption. In particular, the proof of the sufficient stability condition does not rely on Poisson arrivals and could be extended to a general arrival process. The extension of the proof of the asymptotically necessary condition is more involved. Another interesting topic for further research is to extend the developed framework to obtain the stability condition for more general service requirements. Finally, it might be possible to prove the stability condition in a c.o.c. redundancy system with i.i.d. service requirements with the additional feature of fork-join service. In such a system, a replica is created on d servers and k out of d replicas must be served completely. We expect that the notion of synchronicity continues to holds. More specifically, we expect that Proposition 1 (the sufficient stability condition) still holds when replacing $\mathbb{E}[\min\{B_1, \ldots, B_d\}]$ by the expectation of the kth order statistic, i.e., the expectation of the kth smallest value. However, the proof of the necessary stability condition will require significantly more involved arguments.

References

1. Aktas, M.F., Peng, P., Sojanin, E.: Effective straggler mitigation: Which clones should attack and when? ACM SIGMETRICS Perf. Eval. Rev. **45**(2), 12–14 (2017)
2. Aktas, M.F., Peng, P., Sojanin, E.: Straggler mitigation by delayed relaunch of tasks. ACM SIGMETRICS Perf. Eval. Rev. **45**(3), 324–331 (2017)

3. Anton, E., Ayesta, U., Jonckheere, M., Verloop, I.M.: On the stability of redundancy models. arXiv:1903.04414 (2019)
4. Asmussen, S.: Applied Probability and Queues, 2nd edn. Springer, New York (2003)
5. Ayesta, U., Bodas, T., Dorsman, J.L., Verloop, I.M.: A token-based central queue with order-independent service rates. arXiv:1902.02137 (2019)
6. Ayesta, U., Bodas, T., Verloop, I.M.: On a unifying product form framework for redundancy models. Perf. Eval. **127–128**, 93–119 (2018)
7. Gardner, K., Harchol-Balter, M., Scheller-Wolf, A., Van Houdt, B.: A better model for job redundancy: decoupling server slowdown and job size. IEEE/ACM Trans. Netw. **25**(6), 3353–3367 (2017)
8. Gardner, K., Harchol-Balter, M., Scheller-Wolf, A., Velednitsky, M., Zbarsky, S.: Redundancy-*d*: the power of *d* choices for redundancy. Oper. Res. **65**(4), 1078–1094 (2017)
9. Gardner, K.S., Zbarsky, S., Doroudi, S., Harchol-Balter, M., Hyytia, E., Scheller-Wolf, A.: Reducing latency via redundant requests: exact analysis. ACM SIGMETRICS Perf. Eval. Rev. **43**(1), 347–360 (2015)
10. Gardner, K.S., Zbarsky, S., Doroudi, S., Harchol-Balter, M., Hyytia, E., Scheller-Wolf, A.: Queueing with redundant requests: exact analysis. Queueing Syst. **83**(3–4), 227–259 (2016)
11. Grimmett, G., Stirzaker, D.: Probability and Random Processes, 3rd edn. Oxford University Press, Oxford (2001)
12. Hellemans, T., Van Houdt, B.: Analysis of redundancy(*d*) with identical replicas. ACM SIGMETRICS Perf. Eval. Rev. **46**(3), 74–79 (2018)
13. Joshi, G.: Synergy via redundancy: boosting service capacity with adaptive replication. ACM SIGMETRICS Perf. Eval. Rev. **45**(3), 21–28 (2017)
14. Joshi, G., Soljanin, E., Wornell, G.: Efficient replication of queued tasks for latency reduction in cloud systems. In: *Proc. of the Allerton Conf.* (2015)
15. Poloczek, F., Ciucu, F.: Contrasting effects of replication in parallel systems: from overload to underload and back. ACM SIGMETRICS Perf. Eval. Rev. **44**(1), 375–376 (2016)
16. Shah, N.B., Lee, K., Ramchandran, K.: When do redundant requests reduce latency? IEEE Trans. Commun. **64**(2), 715–722 (2016)
17. Squillante, M.S., Xia, C.H., Yao, D.D., Zhang, L.: Threshold-based priority policies for parallel-server systems with affinity scheduling. Proc. Am. Control Conf. **4**, 2992–2999 (2001)
18. Vulimiri, A., Godfrey, P.B., Mittal, R., Sherry, J., Ratnasamy, S., Shenker, S.: Low latency via redundancy. Proc. ACM CoNEXT **2013**, 283–294 (2013)