The Generalized c/μ Rule for Queues with Heterogeneous Server Pools

Zhenghua Long
School of Business, Nanjing University, Nanjing 210093, China
zlong@nju.edu.cn

Hailun Zhang
Institute for Data and Decision Analytics,
The Chinese University of Hong Kong, Shenzhen, Shenzhen 518172, China
zhanghailun@cuhk.edu.cn

Jiheng Zhang
Department of Industrial Engineering & Decision Analytics,
The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong S.A.R, China
jiheng@ust.hk

Zhe George Zhang
Department of Decision Sciences, Western Washington University, Bellingham, Washington 98225; and
Beedie School of Business, Simon Fraser University, Burnaby, British Columbia V5A 1S6, Canada
george.zhang@wwu.edu

We study the optimal control of a queueing model with a single customer class and heterogeneous server pools. The main objective is to strike a balance between the holding cost of the queue and the operating costs of the server pools. We introduce a target-allocation policy, which assigns higher priority to the queue or pools without enough customers, for general cost functions. Although we can prove its asymptotic optimality, implementation requires solving a nonlinear optimization problem. When the cost functions are convex, we propose a dynamic priority policy referred to as the Gc/μ rule, which is much easier to implement. When the cost functions are concave, it turns out that a fixed priority policy is optimal. We also consider an extension to minimize the operating cost of the server pools while satisfying a service-level target for customers waiting in the queue. We develop hybrid routing policies, combining a threshold policy for the queue and the aforementioned policies for the server pools, for different types of operating cost functions. Moreover, the hybrid routing policies coincide with several classical policies in the literature in special cases. Extensive simulation experiments demonstrate the efficacy of our proposed policies.

Key words: inverted-V model, many-server queue, fluid model, general cost, dynamic priority

1. Introduction

Motivated by various service systems in call centers and the healthcare industry (e.g., Tezcan (2008), Mandelbaum et al. (2012)), we study the inverted-V model (a terminology coined by Armony (2005)) with a single customer class who may abandon the system when their patience is exhausted and many heterogeneous server pools handling customers at different service rates and costs. The fundamental problem in the control of the inverted-V model is to decide whether an

arriving customer should be queued in the buffer, and if not, which server pool we should use. In this paper, we design control policies to achieve the following two objectives separately. In the first objective, customers waiting in the queue incur a holding cost; thus, the goal is to minimize the total long-run average holding and operating costs by finding the optimal trade-off. The second objective is to minimize the long-run average operating cost while satisfying a target service level in terms of the long-run abandonment proportion.

Holding and operating costs trade-off. We first consider the problem of routing arrivals to join the queue or enter service in order to strike a balance between the holding cost from the queue (including the queue-length cost and the abandonment penalty) and the operating cost from the server pools (idle servers have no operating cost). Allowing more idle servers may lead to excessive waiting and customer abandonment, whereas keeping more servers busy may increase the operating cost. The idiosyncratic trade-off between holding and operating costs in inverted-V models indicates that work-conserving policies might be suboptimal. This phenomenon has also been observed in a single-class many-server queue, which is a special case of the inverted-V model (see Zhan and Ward (2019), Zhong et al. (2022)).

When a customer is ready to be served, we must decide to which server pool the customer should be routed. Recently, Xia et al. (2022) introduced a fixed priority routing policy, namely the c/μ rule, and proved its optimality for linear costs, indicating that we should give higher priority to the pool with the lower operating cost but faster service rate. The c/μ rule may cause the pools with higher priority to be very busy but those with lower priority to be idle. In this paper, we allow cost functions to be general functions. The corresponding routing policy therefore becomes the generalized c/μ rule (Gc/μ) which assigns dynamic priority to the pools. As such, customers can be dynamically routed to any one of the pools, fairly utilizing all server pools. We show that the Gc/μ rule is asymptotically optimal for nonlinear convex costs. Moreover, our proposed Gc/μ rule is a parsimonious dynamic priority policy oblivious to arrival rate and service capacity information.

In contrast to the Gc/μ rule (a dynamic priority policy with convex costs), we find that for concave cost functions the optimal routing is a fixed priority policy. Akin to the convex queue length costs considered in van Mieghem (1995), convex operating costs are suitable for situations where the marginal cost of serving more customers is much higher than the marginal cost when more servers become idle. Concave operating costs are appropriate for systems where managers have strong preferences for fewer busy servers and become increasingly indifferent to pools with more busy servers (see Ata and Olsen (2009)). In order to find an optimal priority order for systems with concave costs, we need to solve a concave optimization problem, which is usually nontrivial using standard non-linear approaches. We show that the fixed priority routing problem can be

transformed into a knapsack problem, which can be solved more efficiently by using a dynamic programming algorithm.

For nonconvex and nonconcave cost functions, we propose another dynamic scheduling policy referred to as the target-allocation policy. Note that the steady state of customers in the queue and pools can be viewed as the result of allocating all customers in the system. The idea is to assign higher priority to the queue or pools that have not been assigned enough customers, which is determined by solving a nonlinear optimization problem (24). The advantage of this policy is that it is asymptotically optimal for any general cost functions. However, to implement the policy we need to solve the nonlinear programming in advance.

From the above discussion, we can always choose the most appropriate policy for different cost functions; see the last column of Table 1. In Long et al. (2020), where the authors focus on work-conserving policies, three similar control policies (see the second column of Table 1) have been developed to minimize the total holding cost of a V model for general, convex, and concave cost functions, respectively. The routing problems in this paper can be thought of as a dual version of the dynamic scheduling problems in the V model. However, they have distinct characteristics because of different cost structures and operational controls, as reflected in Table 1.

	V model in Long et al. (2020) multiple customer classes single server pool	Inverted-V model in this paper single customer class multiple server pools
Control problem	scheduling different types of customers to service	routing customers to different server pools
Objective	minimize total holding costs	trade off holding and operating costs
Work-conserving or not	work-conserving	non-work-conserving
General cost functions	target-allocation policy	target-allocation policy
Convex cost functions	the $Gc\mu/h$ rule	the Gc/μ rule
Concave cost functions	fixed-priority policy (on buffers)	fixed-priority policy (on server pools)

Table 1 Comparison of the V model in Long et al. (2020) and the inverted-V model in this paper

Minimize the operating cost with a service-level target. Another critical operational decision in the inverted-V model is to minimize the operating cost of the system but also to meet a certain service-level target p. In such a problem, the holding cost is removed from the objective function by adding a service constraint. Such a formulation is more appropriate for service systems where customer service is important but the holding cost is hard to quantify. Specifically, the service constraint is expressed as that the steady-state abandonment probability of the whole system is less than p, which can be any value between 0 to 1. The larger the service-level target, the more customers will be allowed in the buffer. This inspires us to consider a hybrid policy, where the queue follows a threshold policy (determined by p) and the pools still follow one of the aforementioned three types of routing policies to cope with general operating cost functions. Correspondingly, we formulate in Section 4 the hybrid target-allocation policy, the hybrid Gc/μ

rule, and the hybrid fixed-priority policy, under which the service-level target is reached and the total operating cost is also asymptotically optimized for general, convex and concave operating cost functions, respectively.

Different service-level targets also enable us to characterize various practical systems. As illustrated in Figure 2 of Section 6, the holding cost increases with p and the operating cost decreases with p. This implies that for systems with a low p, customers experience a short waiting time and the abandonment rate is relatively low. A typical example is a large call center for a private company's after-sales customer service. In such a system, ensuring a short waiting time is the top requirement. For systems with a medium p, an appropriate queue length is desirable. The costs of holding and operating are balanced. A good example is a tiered security check system (see, for example, Zhang et al. (2011)) or a make-to-order (subcontracting) system. For systems with a relatively high p, customers experience a long waiting time and the abandonment rate can be high. A proper example is a public service system such as a call center for a federal tax service or an elective medical service system. Usually, the servers have specialized skills and can therefore be scarce. Thus, our results have extensive application prospects in real-life operational systems.

1.1. Literature Review

Our paper contributes to three streams of literature i) research on systems under non-work-conserving policies, ii) research on systems under $c\mu$-type rules, and iii) research on systems with a single customer class and multiple server pools.

As a special case of the inverted-V model, the queueing systems with a single customer class and a many-server pool have been extensively studied in the literature from the pioneering work of Whitt (2006) by applying fluid model analysis. Convergence of the stochastic processes to the fluid limits in a many-server regime was proved in Zhang (2013) and Kang and Ramanan (2010) using measure-valued processes. Similar to our analysis, Bassamboo and Randhawa (2010), Bassamboo and Randhawa (2016), and Wu et al. (2019) used the steady states of fluid models to study routing and staffing decisions in $G/G/N+G$ systems. The focus of this line of research is on systems under work-conserving policies (that utilize all the available service resources). However, in the presence of server operating costs, Zhan and Ward (2019) and Zhong et al. (2022) found that work-conserving policies are no longer asymptotically optimal in the $M/M/N+M$ and $G/G/N+G$ settings, respectively. This paper extends such a finding to the inverted-V model showing that intentional idling is a must when designing optimal routing policies to trade off the holding and operating costs.

The $c\mu$-type rules have a long history in the study of scheduling problems in V models (multiple customer classes served by a single server pool) and have recently been applied to the study of

routing problems in inverted-V models (single customer class multiple server pools). As early as Smith (1956), the $c\mu$ rule was proposed and proved to be optimal for a multiclass $M/G/1$ system with linear holding costs. In Atar et al. (2008, 2010, 2011, 2014), it was extended to the $c\mu/\theta$ rule, which is asymptotically optimal for a multiclass many-server queueing system with exponential patience and linear holding costs. The $Gc\mu$ rule of van Mieghem (1995) appears to be the first to consider nonlinear, convex holding costs in the analysis of a multiclass $G/G/1$ queue. Mandelbaum and Stolyar (2004) generalized the $Gc\mu$ rule to a system with heterogeneous servers. The $Gc\mu/h$ rule in Long et al. (2020) extended those studied in van Mieghem (1995) and Atar et al. (2008, 2010, 2011, 2014) to a multiclass many-server queueing system with general patience and nonlinear convex holding costs. The aforementioned literature focuses on the analysis of V models. Recently, Xia et al. (2022) proposed the c/μ rule to control an inverted-V model with linear operating costs. Our Gc/μ rule extends Xia et al. (2022) to an inverted-V model with nonlinear convex operating costs and can be viewed as a counterpart of the $Gc\mu$ rule in van Mieghem (1995). Indeed, the "Gc" in the $Gc\mu$ rule is the marginal cost of the general queue-length cost, and the "Gc" in the Gc/μ rule is the marginal cost of the general server operating cost.

The third literature stream to which we contribute is on the routing policies in inverted-V models. Armony (2005) analyzed the fastest-server-first (FSF) routing policy that assigns customers to the fastest available pool, showing that it asymptotically minimizes the stationary queue length and waiting time. Armony and Mandelbaum (2011) extended this result to accommodate abandonments. Tezcan (2008) proposed the load-balancing (LB) policy in order to have all server pools in the inverted-V model fairly utilized. The idleness-ratio (IR) policy, which is a special case of the queue-and-idleness ratio (QIR) in Gurvich and Whitt (2009a,b, 2010), routes customers to the pool with the highest idleness imbalance. Armony and Ward (2010) analyzed the longest-idle-server-first (LISF) policy in the inverted-V model with two pools. Atar et al. (2011) proposed the longest-idle-pool-first policy that routes a customer to the pool with the longest cumulative idleness among the available pools in order to balance cumulative idleness among the pools. Mandelbaum et al. (2012) introduced the randomized most-idle (RMI) routing policy, which achieves the same server fairness as the LISF policy. As mentioned earlier, Xia et al. (2022) proposed the c/μ rule, which routes customers to the pool with low operating cost but high service rate as much as possible. Most of these papers analyzed the inverted-V model based on diffusion model analysis or by formulating the routing problem as a Markov decision process. In contrast, our work employs fluid model analysis and covers the same policies proposed in the literature, including the fluid version of the LB policy in Tezcan (2008), the IR policy in Gurvich and Whitt (2009a,b, 2010), the c/μ rule in Xia et al. (2022) and the FSF policy in Armony (2005).

1.2. Organization

The remainder of this paper is organized as follows. In Section 2, we introduce the inverted-V model containing a single customer class and multiple server pools. We also establish the corresponding fluid model and then formulate a steady-state optimization problem for the trade-off between the holding and operating costs. Thus, we propose non-work-conserving routing policies in Section 3 that asymptotically minimize the cost of the system. In Section 4, we consider another routing problem with a certain service-level target and develop the corresponding routing policies. We show in Section 5 that our proposed policies have close connections to other routing policies in the literature. In Section 6, we use simulation experiments to test the performance of our proposed policies. Our conclusion is stated in Section 7. Technical proofs and some additional results about the knapsack problem are collected in the e-companion.

2. Model and Asymptotic Framework

2.1. The Stochastic Model

As shown in Figure 1, we consider a sequence of inverted-V queueing systems, where a single type of customer arrives at a system with an unlimited buffer and J server pools. In the nth system, the external arrival process is assumed to be a Poisson process $\Lambda^n(t)$ with rate λ^n. Note that the system parameter of the nth system is denoted by the superscript n. We model customer abandonment from the queue by assuming that each customer has a limited patience time following exponential distribution with rate θ^n. Pool j, $j=1,\ldots,J$, has N_j^n servers and all are capable of handling customers' service requirements. Service times are also assumed to be exponential, with service rates depending on the pool of the particular server. Specifically, the service rate of a server in pool j is μ_j^n. In addition, we assume that the interarrival times, patience times, and service times are independent.

Denote by $Q^n(t)$ the number of customers awaiting service, and by $B_j^n(t)$ the number of busy servers in pool j, $j=1,\ldots,J$, at time t. Let $R^n(t)$ denote the cumulative number of customers who have abandoned the queue by time t. We use $E_j^n(t)$ and $D_j^n(t)$ to denote the cumulative number of customers who have entered pool j and departed from pool j, by time t, respectively. The abandonment process from the queue and the departure process from pool j satisfy

$$R^n(t)=\tilde{R}^n\Big(\int_0^t Q^n(s)ds\Big) \quad \text{and} \quad D_j^n(t)=\tilde{D}_j^n\Big(\int_0^t B_j^n(s)ds\Big),\ j=1,\ldots,J, \tag{1}$$

for some Poisson processes $\tilde{R}^n$ and $\tilde{D}_j^n$ with rates θ^n and μ_j^n, respectively. The above processes are related via the following two balance equations for Q^n and B_j^n:

$$Q^n(t)=Q^n(0)+\Lambda^n(t)-R^n(t)-\sum_{j=1}^{J}E_j^n(t), \tag{2}$$

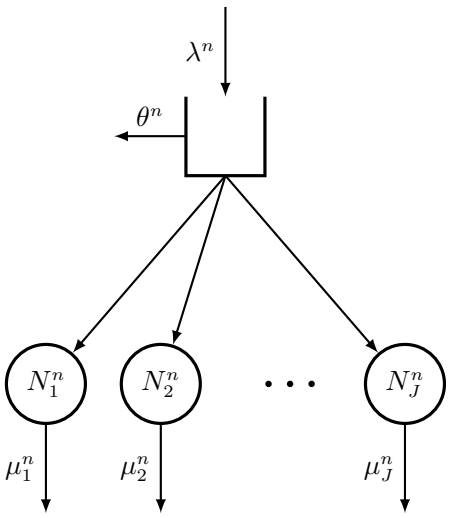


Figure 1 A sequence of inverted-V models

$$B_j^n(t) = B_j^n(0) + E_j^n(t) - D_j^n(t), \quad j = 1 \ldots, J. \tag{3}$$

Other than the index $j = 1, \ldots, J$ for each pool j, we use index $J+1$ to denote the queue. Let

$$E_{J+1}^n(t) = \Lambda^n(t) - \sum_{j=1}^{J} E_j^n(t), \tag{4}$$

which can be considered as the *net* cumulative number of customers who have joined the queue by time t. Thus, (2) can be written as

$$Q^n(t) = Q^n(0) + E_{J+1}^n(t) - R^n(t). \tag{5}$$

Moreover, let

$$I_j^n(t) = N_j^n - B_j^n(t), \ j = 1, \ldots, J, \quad \text{and} \quad I_{J+1}^n(t) = +\infty, \tag{6}$$

where $I_j^n(t)$, $j = 1, \ldots, J$, can be viewed as the number of idle servers in pool j at time t and $I_{J+1}^n(t)$ can be similarly regarded as the available space in the queue, which is infinite owing to the unlimited buffer size.

Due to exponentially distributed patience, system dynamics will not be affected by the order of serving customers. Thus, we can assume that customers are taken from the queue according to a first-come-first-served rule. To complete the full specification of an inverted-V model, we also need to describe the details of policies to route customers to different server pools. The policy need not satisfy the work conservation (i.e., non-idling server) condition. However, we assume that the customer at the head of the queue can enter service either upon customer arrival or upon service completion. Such an assumption is the same as (2) in Zhong et al. (2022). This means that

$$\sum_{j=1}^{J} E_j^n(s,t) \le \Lambda^n(s,t) + \sum_{j=1}^{J} D_j^n(s,t), \quad \text{for all } 0 \le s \le t, \tag{7}$$

where $E_j^n(s,t) := E_j^n(t) - E_j^n(s)$ is the number of customers who have entered pool j during time interval $[s,t]$ and $\Lambda^n(s,t)$, $D_j^n(s,t)$ are similarly defined. The inequality (7) prevents routing a batch of customers waiting in the queue into service and is sufficient for the tightness result in Theorem 1 in Section 2.2 to hold, which is in the same spirit as Assumption 2 in Puha and Ward (2022). Any process

$$\pi^n = (R^n, E^n, D^n, Q^n, B^n, I^n) \tag{8}$$

will be referred to as a policy for the nth system, provided that (1)–(7) hold. Denote by Π^n the collection of all policies for the nth system.

Heavy Traffic Regime and Fluid Scaling. We consider the many-server heavy traffic regime. For the sequence of inverted-V models indexed by n, let the arrival rate and the number of servers in each pool grow in proportion to n but with a fixed abandonment rate $\theta^n = \theta$ and fixed service rates $\mu_j^n = \mu_j$ for all $j = 1 \ldots, J$. Moreover, as n goes to infinity,

$$\frac{\lambda^n}{n} \to \lambda \quad \text{and} \quad \frac{N_j^n}{n} \to N_j, \ j = 1, \ldots, J. \tag{9}$$

The fluid scaling for the arrival process can be defined as

$$\bar{\Lambda}^n(t) = \frac{\Lambda^n(t)}{n}, \tag{10}$$

for all $t \geq 0$. The same scaling also applies to all of the other processes R^n, E^n, D^n, Q^n, B^n and I^n. We assume that the initial states satisfy $\bar{Q}^n(0) \Rightarrow Q(0)$ and $\bar{B}_j^n(0) \Rightarrow B_j(0)$ as n goes to infinity for some $Q(0) \geq 0$ and $B_j(0) \geq 0$, $j = 1, \ldots, J$.

Operating and Holding Costs. Assume that at any time $t \geq 0$ each pool j, $j = 1, \ldots, J$, incurs an instantaneous operating cost $C_j^n(\cdot)$ for busy servers and the queue incurs a per unit time queue-length cost $C_{J+1}^n(\cdot)$ for waiting customers. In detail,

$$C_j^n(B_j^n(t)) = C_j(B_j^n(t)/n), \ j = 1, \ldots, J, \quad \text{and} \quad C_{J+1}^n(Q^n(t)) = C_{J+1}(Q^n(t)/n), \tag{11}$$

where the cost functions are rescaled as the parameter n changes and $C_1, \ldots, C_J, C_{J+1}$ can be any general nondecreasing functions. The same scaling was also used in Section 7 of Mandelbaum and Stolyar (2004). We set $C_j(0) = 0$, $j = 1, \ldots, J, J+1$, meaning that there will not be any cost once there is no customer being served in pool j or waiting in the queue. There is also a penalty cost γ for customer abandonment. Therefore, for any policy $\pi^n \in \Pi^n$, the average total cost over $[0,T]$ is

$$L_T^n(\pi^n) = \frac{1}{T} \sum_{j=1}^{J} \int_0^T C_j(B_j^n(s)/n) ds + \frac{1}{T} \Big[\int_0^T C_{J+1}(Q^n(s)/n) ds + \gamma R^n(T)/n \Big], \tag{12}$$

where the term for the abandonment penalty is also rescaled by n. The idea of the above cost function also follows from Mandelbaum and Stolyar (2004), where the authors study the almost sure convergence of the cost function using Skorohod representation theorem. An alternative way is to consider the convergence in mean, e.g., in Atar et al. (2010, 2014) the authors consider the expectation of the cost function, and the expectation in their papers can be directly appended to the headcount processes due to their assumption of linear costs. One of the main purposes of this paper is to design routing policies that asymptotically minimize the average total cost (12).

For convenience, define the average operating cost and average holding cost (including the queue-length cost and abandonment penalty) over $[0,T]$ as

$$L_T^{O,n}(\pi^n)=\frac{1}{T}\sum_{j=1}^{J}\int_0^T C_j(B_j^n(s)/n)ds \quad \text{and} \quad L_T^{H,n}(\pi^n)=\frac{1}{T}\Big[\int_0^T C_{J+1}(Q^n(s)/n)ds+\gamma R^n(T)/n\Big], \tag{13}$$

respectively.

2.2. The Fluid Model

In this subsection, we introduce a deterministic fluid model, then show that it serves as the fluid limit of the inverted-V model in the many-server heavy traffic regime.

Similar to the stochastic model, the fluid model involves a single type of fluid content that arrives at an inverted-V model consisting of J server pools with the service capacity N_j in each pool j, $j=1,\ldots,J$. The amount of external arrivals over $[0,t]$ is $\Lambda(t)=\lambda t$, where $\lambda>0$. We use $Q(t)$ and $B_j(t)$, $j=1,\ldots,J$, to denote the amount of fluid content waiting in the queue and being served in pool j, respectively. The patience time in the queue follows an exponential distribution with rate $\theta>0$ and the service time in pool j, $j=1,\ldots,J$, also follows an exponential distribution with rate $\mu_j>0$.

Let $R(t)$ denote the cumulative amount of fluid content that has abandoned the queue by time t. We use $E_j(t)$ and $D_j(t)$ to denote the cumulative amount of fluid content that has entered pool j and departed from pool j by time t, respectively. As a counterpart of (1), the fluid abandonment process from the queue and the fluid departure process from pool j satisfy

$$R(t)=\theta\int_0^t Q(s)ds \quad \text{and} \quad D_j(t)=\mu_j\int_0^t B_j(s)ds,\ j=1,\ldots,J, \tag{14}$$

respectively. Analogous to (2) and (3), we have the following two balance equations for Q and B_j:

$$Q(t)=Q(0)+\Lambda(t)-R(t)-\sum_{j=1}^{J}E_j(t), \tag{15}$$

$$B_j(t)=B_j(0)+E_j(t)-D_j(t), \quad j=1,\ldots,J. \tag{16}$$

Similar to (4), the *net* cumulative amount of fluid content that has joined the queue by time t satisfies

$$E_{J+1}(t) = \Lambda(t) - \sum_{j=1}^{J} E_j(t). \tag{17}$$

Then (15) becomes

$$Q(t) = Q(0) + E_{J+1}(t) - R(t). \tag{18}$$

Moreover, in a similar vein to (6), the available service resource in pool j and the available space in the queue at time t satisfy

$$I_j(t) = N_j - B_j(t),\ j = 1, \ldots, J, \quad \text{and} \quad I_{J+1}(t) = +\infty, \tag{19}$$

respectively. Corresponding to (7), we have

$$\sum_{j=1}^{J} E_j(s,t) \le \Lambda(s,t) + \sum_{j=1}^{J} D_j(s,t), \quad \text{for all } 0 \le s \le t, \tag{20}$$

where $E_j(s,t) := E_j(t) - E_j(s)$ and $\Lambda(s,t)$, $D_j(s,t)$ are defined similarly.

We refer to equations (14)–(20) as the *fluid model* of an inverted-V queueing system. As with the stochastic policies introduced in (8), any fluid process

$$\pi = (R, E, D, Q, B, I) \tag{21}$$

will be referred to as a policy for the fluid model given that (14)–(20) hold. Also, denote by Π the collection of all policies for the fluid model.

The following theorem, which we prove in Section EC.1 of the e-companion, proves that the fluid model can be used to approximate the original stochastic model.

Theorem 1 (Fluid Limit). *The sequence of the fluid-scaled stochastic processes* $\{(\bar{\Lambda}^n, \bar{R}^n, \bar{E}^n, \bar{D}^n, \bar{Q}^n, \bar{B}^n, \bar{I}^n) : n \in \mathbb{N}\}$ *satisfying* (1)–(7) *is tight in the Skorohod-J_1 topology, and any subsequential limit of the fluid-scaled stochastic processes satisfies the fluid model equations* (14)–(20).

In view of (12), define the associated fluid total cost as

$$L_T(\pi) = \frac{1}{T} \sum_{j=1}^{J} \int_0^T C_j(B_j(s))ds + \frac{1}{T}\Big[\int_0^T C_{J+1}(Q(s))ds + \gamma R(T)\Big], \tag{22}$$

for any fluid policy $\pi \in \Pi$. Corresponding to (13), the fluid operating and holding costs can be respectively defined as

$$L_T^O(\pi) = \frac{1}{T} \sum_{j=1}^{J} \int_0^T C_j(B_j(s))ds \quad \text{and} \quad L_T^H(\pi) = \frac{1}{T}\Big[\int_0^T C_{J+1}(Q(s))ds + \gamma R(T)\Big]. \tag{23}$$

The cost functions $C_j(\cdot)$, $j=1,\ldots,J$, and $C_{J+1}(\cdot)$ in (11) can now be regarded as the fluid operating cost of the fluid content being served in pool j and the fluid queue-length cost of the fluid content waiting in the queue, respectively.

2.3. Routing Problem

We use a steady-state optimization problem to determine the optimal routing of arrivals to the queue and server pools. The formulation ignores the dynamic impact of variability in the inter-arrival, service, and patience times by only focusing on "expected" arrival, service, and abandonment rates.

Given $\lambda>0$ and $N_j>0$, $j=1,\ldots,J$, we formulate the *routing problem* as follows.

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{J} C_j(b_j) + C_{J+1}(q) + \gamma\theta q \\ \text{subject to} \quad & \sum_{j=1}^{J} \mu_j b_j + \theta q = \lambda, \\ & 0 \le b_j \le N_j,\ j=1,\ldots,J, \\ & q \ge 0. \end{aligned} \tag{24}$$

The objective is to minimize the long-run average total cost by choosing appropriate b_j's, $j=1,\ldots,J$, and q. The decision variables b_j's and q can be intuitively understood as the amount of fluid content that is being served in pool j and is waiting in the queue in the long run, respectively. The first constraint implies that the arrivals must be routed to one of the server pools or the queue. The second constraint states that b_j's must be chosen so that the amount of fluid content being served in pool j does not exceed the service capacity N_j. Moreover, b_j's and q should be nonnegative. Denote by (b^*,q^*), where $b^*=(b_1^*,\ldots,b_J^*)$, an optimal solution to this nonlinear programming and L^* the optimal value. It is clear that L^* serves as the lower bound of any fluid convergent policies. One of the main goals of this paper is to find a routing policy that approaches the lower bound.

Definition 1 (Stationary Optimal Control). A fluid routing policy $\pi\in\Pi$ is said to be *stationary optimal* if the corresponding total cost function (22) satisfies $\lim_{T\to\infty} L_T(\pi)=L^*$.

3. Non-work-conserving Routing Policies

Note that it is possible for the solution to problem (24) to satisfy $q^*\cdot\sum_{j=1}^{J}(N_j-b_j^*)\neq 0$, implying that the queue and idle servers can coexist in the steady state. Thus, in this section, we propose three types of non-work-conserving routing policies to cope with all possible types of cost functions. Indeed, the routing problem (24) can become any type of optimization problem (convex, concave, nonconvex and nonconcave). We first show the fluid routing policies in Section 3.1, then translate them back to the original stochastic systems in Section 3.2.

3.1. Fluid Routing Policies

In this subsection, we first introduce the fluid dynamic priority policy. In Section 3.1.1, the target-allocation policy is proposed for any general cost functions. We then propose in Section 3.1.2 the Gc/μ rule when the cost functions in (24) are convex. On the other hand, if the cost functions are concave, we find it is optimal to apply the fixed priority policy in Section 3.1.3. As shown in Table 1, our proposed three routing policies for inverted-V models actually correspond to those in Long et al. (2020), where three similar work-conserving policies have been developed to minimize the total holding cost of a V model with general, convex, and concave cost functions, respectively. It is also worth pointing out that our proposed policies are non-work-conserving policies for the purpose of balancing the holding and operating costs.

We now introduce the *fluid dynamic priority policy*, which routes the arrival to the queue or a server pool with the smallest priority value at the arrival instants. Here the smallest value represents the highest priority. This means that upon arrival, some amount of fluid content is routed to the pool or queue with index

$$j \in \underset{j=1,\ldots,J,J+1}{\arg\min} P_j(t), \tag{25}$$

where $P_j(t)$, $j = 1, \ldots, J$, is the priority value for pool j and $P_{J+1}(t)$ is the priority value for the queue at time t. Indeed, if the index j is equal to $J+1$ at an arrival instant, then the arrival joins the queue directly, and no fluid content can enter service. Otherwise, if the index j is in the set $\{1, \ldots, J\}$ at an arrival instant, then the arrival joins the queue, and meanwhile, some amount of fluid count at the head of the queue will enter service. The time-dependent index j provides us with a dynamic priority, which introduces challenges to show the convergence of the fluid model, especially starting from any initial state. Comparing (18) with (16), the buffer can be regarded as another server pool with infinite service capacity indexed by $J+1$. Then, $E_{J+1}(t)$ and $R(t)$ in (18) can be regarded as the "entrance into service" process and "departure" process of pool $J+1$, respectively. Only when the pools, including pool $J+1$ (queue), with the smallest priority value are all busy, then the fluid content can be routed to the pools with the second smallest priority value, so on and so forth. Therefore, the fluid dynamic priority policy can also be expressed as

$$\int_0^t \sum_{\{k=1,\ldots,J,J+1:P_k(s)<P_j(s)\}} I_k(s) dE_j(s) = 0, \quad j = 1, \ldots, J, J+1. \tag{26}$$

We take $\sum_{\emptyset} I_k(s) = 0$. Recall that $I_{J+1}(t) = +\infty$ for all $t \geq 0$. Thus, no fluid content will be routed to those pools with priority values larger than that of the queue.

Remark 1. One can find that the routing decisions happen only at the arrival instants, which is different from the work-conserving policies introduced in Long et al. (2020) where some amount of fluid content in the queue will be routed to the server pool upon service completion. Interestingly, we find that it will be suboptimal to allow fluid content to enter service upon service completion in our inverted-V model. Let us consider a special case with $J=1$ and set the initial state to be $B_1(0) \in (b_1^*, \lambda/\mu_1)$ and $Q(0) = (\lambda - \mu_1 B_1(0))/\theta \in (0, q^*)$. If we allow the fluid content in the queue to enter service upon service completion, then it is easy to check that the fluid model will stay at the initial state under the policies introduced in this section. Namely, $B_1(t) = B_1(0)$ and $Q(t) = Q(0)$ for all $t \geq 0$. Since $(B_1(0), Q(0))$ is different from the optimal solution (b_1^*, q^*) of (24), it is better not to make routing decisions upon service completion.

3.1.1. Target-allocation Policy We propose a policy that is suitable for any general cost function. The optimal solution (b^*, q^*) of (24) reveals that there should be b_j^*, $j=1,\ldots,J$, amount of fluid content being served in pool j and q^* amount of fluid content waiting in the queue in the long run. Thus we define the following priority value functions:

$$P_j(t) = B_j(t) - b_j^*, \ j=1,\ldots,J, \quad \text{and} \quad P_{J+1}(t) = Q(t) - q^*. \tag{27}$$

Intuitively, the dynamic priority policy routes the fluid content to the queue or pools that have not been assigned enough fluid content. Eventually, all the B_j's and Q will be close to the optimal solution (b^*, q^*). We refer to this fluid routing policy as the *target-allocation policy* denoted by π_{b^*,q^*} (see (38) in Section 3.2 for the stochastic version). We show its optimality in the following theorem, which is proved in Section EC.2.2 of the e-companion. Note that the target-allocation policy needs the optimal solution of (24) in advance.

Theorem 2 (Optimality of the Target-allocation Policy). *The fluid model* (14)–(20)*, under the target-allocation policy π_{b^*,q^*} with the priority value function* (27)*, satisfies* $\lim_{t\to\infty} B_j(t) = b_j^*$, $j=1,\ldots,J$, $\lim_{t\to\infty} Q(t) = q^*$ *and* $\lim_{T\to\infty} L_T(\pi_{b^*,q^*}) = L^*$.

3.1.2. The Generalized c/μ Rule For convex cost functions, we propose another dynamic priority policy that is easier to implement since the optimal solution of (24) is not required in advance. Consider the Lagrangian function of (24)

$$\begin{aligned}\mathscr{L}(b_j, q, \alpha_0, \alpha_j, \beta_j, \eta) = &\sum_{j=1}^{J} C_j(b_j) + C_{J+1}(q) + \gamma\theta q \\ &+ \alpha_0(\lambda - \sum_{j=1}^{J} \mu_j b_j - \theta q) - \sum_{j=1}^{J} \alpha_j b_j - \sum_{j=1}^{J} \beta_j (N_j - b_j) - \eta q,\end{aligned}$$

where the Lagrange multipliers satisfy $\alpha_0 \in \mathbb{R}$, $\eta \geq 0$ and $\alpha_j, \beta_j \geq 0$ for all $j=1,\ldots,J$. We assume that the cost functions C_j's, $j=1\ldots,J,J+1$, satisfy conditions that are analogous to Assumption 3 in van Mieghem (1995) and Assumption 2 in Huang et al. (2015).

Assumption 1 (Cost Regularity). *The cost functions C_j's, $j=1,\ldots,J,J+1$, are differentiable and strictly convex, and there is an interior solution to the minimization problem* (24).

Denote the derivative of $C_j(\cdot)$ by $c_j(\cdot)$, $j=1,\ldots,J,J+1$. Under Assumption 1, the Karush-Kuhn-Tucker (KKT) conditions then satisfy

$$\frac{c_j(b_j^*)}{\mu_j}=\alpha_0,\quad j=1,\ldots,J, \tag{28}$$

$$\frac{c_{J+1}(q^*)}{\theta}+\gamma=\alpha_0, \tag{29}$$

$$\sum_{j=1}^{J}\mu_j b_j^*+\theta q^*=\lambda. \tag{30}$$

One can find that (28) and (29) are equal to the same constant α_0, which inspires us to consider the following priority value functions:

$$P_j(t)=\frac{c_j(B_j(t))}{\mu_j},\ j=1,\ldots,J,\quad \text{and}\quad P_{J+1}(t)=\frac{c_{J+1}(Q(t))}{\theta}+\gamma. \tag{31}$$

As argued below (25), the buffer can be regarded as another server pool with infinite service capacity indexed by $J+1$. Therefore, with a slight abuse of the terminology, we refer to the above as the priority value functions of the *generalized c/μ rule* (Gc/μ) denoted by π_G (see (39) in Section 3.2 for the stochastic version).

The Gc/μ rule can be viewed as a counterpart of the well-known $Gc\mu$ rule in van Mieghem (1995) for the optimal scheduling of multiple types of customers to a server pool. Symmetrically, the Gc/μ rule is designed for the optimal routing of inverted-V models and the optimality is shown in the following theorem, which we prove in Section EC.2.2 of the e-companion. Here, we require c_j's to be differentiable in the same spirit as the twice differentiability of the cost functions in Section 4 of Mandelbaum and Stolyar (2004).

Theorem 3 (Optimality of the Gc/μ Rule). *Given Assumption 1, if c_j's, $j=1,\ldots,J,J+1$, are differentiable, then the fluid model* (14)–(20) *under the Gc/μ rule π_G with the priority value function* (31) *satisfies* $\lim_{t\to\infty}B_j(t)=b_j^*$, $j=1,\ldots,J$, $\lim_{t\to\infty}Q(t)=q^*$ *and* $\lim_{T\to\infty}L_T(\pi_G)=L^*$.

We find that the proofs of the optimality of the target-allocation policy and the Gc/μ rule are almost the same. Therefore, we will simultaneously prove Theorems 2 and 3 in Section EC.2.2 of the e-companion.

3.1.3. Fixed Priority Policy When $P_j(t)$'s, $j=1,\ldots,J,J+1$, are independent of time t, the fluid dynamic priority policy is known as the *fixed priority policy*. Consider a priority order from pool 1 (highest priority) to pool J (lowest priority). We also set the priority value for the

queue such that the priorities of J pools are separated into two parts. Therefore, the priority value function in (25) can be specified as

$$P_j(t) = j,\ j = 1, \ldots, J, \quad \text{and} \quad P_{J+1}(t) = k + \frac{1}{2},\ k \in \{0, 1, \ldots, J\}. \tag{32}$$

The following proposition shows that the system converges to the steady state under the fixed priority policy (32). The proof is given in Section EC.2.3 of the e-companion.

Proposition 1 (Convergence of the Fixed Priority Policy). *The fluid model* (14)–(20) *under the fixed priority policy with the priority value function* (32) *satisfies*

$$\lim_{t\to\infty} B_j(t) = b_j,\ j = 1, \ldots, J, \quad \textit{and} \quad \lim_{t\to\infty} Q(t) = q, \tag{33}$$

where the limits $b = (b_1, \cdots, b_J)$ *and* q *satisfy the conditions in the following two cases:*

(i) *If* $\lambda > \sum_{l=1}^{k} \mu_l N_l$, *where* $k \in \{0, 1, \ldots, J\}$ *is from the priority value of the buffer in* (32), *then the limits satisfy*

$$b = (N_1, \cdots, N_k, 0, \cdots, 0) \quad \textit{and} \quad q = \frac{1}{\theta}(\lambda - \sum_{l=1}^{k} \mu_l N_l). \tag{34}$$

(ii) *If* $\lambda \le \sum_{l=1}^{k} \mu_l N_l$, *where* $k \in \{0, 1, \ldots, J\}$ *is from the priority value of the buffer in* (32), *then the limits satisfy*

$$b = \Big(N_1, \cdots, N_{j_0-1}, (\lambda - \sum_{j=1}^{j_0-1} \mu_j N_j)/\mu_{j_0}, 0, \cdots, 0\Big) \quad \textit{and} \quad q = 0, \tag{35}$$

where $j_0 = \max\Big\{j \in [1, \cdots, k] : \sum_{l=1}^{j-1} \mu_l N_l < \lambda\Big\}$.

The above limits can be viewed as a solution on the boundary of the feasible region of (24). Therefore, if the nonlinear programming (24) is a concave optimization problem, then the optimal solution (b^*, q^*) surely has the same form as (34) or (35) after reordering the indices if needed. This is associated with an optimal fixed priority order, of which the corresponding fixed priority policy is denoted by π_{P^*} (see (40) in Section 3.2 for the stochastic version). Note that the order among the pools with $b_j^* = N_j$ can be arbitrarily determined. It can also be arbitrary for those with $b_j^* = 0$.

Theorem 4 (Optimality of the Fixed Priority Policy). *If the cost functions* C_j*'s,* $j = 1, \ldots, J, J+1$, *are concave, then the fluid model* (14)–(20) *under the fixed priority policy* π_{P^*} *with the priority value function* (32) *(after reordering the indices if needed) satisfies* $\lim_{t\to\infty} B_j(t) = b_j^*$, $j = 1, \ldots, J$, $\lim_{t\to\infty} Q(t) = q^*$ *and* $\lim_{T\to\infty} L_T(\pi_{P^*}) = L^*$.

Theorem 4 is proved in Section EC.2.3 of the e-companion. This theorem provides a sufficient condition for the optimality of the fixed priority policy. We will show in Section EC.4 the connection between the fixed priority policy and a min-knapsack problem. Consequently, the optimal priority order can be obtained by solving the min-knapsack problem using dynamic programming.

3.2. Stochastic Routing Policies

In this subsection, we show how fluid routing policies can lead to stochastic policies that are optimal in an asymptotic sense.

In the nth system, let $P_j^n(t)$, $j=1,\ldots,J$, be the priority value function of each pool and $P_{J+1}^n(t)$ be the priority value function of the queue. Then the stochastic version of the fluid *dynamic priority policy* (25) is said to be as follows: upon arrival, there will be a customer who is routed to the pool or the queue with index

$$j \in \underset{j=1,\ldots,J,J+1}{\arg\min}\ P_j^n(t). \tag{36}$$

Similar to the argument below (25), in the stochastic system, the buffer can also be regarded as another server pool indexed by $J+1$ with E_{J+1}^n in (4) being the "entrance into service" process and R^n in (2) being the "departure" process. Thus, a routing policy essentially routes the external arrivals to server pools, including pool $J+1$ (queue). If pool i with the smallest priority value is busy, the customer will be routed to pools with the second smallest priority value, and so on and so forth. Ties are broken arbitrarily once there are multiple pools with the same priority value, for example, in favor of the smallest index j. Thus, one can find that the stochastic dynamic priority policy (36) is equivalent to

$$\int_0^t \sum_{\{k=1,\ldots,J,J+1:P_k^n(s)<P_j^n(s)\}} I_k^n(s)dE_j^n(s)=0, \quad j=1,\ldots,J,J+1. \tag{37}$$

We set $\sum_{\emptyset} I_k^n(s)=0$. Since $I_{J+1}^n(t)=+\infty$ for all $t\geq 0$ by (6), no customer will be routed to pools with a lower priority than the queue.

In the following, we consider three stochastic routing policies corresponding to the three fluid routing policies proposed in Section 3.1.

Target-allocation Policy. We denote it by π_{b^*,q^*}^n given the priority value functions

$$P_j^n(t)=B_j^n(t)/n-b_j^*,\ j=1,\ldots,J, \quad \text{and} \quad P_{J+1}^n(t)=Q^n(t)/n-q^*, \tag{38}$$

where we apply the same scaling as in (11) and (b^*,q^*) is an optimal solution of the nonlinear programming (24).

The Generalized c/μ Rule. We denote it by π_G^n given the priority value functions

$$P_j^n(t)=\frac{c_j\big(B_j^n(t)/n\big)}{\mu_j},\ j=1,\ldots,J, \quad \text{and} \quad P_{J+1}^n(t)=\frac{c_{J+1}(Q^n(t)/n)}{\theta}+\gamma, \tag{39}$$

where we apply the same scaling as in (11).

Fixed Priority Policy. We denote it by $\pi^n_{P^*}$ given the priority value function (after reordering the indices if needed)

$$P^n_j(t) = j,\ j = 1, \ldots, J, \quad \text{and} \quad P^n_{J+1}(t) = k + \frac{1}{2},\ k \in \{0, 1, \ldots, J\}. \tag{40}$$

It is worth noting that under the fixed priority policy, the pools with priority lower than pool $J+1$ (queue) can be eliminated (i.e., the system should be downsized or have fewer server pools).

Recall that L^* is the minimum value of the routing problem (24). We have proven in Theorems 2, 3 and 4 that the fluid model can achieve the optimal value L^* under the three fluid routing policies π_{b^*,q^*}, π_G, and π_{P^*}. For the original queueing system, our goal is to find a routing policy such that L^* can also be asymptotically achieved in the many-server heavy traffic regime. We refer to such a routing policy as an *asymptotically stationary optimal* policy. The following theorem shows that the optimal value L^* can actually be asymptotically achieved under any one of the stochastic policies $\pi^n_{b^*,q^*}$, π^n_G and $\pi^n_{P^*}$. In fact, it is exactly the stochastic version of Theorems 2, 3, and 4. The proof is postponed to Section EC.2.4 of the e-companion.

Theorem 5 (Asymptotically Stationary Optimality of Our Policies). *Given the conditions in Theorems 2, 3 and 4 respectively, there is*

$$\liminf_{T\to\infty} \liminf_{n\to\infty} L^n_T(\pi^n) = \limsup_{T\to\infty} \limsup_{n\to\infty} L^n_T(\pi^n) = L^* \tag{41}$$

almost surely, where $\pi^n = \pi^n_{b^*,q^*}$, π^n_G, *and* $\pi^n_{P^*}$, *accordingly.*

4. Routing Problem with a Service-level Target

So far, we have proposed in Section 3 three dynamic priority policies to balance the trade-off between the holding and operating costs. In practice, another critical decision in inverted-V models is to minimize the operating costs of the system but also to meet a certain service-level target. Such a decision is particularly important for situations where customer service level is a major concern, but the waiting cost is difficult to quantify. In this section, we show that this problem can be formulated as another optimization problem that is similar to the routing problem (24).

Assume that the goal is to minimize the operating costs of all the pools subject to keeping the steady-state abandonment probability of the whole system below a service-level target p, which can be any number in $[0, 1]$. Given $\lambda > 0$, $N_j > 0$, $j = 1, \ldots, J$, and $p \in [0, 1]$, consider the following optimization problem.

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{J} C_j(b_j) \\ \text{subject to} \quad & \theta q/\lambda \le p, \\ & \sum_{j=1}^{J} \mu_j b_j + \theta q = \lambda, \\ & 0 \le b_j \le N_j,\ j = 1, \ldots, J, \\ & q \ge 0. \end{aligned} \tag{42}$$

It is clear that (42) is feasible only when $\lambda(1-p) \le \sum_{j=1}^{J} \mu_j N_j$. We assume that this condition is satisfied for the above problem. Similar to the routing problem (24), the above determines the optimal routing of arrivals to the queue and pools and meets the service-level target p. Therefore, we refer to the above optimization problem as the *routing problem with a service-level target*. The decision variables b_j's and q have the same interpretation as in (24). The left-hand side of the first constraint can be understood as the abandonment probability of the whole system, which should be less than or equal to p. The other three constraints are identical to that of (24). The optimal solution of (42) is denoted by $(b^{p,*}, q^{p,*})$, where $b^{p,*} = (b_1^{p,*}, \ldots, b_I^{p,*})$. Here we append a superscript p to emphasize the dependence on the service-level target. We also denote the optimal value by $L^{O,*}$, which is actually the steady state of the operating cost in (23).

Next, we show that if we set the service-level target p to be equal to $\theta q^*/\lambda$, where q^* is the optimal solution of (24), the two routing problems (24) and (42) have a same optimal solution. The proof is straightforward and hence omitted.

Proposition 2 (Connection Between the Two Routing Problems). *For any $\lambda > 0$ and $N_j > 0$, $j = 1, \ldots, J$, if we set the service-level target $p = \theta q^*/\lambda$, then there exist optimal solutions of (24) and (42) that satisfy $(b^*, q^*) = (b^{p,*}, q^{p,*})$.*

In the following, we will design routing policies that attain the optimal value of the routing problem (42), which is $L^{O,*}$, for any given service-level target $p \in [0,1]$. Similar to Definition 1, we have the following definition for the routing problem with a service-level target.

Definition 2 (Stationary Optimal Control with a Service-level Target). For any given $p \in [0,1]$, a fluid routing policy $\pi \in \Pi$ is said to be *stationary optimal with a service-level target p* if the corresponding operating cost function in (23) satisfies $\lim_{T\to\infty} L_T^O(\pi) = L^{O,*}$.

4.1. Fluid Hybrid Routing Policies

It can be easily seen that the optimal solution of (42) satisfies $q^{p,*} = \lambda p/\theta$, which implies that there should be $\lambda p/\theta$ amount of fluid content waiting in the queue in the long run. This inspires us to

consider a hybrid policy, where the queue follows a threshold policy (only when the fluid queue length exceeds $\lambda p/\theta$ can the fluid content at the head of the queue enter service) and the pools still follow one of the three fluid dynamic priority policies introduced in (25).

Specifically, the hybrid policy combines the fixed priority policy on the queue with the dynamic priority policy on the pools. The fixed priority policy on the queue prevents the fluid content from entering service as long as the fluid queue length is no more than $\lambda p/\theta$. This means

$$\int_0^t \big(\lambda p/\theta - Q(s)\big)^+ d\sum_{j=1}^J E_j(s) = 0. \tag{43}$$

When the fluid queue length exceeds $\lambda p/\theta$, the dynamic priority policy on the pools routes some amount of fluid content to the pool with index

$$j \in \arg\min_{j=1,\ldots,J} P_j(t), \tag{44}$$

where $P_j(t)$, $j = 1, \ldots, J$, is the priority value for pool j at time t. If the pools with the highest priority are all busy, then the fluid content can be routed to the pools with the second highest priority value, so on and so forth. Therefore, the fluid dynamic priority policy on the pools (44) can be expressed as

$$\int_0^t \sum_{\{k=1,\ldots,J: P_k(s) < P_j(s)\}} I_k(s) dE_j(s) = 0, \quad j = 1, \ldots, J. \tag{45}$$

Note that $\sum_{\emptyset} I_k(s) = 0$. We refer to (43) and (45) as the fluid *hybrid routing policy*.

It remains to specify the priority value functions for the pools. We find that $P_j(t)$'s, $j = 1, \ldots, J$, perfectly inherit the same form as that of the three routing policies proposed in Section 3. The main difference is that there is no need to define the priority value function for the queue, which is now replaced by (43).

4.1.1. Hybrid Target-allocation Policy For general operating cost functions, we propose a hybrid policy that combines the fixed priority policy on the queue, which is characterized by (43), and the target-allocation policy on the pools. The priority value functions for the pools are almost the same as that of (27). The optimal solution $b^{p,*} = (b_1^{p,*}, \ldots, b_J^{p,*})$ of (42) motivates us to define the following priority value function for the pools:

$$P_j(t) = B_j(t) - b_j^{p,*}, \quad j = 1, \ldots, J. \tag{46}$$

We refer to the fluid routing policy satisfying (43), (45) and (46) as the *hybrid target-allocation policy* denoted by $\pi^h_{b^{p,*}}$ (see (56) in Section 4.2 for the stochastic version). Its optimality is shown in Theorem 6 below, which is the hybrid version of Theorem 2. The proof is given in Section EC.3 of the e-companion.

Theorem 6 (Optimality of the Hybrid Target-allocation Policy). *Given* $\lambda(1-p) \le \sum_{j=1}^{J} \mu_j N_j$, *the fluid model* (14)–(20) *under the hybrid target-allocation policy* $\pi^h_{b^{p,*}}$ *with the priority value function* (46) *satisfies* $\lim_{t\to\infty} B_j(t) = b_j^{p,*}$, $j = 1,\dots,J$, $\lim_{t\to\infty} Q(t) = \lambda p/\theta$ *and* $\lim_{T\to\infty} L_T^O(\pi^h_{b^{p,*}}) = L^{O,*}$.

4.1.2. Hybrid Gc/μ Rule For convex operating cost functions, we propose a hybrid policy that combines the fixed priority policy on the queue, characterized by (43), and the Gc/μ rule on the pools to meet the service-level target p. Similar to Assumption 1, we also have the following assumption for the optimization problem (42).

Assumption 2 (Cost Regularity for the Routing Problem with a Service-level Target). *The operating cost functions* C_j*'s,* $j = 1,\dots,J$, *are differentiable and strictly convex, and there is an interior solution to the minimization problem* (42).

Under Assumption 2, the KKT conditions (28)–(30) then reduce to

$$\frac{c_j(b_j^{p,*})}{\mu_j} = \alpha_0, \quad j = 1,\dots,J, \tag{47}$$

$$\sum_{j=1}^{J} \mu_j b_j^{p,*} = (1-p)\lambda. \tag{48}$$

In line with (31), we consider the following priority value function:

$$P_j(t) = \frac{c_j(B_j(t))}{\mu_j}, \ j = 1,\dots,J. \tag{49}$$

This equation is referred to as the priority value function of the *hybrid generalized c/μ rule* (hybrid Gc/μ) denoted by π^h_G (see (57) in Section 4.2 for the stochastic version).

The optimality of the hybrid Gc/μ rule is shown in the following theorem. The proof is given in Section EC.3 of the e-companion.

Theorem 7 (Optimality of the Hybrid Gc/μ Rule). *Given Assumption 2 and* $\lambda(1-p) \le \sum_{j=1}^{J} \mu_j N_j$, *if* c_j*'s are differentiable, then the fluid model* (14)–(20) *under the hybrid Gc/μ rule* π^h_G *with the priority value function* (49) *satisfies* $\lim_{t\to\infty} B_j(t) = b_j^{p,*}$, $j = 1,\dots,J$, $\lim_{t\to\infty} Q(t) = \lambda p/\theta$ *and* $\lim_{T\to\infty} L_T^O(\pi^h_G) = L^{O,*}$.

4.1.3. Hybrid Fixed Priority Policy For concave operating cost functions, we propose a hybrid policy that combines the fixed priority policy on the queue, which is characterized by (43), and the fixed priority policy on the pools to meet the service-level target p. The fixed priority policy on the pools essentially prevents the fluid content from entering server pools as long as other pools with higher priority are still available. Consider a priority order from class 1 (highest priority) to

class J (lowest priority). Then, the same as (32), the priority value function for each pool can be specified as

$$P_j(t) = j, \quad j = 1, \ldots, J. \tag{50}$$

The following proposition shows that the system converges to the steady state under the hybrid fixed priority policy that satisfies (43), (45) and (50).

Proposition 3 (Convergence of the Hybrid Fixed Priority Policy). *Given* $\lambda(1-p) \leq \sum_{j=1}^{J} \mu_j N_j$, *the fluid model* (14)–(20) *under the hybrid fixed priority policy with the priority value function* (50) *converges to the following steady state*

$$\lim_{t\to\infty} B_j(t) = b_j, \ j = 1, \ldots, J, \quad \textit{and} \quad \lim_{t\to\infty} Q(t) = \lambda p/\theta, \tag{51}$$

where the limit $b = (b_1, \cdots, b_J)$ *satisfies*

$$b = \Big(N_1, \cdots, N_{j_0-1}, \big(\lambda(1-p) - \sum_{j=1}^{j_0-1} \mu_j N_j\big)/\mu_{j_0}, 0, \cdots, 0\Big), \tag{52}$$

where $j_0 = \max\Big\{j \in [1, \cdots, J] : \sum_{l=1}^{j-1} \mu_l N_l < \lambda(1-p)\Big\}$.

The allocation of the service resource (52) takes a special form such that $b_j = N_j$ for all pools $j < j_0$ providing full service resource, $b_j = 0$ for all pools $j > j_0$ without providing any service, and $b_{i_0} = \big(\lambda(1-p^{Ab}) - \sum_{i=1}^{i_0-1} \mu_i N_i\big)/\mu_{i_0}$ for at most one pool i_0 providing partial service resource. Combined with the fact that $q^{p,*} = \lambda p/\theta$, this is virtually a solution on the boundary of the feasible region of (42). Therefore, if the nonlinear programming (42) is a concave optimization problem, then the optimal solution $b^{p,*} = (b_1^{p,*}, \ldots, b_J^{p,*})$ has the same form as (52) after reordering the pool indices if needed. This is associated with an optimal fixed priority order on the pools, of which the corresponding hybrid fixed priority policy is denoted by π^h_{P*} (see (58) in Section 4.2 for the stochastic version).

Theorem 8 (Optimality of the Hybrid Fixed Priority Policy). *Given* $\lambda(1-p) \leq \sum_{j=1}^{J} \mu_j N_j$, *if the operating cost functions* C_j*'s,* $j = 1, \ldots, J$, *are concave, then the fluid model* (14)–(20) *under the hybrid fixed priority policy* π^h_{P*} *with the priority value function* (50) *(after reordering the pool indices if needed) satisfies* $\lim_{t\to\infty} B_j(t) = b_j^{p,*}$, $j = 1, \ldots, J$, $\lim_{t\to\infty} Q(t) = \lambda p/\theta$ *and* $\lim_{T\to\infty} L_T^O(\pi^h_{P*}) = L^{O,*}$.

The above theorem is proved in Section EC.3 of the e-companion and is the hybrid version of Theorem 4.

4.2. Stochastic Hybrid Routing Policies

Now we translate the fluid hybrid routing policies back to the stochastic ones and show that they asymptotically minimize the operating cost but still meet the service-level target p.

The hybrid policy combines the fixed priority policy on the queue and the dynamic priority policy on the pools. The stochastic version of the fluid fixed priority policy on the queue (43) can be expressed as

$$\int_0^t \left(\lambda^n p/\theta - Q^n(s)\right)^+ d\sum_{i=1}^{I} E_i^n(s) = 0, \tag{53}$$

which means customers can be routed into server pools only when the queue length exceeds a threshold $\lambda^n p/\theta$. Moreover, the stochastic version of the fluid dynamic priority policy on the pools (44) is said to be: at time t, given that a customer is to be served, this customer is routed to the pool with the index

$$j \in \arg\min_{j=1,\ldots,J} P_j^n(t), \tag{54}$$

where $P_j^n(t)$, $j = 1, \ldots, J$, is the priority value function of each pool. If pool j with the smallest priority value is busy, the customer will be routed to pools with the second smallest priority value, and so on and so forth. Ties are broken arbitrarily once there are multiple pools with the same priority value, for example, in favor of the smallest index j. It can be easily seen that the stochastic dynamic priority policy on the pools (54) is equivalent to

$$\int_0^t \sum_{\{k=1,\ldots,J: P_k^n(s) < P_j^n(s)\}} I_k^n(s) dE_j^n(s) = 0, \quad j = 1, \ldots, J. \tag{55}$$

We set $\sum_{\emptyset} I_k^n(s) = 0$. We refer to (53) and (55) as a *hybrid routing policy*.

In line with the routing policies in Section 3, we also consider three stochastic hybrid routing policies that correspond to the three fluid hybrid routing policies proposed in Section 4.1.

Hybrid Target-allocation Policy. We denote it by $\pi_{b^{p,*}}^{h,n}$ given the priority value function

$$P_j^n(t) = B_j^n(t)/n - b_j^{p,*}, \quad j = 1, \ldots, J, \tag{56}$$

where we apply the same scaling as in (11) and $b^{p,*}$ is an optimal solution of the nonlinear programming (42).

Hybrid Generalized c/μ Rule. We denote it by $\pi_G^{h,n}$ given the priority value function

$$P_j^n(t) = \frac{c_j\left(B_j^n(t)/n\right)}{\mu_j}, \quad j = 1, \ldots, J, \tag{57}$$

where we apply the same scaling as in (11).

Hybrid Fixed Priority Policy. We denote it by $\pi^{h,n}_{P^*}$ given the priority value function (after reordering the indices if needed)

$$P^n_j(t) = j, \quad j = 1, \ldots, J. \tag{58}$$

The following theorem is the stochastic version of Theorems 6, 7 and 8. The proof is the same as that of Theorem 5. Thus, we omit it for brevity.

Theorem 9 (Asymptotically Stationary Optimality of Hybrid Policies). *Given the conditions in Theorems 6, 7 and 8 respectively, there is*

$$\liminf_{T\to\infty} \liminf_{n\to\infty} L^{O,n}_T(\pi^n) = \limsup_{T\to\infty} \limsup_{n\to\infty} L^{O,n}_T(\pi^n) = L^{O,*} \tag{59}$$

almost surely, where $\pi^n = \pi^{h,n}_{b^{p,*}}$, $\pi^{h,n}_G$ *and* $\pi^{h,n}_{P^*}$ *accordingly.*

5. Connection to Other Routing Policies

In this section, we consider the routing problem with a service-level target $p = 0$. Then the optimization problem (42) is feasible only when $\lambda \le \sum_{j=1}^J \mu_j N_j$. By Theorems 6, 7 and 8, the fluid queue length vanishes under any one of the three hybrid routing policies proposed in Section 4.

Furthermore, given $p = 0$ the hybrid Gc/μ rule can be specialized to the policies proposed in Tezcan (2008) and Gurvich and Whitt (2009a,b, 2010), and the hybrid fixed priority policy can be specified as the policies developed in Armony (2005) and Xia et al. (2022).

5.1. The Load-balancing (LB) Policy

The key idea of the load-balancing (LB) policy proposed in Tezcan (2008) is to have all servers in the inverted-V model be utilized fairly. To show its connection to the hybrid Gc/μ rule, we set the service-level target $p = 0$ and the operating cost function to be

$$C_j(x) = \frac{x^2}{2N_j}\mu_j, \quad j = 1, \ldots, J. \tag{60}$$

Then the priority value function of the hybrid Gc/μ rule (49) becomes

$$P_j(t) = \frac{B_j(t)}{N_j}, \quad j = 1, \ldots, J, \tag{61}$$

which is the utilization of pool j at time t. With regard to (43) and (44), upon arrival, the fluid content at the head of the queue is routed to the least utilized pool with the index

$$\arg\min_{j=1,\ldots,J} \frac{B_j(t)}{N_j}. \tag{62}$$

We refer to (61) as the priority value function of the *load-balancing (LB) policy* denoted by π_{LB}. As a special case of the hybrid Gc/μ rule, the LB policy with the operating cost function (60) can also attain the optimal value of the optimization problem (42). We formally state it in the following corollary of Theorem 7.

Corollary 1 (Optimality of the Load-balancing (LB) Policy). *Given* $p = 0$ *and* $\lambda \leq \sum_{j=1}^{J} \mu_j N_j$, *if the operating cost function satisfies* (60), *then the fluid model* (14)–(20) *under the load-balancing policy with the priority value function* (61) *satisfies* $\lim_{t\to\infty} B_j(t) = b_j^{p,*}$, $j = 1, \ldots, J$, $\lim_{t\to\infty} Q(t) = 0$ *and* $\lim_{T\to\infty} L_T^O(\pi_{LB}) = L^{O,*}$. *In detail,* $b_j^{p,*} = \frac{\lambda}{\sum_{j=1}^{J} \mu_j N_j} N_j$ *and*

$$\frac{B_j(t)}{N_j} - \frac{B_k(t)}{N_k} \to 0 \quad \text{as } t \to \infty, \tag{63}$$

for all $j, k = 1, \ldots, J$.

5.2. The Idleness-ratio (IR) Policy.

The idleness-ratio (IR) policy is a special case of queues-and-idleness ratio (QIR) policies for service systems with multiple server pools and multiple customer classes that were proposed and analyzed by Gurvich and Whitt (2009a,b, 2010). Adopting the QIR policy to the inverted-V model, the IR policy routes customers to the pool with the highest idleness imbalance. Indeed, given the service-level target $p = 0$, our proposed hybrid Gc/μ rule can also degenerate to the IR policy by setting the operating cost function to be

$$C_j(x) = \frac{\mu_j}{2w_j}(x - N_j)^2, \quad j = 1, \ldots, J, \tag{64}$$

where $\sum_{j=1}^{J} w_j = 1$ and $0 < w_j < 1$ is a priori fixed constant. The priority value function of the hybrid Gc/μ rule (49) will be

$$P_j(t) = \frac{B_j(t) - N_j}{w_j} = -\frac{I_j(t)}{w_j}, \quad j = 1, \ldots, J. \tag{65}$$

This together with (43) and (44) implies that upon arrival the fluid content at the head of the queue is routed to the highest idleness imbalance pool with the index

$$\arg\max_{j=1,\ldots,J} \frac{I_j(t)}{w_j},$$

which is in the same spirit as (12) in Gurvich and Whitt (2010). We refer to (65) as the priority value function of the *idleness-ratio (IR) policy* denoted by π_{IR}. Let $I(t) = \sum_{j=1}^{J} I_j(t)$ be the total available service resource among all server pools. The basic idea of the IR policy is to route the arrivals in such a way that the vector $(I_1(t), \ldots, I_J(t))$ is as close to $(w_1 I(t), \ldots, w_J I(t))$ as possible. This is shown in the following corollary of Theorem 7.

Corollary 2 (Optimality of the Idleness-ratio (IR) Policy). *Given* $p = 0$ *and* $\lambda \leq \sum_{j=1}^{J} \mu_j N_j$, *if the operating cost function satisfies* (64), *then the fluid model* (14)–(20) *under the idleness-ratio (IR) policy with the priority value function* (65) *satisfies* $\lim_{t\to\infty} B_j(t) = b_j^{p,*}$, $j = 1, \ldots, J$, $\lim_{t\to\infty} Q(t) = 0$ *and* $\lim_{T\to\infty} L_T^O(\pi_{IR}) = L^{O,*}$. *In detail,* $b_j^{p,*} = N_j - w_j \frac{\sum_{j=1}^{J} \mu_j N_j - \lambda}{\sum_{j=1}^{J} w_j \mu_j}$ *and*

$$\lim_{t\to\infty} \frac{I_j(t)}{I(t)} = w_j,$$

for all $j = 1, \ldots, J$.

5.3. The c/μ Rule

We consider a special case of linear operating cost functions by setting $C_j(x) = c_j x$ for all $j = 1, \ldots, J$. Then the routing problem (42) with service-level target $p = 0$ becomes the following linear programming:

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{J} c_j b_j \\ \text{subject to} \quad & \sum_{j=1}^{J} b_j \mu_j = \lambda, \\ & 0 \le b_j \le N_j, \ j = 1, \ldots, J. \end{aligned} \tag{66}$$

Apparently, the above programming is feasible only when $\lambda \le \sum_{j=1}^{J} b_j N_j$. After replacing b_j with y_j/μ_j, the linear programming (66) can be rewritten as

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{J} \frac{c_j}{\mu_j} y_j \\ \text{subject to} \quad & \sum_{j=1}^{J} y_j = \lambda, \\ & 0 \le y_j \le \mu_j N_j, \ j = 1, \ldots, J. \end{aligned} \tag{67}$$

Due to the simple form of the above objective function, to minimize (67), the obvious solution is to assign as much value (namely $\mu_j N_j$) as possible to y_j with the smaller coefficient c_j/μ_j, and, equivalently, to assign as much value (namely N_j) as possible to b_j with smaller coefficient c_j/μ_j. For convenience, we relabel the indices such that $c_1/\mu_1 \le \cdots \le c_J/\mu_J$. After reordering the indices, the linear programming (66) admits an optimal solution with the same form as (52). Thus, it is straightforward to design a fixed priority policy that assigns higher priority to pools with smaller c_j/μ_j. The optimality of the c/μ rule can easily be seen from Theorem 8. This is exactly the fluid version of the c/μ rule studied in Xia et al. (2022).

Moreover, if we set $c_1 = c_2 = \cdots = c_J$ or simply replace the objective function (66) by

$$\text{minimize} \quad \sum_{j=1}^{J} b_j \tag{68}$$

for the purpose of minimizing the total number of busy servers, then the c/μ rule coincides with the fastest-server-first policy proposed in Armony (2005).

6. Numerical Experiments

In this section, we present some of the numerical experiments we have carried out on the inverted-V model. The main purpose is to confirm our understanding of how the dynamic priority policy works and to test the approximations obtained from the asymptotic analysis. In order to meet a certain service-level target, we also illustrate through some examples how to apply the hybrid policy to improve the design and operation of inverted-V service systems.

6.1. Simulation Parameters

We now explain the parameters in Table 2, where we provide the simulation parameters of an inverted-V model with a buffer and three heterogeneous server pools; this means $J = 3$. In Table 2(a), we present the server pool size N_j's, the pool dependent service rates μ_j's and the operating cost functions C_j's for three pools. The parameters for the buffer are presented in Table 2(b), where we display the arrival rate λ, the abandon rate θ from the queue, the reneging penalty γ for each abandonment from the queue, and the queue-length cost function C_{J+1}.

Server Pool	Size N_j	Service rate μ_j	Operating Cost $C_j(x)$
Pool 1	75	1	$x^2/150$
Pool 2	50	2	$x^2/50$
Pool 3	25	3	$3x^2/50$

(a) Pool sizes, service rates and operating costs for three server pools

Buffer	Arrival rate λ	Abandon rate θ	Reneging penalty γ	Queue-length Cost $C_{J+1}(x)$
Queue	200	2	0.2	$x^2/200$

(b) Arrival rate, abandon rates, reneging penalty and queue-length cost

Table 2 Simulation parameters for an inverted-V model

We assume that the arrivals follow a Poisson process with rate λ and that customers' patience for waiting in the queue has an exponential distribution with rate θ. For notational simplicity, we use "expo(x)" to denote an exponential distribution with mean x, "$E_2(x)$" to denote an Erlang E_2 distribution with mean x, and "$\ln(x, y)$" to denote a log-normal distribution with mean x and variance y. It is well known that the steady state of the fluid approximation depends upon the service time distributions only through their expectations (Whitt (2006)). Thus we simulate the system with three different service time distributions, i.e, "expo($1/\mu_j$)", "$E_2(1/\mu_j)$" and "$\ln(1/\mu_j, 1/\mu_j^2)$", which have the same service rate μ_j for any $j = 1, 2, 3$.

We run each simulation long enough to observe 2 million arrivals with the given parameters and use 10 independent simulation runs to obtain confidence intervals. The first 10% and the last 10% of the simulation period are regarded as the warm-up and close-down periods of the system; thus, they are discarded when computing the steady-state performance metrics.

6.2. Performance under the Gc/μ Rule

The operating and queue-length cost functions specified in Table 2 are all convex. Thus, it is suitable to apply the Gc/μ rule to minimize the long-run average total cost. Considering the Gc/μ rule and applying the parameters in Table 2 to (31) yield

$$P_1(t) = \frac{B_1(t)}{75}, \quad P_2(t) = \frac{B_2(t)}{50}, \quad P_3(t) = \frac{B_3(t)}{25} \quad \text{and} \quad P_4(t) = \frac{Q(t)}{200} + 0.2, \tag{69}$$

where $P_4(t)$ is the priority value function for the queue.

We present the results of our simulation experiments under the Gc/μ rule in Table 3. The steady state of the fluid model under the Gc/μ rule can easily be computed given the experimental setting in Table 2 and the priority value functions for three pools and the queue in (69). Indeed, the steady state (b^*, q^*) can be obtained by solving the KKT conditions (28)–(30). Then the holding cost, operating cost, and total cost follow directly from the objective function of (24). This yields the fluid approximation of the system, which is displayed in the last column of Table 3 for comparison with the simulation results. In Table 3, we also present the simulation approximations for Q, B_j's, the long-run average holding, operating, and total costs, along with their relative errors and 95% confidence intervals for three different service time distributions. One can find that, under the Gc/μ rule, there are customers waiting in the queue and all three pools are not fully occupied. This means that we need to intentionally allow some idle servers to attain the optimal total cost of the system.

	Exponential expo$(1/\mu_i)$		Erlang $E_2(1/\mu_i)$		Log-normal $\ln(1/\mu_i, 1/\mu_i^2)$		
Performance	Simulation	Relative Error(%)	Simulation	Relative Error(%)	Simulation	Relative Error(%)	Approximation
Q	45.459	1.51	45.467	1.49	45.479	1.46	46.154
	±0.213		±0.177		±0.209		
B_1	32.661	1.09	32.658	1.08	32.664	1.10	32.308
	±0.080		±0.068		±0.078		
B_2	21.720	0.85	21.722	0.85	21.724	0.86	21.538
	±0.054		±0.041		±0.052		
B_3	10.588	1.68	10.980	1.96	10.983	1.99	10.769
	±0.066		±0.020		±0.026		
Holding cost	28.690	1.45	28.701	1.42	28.692	1.45	29.113
	±0.150		±0.150		±0.129		
Operating cost	23.923	3.14	23.927	3.16	23.921	3.13	23.195
	±0.115		±0.114		±0.093		
Total cost	52.614	0.58	52.613	0.58	52.628	0.61	52.308
	±0.265		±0.221		±0.263		

Table 3 The Gc/μ rule: comparison of the simulation results and approximations

It is worth pointing out that the approximation using the fluid steady state performs well, not only for systems with exponential service times but also for systems with general service times. For example, the value of B_1 is 32.661 when service time distributions in different pools are exponential. The corresponding values of B_1 for Erlang E_2 and log-normal distributions are 32.658 and 32.664, respectively. The relative errors of the approximations for B_1 are all less than 1.10% in the three cases with different service time distributions. The results of other performance metrics are also close to each other and remarkably accurate.

6.3. Performance under the Hybrid Gc/μ Rule

Now consider a routing problem with any service-level target $p \in [0,1]$. As an optimal solution of (42), the value of $b^{p,*} = (b_1^{p,*}, \ldots, b_J^{p,*})$ can be solved by applying the KKT conditions (47) and (48). In view of the objective function of (42), in the fluid steady state, the optimal operating cost is $\sum_{j=1}^{J} C_j(b_j^{p,*})$. It is clear that the optimal solution of (42) is $q^{p,*} = p\lambda/\theta$, where λ and θ are given in Table 2(b). With regard to the objective function of (24), the corresponding fluid holding cost should be $C_J(q^{p,*}) + \gamma\theta q^{p,*} = C_J(p\lambda/\theta) + \gamma p\lambda$. Consequently, the fluid total cost is calculated as the sum of the holding and operating costs.

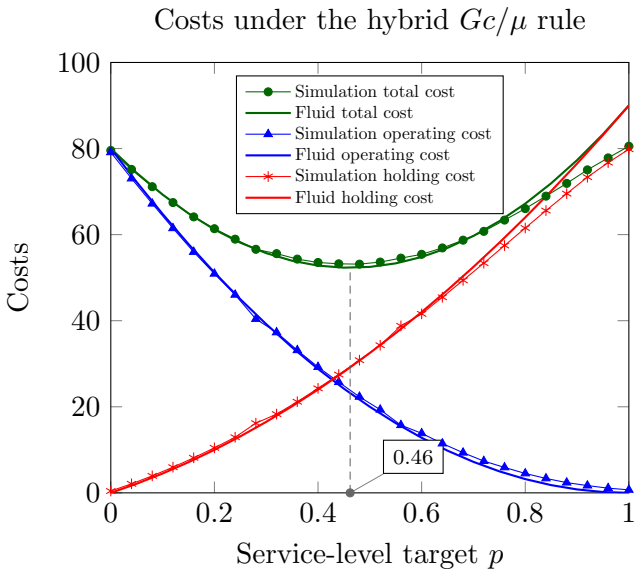


Figure 2 Costs of the inverted-V models under the hybrid Gc/μ rule for any service level target $p \in [0,1]$

Given the parameters in Table 2, we apply the hybrid Gc/μ rule with any possible service-level target p between 0 and 1. Since the simulation results are almost identical for different service time distributions, we only consider the case with exponential service times. Figure 2 depicts the simulation results of the operating cost together with the corresponding holding and total costs. The fluid holding, operating, and total costs are also plotted in Figure 2. The graph shows that the fluid approximation is suitable for most service-level targets $p \in [0,1]$. However, for some values of p (i.e., a small value range near 1), the approximation is not close enough to obtain an accurate performance evaluation. For example, when p is very close to 1, our fluid approximation overestimates the holding and total costs, which will be explained in Section 6.3.3.

One can find that the (fluid) holding cost increases with p but the (fluid) operating cost decreases with p. Moreover, we can use Proposition 2 to serve as a quantitative guide to find the service-level target p that minimizes the total cost. It turns out to be $p = \theta q^*/\lambda = 2 \times 46.154/200 = 0.46$, where

$q^* = 46.154$ is the fluid approximation of the queue length displayed in Table 3. It is also important to point out that both the fluid and simulation costs exhibit a "flat bottom", suggesting that the choice of the service-level target can be quite robust. A relatively wide range of choices of the service-level target gives similar costs that are close to the optimum.

6.3.1. Hybrid Gc/μ Rule with the Service-level Target $p = \theta q^*/\lambda$ We have shown above that $p = \theta q^*/\lambda = 0.46$, which is the point that minimizes the total cost as illustrated in Figure 2. To show its connection with the result in Section 6.2, we display the simulation results and the fluid approximations of an inverted-V model under the hybrid Gc/μ rule with $p = \theta q^*/\lambda$ in Table 4. Clearly, our approximations using the fluid steady state are fairly accurate for all three different service time distributions.

	Exponential expo$(1/\mu_i)$		Erlang $E_2(1/\mu_i)$		Log-normal $\ln(1/\mu_i, 1/\mu_i^2)$		
Performance	Simulation	Relative Error(%)	Simulation	Relative Error(%)	Simulation	Relative Error(%)	Approximation
Q	46.170	0.03	46.170	0.03	46.170	0.03	46.154
	±0.008		±0.008		±0.008		
B_1	33.079	2.39	33.100	2.45	32.656	2.43	32.308
	±0.203		±0.017		±0.005		
B_2	21.420	0.55	21.403	0.59	21.402	0.55	21.538
	±0.124		±0.011		±0.012		
B_3	10.588	1.68	10.575	1.76	10.575	1.76	10.769
	±0.066		±0.005		±0.006		
Holding cost	29.131	0.06	29.131	0.06	29.131	0.06	29.113
	±0.033		±0.033		±0.033		
Operating cost	23.908	3.07	23.992	3.44	23.938	3.20	23.195
	±0.292		±0.294		±0.245		
Total cost	53.039	1.40	53.084	1.45	53.048	1.56	52.308
	±0.268		±0.026		±0.027		

Table 4 Hybrid Gc/μ Policy with $p = \theta q^*/\lambda$: comparison of the simulation results and approximations

Another important finding is that the system performance under the hybrid Gc/μ rule with the service-level target $p = \theta q^*/\lambda$ is quite close to that of the Gc/μ rule (see Tables 3 and 4). This verifies Proposition 2 showing that once the service-level target p is set to be $\theta q^*/\lambda$ the hybrid Gc/μ rule and the Gc/μ rule attain the same steady state; as do the long-run average holding, operating, and total costs.

6.3.2. Hybrid Gc/μ Rule with the Service-level Target $p = 0$ In Table 5, we report the simulation results and their fluid approximations for an inverted-V model under the hybrid Gc/μ rule with $p = 0$ for three different service time distributions. We find that our approximations using the fluid steady state are still very accurate. The relative errors for Q and the holding cost are

omitted since their fluid approximations are 0. For other performance metrics, the relative errors are less than 0.83% across all simulations with different service time distributions.

Given the operating cost functions in Table 2(a), the hybrid Gc/μ rule with $p = 0$ actually degenerates to the load-balancing policy (62), which means that the servers in different server pools are utilized fairly. As shown in Table 5, the simulation results with exponential service time distributions satisfy $B_1/N_1 = 60.447/75 = 0.806$, $B_2/N_2 = 39.874/50 = 0.797$ and $B_3/N_3 = 19.899/25 = 0.796$, showing that the utilization of every server pool is close to 0.8. This verifies the key feature of the load-balancing policy shown in (63).

	Exponential expo$(1/\mu_i)$		Erlang $E_2(1/\mu_i)$		Log-normal $\ln(1/\mu_i, 1/\mu_i^2)$		
Performance	Simulation	Relative Error(%)	Simulation	Relative Error(%)	Simulation	Relative Error(%)	Approximation
Q	0.114	—	0.108	—	0.110	—	0
	±0.016		±0.012		±0.021		
B_1	60.447	0.75	60.455	0.76	60.457	0.76	60
	±0.151		±0.161		±0.143		
B_2	39.874	0.32	39.873	0.32	39.875	0.31	40
	±0.097		±0.105		±0.098		
B_3	19.899	0.51	19.898	0.51	19.897	0.52	20
	±0.048		±0.053		±0.050		
Holding cost	0.049	—	0.045	—	0.046	—	0
	±0.009		±0.005		±0.009		
Operating cost	80.604	0.76	80.607	0.76	80.617	0.77	80
	±0.382		±0.418		±0.385		
Total cost	80.652	0.82	80.653	0.82	80.663	0.83	80
	±0.382		±0.419		±0.380		

Table 5 Hybrid Gc/μ rule with $p = 0$: comparison of the simulation results and approximations

6.3.3. Hybrid Gc/μ Rule with the Service-level Target $p = 1$ In Figure 2, we can see that our fluid approximations are not close enough to the simulations results when p is close to 1. Table 6 presents the simulation results and fluid approximations for an inverted-V model under the hybrid Gc/μ rule with $p = 1$ for three different service time distributions. The approximation indicates that, eventually, 100 customers will wait in the queue and no customer will be serviced in the server pools. However, in the simulations, the queue length oscillates around 100 from below because of the randomness in arrivals and customer abandonment. Furthermore, the new arrivals will be routed to one of the server pools whenever the queue length exceeds 100. This is the key reason why our fluid approximation overestimates the holding and total costs. Note that the $p = 1$ case reveals the fluid approximation for an inverted-V model with the least accuracy. The relative errors for the queue length, holding cost, and total cost shown in Table 6 can be considered as the upper bounds for these relative errors when using the fluid approximation for any given p value. For a more accurate performance evaluation, more refined approximations such as a diffusion approximation are required.

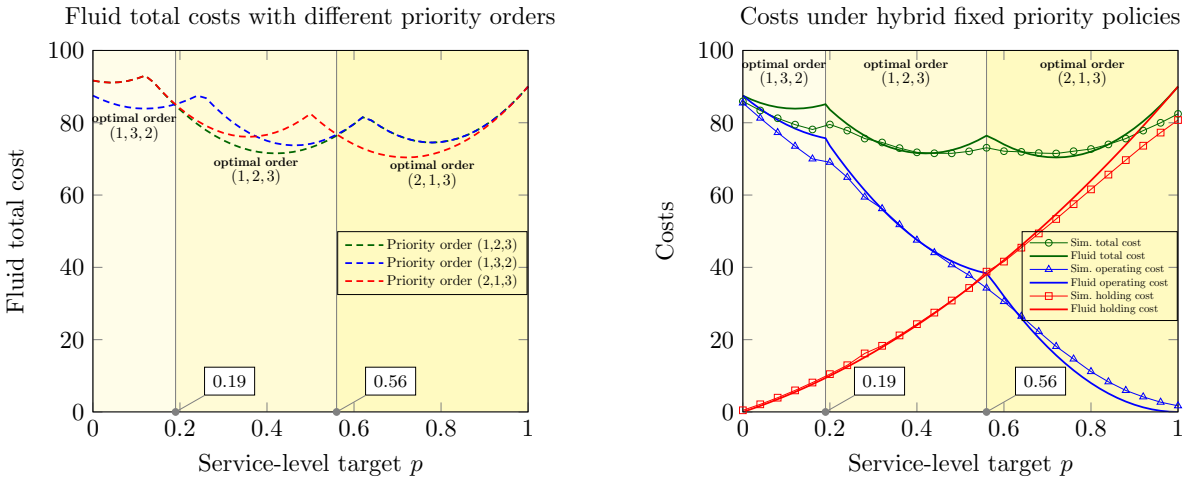
	Exponential expo$(1/\mu_i)$		Erlang $E_2(1/\mu_i)$		Log-normal $\ln(1/\mu_i, 1/\mu_i^2)$		
Performance	Simulation	Relative Error(%)	Simulation	Relative Error(%)	Simulation	Relative Error(%)	Approximation
Q	92.284	7.72	92.238	7.76	92.273	7.73	100
	±0.208		±0.221		±0.217		
B_1	5.120	—	5.177	—	5.137	—	0
	±0.232		±0.222		±0.217		
B_2	2.819	—	2.844	—	2.825	—	0
	±0.124		±0.120		±0.124		
B_3	1.539	—	1.547	—	1.542	—	0
	±0.051		±0.047		±0.043		
Holding cost	79.716	11.43	79.656	11.49	79.702	11.44	90
	±0.196		±0.212		±0.208		
Operating cost	0.697	—	0.739	—	0.715	—	0
	±0.060		±0.060		±0.057		
Total cost	80.413	10.65	80.394	10.67	80.417	10.65	90
	±0.254		±0.269		±0.262		

Table 6 Hybrid Gc/μ rule with $p = 1$: comparison of the simulation results and approximations

6.4. Performance under the Hybrid Fixed Priority Policy

In this subsection, we restrict ourselves to the family of hybrid fixed priority policies, using the parameters in Table 2, and find the optimal priority order for any service-level target $p \in [0, 1]$. Since there are only three pools ($J = 3$) in this example, we can use the brute-force algorithm to evaluate all of the six possible priority orders. According to (22) and Proposition 3, we plot in Figure 3(a) the fluid total costs with three different priority orders. We omit the other three priority orders since their corresponding costs are larger. We can see from Figure 3(a) that the optimal priority order varies as p changes. If $0 \le p < 0.19$, the optimal priority order is $(1, 3, 2)$, whereby pool 1 has the highest priority, pool 3 has the second highest priority, and pool 2 has the lowest priority. If $0.19 \le p < 0.56$, the optimal priority order becomes $(1, 2, 3)$. Otherwise, $0.56 \le p \le 1$, the optimal priority order is $(2, 1, 3)$. The change in the optimal priority order with p reflects the complicated trade-off between service speeds and operating costs among the server pools. Thus, we divide the graph into three regions, as shown in Figure 3. Since the brute-force algorithm is NP-hard, when the number of pools J is large, we can utilize the dynamic programming algorithm introduced in Section EC.4 to obtain the optimal priority order more efficiently.

Once the optimal priority order is determined for a given service-level target p, we run simulations to obtain the total cost, holding cost, and operating cost, using the parameters in Table 2 under the hybrid fixed priority policy. Their fluid approximations can be calculated using the results in Proposition 3. Similar to Figure 2, the simulated and fluid costs under the hybrid fixed priority policies are plotted in Figure 3(b). The graph shows that the fluid approximations of the total, operating, and holding costs are suitable because the expected cost of the stochastic systems dips and peaks with the corresponding fluid costs. However, at some service-level targets, the



(a) A means of finding the optimal priority order

(b) Costs under hybrid fixed priority policies for any p

Figure 3 The optimal priority order and costs of the inverted-V model under hybrid fixed priority policies

approximation is not close enough to obtain an accurate performance evaluation, particularly for the turning points (where the optimal priority order changes). For these particular points, more refined approximations such as a diffusion approximation are required, which is beyond the aim of this paper.

7. Conclusion

In this paper, we address some of the fundamental issues that arise in the operations of inverted-V models based on fluid model analysis. Three non-work-conserving policies are proposed to cope with any general cost functions in order to trade off the holding and operating costs. Specifically, the target-allocation policy works for any general cost functions. Our Gc/μ rule optimally controls an inverted-V model with convex costs and can be viewed as a counterpart of the $Gc\mu$ rule in van Mieghem (1995). The fixed priority policy is asymptotically optimal for concave cost functions. We also develop a dynamic programming algorithm to find the optimal priority order. This will be much more efficient when the number of server pools is large.

To minimize the operating cost of the system but still meet a certain service-level target, we propose another three hybrid routing policies; the hybrid target-allocation policy, the hybrid Gc/μ rule, and the hybrid fixed priority policy. The hybrid routing policies not only inherit the advantages of the aforementioned three routing policies but also allow us to characterize various practical systems with different service-level targets. More specifically, the hybrid Gc/μ rule covers the LB policy in Tezcan (2008) and the IR policy in Gurvich and Whitt (2009a,b, 2010), while the hybrid fixed priority policy extends the c/μ rule in Xia et al. (2022) and the FSF policy in Armony (2005).

Managerial insights for systems with different service-level targets are also obtained from extensive numerical experiments.

An interesting direction for future research would be to study how to adapt the routing problem and extend the results to more general queueing systems with heterogeneous server pools and heterogeneous customers. In addition, the convergence of the fluid model to the steady state with non-exponential service time distributions remains to be established. Although our Gc/μ rule adapts automatically to the changes in the arrival rate and the service capacity in each pool, analysis similar to Liu and Whitt (2014) might result in more accurate performance evaluation in rapidly changing environments.

Acknowledgments

The authors thank the area editor Ramandeep Randhawa, an anonymous associate editor, and two anonymous reviewers for many valuable comments and suggestions that greatly helped improve the paper.

References

Armony, M. (2005). Dynamic routing in large-scale service systems with heterogeneous servers. *Queueing Systems 51*(3-4), 287–329.

Armony, M. and A. Mandelbaum (2011). Routing and staffing in large-scale service systems: The case of homogeneous impatient customers and heterogeneous servers. *Operations Research 59*(1), 50–65.

Armony, M. and A. R. Ward (2010). Fair dynamic routing in large-scale heterogeneous-server systems. *Operations Research 58*(3), 624–637.

Ata, B. and T. L. Olsen (2009). Near-optimal dynamic lead-time quotation and scheduling under convex-concave customer delay costs. *Operations Research 57*(3), 753–768.

Atar, R., C. Giat, and N. Shimkin (2008). The $c\mu/\theta$ rule. In *Proceedings of the 3rd International Conference on Performance Evaluation Methodologies and Tools*, ValueTools '08, ICST, Brussels, Belgium, Belgium, pp. 58:1–58:4. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

Atar, R., C. Giat, and N. Shimkin (2010). The $c\mu/\theta$ rule for many server queues with abandonment. *Operations Research 58*(5), 1427–1439.

Atar, R., C. Giat, and N. Shimkin (2011). On the asymptotic optimality of the $c\mu/\theta$ rule under ergodic cost. *Queueing Systems 67*(2), 127–144.

Atar, R., H. Kaspi, and N. Shimkin (2014). Fluid limits for many-server systems with reneging under a priority policy. *Mathematics of Operations Research 39*(3), 672–696.

Atar, R., Y. Y. Shaki, and A. Shwartz (2011). A blind policy for equalizing cumulative idleness. *Queueing Systems 67*(4), 275–293.

Bassamboo, A. and R. S. Randhawa (2010). On the accuracy of fluid models for capacity sizing in queueing systems with impatient customers. *Operations Research 58*(5), 1398–1413.

Bassamboo, A. and R. S. Randhawa (2016). Scheduling homogeneous impatient customers. *Management Science 62*(7), 2129–2147.

Billingsley, P. (1968). *Convergence of probability measures*. Wiley Series in Probability and Statistics: Probability and Statistics. New York: John Wiley & Sons Inc.

Chen, H. and D. D. Yao (2001). *Fundamentals of queueing networks*, Volume 46 of *Applications of Mathematics (New York)*. New York: Springer-Verlag.

Dupuis, P. and R. S. Ellis (1997). *A weak convergence approach to the theory of large deviations*. Wiley Series in Probability and Statistics: Probability and Statistics. John Wiley & Sons, Inc., New York.

Gurvich, I. and W. Whitt (2009a). Queue-and-idleness-ratio controls in many-server service systems. *Mathematics of Operations Research 34*(2), 363–396.

Gurvich, I. and W. Whitt (2009b). Scheduling flexible servers with convex delay costs in many-server service systems. *Manufacturing & Service Operations Management 11*(2), 237–253.

Gurvich, I. and W. Whitt (2010). Service-level differentiation in many-server service systems via queue-ratio routing. *Operations Research 58*(2), 316–328.

Huang, J., B. Carmeli, and A. Mandelbaum (2015). Control of patient flow in emergency departments, or multiclass queues with deadlines and feedback. *Operations Research 63*(4), 892–908.

Kang, W. and K. Ramanan (2010). Fluid limits of many-server queues with reneging. *The Annals of Applied Probability 20*(6), 2204–2260.

Liu, Y. and W. Whitt (2014). Algorithms for time-varying networks of many-server fluid queues. *INFORMS Journal on Computing 26*(1), 59–73.

Long, Z., N. Shimkin, H. Zhang, and J. Zhang (2020). Dynamic scheduling of multiclass many-server queues with abandonment: The generalized $c\mu/h$ rule. *Operations Research 68*(4), 1218–1230.

Mandelbaum, A., P. Momčilović, and Y. Tseytlin (2012). On fair routing from emergency departments to hospital wards: QED queues with heterogeneous servers. *Management Science 58*(7), 1273—1291.

Mandelbaum, A. and A. L. Stolyar (2004). Scheduling flexible servers with convex delay costs: heavy-traffic optimality of the generalized $c\mu$-rule. *Operations Research 52*(6), 836–855.

Puha, A. L. and A. R. Ward (2022). Fluid limits for multiclass many-server queues with general reneging distributions and head-of-the-line scheduling. *Mathematics of Operations Research 47*(2), 1192–1228.

Smith, W. E. (1956). Various optimizers for single-stage production. *Naval Research Logistics Quarterly 3*(1-2), 59–66.

Tezcan, T. (2008). Optimal control of distributed parallel server systems under the Halfin and Whitt regime. *Mathematics of Operations Research 33*(1), 51–90.

van Mieghem, J. A. (1995). Dynamic scheduling with convex delay costs: The generalized $c\mu$ rule. *The Annals of Applied Probability 5*(3), 809–833.

Whitt, W. (2006). Fluid models for multiserver queues with abandonments. *Operations Research 54*(1), 37–54.

Wu, C. A., A. Bassamboo, and O. Perry (2019). Service system with dependent service and patience times. *Management Science 65*(3), 1151–1172.

Xia, L., Z. G. Zhang, and Q.-L. Li (2022). A c/μ-rule for job assignment in heterogeneous group-server queues. *Production and Operations Management 31*(3), 1191–1215.

Zhan, D. and A. R. Ward (2019). Staffing, routing, and payment to trade off speed and quality in large service systems. *Operations Research 67*(6), 1738–1751.

Zhang, J. (2013). Fluid models of many-server queues with abandonment. *Queueing Systems 73*(2), 147–193.

Zhang, Z. G., H. P. Luh, and C.-H. Wang (2011). Modeling security-check queues. *Management Science 57*(11), 1979–1995.

Zhong, Y., A. R. Ward, and A. L. Puha (2022). Asymptotically optimal idling in the $GI/GI/N+GI$ queue. *Operations Research Letters 50*(3), 362–369.

Electronic Companion of "The Generalized c/μ Rule for Queues with Heterogeneous Server Pools"

We prove Theorem 1 in Section EC.1. In Section EC.2, we prove the optimality of the target-allocation policy, the Gc/μ rule, and the fixed priority policy. The optimality of the corresponding three hybrid routing policies is provided in Section EC.3. We show in Section EC.4 the connection between inverted-V models and knapsack problems.

EC.1. Proofs of the Asymptotic Analysis

Proof of Theorem 1. We first show that $\bar{R}^n$ is tight. With regard to (1) and (10), there is $\bar{R}^n(t) = \frac{1}{n}\tilde{R}^n\big(\int_0^t Q^n(s)ds\big)$, where $\tilde{R}^n(\cdot)$ is a Poisson process with rate $\theta^n = \theta$. It follows from the functional strong law of large numbers that $\frac{1}{n}\tilde{R}^n(nt) \to \theta t$ almost surely as n goes to infinity. For any $T > 0$, by (2), (9) and the initial state $\bar{Q}^n(0) \Rightarrow Q(0)$ we have $\frac{1}{n}\int_0^t Q^n(s)ds \le 2(Q(0) + \lambda T)t$, $0 \le t < T$, and $\frac{1}{n}\int_\tau^t Q^n(s)ds \le 2(Q(0)+\lambda T)(t-\tau)$, $0 \le \tau \le t < T$, for all large n. Thus, $\frac{1}{n}\int_0^t Q^n(s)ds$ is tight by Theorem 15.5 in Billingsley (1968). It then follows from the random-time-change theorem (see Theorem 5.3 in Chen and Yao (2001)) that $\bar{R}^n$ is tight. Next, we show that $\bar{D}_j^n$ is tight. The proof is almost the same as that of $\bar{R}^n$. In view of (1) and (10), we have $\bar{D}_j^n(t) = \frac{1}{n}\tilde{D}_j^n\big(\int_0^t B_j^n(s)ds\big)$, where $\tilde{D}_j^n(\cdot)$ is a Poisson process with rate $\mu_j^n = \mu_j$. It follows from the functional strong law of large numbers that $\frac{1}{n}\tilde{D}_j^n(nt) \to \mu_j t$ almost surely as n goes to infinity. By (9), for all large n there is $\frac{1}{n}\int_0^t B_j^n(s)ds \le 2N_j t$, $0 \le t$, and $\frac{1}{n}\int_\tau^t B_j^n(s)ds \le 2N_j(t-\tau)$, $0 \le \tau \le t$. We know that $\frac{1}{n}\int_0^t B_j^n(s)ds$ is tight by Theorem 15.5 in Billingsley (1968). It then follows from the random-time-change theorem that $\bar{D}_j^n$ is tight. It directly follows from (9) that $\bar{\Lambda}^n(t) \Rightarrow \lambda t$. Thus, the tightness of $\bar{\Lambda}^n$ holds. Then the tightness of $\sum_{j=1}^J \bar{E}_j^n(t)$ follows from (7). Since the entrance into service process $E_j^n(t)$ is nondecreasing, it follows that each $\bar{E}_j^n$, $j = 1, \ldots, J$, must be tight. Then, the tightness of $\bar{E}_{J+1}^n$ follows from (4). As a consequence of (2), $\bar{Q}^n(t)$ is also tight. The tightness of $\bar{B}_j^n$ follows from (3). Then, $\bar{I}_j^n(t) = N_j^n/n - \bar{B}_j^n(t)$, $j = 1, \ldots, J$, is also tight.

We have so far proven the tightness of the fluid-scaled stochastic processes. This shows the existence of the fluid limit implying that the sequence of the fluid-scaled processes $\{(\bar{\Lambda}^n, \bar{R}^n, \bar{E}^n, \bar{D}^n, \bar{Q}^n, \bar{B}^n, \bar{I}^n) : n \in \mathbb{N}\}$ has a subsequence which converges to some limit, denoted by $(\Lambda, R, E, D, Q, B, I)$. We have shown in the above that $\frac{1}{n}\tilde{R}^n(nt) \to \theta t$ and $\frac{1}{n}\tilde{D}_i^n(nt) \to \mu_i t$ almost surely as n goes to infinity. Thus, the fluid dynamic equations (14)–(20) can be verified by the corresponding stochastic equations (1)–(7). This completes the proof. □

Lemma EC.1. *Consider the fluid model* (14)–(20). *Then all the fluid processes* $\Lambda(t), R(t), E(t), D(t), Q(t), B(t)$ *and* $I(t)$ *are absolutely continuous.*

Proof. It is clear that the arrival process $\Lambda(t)$ is absolutely continuous. The absolute continuity of R and D_j follows from (14). Then the absolute continuity of $\sum_{j=1}^{J} E_j(t)$ follows from (20). Since the entrance into service process $E_j(t)$ is nondecreasing, it follows that each E_j, $j=1,\dots,J$, must be absolutely continuous. By (17), E_{J+1} is also absolutely continuous. The absolute continuity of Q follows from (15). Moreover, the absolute continuity of B_j follows from (16). Then, $I_j(t) = N_j - B_j(t)$, $j=1,\dots,J$, is also absolutely continuous. □

EC.2. Proofs of the Optimality of the Routing Policies

EC.2.1. Flow Rates of the Fluid Model

Let ${}_*J_k(t)$ be the collection of indices with the first kth smallest priority value at time t recursively defined as follows:

$$ {}_*J_1(t) = \operatorname*{arg\,min}_{j\in\{1,\dots,J,J+1\}} P_j(t), \tag{EC.1}$$

and for $1 \le k \le J$,

$$ {}_*J_{k+1}(t) = {}_*J_k(t) \cup \operatorname*{arg\,min}_{j\in\{1,\dots,J,J+1\}\setminus {}_*J_k(t)} P_j(t).$$

Lemma EC.2. *Consider the fluid model* (14)–(20) *given any continuous priority value function* $P_j(t)$. *Then the entrance into service processes* $E_j(t)$ *are absolutely continuous, and the derivatives* $E_j'(t) := (d/dt)E_j(t)$ *satisfy a.e. for* $j=1,\dots,J,J+1$,

$$ \sum_{j\in {}_*J_k(t)} E_j'(t) = \begin{cases} \lambda & \text{if } \sum_{j\in {}_*J_k(t)} I_j(t) > 0, \\ \lambda \wedge \sum_{j\in {}_*J_k(t)} \mu_j N_j & \text{if } \sum_{j\in {}_*J_k(t)} I_j(t) = 0, \end{cases} \tag{EC.2}$$

where $a\wedge b$ *is the minimum of* a *and* b.

Proof. The absolute continuity of E_j has been proven in Lemma EC.1.

First we consider the case with $\sum_{j\in {}_*J_k(t)} I_j(t) > 0$ for some t. By (17), we have $\sum_{j=1}^{J+1} E_j'(t) = \lambda$. On the other hand, there must be $E_j'(t) = 0$ for all $j \notin {}_*J_k(t)$ by (26). This yields that $\sum_{j\in {}_*J_k(t)} E_j'(t) = \lambda$ if $\sum_{j\in {}_*J_k(t)} I_i(t) > 0$.

Next we consider the case with $\sum_{j\in {}_*J_k(t)} I_j(t) = 0$ for some t. It is clear that $J+1 \notin {}_*J_k(t)$ by (19). Since I_i's are absolutely continuous, it follows that $\sum_{j\in {}_*J_k(t)} I_j'(t) = 0$ a.e. on $S := \{t : \sum_{j\in {}_*J_k(t)} I_j(t) = 0\}$ by Theorem A.6.3 in Dupuis and Ellis (1997). Moreover, from (16) and (19) we have

$$ \sum_{j\in {}_*J_k(t)} B_j'(t) = \sum_{j\in {}_*J_k(t)} E_j'(t) - \sum_{j\in {}_*J_k(t)} \mu_j B_j(t),$$
$$ B_j(t) = N_j - I_j(t) = N_j, \quad \text{for all } j \in {}_*J_k(t).$$

Thus a.e. on S, we have $\sum_{j\in {}_*J_k(t)} E_j'(t) = \sum_{j\in {}_*J_k(t)} \mu_j N_j$. This together with the fact that $\sum_{j=1}^{J+1} E_j'(t) = \lambda$ by (17) yields a.e. on S, $\sum_{j\in {}_*J_k(t)} E_j'(t) = \sum_{j\in {}_*J_k(t)} \mu_j N_j = \lambda \wedge \sum_{j\in {}_*J_k(t)} \mu_j N_j$. This completes the proof. □

EC.2.2. Optimality of the Target-allocation Policy and the Gc/μ Rule

In view of the fact that the priority value functions go to an equal constant under both policies. We will show that the proofs of the optimality of the target-allocation policy and the Gc/μ rule are exactly the same. Thus we prove Theorems 2 and 3 simultaneously, which is presented in the end of this subsection. Prior to that, some auxiliary Lemmas EC.3–EC.5 are analyzed. First, we introduce some additional notations.

As we have argued below (25), in the fluid model the buffer can be regarded as another server pool indexed by $J+1$ with E_{J+1} in (18) being the "entrance into service" process and R in (18) being the "departure" process. The fluid queue length Q can also be considered as the amount of fluid content "being served" in pool $J+1$. For notational simplicity, we set

$$B_{J+1}(t) := Q(t), \quad D_{J+1}(t) := R(t), \quad \mu_{J+1} := \theta, \quad b^*_{J+1} := q^*. \tag{EC.3}$$

Then the balance equation for the fluid queue length (18) can be written as

$$B_{J+1}(t) = B_{J+1}(0) + E_{J+1}(t) - D_{J+1}(t). \tag{EC.4}$$

For the target-allocation policy π_{b^*,q^*} proposed in Section 3.1.1, let

$$A_j(x) = x - b^*_j + \alpha_0, \quad j = 1, \ldots, J, J+1, \tag{EC.5}$$

where α_0 can be chosen as any constant. Note that in (EC.3) we set $b^*_{J+1} = q^*$ when $j = J+1$. To have the same proof as the optimality of the Gc/μ rule, we choose α_0 to be the one in (28) and (29). With a slight abuse of the notation, for the Gc/μ rule we also introduce $A_j(\cdot)$ as follows:

$$A_j(x) = \frac{c_j(x)}{\mu_j} + \gamma \mathbb{1}_{\{j=J+1\}}, \quad j = 1, \ldots, J, J+1. \tag{EC.6}$$

Note that by (28), (29) and (EC.5), we have

$$A_j(b^*_j) = \alpha_0, \quad j = 1, \ldots, J, J+1, \tag{EC.7}$$

for both $A_j(\cdot)$ in (EC.5) and (EC.6). Obviously, $A_j(\cdot)$ in (EC.5) is strictly increasing, and $A_j(\cdot)$ in (EC.6) is also strictly increasing under Assumption 1. Thus, within this subsection $A_j(\cdot)$ could either be (EC.5) or (EC.6). Now we introduce

$${}_*A(B(t)) := \min_{j=1,\ldots,J,J+1} A_j(B_j(t)). \tag{EC.8}$$

It should be pointed out that $B_{J+1}(t) = Q(t)$ by (EC.3). In view of (27), (EC.3) and (EC.5), for the target-allocation policy, we can consider $A_j(B_j(t))$ as the priority value function instead of the one in (27). Then ${}_*J_1(t)$ in (EC.1) can be replaced by

$${}_*J_1(t) := \{j \in \{1, \ldots, J, J+1\} : A_j(B_j(t)) = {}_*A(B(t))\}, \tag{EC.9}$$

which is the collection of indices with the smallest priority value at time t. And define

$$ {}_*B_j(t) := \{\zeta \geq 0 : A_j(\zeta) = {}_*A(B(t))\}. \tag{EC.10} $$

Lemma EC.3. *Consider the fluid model* (14)–(20) *given the priority value function* (27) *or* (31). *If* ${}_*A(B(t)) \leq \alpha_0$, *then we have*

$$ \sum_{j \in {}_*J_1(t)} B_j'(t) \geq 0, \tag{EC.11} $$

$$ \frac{d}{dt}[{}_*A(B(t))] \geq 0. \tag{EC.12} $$

Proof. Since ${}_*A(B(t)) \leq \alpha_0$, we have $B_j(t) \leq b_j^*$ for all $j \in {}_*J_1(t)$ by (EC.7) and (EC.9). From (16),

$$ \sum_{j \in {}_*J_1(t)} B_j'(t) = \sum_{j \in {}_*J_1(t)} E_j'(t) - \sum_{j \in {}_*J_1(t)} D_j'(t). \tag{EC.13} $$

By Lemma EC.2, the above expression is nonnegative once $\sum_{j \in {}_*J_1(t)} E_j'(t) = \sum_{j \in {}_*J_1(t)} \mu_j N_j$. So we only need to consider the other possible case $\sum_{j \in {}_*J_1(t)} E_j'(t) = \lambda$ when proving (EC.11), which still holds since $\sum_{j \in {}_*J_1(t)} D_j'(t) = \sum_{j \in {}_*J_1(t)} \mu_j B_j(t) \leq \sum_{j \in {}_*J_1(t)} \mu_j b_j^* \leq \lambda$. Here the last inequality follows from the first constraint of (24) and (EC.3). Thus (EC.11) holds. We cannot have $(d/dt){}_*A(B(t)) < 0$ since this would imply $B_j'(t) < 0$ for all $j \in {}_*J_1(t)$ by (EC.9). This contradicts (EC.11). So we have (EC.12). □

Lemma EC.4. *Consider the fluid model* (14)–(20) *given the priority value function* (27) *or* (31). *If there exists a* $\tau_0 \geq 0$ *such that* ${}_*A(B(\tau_0)) \geq \alpha_0$, *then we have*

$$ {}_*A(B(t)) \geq \alpha_0 \quad \textit{for all } t \geq \tau_0, \tag{EC.14} $$

$$ B_j(t) \geq b_j^* \quad \textit{for all } t \geq \tau_0 \textit{ and } j = 1, \ldots, J, J+1. \tag{EC.15} $$

Proof. Suppose contrarily there exists a $t_1 > \tau_0$ such that ${}_*A(B(t_1)) < \alpha_0$. Let $t_0 = \sup\{t_0 < t_1 : {}_*A(B(t_0)) \geq \alpha_0\}$. It is clear that ${}_*A(B(t_0)) = \alpha_0$ and ${}_*A(B(t)) < \alpha_0$ for all $t \in (t_0, t_1]$. This together with (EC.12) yields ${}_*A(B(t_1)) = {}_*A(B(t_0)) + \int_{t_0}^{t_1} d[{}_*A(B(t))] \geq \alpha_0$. It contradicts the assumption that ${}_*A(B(t_1)) < \alpha_0$. Thus (EC.14) holds. Then, (EC.15) directly follows from (EC.5), (EC.7), (EC.8) and (EC.14). □

Lemma EC.5. *Consider the fluid model* (14)–(20) *given the priority value function* (27) *or* (31). *If* $\sum_{j=1}^{J+1} {}_*B_j(t) \leq \sum_{j=1}^{J+1} b_j^* - \delta$ *for some* $\delta > 0$, *then there exists a constant* $\epsilon_0 > 0$ *depending only on* δ *such that*

$$ B_j(t) \leq b_j^* - \epsilon_0 \quad \textit{for all } j \in {}_*J_1(t), \tag{EC.16} $$

and there also exists a constant $\epsilon_1 > 0$ depending only on δ such that

$$\sum_{j \in_* J_1(t)} B_j'(t) \geq \epsilon_1, \tag{EC.17}$$

$$\frac{d}{dt}[\sum_{j=1}^{J+1} {}_*B_j(t)] \geq \epsilon_1. \tag{EC.18}$$

Proof. First, we show that there must be $B_j(t) < b_j^*$ for all $j \in {}_*J_1(t)$ with strict inequalities. Otherwise, we will have $B_j(t) \geq b_j^*$ for at least one $j \in {}_*J_1(t)$, which causes ${}_*A(B(t)) \geq \alpha_0$ following from (EC.7) and (EC.9). Then ${}_*B_j(t) \geq b_j^*$ for all $j = 1, \ldots, J, J+1$ deduced from (EC.10). This is a contradiction to the assumption $\sum_{j=1}^{J+1} {}_*B_j(t) < \sum_{j=1}^{J+1} b_j^*$. Therefore ${}_*A(B(t)) = \alpha_0 - \varepsilon$ for some $\varepsilon > 0$. From (EC.10), we have

$$\sum_{j=1}^{J+1} {}_*B_j(t) = \sum_{j=1}^{J+1} A_j^{-1}(\alpha_0 - \varepsilon) \leq \sum_{j=1}^{J+1} b_j^* - \delta.$$

Let ε^* satisfy $\sum_{j=1}^{J+1} A_j^{-1}(\alpha_0 - \varepsilon^*) = \sum_{j=1}^{J+1} b_j^* - \delta$. There must be $0 < \varepsilon^* \leq \varepsilon$ since A_j^{-1}, $j = 1 \ldots, J, J+1$, are increasing. By (EC.9), for all $j \in {}_*J_1(t)$, $B_j(t) = A_j^{-1}(\alpha_0 - \varepsilon) \leq A_j^{-1}(\alpha_0 - \varepsilon^*) = b_j^* - (b_j^* - A_j^{-1}(\alpha_0 - \varepsilon^*))$. Now let $\epsilon_0 = \min_{j \in_* J_1(t)} (b_j^* - A_j^{-1}(\alpha_0 - \varepsilon^*))$ which is positive and depends only on δ. This proves (EC.16).

By (EC.16), we have $\sum_{j \in_* J_1(t)} I_j(t) > 0$. This together with (EC.2) and (EC.13) yields

$$\begin{aligned}
\sum_{j \in_* J_1(t)} B_j'(t) &= \lambda - \sum_{j \in_* J_1(t)} B_j(t)\mu_j \\
&\geq \lambda - \sum_{j \in_* J_1(t)} (b_j^* - \epsilon_0)\mu_j \\
&\geq \sum_{j \in_* J_1(t)} \epsilon_0 \mu_j \\
&\geq \min_{j = \{1, \ldots, J, J+1\}} \epsilon_0 \mu_j,
\end{aligned}$$

where the first inequality uses (EC.16) and the second inequality is due to the first constraint of (24) and (EC.3). This proves (EC.17).

Next, we prove (EC.18). Using (EC.9) and (EC.10) yields ${}_*B_j'(t) = B_j'(t)$ for all $j \in {}_*J_1(t)$. We have shown in the above that ${}_*A(B(t)) = \alpha_0 - \varepsilon \leq \alpha_0$. Then, by (EC.10) and (EC.12), ${}_*B_j'(t) \geq 0$ for all $j = 1 \ldots, J, J+1$. Therefore,

$$\sum_{j=1}^{J+1} {}_*B_j'(t) \geq \sum_{j \in_* J_1(t)} {}_*B_j'(t) = \sum_{j \in_* J_1(t)} B_j'(t) \geq \epsilon_1,$$

where the second inequality follows from (EC.17). Thus (EC.18) also holds. □

Proof of Theorems 2 and 3. We first show that

$$\liminf_{t\to\infty} B_j(t) \geq b_j^* \quad \text{for all } j=1,\ldots,J,J+1. \tag{EC.19}$$

If there exists a $\tau_0 \geq 0$ such that ${}_*A(B(\tau_0)) \geq \alpha_0$, then (EC.19) directly follows from (EC.15). It suffices to consider the case ${}_*A(B(t)) < \alpha_0$ for all $t \geq 0$. This together with (EC.7) and (EC.10) implies that ${}_*B_j(t) < b_j^*$ for all $j=1,\ldots,J,J+1$ and $t \geq 0$. It then follows from (EC.18) that $\lim_{t\to\infty} {}_*B_j(t) = b_j^*$ for all $j=1,\ldots,J,J+1$. From the definition of ${}_*B_j(t)$ in (EC.10), we have $A_j({}_*B_j(t)) \leq A_j(B_j(t))$. Since A_j is increasing, this inequality implies ${}_*B_j(t) \leq B_j(t)$ for all $j=1,\ldots,J,J+1$. Thus (EC.19) holds. We can conclude from (EC.19) that for any $\epsilon_0 > 0$ and $i=1,\ldots,J,J+1$,

$$B_j(t) \geq b_j^* - \epsilon_0 \quad \text{for all large enough } t. \tag{EC.20}$$

Now we use (EC.20) to prove

$$\lim_{t\to\infty} \sum_{j=1}^{J+1} B_j(t) = \sum_{j=1}^{J+1} b_j^*. \tag{EC.21}$$

To this end, we show that for any $\varepsilon > 0$ there exists a $\delta > 0$ such that for all large enough t

$$\sum_{j=1}^{J+1} B_j'(t) \leq -\delta \quad \text{whenever } \sum_{j=1}^{J+1} B_j(t) \geq \sum_{j=1}^{J+1} b_j^* + \varepsilon. \tag{EC.22}$$

It is clear that there must exist $j_1 \in \{1,\ldots,J,J+1\}$ such that $B_{j_1}(t) \geq b_{j_1}^* + \frac{\varepsilon}{J+1}$. Then we can choose the ϵ_0 in (EC.20) small enough such that

$$\sum_{j=1}^{J+1} D_j'(t) = \sum_{j\neq j_1} \mu_j B_j(t) + \mu_{j_1} B_{j_1}(t) \geq \sum_{j=1}^{J+1} \mu_j b_j^* + m\varepsilon,$$

for all large enough t, where $m > 0$ is a small enough constant. It follows from (17) that $\sum_{j=1}^{J+1} E_j'(t) = \lambda = \sum_{j=1}^{J+1} \mu_j b_j^*$ for all $t \geq 0$, where the second inequality follows the first constraint of (24) and (EC.3). Thus, $\sum_{j=1}^{J+1} B_j'(t) \leq -m\varepsilon$ is strictly negative deduced from the above inequality and (16) and (EC.4). Let $\delta = m\varepsilon$, then (EC.22) holds. This together with (EC.19) yields (EC.21). Moreover, we can conclude from (EC.19) and (EC.21) that $\lim_{t\to\infty} B_j(t) = b_j^*$ for all $j=1,\ldots,J,J+1$. Thus, $\lim_{t\to\infty} Q(t) = q^*$ by (EC.3). This together with (22) yields $\lim_{T\to\infty} L_T(\pi_{b^*,q^*}) = \lim_{T\to\infty} L_T(\pi_G) = L^*$. This completes the proof. □

EC.2.3. Optimality of the Fixed Priority Policy

Proof of Proposition 1. We prove this result in the following two cases.

Case 1: $\lambda > \sum_{l=1}^{k} \mu_l N_l$, $k \in \{0,1,\ldots,J\}$. We first show that there exists $T_0 > 0$ such that

$$B_j(t) = N_j \quad \text{for } j=1,\ldots,k \text{ and } t \geq T_0. \tag{EC.23}$$

It follows from (16) and (EC.2) that whenever $\sum_{j=1}^{k} B_j(t) < \sum_{j=1}^{k} N_j$ there must be

$$\sum_{j=1}^{k} B_j'(t) = \lambda - \sum_{j=1}^{k} \mu_j B_j(t) > \lambda - \sum_{j=1}^{k} \mu_j N_j > 0.$$

The above implies that there exists a $T_0 > 0$ such that $\sum_{j=1}^{k} B_j(t) = \sum_{j=1}^{k} N_j$ for all $t \geq T_0$. Thus, (EC.23) follows. It then follows from (EC.2) that for all $t \geq T_0$, $E_{J+1}'(t) = \lambda - \sum_{j=1}^{k} \mu_j N_j$. Then we can see from (14) and (18) that, for all $t \geq T_0$, $Q'(t) = \lambda - \sum_{j=1}^{k} \mu_j N_j - \theta Q(t)$, of which the solution is $Q(t) = Q(T_0)e^{-\theta(t-T_0)} + \theta^{-1}(\lambda - \sum_{j=1}^{k} \mu_j N_j)(1 - e^{-\theta(t-T_0)})$, $t \geq T_0$. It immediately follows that $\lim_{t\to\infty} Q(t) = (\lambda - \sum_{j=1}^{k} \mu_j N_j)/\theta$. It also follows from (EC.2) that $E_j'(t) = 0$ for all $j = k+1, \ldots, J$ and $t \geq 0$. Similarly, we can see from (14) and (16) that, for all $t \geq 0$, $B_j(t) = B_j(0)e^{-\mu_j t}$, $j = k+1, \ldots, J$. Consequently, $\lim_{t\to\infty} B_j(t) = 0$ for all $j = k+1, \ldots, J$. This proves (34).

Case 2: $\lambda \leq \sum_{l=1}^{k} \mu_l N_l$, $k \in \{0, 1, \ldots, J\}$. We next prove that there exists $T_1 > 0$ such that

$$B_j(t) = N_j \quad \text{for } j = 1, \ldots, j_0 - 1 \text{ and } t \geq T_1. \tag{EC.24}$$

It can be seen from (16) and (EC.2) that whenever $\sum_{j=1}^{j_0-1} B_j(t) < \sum_{j=1}^{j_0-1} N_j$ there must be

$$\sum_{j=1}^{j_0-1} B_j'(t) = \lambda - \sum_{j=1}^{j_0-1} \mu_j B_j(t) > \lambda - \sum_{j=1}^{j_0-1} \mu_j N_j > 0,$$

where the last inequality follows from the definition of j_0. This implies that there exists a $T_1 > 0$ such that $\sum_{j=1}^{j_0-1} B_j(t) = \sum_{j=1}^{j_0-1} N_j$ for all $t \geq T_1$. Thus, (EC.24) holds. According to the definition of j_0 in (35), there must be $\lambda - \sum_{j=1}^{j_0-1} \mu_j N_j \leq \mu_{j_0} N_{j_0}$. It then follows from (EC.2) that for all $t \geq T_1$, $E_{j_0}'(t) = \lambda - \sum_{j=1}^{j_0-1} \mu_j N_j$. Then we can see from (14) and (16) that, for all $t \geq T_1$, $B_{j_0}'(t) = \lambda - \sum_{j=1}^{j_0-1} \mu_j N_j - \mu_{j_0} B_{j_0}(t)$, of which the solution is $B_{j_0}(t) = B_{j_0}(T_1)e^{-\mu_{j_0}(t-T_1)} + \mu_{j_0}^{-1}(\lambda - \sum_{j=1}^{j_0-1} \mu_j N_j)(1 - e^{-\mu_{j_0}(t-T_1)})$, $t \geq T_1$. It immediately follows that $\lim_{t\to\infty} B_{j_0}(t) = (\lambda - \sum_{j=1}^{j_0-1} \mu_j N_j)/\mu_{j_0}$. It follows from (EC.2) that $E_j'(t) = 0$ for all $j = j_0 + 1, \ldots, J, J+1$ and $t \geq T_1$. Similarly, we can see from (14), (16) and (18) that $\lim_{t\to\infty} B_j(t) = 0$ for all $j = j_0 + 1, \ldots, J$ and $\lim_{t\to\infty} Q(t) = 0$. This proves (35). We have therefore completed the proof. □

Proof of Theorem 4. It is clear that the nonlinear programming (24) is a concave optimization problem if the cost functions C_j's, $j = 1, \ldots, J, J+1$, are concave. Note that the constraint set is a convex set (actually it is a convex polytope). Then, it follows that the optimization problem admits a global minimum at an extreme point, i.e., at one of the vertices of this polytope. At a vertex we have that $0 < b_j < N_j$, $j = 1, \ldots, J$, and $q > 0$ for at most one b_j or q. Corresponding to any optimal vertex, we can define an optimal fixed priority order. Then this theorem immediately follows from (14), (22) and Proposition 1. □

EC.2.4. Proofs of the Optimality of the Stochastic Routing Policies

Proof of Theorem 5. In addition to the fluid limit proved in Theorem 1, we also show that (26) under the fluid routing policies π_{b^*,q^*}, π_G, π_{P^*} serves as the fluid limit of (37) under the stochastic routing policies $\pi^n_{b^*,q^*}$, π^n_G, $\pi^n_{P^*}$. By Lemma EC.1, let $E'_j(t) = (d/dt)E_j(t)$. For any fixed $j \in \{1,\ldots,J,J+1\}$, it suffices to prove that $E'_j(t) = 0$ if $\sum_{\{k=1,\ldots,J,J+1:P_k(t)<P_j(t)\}} I_k(t) > 0$, which gives (26). So assume that there exists $t > 0$ and $j \in \{1,\ldots,J,J+1\}$ such that $P_k(t) < P_j(t)$ and $I_k(t) > 0$. Due to the continuity of P_k and P_j (which are defined in (27), (31) and (32) for our proposed fluid policies π_{b^*,q^*}, π_G and π_{P^*}, respectively) and the continuity of I_k by Lemma EC.1, we can conclude that for n large enough $P^n_k(s) < P^n_j(s)$ and $\bar{I}^n_k(s) > 0$ for $|s-t| < \delta$ and some $\delta > 0$. According to the stochastic dynamic priority policy (36) (or equivalently (37)), $\bar{E}^n_j(t+\delta) - \bar{E}^n_j(t-\delta) = 0$, and therefore $E_j(t+\delta) - E_j(t-\delta) = 0$. This proves that (26) serves as the fluid limit of (37) under our proposed policies.

Now we start to prove (59). We first consider the target-allocation policy $\pi^n_{b^*,q^*}$. By Theorem 1 and the above discussion, for the sequence of the target-allocation policies $\{\pi^n_{b^*,q^*}\}$ we can always choose a convergent subsequence as the supremum. Using Skorohod representation theorem (see, for example, Lemma C.1 in Zhang (2013)) we can map all of the random objects to the same probability space so that all weak convergence becomes almost sure convergence. Thus, there is a fluid target-allocation policy π_{b^*,q^*} such that $\limsup_{n\to\infty} L^n_T(\pi^n_{b^*,q^*}) = L_T(\pi_{b^*,q^*})$ almost surely. It then follows from Theorem 2 that the second equation in (59) holds. The limit inferior in (59) follows for the same reason. The proof for the other two policies π^n_G and $\pi^n_{P^*}$ is exactly the same. □

EC.3. Proofs of the Optimality of the Hybrid Routing Policies

With regards to (43), the decisions on routing customers to the queue are the same for the three hybrid routing policies. Their decisions on routing customers to pools correspond to the three routing policies proposed in Section 3. Therefore, we can simultaneously prove Theorems 6, 7 and 8, and Proposition 2.

Proof of Theorems 6, 7 and 8, and Proposition 2. If $Q(t) < \lambda p/\theta$, then by (43) we have $\sum_{j=1}^J E'_j(t) = 0$. It then follows from (14) and (15) that $Q'(t) = \lambda - \theta Q(t) > 0$. Otherwise, if $Q(t) > \lambda p/\theta$, then due to the same reason as (EC.2) we have $\sum_{j=1}^J E'_j(t) = \lambda$ whenever $\sum_{j=1}^J I_j(t) > 0$ and $\sum_{j=1}^J E'_j(t) = \lambda \wedge \sum_{j=1}^J \mu_j N_j$ whenever $\sum_{j=1}^J I_j(t) = 0$. By (14), (15) and the fact that $\lambda(1-p) \le \sum_{j=1}^J \mu_j N_j$, there will be $Q'(t) = \lambda - \theta Q(t) - \sum_{j=1}^I E'_j(t) < 0$ in this case. Thus, we can conclude that $\lim_{t\to\infty} Q(t) = \lambda p/\theta$. Moreover, if there exists a $T_0 \ge 0$ such that $Q(T_0) = \lambda p/\theta$, then there will be $Q(t) = \lambda p/\theta$ for all $t \ge T_0$. Otherwise, if there is no such a T_0, there will be $\lim_{t\to\infty} Q'(t) = 0$ deducing from the previous two ordinary differential equations about $Q(t)$. Then by (14) and (15) there will be $\lim_{t\to\infty} \sum_{j=1}^J E'_j(t) = \lambda(1-p)$. Since the three routing policies in Section 3 and the other three

hybrid routing policies in Section 4 perform the same dynamics for customers to be serviced in the pools. Then the convergence of B_j's, $j=1,\ldots,J$, can be proved by applying the same argument as that of Theorems 2, 3 and 4, and Proposition 1, respectively. Thus, we omit it here for brevity. □

EC.4. Min-knapsack Problems

In this section, we show the connection between inverted-V models and knapsack problems. We show that the c/μ rule derived from (67) is identical to the Fractional Min-knapsack problem. We also introduce the Fractional 0-1 Min-knapsack Problem in (EC.26), which turns out to be consistent with the fixed priority routing problem in Section 3.1.3. Moreover, in Section EC.4.3 we propose a dynamic programming algorithm to solve it efficiently.

EC.4.1. Fractional Min-knapsack Problem

Let there be K items, indexed by $k=1,\ldots,K$, with price p_k and weight w_k for item k. The knapsack problem allows every item to be divided. The amount of item k that is packed in the knapsack will be denoted by y_k being a real number between 0 and w_k. The minimum weight that should be carried in the knapsack is W. More specifically, we wish to solve the following minimization problem:

$$\begin{aligned} \text{minimize} \quad & \sum_{k=1}^{K} \frac{p_k}{w_k} y_k \\ \text{subject to} \quad & \sum_{k=1}^{K} y_k \geq W, \\ & 0 \leq y_k \leq w_k,\ k=1,\ldots,K. \end{aligned} \tag{EC.25}$$

Because of its straightforward form, it admits an immediate algorithm: order the items according to their price-to-weight ratio, $\frac{p_1}{w_1} \leq \cdots \leq \frac{p_K}{w_K}$, then apply a greedy algorithm to pack as many low ratio items into the knapsack as possible. It can be easily seen that the form of the optimal solution is either 0 or w_k for each item, with at most one exception to choosing the fractional part of its weight. Now comparing the minimization problems (67) and (EC.25), there is no doubt that the c/μ rule is virtually a Fractional Min-knapsack Problem. We formally state it in the following proposition and omit its proof for brevity.

Proposition EC.1. *For linear operating cost functions, the c/μ rule problem* (66) *is identical to the Fractional Min-knapsack Problem* (EC.25).

EC.4.2. Fractional 0-1 Min-knapsack Problem

Instead of the linear objective function in (EC.25), we consider a nonlinear cost function $P_k(y_k)$ being the price of item k with weight y_k packed into the knapsack. Moreover, in addition to the K items with finite maximum weight w_k, $k=1,\dots,K$, we add one more item indexed by $K+1$ of which the maximum $w_{K+1}=+\infty$, showing that y_{K+1} can be any number in $[0,+\infty)$. For standardization, we set $P_k(0)=0$. Also $P_k(y_k)$ is assumed to be a nondecreasing function in y_k. Among all of the possible choices of $\{y_1,\dots,y_K,y_{K+1}\}$, we allow at most one item to be strictly between 0 and its maximum weight. Hence, the problem is extended to

$$\begin{aligned} \text{minimize} \quad & \sum_{k=1}^{K+1} P_k(y_k) \\ \text{subject to} \quad & \sum_{k=1}^{K+1} y_k \geq W, \\ & 0 \leq y_k \leq w_k,\ k=1,\dots,K, \\ & 0 \leq y_{K+1} \leq w_{K+1},\ \text{where } w_{K+1}=+\infty, \\ & 0 < y_k < w_k \text{ for at most one } k \in \{1,\dots,K+1\}. \end{aligned} \tag{EC.26}$$

We refer to the minimization problem (EC.26) as the *Fractional 0-1 Min-knapsack Problem* since it allows at most one item to be divided, as in the Fractional Min-knapsack Problem, and requires other items to be packed in their entirety or not packed at all, as in the classical 0-1 Knapsack Problem. Obviously, the last constraint can be eliminated when (EC.26) is a concave optimization problem. Now, it becomes clear that to find an optimal fixed priority order, it is essential to solve the Fractional 0-1 Min-knapsack Problem. Therefore, the proposition below immediately follows.

Proposition EC.2. *For general cost functions, the fixed priority routing problem in Section 3.1.3 is identical to the Fractional 0-1 Min-knapsack Problem* (EC.26)*. Moreover, the hybrid fixed priority routing problem in Section 4.1.3 is also identical to the Fractional 0-1 Min-knapsack Problem* (EC.26) *after setting $P_{K+1}(y_{K+1})=+\infty$ for all $y_{K+1}>0$.*

It is worth noting that two similar maximization knapsack problems have been studied in Section 5 of Long et al. (2020), showing the connection between V models and knapsack problems. Indeed, the Fractional Min-knapsack Problem (EC.25) can be simply transformed into the Fractional Knapsack Problem (27) in Long et al. (2020) after replacing y_k in (EC.25) by $w_k - x_k$. However, the Fractional 0-1 Min-knapsack Problem (EC.26) can not be transformed to the Fractional 0-1 Knapsack Problem (28) in Long et al. (2020) since the objective function of (EC.26) is nonlinear. Thus, we need to design a specific algorithm to solve (EC.26).

EC.4.3. Dynamic Programming Algorithm

In this subsection, we develop a dynamic programming (DP) algorithm to solve the Fractional 0-1 Min-knapsack Problem (EC.26) using a four-step procedure, which can be considered as the dual version of the one developed in Section EC.4 of Long et al. (2020).

Step 1: Decompose the problem into subproblems.

In view of (EC.26), any feasible solution contains at most one fractionally packed item. This suggests constructing a three-dimensional array $M[0..K+1, 0..W, 0..K+1]$, where the third dimension is used to track the fractionally packed item. For $1 \le k \le K+1$, $0 \le w \le W$ and $0 \le l \le K+1$, we consider the following two cases:

Case 1: $l = 0$. The entry $M[k, w, 0]$ stores the minimum cost of items packed in their entirety from any subset of items $\{1, 2, \ldots, k\}$ with a total weight of at least w. The component 0 in $M[k, w, 0]$ indicates that there is no fractionally packed item.

Case 2: $l > 0$. The entry $M[k, w, l]$ stores the minimum cost of the fractionally packed item l and the items packed in their entirety from any subset of items $\{1, 2, \ldots, k\} \setminus \{l\}$ with a total weight of at least w.

We also need the following initial setting for $k = 0$,

$$M[0, w, l] = \begin{cases} 0 & \text{if } l = 0 \text{ and } w = 0, \\ +\infty & \text{if } l = 0 \text{ and } w > 0, \\ P_l(w) & \text{if } l > 0 \text{ and } w_l > w > 0, \\ +\infty & \text{if } l > 0 \text{ and } w_l \le w, \\ +\infty & \text{if } l > 0 \text{ and } w = 0. \end{cases} \tag{EC.27}$$

For the case with weight limit $w < 0$, which is also illegal, we set

$$M[k, w, l] = +\infty, \quad \text{for all } w < 0 \text{ and } k, l \ge 0. \tag{EC.28}$$

Step 2: Recursively define the value of an optimal solution.

For $l = 0$, which means no item is fractionally packed, the optimal solution corresponding to $M[k, w, 0]$ is to either leave item k behind, in which case $M[k, w, 0] = M[k-1, w, 0]$, or to pack item k, in which case $M[k, w, 0] = V_k(w_k) + M[k-1, w-w_k, 0]$. Due to the penalty for a negative weight in (EC.28), we conclude that

$$M[k, w, 0] = \min\{M[k-1, w, 0], P_k(w_k) + M[k-1, w-w_k, 0]\} \tag{EC.29}$$

for all $1 \le k \le K+1$, $0 \le w \le W$.

For $l = 1, \ldots, K+1$, where item l is exactly the fractionally packed item, we can similarly derive

$$M[k, w, l] = \begin{cases} M[k-1, w, l] & \text{if } k = l, \\ \min\{M[k-1, w, l], P_k(w_k) + M[k-1, w-w_k, l]\} & \text{if } k \ne l \end{cases} \tag{EC.30}$$

for all $1 \le k \le K+1$, $0 \le w \le W$, where the first entry means that item k has been fractionally packed, and thus cannot also be packed in its entirety. The second entry relies on a similar explanation to that of (EC.29). Since this time item k is not the fractionally packed item, it can be either left behind or packed in the optimal solution corresponding to the minimum value $M[k,w,l]$.

We show in the proposition below that these recursions can indeed be described by a single recursive equation.

Proposition EC.3 (Recursive Equation). *The Fractional 0-1 Knapsack Problem* (EC.26) *can be solved using dynamic programming, namely for any $l \in \{0,1,\ldots,K+1\}$, we have the following recursive equation*

$$M[k,w,l] = \min\left\{M[k-1,w,l], P_k(w_k) + M[k-1,w-w_k,l] + \text{Inf}\mathbb{1}_{\{k=l\}}\right\}, \tag{EC.31}$$

holds for all $k \in \{1,\ldots,K+1\}$ and $w \in \{0,1,\ldots,W\}$, where $\text{Inf} = +\infty$.

Proof. From the condition of this proposition, only $k \ge 1$ should be considered and $k = 0$ for the boundary condition has been given in (EC.27). Thus, it is easy to see that the recursions (EC.29) and (EC.30) can be expressed as a unified equation (EC.31). To prove (EC.31), we first consider a possible case $k = l$, which implies that item k is the fractionally added item. Then $M[k,w,l] = M[k-1,w,l]$ since in this case item k cannot be wholly taken. It remains to prove the case $k \ne l$. To compute $M[k,w,l]$ we note that there are only two choices for item k. If we leave the whole item k, then limited by the minimum weight w the minimum cost with the wholly added items taken from $\{1,2,\cdots,k-1\}$ and the fractionally added item l is $M[k-1,w,l]$. Instead, if we take the whole item k (only possible if $w \ge w_k$), then we gain $V_k(w_k)$ immediately, but consume w_k weight of our storage. Now, the rest weight limit becomes $w - w_k$ and the minimum cost with the remaining items $\{1,2,\cdots,k-1\}$ is $M[k-1,w-w_k,l]$. In all, we obtain $V_k(w_k) + M[k-1,w-w_k,l]$. Note that if $w < w_k$, then $M[k-1,w-w_k,l] = +\infty$ from (EC.28). So the recursion (EC.31) holds in both cases. □

Step 3: Compute the value of an optimal solution.

For any fixed $l \in \{0,1\ldots,K+1\}$, the above recursive equation (EC.31) suggests a two-dimensional recursive equation. In all, there are $K+2$ independent recursive equations. To reach our goal, we only need to recursively calculate $K+2$ two-dimensional recursions for $k \in \{1,\ldots,K+1\}$ and $w \in \{0,1,\ldots,W\}$ based on the boundary conditions (EC.27) and (EC.28). Thus the running time of the dynamic programming algorithm is $O(K^2W)$. Finally, the optimal value of the Fractional 0-1 Min-knapsack Problem (EC.26) is obtained as follows:

$$\min \sum_{k=1}^{K+1} P_k(y_k) = \min_{l \in \{0,1,\ldots,K+1\}} M[K+1,W,l]. \tag{EC.32}$$

Step 4: Construct an optimal solution.

From (EC.32), we find that $Frac := \arg\min_{l\in\{0,1,\ldots,K+1\}} M[K+1,W,l]$ is the index of the fractionally packed item of the optimal solution. The only remaining problem is to obtain the indices of the items that are packed in their entirety. To that end, we need one auxiliary three-dimensional array $\mathcal{T}[0..K+1,0..W,0..K+1]$ to be a Boolean array to find their indices. Each entry $\mathcal{T}[k,w,l]$ records whether item k is packed in its entirety in realizing the smallest value $M[k,w,l]$. That is, $\mathcal{T}[k,w,l]=1$ if item k is packed in its entirety and $\mathcal{T}[k,w,l]=0$ otherwise. In the optimal solution, item $K+1$ is packed in its entirety if $\mathcal{T}[K+1,W,Frac]=1$. We can now repeat this argument for $\mathcal{T}[K,W-w_K,Frac]$. And item $K+1$ is not packed in its entirety if $\mathcal{T}[K+1,W,Frac]=0$. In this case, we can repeat the argument for $\mathcal{T}[K,W,Frac]$. Iterating the argument $K+1$ times from item $K+1$ downward to item 1 will give the indices of all items that are packed in their entirety. Thus far we have identified the optimal value and the solution to (EC.26).

References

Billingsley, P. (1968). *Convergence of probability measures*. Wiley Series in Probability and Statistics: Probability and Statistics. New York: John Wiley & Sons Inc.

Chen, H. and D. D. Yao (2001). *Fundamentals of queueing networks*, Volume 46 of *Applications of Mathematics (New York)*. New York: Springer-Verlag.

Dupuis, P. and R. S. Ellis (1997). *A weak convergence approach to the theory of large deviations*. Wiley Series in Probability and Statistics: Probability and Statistics. John Wiley & Sons, Inc., New York.

Long, Z., N. Shimkin, H. Zhang, and J. Zhang (2020). Dynamic scheduling of multiclass many-server queues with abandonment: The generalized $c\mu/h$ rule. *Operations Research 68*(4), 1218–1230.

Zhang, J. (2013). Fluid models of many-server queues with abandonment. *Queueing Systems 73*(2), 147–193.