# Capacity Analysis of Distributed Computing Systems with Multiple Resource Types

Pengchao Han*†, Shiqiang Wang‡, Kin K. Leung†
*College of Computer Science and Engineering, Northeastern University, China
†Department of Electrical and Electronic Engineering, Imperial College London, UK
‡IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
Email: hanpengchao199@gmail.com, wangshiq@us.ibm.com, kin.leung@imperial.ac.uk

***Abstract*—In cloud and edge computing systems, computation, communication, and memory resources are distributed across different physical machines and can be used to execute computational tasks requested by different users. It is challenging to characterize the capacity of such a distributed system, because there exist multiple types of resources and the amount of resources required by different tasks is random. In this paper, we define the capacity as the number of tasks that the system can support with a given overload/outage probability. We derive theoretical formulas for the capacity of distributed systems with multiple resource types, where we consider the power of $d$ choices as the task scheduling strategy in the analysis. Our analytical results describe the capacity of distributed computing systems, which can be used for planning purposes or assisting the scheduling and admission decisions of tasks to various resources in the system. Simulation results using both synthetic and real-world data are also presented to validate the capacity bounds.**

***Index Terms*—Capacity analysis, cloud and edge computing, distributed systems, multiple resource types, power of $d$ choices**

## I. Introduction

Cloud computing allows flexible configuration of high-level services and sharing of resources such as computation, memory, and communication among users. When the resources are in close proximity to end users, they can be shared in a similar manner in an edge computing system [1], [2]. From the system perspective, the infrastructure of cloud and edge computing corresponds to a distributed computing network with different types of servers (machines) and communication resources for executing computational tasks. These interconnected machines may belong to one or multiple owners, where each owner corresponds to a domain as shown in Fig. 1. An owner can be a cloud/edge platform provider or any entity that owns a part of the physical system. Such scenarios exist in both business/civilian applications as well as in defense applications involving coalitions [3].

In such a multi-domain distributed computing system, a natural question to ask is: *how many tasks can each domain process simultaneously at any time instant?* The answer to this question is important for system planning purposes. For example, a service provider may need to decide whether to sign a contract with a domain owner depending on the amount of resources that the domain can provide in a particular region; a domain owner may need to pre-compute the number of servers it needs to deploy depending on the expected demand. Furthermore, it is often useful to characterize the relationship between system configuration and the number of tasks that the system can support, for both theoretical and practical understanding. Toward this goal, we analyze the system *capacity* in this paper, which is defined as the maximum number of tasks that a domain can process simultaneously.

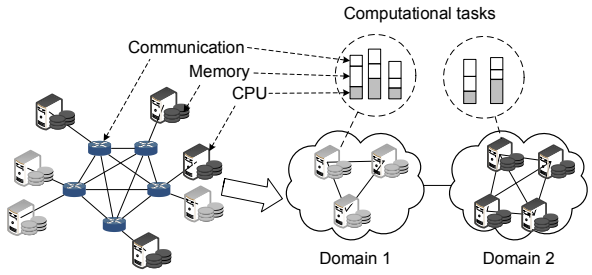


Fig. 1. Distributed computing systems with multiple resource types.

Since the capacity of the system depends on the task assignment strategy, in this paper we consider the power of $d$ choices (PODC) for assigning tasks, which is widely used in theoretical analysis and practical systems to achieve the tradeoff between load balancing and communication overhead [4], [5]. Generally, $d \geq 2$ machines[1] are selected uniformly and randomly from $N$ machines for each task and the least-loaded one among the $d$ selected machines will be chosen to process the task. The benefit of PODC is that the tradeoff between system capacity and control overhead is controllable, i.e., a larger $d$ gives a larger system capacity but also requires higher control overhead.

Using PODC as the task assignment strategy, we derive analytical expressions of the capacity of distributed systems with multiple resource types and random resource requirements of tasks. Because the exact capacity expression is very difficult to obtain, we present an *achievable lower bound* and an *upper bound* of capacity. Through simulations with both synthetic

P. Han's contribution to this work was made when she was at Imperial College London.

This research was sponsored in part by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-16-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

[1]PODC can be generalized to $d \geq 1$, we will consider PODC with $d \geq 1$ later in this paper.

task arrivals and real-world task arrival traces collected in a data center, we show that these capacity bounds can closely approximate the actual capacity of the system when parameters in the expressions are properly tuned.

The remainder of this paper is organized as follows. In Section II, we review the related work. Section III presents the system model. In Section IV, we describe our capacity analysis with theoretical results. Section V presents numerical results. Section VI draws conclusion.

## II. Related Work

Although many capacity-achieving scheduling mechanisms exist in the literature [6]–[12], they do not provide an analytical expression of the capacity value. Instead, they consider a capacity region of the system that is described with multiple constraints, which is much more difficult to interpret than a single value. In this paper, we take a different approach (and use a slightly different capacity definition) and focus on obtaining analytical expressions of the capacity value.

Our capacity definition shares some similarity with the effective capacity [13]–[15] and Erlang capacity [16], [17], both of which are widely explored in wireless communication to find the maximum arrival rate that a wireless link can support with a constrained probability of delay violation or blocking users. However, the effective and Erlang capacity notions do not capture the case with multiple resource types.

Literature related to PODC focuses on analyzing the maximum load among machines for single resource type [18]–[20] or identical tasks with the same requirement for each kind of resource [21], [22]. To the best of our knowledge, the scenario where different tasks can require different (random) amounts of resources, in the case with multiple resource types, has not been studied. The capacity notion in this scenario has not been defined in existing literature either. We fill the gap by addressing these problems.

## III. System Model and Definitions

We consider a system with multiple types of distributed resources. There are $N$ distributed machines, each has $R$ types of resources (e.g., computation, communication, and memory). The available amount of type-$r$ resource at machine $n$ is normalized to 1 for any $n$ and $r$. There are $T$ tasks running on the machines in total. The requirement of task $t$ for resource type $r$ is a random variable $X_{t,r} \in [0,1]$. For each resource type, we assume that the requirements of all tasks for this type of resource (i.e., $X_{t,r}, \forall t$) are independent and identically distributed (i.i.d.), while different distributions may apply to different resource types (i.e., the distributions of $X_{t,r}$ and $X_{t,r'}$ for $r \neq r'$ may be different). Note that although this i.i.d. assumption is required for theoretical analysis, the results in Section V-B empirically validate that our capacity bounds still hold on real datasets where this assumption may not hold.

In the system, each task is assigned to one machine that provides resources for this task. Denote $\rho_{n,r}$ as the amount of currently utilized type-$r$ resource at machine $n$, which is equal to the total requirements for type-$r$ resource of tasks allocated to machine $n$. To facilitate the capacity analysis later, let $\rho_n := \max_r \rho_{n,r}$ denote the maximum resource utilization among all resources on machine $n$.

The task allocation follows PODC with $d \geq 1$. When a new task arrives, $d \geq 1$ machines are randomly chosen according to a uniform distribution. The task is assigned to the machine with the minimum $\rho_n$, among the $d$ selected machines. During the task assignment process, information on resource utilization is exchanged between the $d$ randomly selected machines and a controller, so that the controller can determine which machine is the least-occupied, i.e., has the smallest $\rho_n$. Different values of $d$ have different control overheads and abilities to balance workload, thus leading to different numbers of tasks that the system can process.

**Definition 1** (Capacity)**.** We define the $\varepsilon$-capacity of a distributed computing system as the maximum number of tasks, denoted by $M$, that the system can serve simultaneously, such that the overload probability is not higher than $\varepsilon$ ($\varepsilon > 0$), i.e.,

$$\Pr\left\{\bigcup_{r=1}^{R}\left(\bigcup_{n=1}^{N}\left[\rho_{n,r} \geq 1\right]\right)\right\} \leq \varepsilon. \tag{1}$$

We analyze this capacity in the next section.

## IV. Capacity Analysis

The goal of our capacity analysis is to obtain upper bounds of $M$ that serve as sufficient and necessary conditions of (1). The sufficient and necessary conditions give lower and upper bounds of capacity, respectively.

### A. Preliminary Lemma

The following lemma is used for the approximation of PODC in the derivation of sufficient and necessary conditions.

**Lemma 1.** *For vector $\boldsymbol{a} = (a_1, \cdots, a_N)$ with $a_n \geq a_{n+1}$ for $n \in \{1, \cdots, N-1\}$ and probability vector $\boldsymbol{p} = (p_1, \cdots, p_N)$ with $p_n \leq p_{n+1}$ for $i \in \{1, \cdots, N-1\}$ and $\sum_{n=1}^{N} p_n = 1$, define the weighted average of $\boldsymbol{a}$ as $\sum_{n=1}^{N} p_n a_n$, then*

$$\sum_{n=1}^{N} p_n a_n \leq \bar{p} \sum_{n=1}^{N} a_n, \tag{2}$$

*where $\bar{p} = 1/N$ is the mean of all elements in $\boldsymbol{p}$.*

*Proof.* Assume we have $p_i$ such that $p_i \leq \bar{p} \leq p_{i+1}$, the difference between $\bar{p}\sum_{n=1}^{N} a_n$ and $\sum_{n=1}^{N} p_n a_n$ is

$$\begin{aligned}\bar{p}\sum_{n=1}^{N} a_n - \sum_{n=1}^{N} p_n a_n &= \sum_{n=1}^{i} (\bar{p} - p_n)\, a_n - \sum_{n=i+1}^{N} (p_n - \bar{p})\, a_n \\ &\geq \sum_{n=1}^{i} (\bar{p} - p_n)\, a_i - \sum_{n=i+1}^{N} (p_n - \bar{p})\, a_i. \end{aligned} \tag{3}$$

Based on the fact that $\sum_{n=1}^{N} \bar{p} = \sum_{n=1}^{N} p_n$, we have $\bar{p}\sum_{n=1}^{N} a_n - \sum_{n=1}^{N} p_n a_n \geq 0$, and the claim follows. □

### B. Markov chain

Define vector $\boldsymbol{\rho}(t) := (\rho_n(t), \forall n)$, where $\rho_n(t) := \max_r \rho_{n,r}(t)$ is the maximum utilization among all resource

types at machine $n$, after the $t$-th task has been assigned. The task assignment process defines a Markov chain over the vector $\boldsymbol{\rho}(t)$ as follows:

1) Choose the machine $i \in \{1, \cdots, N\}$ for the newly arriving $t$-th task according to PODC.
2) For all $r = 1, 2, ..., R$:
   a) Sample $X_{t+1,r}$ as the requirement of type-$r$ resource of the $t$-th task, from a given probability distribution.
   b) Set $\rho_{n,r}(t+1) = \rho_{n,r}(t) + X_{t+1,r}$ for $n = i$ (i.e., machine $i$ is chosen for the $t$-th task) and $\rho_{n,r}(t+1) = \rho_{n,r}(t)$ for $n \neq i$.

We also define a probability vector $\boldsymbol{p} := (p_1, \cdots, p_N)$, where $p_i$ denotes the probability that the new task $t+1$ is assigned to the $i$-th most loaded machine in terms of $\{\rho_n(t) : \forall n\}$. If we rank $\boldsymbol{\rho}(t)$ in a non-increasing order such that $\rho_1(t) \geq \rho_2(t) \geq \cdots \geq \rho_N(t)$, we have $p_1 \leq p_2 \leq \cdots \leq p_N$ due to the use of PODC strategy. Whenever the context is clear, we write $\boldsymbol{\rho}$, $\rho_n$ and $\rho_{n,r}$ instead of $\boldsymbol{\rho}(t)$, $\rho_n(t)$ and $\rho_{n,r}(t)$.

### C. Sufficient Condition for (1)

We first provide the following lemma that serves as an intermediate sufficient condition for further analysis.

**Lemma 2.** *For any task assignment strategies, if*

$$\sum_{n=1}^{N} E\left(e^{\theta\rho_n}\right) \leq \frac{\varepsilon e^{\theta}}{R} \tag{4}$$

*for some $\theta > 0$, then the overload probability less than $\varepsilon$ in (1) is guaranteed.*

*Proof.* We note that $E\left(e^{\theta\rho_n}\right)$ is the moment generating function (MGF) of $\rho_n$ for $\theta > 0$, Chernoff's bound can be applied. Thus, we have

$$\Pr(\rho_n \geq 1) \leq \frac{E\left(e^{\theta\rho_n}\right)}{e^{\theta}}, \theta > 0. \tag{5}$$

Substituting (5) into (4) gives

$$R\sum_{n=1}^{N} \Pr(\rho_n \geq 1) \leq \varepsilon.$$

Applying Boole's inequality to the left side of above, we have

$$\varepsilon \geq R\sum_{n=1}^{N} \Pr(\rho_n \geq 1) \geq \sum_{n=1}^{N}\sum_{r=1}^{R} \Pr(\rho_{n,r} \geq 1)$$
$$\geq \Pr\left\{\bigcup_{r=1}^{R}\left(\bigcup_{n=1}^{N}[\rho_{n,r} \geq 1]\right)\right\}.$$

Thus, the overload probability in (1) is confirmed. □

Denote $G(\theta) := E\left(e^{\theta \max_r X_{t,r}}\right)$ with $\theta > 0$ for any task $t$ and resource type $r$ as the MGF of $\max_r X_{t,r}$, which can be calculated based on the probability density functions (PDFs) of $X_{t,r}$ for all $r$ (recall that the PDFs of $X_{t,r}$ for different $t$ values are the same, for some given $r$). Using Lemma 2, we obtain the following sufficient condition for (1).

**Theorem 1.** *For the PODC strategy with $d \geq 1$, if the number of tasks is less than or equal to*

$$T_l := \frac{\log\frac{\varepsilon}{NR} + \theta}{\log(\omega G(\theta) + N - \omega) - \log N}, \tag{6}$$

*for some $\theta > 0$ and $0 < \omega \leq 1$, then the overload probability in (1) is guaranteed.*

*Proof.* We first define $\Phi(t) := \sum_{n=1}^{N} e^{\theta\rho_n(t)}$ and calculate the mean of $\Phi(t)$. Rank $\boldsymbol{\rho}(t)$ in the non-increasing order such that $p_1 \leq p_2 \leq \cdots \leq p_N$ for task $t+1$. The mean increment of $\Phi(t)$ can be calculated as follows:

$$\begin{aligned} &E[\Phi(t+1) - \Phi(t) | \boldsymbol{\rho}(t)] \\ &= E\left[\sum_{n=1}^{N}\left(e^{\theta\rho_n(t+1)} - e^{\theta\rho_n(t)}\right)\middle|\boldsymbol{\rho}(t)\right] \\ &\leq \sum_{i=1}^{N} p_i E\left[e^{\theta(\rho_i(t)+\max_r X_{t+1,r})} - e^{\theta\rho_i(t)}\middle|\boldsymbol{\rho}(t)\right] \qquad (7) \\ &= (G(\theta) - 1)\sum_{i=1}^{N} p_i\left(e^{\theta\rho_i(t)}\right) \leq \frac{\omega G(\theta) - \omega}{N}\Phi(t). \end{aligned}$$

In the above, with probability $p_i$ a newly arrived task is allocated to machine $i$, whose utilization will increase by $X_{t+1,r}$ for each resource type $r$. The change of $e^{\theta\rho_n(t)}$ on other machines is 0, i.e., $e^{\theta\rho_n(t+1)} - e^{\theta\rho_n(t)} = 0$ for $n \neq i$, which gives (7), where the inequality is because $\rho_i(t+1) \leq \rho_i(t) + \max_r X_{t+1,r}$. Moreover, since $\sum_{i=1}^{N}\left(p_i e^{\theta\rho_i(t)}\right)$ is a weighted average of $e^{\theta\rho_i}$ with higher weights for smaller elements and $\sum_{i=1}^{N} p_i = 1$, we have $\sum_{i=1}^{N}\left(p_i e^{\theta\rho_i(t)}\right) \leq \Phi(t)/N$ according to Lemma 1. This gives the last inequality for $\omega = 1$.

Because $E[E(X|Y)] = E[X]$, from the above we have

$$\begin{aligned} E[\Phi(t+1) - \Phi(t)] &= E[E[\Phi(t+1) - \Phi(t)|\boldsymbol{\rho}(t)]] \\ &\leq \frac{\omega G(\theta) - \omega}{N} E[\Phi(t)]. \end{aligned}$$

That is,

$$E[\Phi(t+1)] \leq \frac{\omega G(\theta) + N - \omega}{N} E[\Phi(t)].$$

For $T$ tasks in total ($T \leq T_l$), because $E[\Phi(0)] = \Phi(0) = \sum_{n=1}^{N} e^{\theta 0} = N$, we have

$$\begin{aligned} \sum_{n=1}^{N} E\left[e^{\theta\rho_n(T)}\right] = E[\Phi(T_l)] &\leq N\left(\frac{\omega G(\theta) + N - \omega}{N}\right)^{T} \\ &\leq N\left(\frac{\omega G(\theta) + N - \omega}{N}\right)^{T_l} = \frac{\varepsilon e^{\theta}}{R} \end{aligned}$$

for some properly chosen $\omega$, where the first equality is due to the linearity of expectation, the last inequality is because $T \leq T_l$ and $\frac{\omega G(\theta)+N-\omega}{N} \geq 1$ for $\omega = 1$, and the last equality is from the definition of $T_l$ in (6). The above is equivalent to (4) in Lemma 2, hence we have proved the theorem. □

### D. Necessary Condition for (1)

We provide the following lemma that serves as an intermediate necessary condition for further analysis.

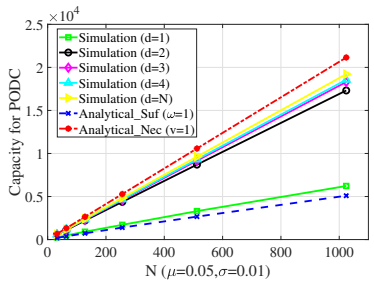


Fig. 2. Performance of capacity bounds for Gaussian resources with $\omega = v = 1$.

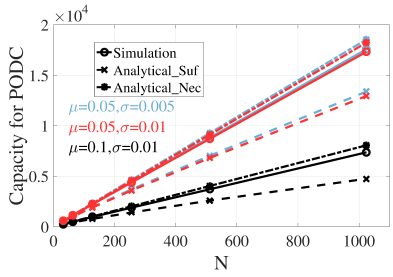


Fig. 3. Comparison of capacity bounds with different $\mu$ and $\sigma$ for a single resource type (i.e., $R = 1$) and tuned $\omega$ and $v$.

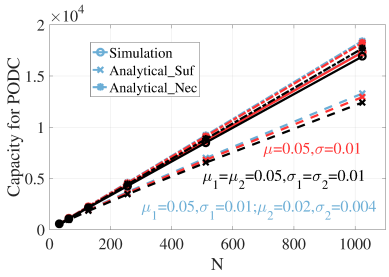


Fig. 4. Comparison of capacity bounds with different $R$ ($R = 1$ for $\mu = 0.05, \sigma = 0.01$ and $R = 2$ elsewhere) and tuned $\omega$ and $v$.

**Lemma 3.** *For any task assignment that satisfies (1) with some given $\varepsilon$, we have*

$$\frac{1}{N}\sum_{n=1}^{N} E\left(e^{-\theta\rho_n}\right) \geq \frac{1-\varepsilon}{e^{\theta}} \tag{8}$$

*for any $\theta > 0$.*

*Proof.* A necessary condition for (1) is

$$\frac{1}{N}\sum_{n=1}^{N} \Pr\left(\rho_n \geq 1\right) \leq \max_n \Pr\left(\rho_n \geq 1\right) \tag{9}$$
$$\leq \Pr\left\{\bigcup_{r=1}^{R}\left(\bigcup_{n=1}^{N}\left[\rho_{n,r} \geq 1\right]\right)\right\} \leq \varepsilon.$$

Using Chernoff's bound with $\theta > 0$, we have

$$1 - E\left[e^{-\theta\rho_n}\right] e^{\theta} \leq \Pr\left(\rho_n \geq 1\right). \tag{10}$$

We then apply (10) to the left-hand side of (9) to prove the lemma. □

Denote $H(\theta) := E\left(e^{-\theta \min_r X_{t,r}}\right)$ with $\theta > 0$ for any task $t$ and resource type $r$ as the MGF (with negative parameter) of $\min_r X_{t,r}$, which can be obtained with the PDFs of $X_{m,r}$ for all $r$. Note that although we take the minimum of $\{X_{m,r} : \forall r\}$ here, $\rho_n$ is still defined as the maximum of $\{\rho_{n,r} : \forall r\}$ (see Section III). We have the following result.

**Theorem 2.** *For the PODC strategy with $d \geq 1$ that satisfies (1) with some given $\varepsilon$, the system capacity $M$ satisfies*

$$M \leq T_u := \frac{\log\left(1-\varepsilon\right) - \theta}{\log\left(vH(\theta) + N - v\right) - \log N} \tag{11}$$

*for any $\theta > 0$ and some $v \geq 1$.*

*Proof.* Define $\Psi_n\left(t\right) := e^{-\theta\rho_n(t)}$ for machine $n$ and rank $\boldsymbol{\rho}\left(t\right)$ in non-increasing order such that $p_1 \leq p_2 \leq \cdots \leq p_N$ for task $t+1$. The mean increment of $\Psi_n\left(t\right)$ for machine $n$ is

$$\begin{aligned} &E\left[\Psi_n\left(t+1\right) - \Psi_n\left(t\right)|\boldsymbol{\rho}\left(t\right)\right] \\ &\leq p_n e^{-\theta\left(\rho_n(t) + \min_r X_{t+1,r}\right)} + \left(1-p_n\right)e^{-\theta\rho_n(t)} - e^{-\theta\rho_n(t)} \\ &= \left(H(\theta) - 1\right)p_n\Psi_n\left(t\right). \end{aligned}$$

Define $\Psi\left(t\right) := \frac{1}{N}\sum_{n=1}^{N}\Psi_n\left(t\right)$, we have

$$\begin{aligned} &E\left[\Psi\left(t+1\right) - \Psi\left(t\right)|\boldsymbol{\rho}\left(t\right)\right] \\ &= \frac{1}{N}\sum_{n=1}^{N} E\left[\Psi_n\left(t+1\right) - \Psi_n\left(t\right)|\boldsymbol{\rho}\left(t\right)\right] \\ &\leq \frac{1}{N}\sum_{n=1}^{N}\left(H(\theta)-1\right)p_n\Psi_n\left(t\right) \leq \left(H(\theta)-1\right)\frac{v}{N}\Psi\left(t\right), \end{aligned}$$

where Lemma 1 is used in the last step by considering that $\left(H(\theta)-1\right)p_n$ decreases with $n$ (note that $H(\theta)-1<0$) and $\Psi_n\left(t\right)$ increases with $n$, hence the result holds for $v = 1$. Using $E\left[E\left(X|Y\right)\right] = E\left[X\right]$, we have

$$\begin{aligned} E\left[\Psi\left(t+1\right)\right] &= E\left[E\left[\Psi\left(t+1\right) - \Psi\left(t\right)|\boldsymbol{\rho}\left(t\right)\right]\right] + E\left[\Psi\left(t\right)\right] \\ &\leq \left(\left(H(\theta)-1\right)\frac{v}{N} + 1\right)E\left[\Psi\left(t\right)\right]. \end{aligned}$$

For $M$ tasks in total, based on the above and using $E\left[\Psi\left(0\right)\right] = \sum_{n=1}^{N} e^{-\theta 0}/N = 1$, we have

$$\left(\frac{vH(\theta)+N-v}{N}\right)^{M} \geq E\left[\Psi(M)\right] = \frac{1}{N}\sum_{n=1}^{N} E\left[e^{-\theta\rho_n(M)}\right] \geq \frac{1-\varepsilon}{e^{\theta}}$$

where the equality is due to the linearity of expectation and the last inequality is from Lemma 3. Rearranging the above to solve for $M$ proves the theorem. □

### E. Discussion

The sufficient condition for (1) given by Theorem 1 represents a lower bound of capacity, because the system is guaranteed to support less than or equal to $T_l$ tasks with an overload probability of $\varepsilon$, where we recall that the capacity is defined as the *maximum* number of tasks the system can support. The necessary condition for (1) given by Theorem 2 gives an upper bound of capacity. Hence, the actual capacity $M$ is bounded by $T_l \leq M \leq T_u$. We note that Theorems 1 and 2 always hold when $\omega = v = 1$, but the parameters $\omega$ and $v$ can be tuned to obtain a tighter bound.

## V. Numerical Results

We compare our analytical lower and upper bounds of capacity with the actual capacity obtained from simulation, where both Gaussian-distributed and real-world task resource requirements are considered. The MGFs $G(\theta)$ and $H(\theta)$ are computed numerically according to the distribution in each case (the distribution is explained in further details below). The parameter $\theta$ in the MGF used in the computation of capacity bounds is also determined numerically via linear search, where

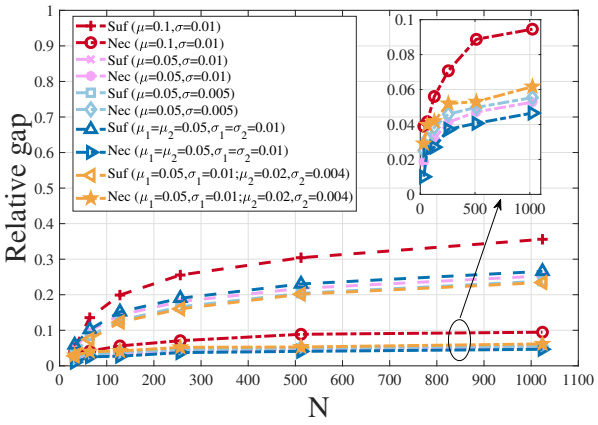


Fig. 5. Relative gaps of capacity bounds for Gaussian resource requirements with tuned $\omega$ and $v$.

TABLE I
VALUES OF $w$ AND $v$ FOR GAUSSIAN RESOURCE REQUIREMENTS

| **Parameter settings** | $\omega$ | $v$ |
|---|---|---|
| $R=1, \mu=0.1, \sigma=0.01$ | 0.28 | 1.33 |
| $R=1, \mu=0.05, \sigma=0.01$ | 0.41 | 1.16 |
| $R=1, \mu=0.05, \sigma=0.005$ | 0.41 | 1.14 |
| $R=2, \mu_1=\mu_2=0.05, \sigma_1=\sigma_2=0.001$ | 0.34 | 1.34 |
| $R=2, \mu_1=0.05, \sigma_1=0.01, \mu_2=0.01, \sigma_2=0.004$ | 0.28 | 2.84 |

we choose $\theta$ that gives the largest lower bound and smallest upper bound, so that the bound remains as tight as possible. In all simulations, we fix $\varepsilon = 0.01$.

For simplicity, we use "simulation" to denote the simulated capacity and "analytical" as the results of analytical bounds in the figures presented in the following. We also use "Suf" and "Nec" to denote the sufficient condition (lower bound) and the necessary condition (upper bound) of capacity, respectively.

### A. Performance of Gaussian Resource Requirements

We first consider a single resource type where the amount of resource requested by each task follows a Gaussian distribution with parameters $\mu$ and $\sigma^2$. Fixing $\mu = 0.05$ and $\sigma = 0.01$, Fig. 2 shows the capacity for different $d$ in PODC, when the number of machines varies. We see that $d = N$ performs the best and $d = 2$ outperforms $d = 1$ significantly, which is consistent with known results [4]. Moreover, for this case where $\omega = v = 1$, the theoretical upper and lower capacity bounds hold for all $d$ values with $1 \leq d \leq N$.

The parameters $\omega$ and $v$ in the capacity bounds can be tuned empirically so that the theoretical bounds provide a better approximation for the actual capacity values. For example, we can find the appropriate values of $\omega$ and $v$ by minimizing the gaps between the simulation results and the analytical bounds when the number of machines $N \in \{20, 40, 60, 80, 100\}$ (a set of settings with small number of machines) . The best $\omega$ and $v$ values found with this approach are shown in Table I, for different resource requirement distributions, where we recall that $R$ denotes the number of resource types. We use these tuned $\omega$ and $v$ parameters in cases with much larger $N$ in our simulations presented next.

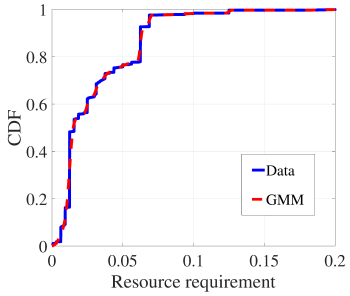


Fig. 6. GMM distribution fitting results for CPU resource.

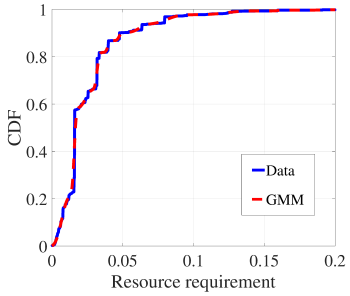


Fig. 7. GMM distribution fitting results for memory resource.

Fig. 3 shows the results with different values of $\mu$ and $\sigma$ for $d = 2$, where a lower value of mean ($\mu$) and standard deviation ($\sigma$) in the amount of resource required by each task leads to a larger capacity (i.e., more tasks can be served), as one would intuitively expect. The comparison of single and multiple (two) resource types for $d = 2$ is shown in Fig. 4. We can see that when there are multiple types of resources, the capacity of is dominated by the resource type with larger $\mu$ (i.e., the most heavily utilized resource). However, it is uncertain whether multiple resource types will cause higher or lower capacity compared to the case with a single resource type, when the maximum mean value is the same.

We also see that our analytical bounds are close to the simulation results in Figs. 3 and 4. The relative gaps (defined as the difference between the simulated capacity and analytical bounds, divided by the simulated capacity), as shown in Fig. 5, are around 0.2 for sufficient conditions (lower bounds) and below 0.1 for necessary conditions (upper bounds). In addition, the analytical bounds can capture the capacity differences when the parameters $\mu$, $\sigma$, $R$, and $N$ are different.

### B. Performance of Real-World Resource Requirements

The Google cluster dataset [23] captures the task request dynamics in a real-world computing cluster. It includes the requirements for CPU and memory resources of over $45,000,000$ tasks. To predict the capacity of a distributed system with such task resource requirements, we fit a Gaussian mixture model (GMM) using the dataset and use the MGF of this GMM. Figs. 6 and 7 show the results of GMM fitting for CPU and memory resources respectively. We see that the GMM closely captures the underlying data distribution.

Based on the fitted GMM, Figs. 8 and 9 show the performance of the analytical bounds without or with parameter

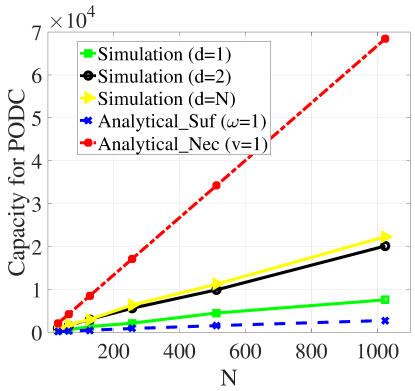


Fig. 8. Performance of capacity bounds for real data with $\omega = v = 1$.

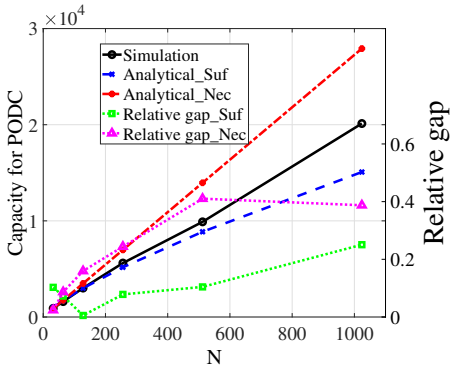


Fig. 9. Performance and relative gaps of capacity bounds for real data with tuned parameters $\omega$ and $v$.

tuning as well as the relative gaps. In the case where $\omega$ and $v$ are tuned, their values are empirically determined as $\omega = 0.14, v = 2.42$, where the tuning follows the same approach as in Section V-A. The comparison between different capacity results follow a similar trend as in the Gaussian case in Section V-A. We anticipate that this GMM fitting approach can be applied to other real-world datasets too.

## VI. Conclusion

We have derived theoretical bounds of the capacity of distributed computing systems with multiple resources types. The lower and upper bounds correspond to the sufficient and necessary conditions for the overload probability, respectively. The PODC task assignment strategy has been considered, where the trade-off between control overhead and system capacity can be adjusted using the parameter $d$. The numerical results have shown that our proposed capacity bounds can capture the key characteristics of system capacity.

Our results in this paper are useful for describing the capacity of distributed computing systems with multiple resource types, which can be helpful for system planning, analysis, and resource allocation, where our analytical capacity bounds can be used for approximating the actual system capacity. Future work can study the derivation of tighter capacity bounds with less limiting assumptions, and specific ways of applying such bounds to planning and allocation problems.

## References


[1] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Communications Surveys Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[2] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Communications Surveys Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[3] "International technology alliance in distributed analytics and information sciences (dais-ita)." https://en.wikipedia.org/wiki/DAIS-ITA.

[4] M. Mitzenmacher, A. W. Richa, and R. Sitaraman, "The power of two random choices: A survey of techniques and results," in *Handbook of Randomized Computing*, pp. 255–312, 2000.

[5] L. Ying, R. Srikant, and X. Kang, "The power of slightly more than one sample in randomized load balancing," *Mathematics of Operations Research*, vol. 42, no. 3, pp. 692–722, 2017.

[6] S. T. Maguluri, R. Srikant, and L. Ying, "Stochastic models of load balancing and scheduling in cloud computing clusters," in *IEEE INFOCOM*, pp. 702–710, 2012.

[7] M. J. Neely, E. Modiano, and C. Li, "Fairness and optimal stochastic control for heterogeneous networks," *IEEE/ACM Transactions on Networking*, vol. 16, no. 2, pp. 396–409, 2008.

[8] M. J. Neely, E. Modiano, and C. E. Rohrs, "Dynamic power allocation and routing for time-varying wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 1, pp. 89–103, 2005.

[9] S. T. Maguluri, R. Srikant, and L. Ying, "Heavy traffic optimal resource allocation algorithms for cloud computing clusters," *Performance Evaluation*, vol. 81, pp. 20–39, 2014.

[10] R. Urgaonkar, S. Wang, T. He, M. Zafer, K. Chan, and K. K. Leung, "Dynamic service migration and workload scheduling in edge-clouds," *Performance Evaluation*, vol. 91, pp. 205–228, 2015.

[11] C.-K. Huang, S.-H. Shen, C.-Y. Huang, T.-L. Chin, and C.-A. Shen, "S-cache: Toward an low latency service caching for edge clouds," in *Proceedings of the ACM MobiHoc Workshop on Pervasive Systems in the IoT Era*, pp. 49–54, 2019.

[12] A. Khalili, S. Zarandi, and M. Rasti, "Joint resource allocation and offloading decision in mobile edge computing," *IEEE Communications Letters*, vol. 23, no. 4, pp. 684–687, 2019.

[13] D. Wu and R. Negi, "Effective capacity: a wireless link model for support of quality of service," *IEEE Transactions on Wireless Communications*, vol. 2, no. 4, pp. 630–643, 2003.

[14] A. Helmy, L. Musavian, and T. Le-Ngoc, "Energy-efficient power adaptation over a frequency-selective fading channel with delay and power constraints," *IEEE Transactions on Wireless Communications*, vol. 12, no. 9, pp. 4529–4541, 2013.

[15] X. Zhang and Q. Zhu, "Statistical QoS provisioning over D2D-offloading based 5G multimedia big-data mobile wireless networks," in *IEEE INFOCOM WKSHPS*, pp. 742–747, 2018.

[16] S. Thaherbasha and S. N. parveen, "Erlang capacity estimation of ofdma-based cellular systems under co-channel interference," in *International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, 2016.

[17] R. Framjee, *Some Considerations of Performance and Capacity of Voice Over IP High Bit Rate Wireless Reverse Links*. PhD thesis, University of Texas at Arlington, 2014.

[18] P. Berenbrink, T. Friedetzky, Z. Hu, and R. Martin, "On weighted balls-into-bins games," *Theoretical Computer Science*, vol. 409, no. 3, pp. 511–520, 2008.

[19] Y. Peres, K. Talwar, and U. Wieder, "The (1 + $\beta$)-choice process and weighted balls-into-bins," in *Proceedings of ACM-SIAM Symposium on Discrete Algorithms*, pp. 1613–1619, 2010.

[20] U. Wieder, "Hashing, load balancing and multiple choice," *Foundations and Trends in Theoretical Computer Science*, vol. 12, no. 3-4, pp. 275–379, 2017.

[21] Q. Xie, X. Dong, Y. Lu, and R. Srikant, "Power of d choices for large-scale bin packing: A loss model," *SIGMETRICS Perform. Eval. Rev.*, vol. 43, no. 1, pp. 321–334, 2015.

[22] Y. Azar, I. R. Cohen, and D. Panigrahi, "Randomized algorithms for online vector load balancing," in *Proceedings of ACM-SIAM Symposium on Discrete Algorithms*, pp. 980–991, 2018.

[23] C. Reiss, J. Wilkes, and J. L. Hellerstein, "Google cluster-usage traces: format+ schema," *Google Inc., White Paper*, pp. 1–14, 2011.