Multiservice-based Network Slicing Orchestration with Impatient Tenants

Bin Han, *Member, IEEE,* Vincenzo Sciancalepore, *Senior Member, IEEE,* Xavier Costa-Pérez, *Senior Member, IEEE,* Di Feng, and Hans D. Schotten, *Member, IEEE*

***Abstract*—The combination of recent emerging technologies such as network function virtualization (NFV) and network programmability (SDN) gave birth to the novel Network Slicing paradigm. 5G networks consist of multi-tenant infrastructures capable of offering leased network "slices" to new customers (e.g., vertical industries) enabling a new telecom business model: Slice-as-a-Service (SlaaS). However, as the service demand gets increasingly dense, slice requests congestion may occur leading to undesired waiting periods. This may turn into impatient tenant behaviors that increase potential loss of the business attractiveness to customers. In this paper, we aim to *i*) study the slicing admission control problem by means of a multi-queuing system for heterogeneous tenant requests, *ii*) derive its statistical behavior model, *iii*) find out the rational strategy of impatient tenants waiting in queue-based slice admission control systems, *iv*) prove mathematically and empirically the benefits of allowing infrastructure providers to share its information with the upcoming tenants, and *v*) provide a utility model for network slices admission optimization. Our results analyze the capability of the proposed SlaaS system to be approximately Markovian and evaluate its performance as compared to a baseline solution.**

***Index Terms*—Beyond-5G, virtualization, network slicing, impatience, NFV, cloud service, resource management, multi-tenancy, multi-service**

I. Introduction

Network Slicing [1] is an emerging 5G technology that allows infrastructure providers to offer "slices" of resources (computational, storage and networking) to network tenants. In this way a new business game [2] is introduced as infrastructure providers (sellers) strategically decide which tenants (buyers) get granted slices to deliver their services. Intuitively, this involves a number of challenges that fall in the economic research field, which, in turn, requires a detailed understanding of the context. In particular, the infrastructure provider may rely on this emerging technology as a means to increase its revenue sources. However, to achieve the overall revenue maximization, advanced admission control policies are required as tenants compete for a limited set of available resources.

In this competing environment, a brokering solution may act as a mediator between seller and buyers while providing service level agreements (SLAs) guarantees to granted running slices [3]. Admission control policies will guide the broker in the process of deciding the set of network slices that can be installed on the system and the ones to be rejected. As the number of network slices grows—as envisioned for the next few years [4]—it will be necessary to design an automated solution that dynamically decides on the received slice requests while guaranteeing a certain degree of fairness among network tenants. Indeed, network slice requests may be queued while waiting for the next available resources, or may be re-issued.

To properly design such a *slicing brokering process*, a deep understanding of the slice queuing behavior is needed that accounts, e.g. the average slice duration (based on the slice type), the frequency of slice requests (based on the tenant), etc. This enables a Slice-as-a-Service (SlaaS) [5] solution that fully supports on-demand slices requests: tenants issue slice requests for given periods of time and decide whether to re-issue the same request upon rejection based on service level agreements. Advanced slicing admission control solutions may have different policies for tenants frequently asking for short-term slices—such as Internet-of-Things (IoT), or crowded event-based network slices—as they will automatically re-issue the same request in the near future, with respect to those that require only few longer network slices—such as Mobile Virtual Network Operators (MVNOs) or Industrial Network Slices [6]—which may be probably lost if not accepted. Moreover, similar as widely recognized in all kinds of queuing systems for service scheduling, tenants may be *impatient* and choose to leave for another available infrastructure provider instead of waiting in a queue, especially when the expected waiting time is long. This relies on the reasonable assumption that multiple network operators may offer similar slice services over the same spatial area opening up to new business models as per the SlaaS concept. A concrete example could be a stadium area covered by multiple telco operators, where a short-term crowded event may require a tenant to ask for a slice that, in turn, can be offered by different operators based on different offers. Such impatient behavior shall also be taken into account while designing a slicing admission control solution to mitigate potential revenues loss in case of resource congestion.

While conventional admission control problems have been extensively studied in the literature, as our main contribution in this study, we pioneer a new stochastic model for network slicing that leverages on the multi-queuing system to optimally design an admission control of on-demand network slices as well as to orchestrate them once accepted. This also allows to account for impatient tenant behaviors and heterogeneous network slice characteristics while, at the same time, enforcing

Bin Han and Hans D. Schotten are with Institute of Wireless Communication, Technische Universität Kaiserslautern, 67655 Kaiserslautern, Germany. Emails: {binhan, schotten}@eit.uni-kl.de

V. Sciancalepore and X. Costa-Pérez are with NEC Laboratories Europe, 69115 Heidelberg, Germany. Emails: {vincenzo.sciancalepore, xavier.costa}@neclab.eu

D. Feng is with Faculty of Business and Economics, Université de Lausanne, CH-1015 Lausanne, Switzerland. Email: di.feng@unil.ch

given performance metrics, such as fairness between tenants or between network slice types or utility-based maximization. We also numerically demonstrate this multi-queuing framework to outperform conventional single-queue solution.

The remainder of this article is structured as follows. In Section II, we introduce our assumptions and we formulate the network slicing admission control problem for on-demand slice request arrivals. In Section III, we provide a simple use case to cast our problem into a queuing system. In Section IV, we model the problem as a multi-queue problem where each queue may host slice requests of the same type while waiting for being granted. In Section V, we devise and analyze the multi-queuing controller with additional metrics by proving the capability of conventional queuing models on such system with impatient tenants taken into account, whereas in Section VI we further deepen our analysis on the rational impatient behavior of tenants. In Section VII we briefly introduce a scheme to optimize the MNO's admission control strategy. In Section VIII, we carry out an exhaustive simulation campaign to prove our findings and validate our model. In Section IX we discuss the applicability of some assumptions whereas in Section X we outline the main related works on this topic. Finally, in Section XI we provide concluding remarks.

II. Model design

We cast our problem into a typical network slicing scenario, where the Mobile Network Operator (MNO) decides to lease infrastructure resources to tenants, willing to pay to take over the control of an independent network slice so as to deliver an end-service to their own users. Hereafter, we describe our assumptions and mathematically formulate the problem.

A. Resource pool and slice types

Let us consider a single MNO that possesses a static resource pool of M different resources and offers $N = |\mathcal{N}|$ pre-defined types of slices. Depending on the slice type $n \in \mathcal{N}$, it costs a certain resource bundle to create and maintain a slice. Let $\mathbf{r} = [r_1, r_2, \dots, r_M]^{\mathrm{T}}$, $\mathbf{s} = [s_1, s_2, \dots, s_N]^{\mathrm{T}}$ and $\mathbf{c}_n = [c_{1,n}, c_{2,n}, \dots, c_{M,n}]^{\mathrm{T}}$ denote the resource pool, the set of active slices (i.e. slices created upon accepted requests), and the resource bundle required to maintain a slice of type $n \in \mathcal{N}$, respectively. The assigned resources can be then represented as

$$\mathbf{a} \triangleq [a_1, a_2, \dots, a_M]^{\mathrm{T}} = \mathbf{C} \times \mathbf{s}, \tag{1}$$

where $\mathbf{C} = [\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_N]$. At any time instance, the MNO cannot simultaneously maintain more slices than its resources may support. This constraint is expressed by the *feasibility region* [7]:

$$\mathbb{S} = \{\mathbf{s} | r_m - a_m \geq 0, \quad \forall 1 \leq m \leq M\}. \tag{2}$$

Note that $\mathbb{S}$ is a finite discrete set, thus the MNO can be characterized as a finite-state machine (FSM) where each active slice set represents the system state $\mathbf{s} \in \mathbb{S}$.

B. Slice admission in SlaaS

We consider a certain number of tenants randomly generating network slice requests. Slices requested by a certain tenant are of the same type. For each tenant, the inter-arrival time between two requests is drawn from an exponential distribution. The request arrivals of different tenants are independent and identically distributed (i.i.d.).

Once a request for slice creation is triggered, the MNO makes a binary decision, i.e., the MNO either accepts or declines it. Upon acceptance, the requested slice is created, and continuously maintained so that a corresponding resource bundle is occupied until the slice is terminated (at the end of its lifetime) and the resource bundle is released. It should be noted that the constraint of feasibility region forbids the MNO to accept any request when its current state is close to the border of $\mathbb{S}$. In other words, if the current MNO resource pool is close to be saturated by active slices, it does not accept additional network slice requests that might experience a service disruption. This introduces the well-known concept of *admissibility region*, where the idle resources are sufficient to allow to accept at least one new request[1]:

$$\mathbb{A} = \{\mathbf{s} | \mathbf{s} \in \mathbb{S}, \exists n : \mathbf{s} + \Delta \mathbf{s}_n \in \mathbb{S}\}, \tag{3}$$

where $\Delta \mathbf{s}_n$ is the *unit slice incremental vector* of type n

$$\Delta \mathbf{s}_n = [\underbrace{0, \dots, 0}_{n-1}, 1, \underbrace{0, \dots, 0}_{N-n}], \quad n \in \{1, 2, \dots, N\}. \tag{4}$$

Fig. 1 briefly illustrates the concepts of $\mathbb{S}$ and $\mathbb{A}$ with an example where $M = 2, N = 2$.

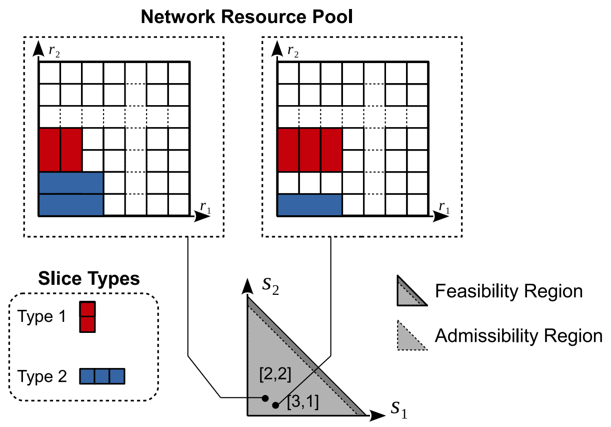


Fig. 1: The network resource utilization can be described with the set of active slices $\mathbf{s}$, which always falls in the feasibility region $\mathbb{S}$. The admissibility region $\mathbb{A}$ is a proper subset of $\mathbb{S}$.

We assume that the lifetime of every slice is an i.i.d. exponentially distributed variable and the expected lifetime depends on the slice type. We also consider that the MNO makes every decision according to a consistent slicing policy, i.e., the decision depends only on the type of requested slice n and the current system state $\mathbf{s}$ that defines the current set of active slices.

[1]The admissibility region has been exhaustively studied in the literature for different use cases and scenarios. We refer the reader to [8], where a stochastic admissibility region is derived for a network slicing admission control.

TABLE I: Key Notations

Notation	Meaning
$M, N, \mathcal{N}$	Amount of resource types, amount of slice types, set of slice types
$\mathbf{r}, \mathbf{s}, \mathbf{a}$	Resource pool, set of active slices, assigned resources
$\mathbf{C}, \mathbf{c}_n$	Resource cost matrix, resource cost of a type-n slice
$\mathbb{S}, \mathbb{A}$	Feasibility region, admissibility region
$\Delta\mathbf{s}_n$	unit slice incremental vector of type-n
$\mathbf{\Phi}$, Φ_i	MNO's preference matrix, MNO's preference vector in state $\mathbf{s}_{(i)}$
$l_n, L_n, \overline{W}_n$,	Current queue length, average length, and average waiting time in queue n
λ_n, μ_n, ρ_n	Request arrival rate, serving rate, and work load rate in queue n
b_n, β_n	Tenants' balking rate and balking exponent in queue n
$W_{\max,n}, \alpha_n$	Tenant's maximal waiting time and reneging exponent in queue n
$u_{0,n}, u_n$	Tenants' one-time cost to issue a request and periodical cost to wait in queue n
ζ_n, τ_n	A tenant's periodical profit from a type-n slice, the slice's random lifetime
η_n	Releasing rate of type-n slices
$\ni$	Maximal waiting cost w.r.t. expected achievable profit of a blind reneging tenant
ΔK	Minimal waiting time of a tenant only aware of its position in queue

C. Delayed reattempt upon request denial

If a request for slice creation is declined—because of a temporary shortage of available resources due to many other active slices—the tenant is not able to obtain the requested slice immediately. Instead, its request may be sent to the MNO again for a reconsideration after some delay with the hope that some running slice has expired (i.e., resources have been released). Generally, there are two critical features of the delaying mechanism, which should be taken into account: *i*) resource efficiency and *ii*) fairness. The former requires that the chosen mechanism purses the resource pool utilization maximization whereas the latter requires that the expected delay for different requests is normalized.

Two categories of approaches are commonly used to solve this kind of problem:

Random delay. Every declined request is re-proposed to the MNO after a random delay. This approach provides a good fairness, but generates extra signaling overhead while being not able to provide the discipline of "First Come, First Served" (FCFS), as described in the next section.

Queuing. Declined requests wait in one or multiple queue(s) for the next opportunity during the MNO's decisional process. This is the most common solution in cloud service scheduling.

Hereafter, we show how a multi-queuing system may be fully exploited to provide insights on the system behaviors and pave the road towards a slicing orchestration solution.

III. Network slicing queuing

In the literature a number of disciplines have been studied to serve the request queues. Among the others, the most common policies are *i*) First come, first served (FCFS), *ii*) Last come, first served (LCFS), *iii*) Random selection for service (RSS) and *iv*) Priority-based (PR). All of them analyze different behaviors and are used to achieve distinct performance metrics. For instance, the LCFS is used to allow the latest arriving request to override its awaiting predecessors, such as in information-freshness-critical scenarios [9]. The PR is implemented when there is some high-level preference to be considered in centralized scheduling [10]. RSS shows huge complexity in the implementation without bringing any significant advantage with respect to the others. Hereafter, we discuss different queuing schemes with focus on the FCFS case. However, any other discipline may be easily adapted to our analysis.

A. Queuing schemes

We differentiate the queuing systems into two different categories: *i*) single-queue and *ii*) multi-queue systems. When considering the single-queue, only one queue is implemented for all declined requests that need to wait for the next acceptance opportunity, an example was given in [11]. Conversely, the multi-queue system implements multiple queue for declined requests. Specifically, such queues may show different features. We consider homogeneous-mixed queues, wherein each queue consists of requests for slices of different types, and heterogeneous queues, where each queue is specified for only one unique slice type. We next show a simple case-study to justify that the queuing system is suitable for this kind of problems.

B. Resource efficiency: a simple case-study

Consider a simplified case where $M = 1$, $N = 2$, $\mathbf{r} = [1]$, $\mathbf{c}_1 = [0.6]$, $\mathbf{c}_2 = [0.2]$ and $\mathbf{s} = [1, 0]^{\mathrm{T}}$. The first four requests awaiting in the queue(s) are in the sequential order $[1, 1, 2, 2]$. The MNO takes a greedy strategy, i.e., it intends to accept all requests so far the resource pool supports.

In both the schemes of single-queue and homogeneous multi-queue, the awaiting requests of type 2 are blocked from acceptance due to the type-1 requests ahead of them, and therefore have to wait in queue, although the MNO has both enough idle resource and the intention to accept them immediately. The heterogeneous multi-queue scheme, in contrast, enables the MNO to fully utilize its resources as shown in Fig. 2.

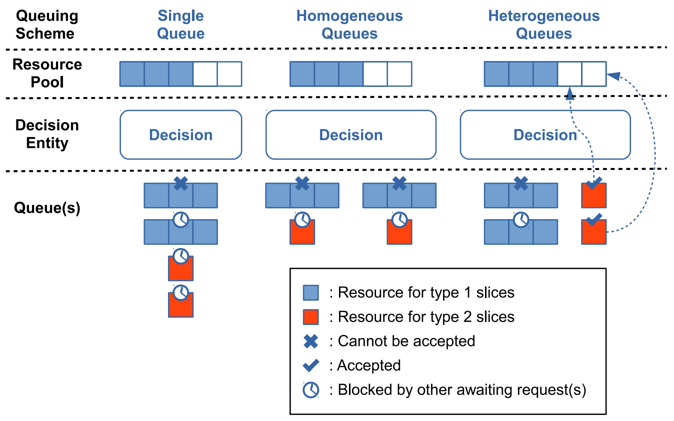


Fig. 2: A simple case study on different queuing schemes.

Obviously, both the single-queue and the homogeneous multi-queue schemes can also overcome this issue by introducing a "queue-jumping" mechanism. However, this may require an extra design of (more complex) logic that automatically (and dynamically) selects the queue jumper(s). Therefore, in this study we consider the scheme with N FCFS heterogeneous queues. Note that despite the intuitiveness and case-driven nature of this motivation, in Section VIII we demonstrate the performance superiority of this multi-queuing scheme through numerical simulations. Furthermore, as a well-studied extension of single-queue, the homogeneous multi-queue scheme is known to benefit only from its linearly increased serving rate in comparison to single-queue. Unfortunately, this gain does not apply in slice admission control where the request serving rates of different queues are jointly limited by the shared resource pool, while the implementation complexity is significantly higher than single-queue. Hence, in this study the performance of homogeneous multi-queue scheme will not be discussed.

IV. Heterogeneous multi-queue admission control

Based on the heterogeneous multi-queue scheme, we propose in this section a novel code to present the MNO's preference for different slice types in variable states, a multi-queue admission controller for SlaaS, and analyze its queue model.

A. Slice-type preference encoder

Differing from existing studies that do not consider queuing and the single-queue scheme, in the multi-queue scheme, the MNO may receive multiple requests for slices of different types within the same operations period (which is usually synchronized to the billing cycle, e.g. a month or longer). Therefore, instead of making a binary decision to accept or decline a request, it has to either accept one among the simultaneously awaiting requests while declining the rest, or decline all of them[2]. Especially, with heterogeneous queues, the MNO's preference for some request queue(s) over the others implies its proclivity to some slice type(s) against the others.

[2]Note that any simultaneous acceptance of multiple requests, which is technically applicable, can be decomposed into a sequence of atomic single-request acceptance operations. By considering only atomic acceptance operations, the action space is minimized.

It shall be noted that the decision of accepting a request from a certain queue requires the queue to be non-empty, which makes the MNO's decision dependent not only on the active slice set, but also on the status of queues. This extends the state space and increases the problem complexity. To simplify the analysis, we propose to investigate not the decision but the MNO's preference, which is decoupled from the queue status. However, it would be possible to uniquely determine the decision when any queue status is provided. Specifically, for an MNO offering N different slice types, we can encode an arbitrary preference of the MNO into a *preference vector* of length $N+1$:

$$\Phi = [\varphi_1, \varphi_2, \dots, \varphi_{N+1}]^{\mathrm{T}}, \tag{5}$$

which is a permutation of $\{0, 1, \dots, N\}$. Every element $\varphi_i > 0$ indicates a slice type and its position in Φ represents the MNO's preference. More specifically, $\forall i, j \in \{1, 2, \dots N+1\}$, $i < j$ denotes that MNO prefers slice type φ_i over φ_j. The zero-element denotes reserving resource for potential opportunities in future, so that all the requests of type φ_j will not be served by the MNO at all, if $j > i$ and $\varphi_i = 0$.

While being in states on (or close to) the border of feasibility region $\mathbf{s} \in \mathbb{S} - \mathbb{A}$, the MNO cannot accept further request from any queue, hence the preference does not make any impact. Thus, we focus on the admissibility region $\mathbb{A}$ and assume that the MNO's preference is consistent and depends only on its current state $\mathbf{s} \in \mathbb{A}$. Thus, we can characterize the MNO's admission strategy with a $(N+1) \times |\mathbb{A}|$ *preference matrix* as the following

$$\begin{aligned} \mathbf{\Phi} &= [\Phi_1, \Phi_2, \dots, \Phi_{|\mathbb{A}|}] \\ &= \begin{bmatrix} \phi_{1,1} & \phi_{1,2} & \cdots & \phi_{1,|\mathbb{A}|} \\ \phi_{2,1} & \phi_{2,2} & \cdots & \phi_{2,|\mathbb{A}|} \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{N+1,1} & \phi_{N+1,2} & \cdots & \phi_{N+1,|\mathbb{A}|} \end{bmatrix}, \end{aligned} \tag{6}$$

where each column Φ_i represents the preference for slice types in a specific admissible state $\mathbf{s}_{(i)} \in \mathbb{A}$, for which the index i is arbitrarily mapped by $\mathcal{I} : \mathbb{A} \xrightarrow{\mathcal{I}} \{1, 2, \dots |\mathbb{A}|\}$.

B. Mechanism overview

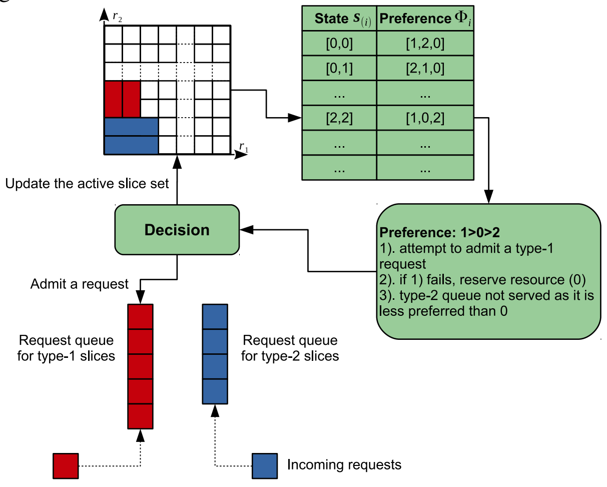
Let l_n denote the length of queue n, the decision entity executes the algorithm described in Fig. 3. The MNO keeps waiting for incoming tenant issues and responses upon issue arrivals. If the tenant issues to release a slice of its own, the MNO always releases it. If the tenant issues to create a new slice, the request will be pushed into a queue with respect to the type of requested slice. After responding to the issue, the MNO will recursively serve the request queues in a sequence determined by its admission strategy and active slice set, until no more waiting request can be accepted. Then it stops serving the queues and waits for the next tenant issue.

V. Network slicing controller design

Hereafter we analyze different characteristics of the conventional queuing models, highlighting the novel features applied

```
Initialize with certain $N$, $\mathbb{S}$, $\mathbb{A}$, $\mathbf{\Phi}$ and $\mathbf{s}$;
while True do                                                    Main loop
    Wait for the next incoming tenant issue;
    if Slice of type $n$ released then                           Releasing a slice
        $\mathbf{s} \leftarrow \mathbf{s} - \Delta\mathbf{s}_n$;
    else if Slice of type $n$ requested then                     Request arrives
        $l_n \leftarrow l_n + 1$;
    end
    while $\mathbf{s} \in \mathbb{A}$ do      Recursively serving the queues until blocked
        $\tilde{\mathbf{s}} \leftarrow \mathbf{s}$;
        Find the current preference vector $\Phi$ according to $\mathbf{\Phi}$ and $\mathbf{s}$;
        for $1 \leq n \leq N$ do                                 Serve queues w.r.t. preference
            if $\varphi_n = 0$ then                              Omitting queues after 0
                break;
            else if $l_n > 0$ AND $(\mathbf{s} + \Delta\mathbf{s}_n) \in \mathbb{S}$ then     Acceptance
                $l_n \leftarrow l_n - 1$;
                $\mathbf{s} \leftarrow \mathbf{s} + \Delta\mathbf{s}_n$;
            end
        end
        if $\tilde{\mathbf{s}} = \mathbf{s}$ then                Blockage detection
            Break;
        end
    end
end
```

(a) Pseudo code of the multi-queue slice admission controlling algorithm.



r_2
r_1
State $s_{(i)}$
Preference Φ_i
[0,0]
[1,2,0]
[0,1]
[2,1,0]
...
[2,2]
[1,0,2]
Update the active slice set
Decision
Preference: 1>0>2
1). attempt to admit a type-1 request
2). if 1) fails, reserve resource (0)
3). type-2 queue not served as it is less preferred than 0
Admit a request
Request queue for type-1 slices
Request queue for type-2 slices
Incoming requests

(b) Graphical illustration.

Fig. 3: A heterogeneous multi-queue mechanism can be implemented to enable delayed slice admission, in which a specific preference among different slice types is addressed to every state of active slice set.

to our model while designing the network slicing controller. This helps to shed light on the main advantages and limitations of our novel admission control model.

A. Analysis of inter-acceptance time

As per existing works [7], [8], [12], we consider request arrivals of every slice type as an independent Poisson process, so that the inter-arrival time between requests in every queue is an independent exponential random process. Conversely, the request acceptance rate of every queue is jointly determined by the slice releases of all types, and the MNO's preference strategy.

Remark 1. *Consider a heterogeneous multi-queue slice admission controller that executes the algorithm in Fig. 3(a) with a consistent preference matrix. The acceptance in different queues are mutually independent Poisson processes, if: 1) the arrivals of new requests and releases of active slices are mutually independent Poisson processes for every individual slice type; 2) the arrivals of different slice types are mutually independent from each other, the releases of different slice types are mutually independent from each other.*[3]

B. Queuing-theoretic analysis

While considering both request arrivals and request acceptances (service) as Poisson processes, every request queue is a classic M/M/1 queuing system, known as single-server birth-death system [14]. Hence, many features of birth-death model can be directly applied.

1) Little's Formula: For slice type (queue) n, given its request arrival rate λ_n, according to the famous Little's formula [15] the mean length of queue n is

$$L_n = \lambda_n \overline{W}_n, \tag{7}$$

where $\overline{W}_n$ represents the average waiting time in queue n.

2) Steady Queue State Probability: Given the request arrival rate λ_n and serving rate μ_n of queue n, the probability that the queue steadily consists of l requests at an arbitrary time instant is geometrically distributed, i.e.,

$$p_n(l) = (1-\rho)\rho^l, \tag{8}$$

where $\rho_n = \lambda_n/\mu_n < 1$ is the *work load rate* of queue n.

3) Waiting Time Distribution: The probability density function (PDF) and the cumulative density function (CDF) of an arbitrary type-n request's waiting time are

$$f(W_n) = \begin{cases} 0 & W_n < 0 \\ (\mu_n - \lambda_n)e^{-(\mu_n-\lambda_n)W_n} & W_n \geq 0 \end{cases}, \tag{9}$$

$$F(W_n) = \begin{cases} 0 & W_n < 0 \\ 1 - e^{-(\mu_n-\lambda_n)W_n} & W_n \geq 0 \end{cases}. \tag{10}$$

C. Extension: impatient tenants

From Eqs. (7–10) it is clear that both L_n and W_n converge only when $\lambda_n < \mu_n$. Otherwise, when the request acceptance rate is below the arrival rate in queue n, the queue length will infinitely increase, and therefore also the mean waiting time. This is known as the necessary and sufficient condition of statistical equilibrium in queuing processes, as proven in [16].

However, in a real slice admission controller, there are various situations where $\lambda_n \geq \mu_n$ for some n, including cases

- when the controller is specified with an inappropriate strategy, so that requests in the queue n is rarely or even never accepted despite of resource feasibility;
- when the release rates of active slices are low, so that the resource pool fails to support a sufficiently high μ_n regardless of any admission strategy.

There are two mechanisms that prevent queuing systems from such divergence. On the one hand, the system may force to truncate a queue at some maximal length, and forbid this queue to take any new request before it is shortened. On the other hand, the clients may lose patience while waiting, and leave the queues before being served (e.g., for looking for

[3]We refer the readers to [13] for a detailed proof of Remark 1.

some other MNO with resource availability). In the scenario of SlaaS, the system (MNO) is probably very cautious with refusing requests, while the waiting time can be critical to the customers (tenants). Therefore, here we consider no queue truncation but queues with impatience.

Usually, impatience in queues can occur in three different behaviors: *i*) balking, i.e. customers being reluctant to join a queue upon arrival, *ii*) reneging, i.e. customers leaving the queue after joining and waiting, and *iii*) jockeying from long lines to shorter ones. As the heterogeneous multi-queue design disables jockeying, here we consider the balking and reneging phenomena.

Balking Model. The phenomenon of balking can be modeled in such a way, that every arrival request of slice type n enters the queue with a probability b_n, which is a monotonically decreasing function of the current queue length l_n. *Ancker* and *Gafarian* have proposed two different balking models in [17], [18]. The first model considers a linear balking factor:

$$1 - b_n = l_n / l_{n,\max}, \tag{11}$$

where $l_{n,\max}$ is the truncation length of queue n. The second considers a non-linear balking factor:

$$1 - b_n = \begin{cases} 0 & l_n = 0; \\ 1 - \beta_n / l_n & l_n \in \mathbb{N}^+, \end{cases} \tag{12}$$

where $\beta_n \in [0, 1]$ measures the willingness of tenants requesting type-n slices to wait. In cases that the tenant has knowledge about μ_n, *Shortle* et al. suggest another non-linear balking model [14]:

$$1 - b_n = 1 - e^{-\beta_n l_n / \mu_n}, \quad \beta_n > 0. \tag{13}$$

Reneging Model. The phenomenon of reneging can be modeled by randomly assigning an individual maximal waiting time to every request when it joins the queue. The request will leave the queue after that maximal waiting time if it has not been accepted yet. *Ancker* and *Gafarian* [18] proposed to consider exponentially distributed random maximal waiting time:

$$W_{\max,n} \sim \mathrm{Exp}(\alpha_n), \tag{14}$$

where $1/\alpha_n > 0$ is the mean maximal waiting time in queue n.

Here we consider the exponential balking and reneging models described by Eq. (13) and Eq. (14), respectively. We will justify this choice later in Section VI. Before that, we first continue analyzing the performance of out heterogeneous multi-queue slice admission controller, taking into account the impatience of tenants.

D. Performances with balking and reneging

It should be noted that the balking and reneging processes are with memory, leading to a non-Markovian behavior of request acceptances. However, under low balking and reneging rates, this impact can be negligible and the acceptance process can still be approximated as Poissonian. When the balking and reneging rates rise to significant levels, the memory of acceptance process shall be considered, as demonstrated in Section VIII-B1 by means of simulations.

Under a combination of exponential balking and exponential reneging, the steady state probability of having l requests in the queue n is

$$p_n(l) = \begin{cases} \left(1 + \sum_{j=1}^{+\infty} \frac{\Gamma(\gamma_n+1)(\frac{\lambda_n \delta_n^j}{\alpha_n})^l}{\Gamma(l+\gamma_n+1)}\right)^{-1} & l = 0 \\ p_n(0) \prod_{j=1}^{l} \frac{\lambda_n \delta_n^j}{\mu + j\alpha_n} & l \in \mathbb{N}^+ \end{cases}, \tag{15}$$

where $\gamma_n = \mu_n/\alpha_n$, $\delta_n = e^{-\beta_n/\mu_n}$ and $\Gamma(\cdot)$ is the Gamma function. A detailed calculation is provided in the appendix.

Meanwhile, we are interested in three different distributions of waiting time spent in a queue n: *i*) $f_\mathrm{a}(W_n)$ for requests that are eventually accepted, *ii*) $f_\mathrm{r}(W_n)$ for requests that renege and *iii*) $f_\mathrm{q}(W_n)$ for all requests that join the queue. Let us define A_n and J_n as the events of request being accepted and joining the queue n, respectively. Following the analysis approach used in [17], [18], there are

$$P(J_n) = \sum_{j=1}^{+\infty} p_n(j)\delta_n^j \tag{16}$$

$$P(A_n) = p_n(0) + \sum_{j=1}^{+\infty} p_n(j)\delta_n^j \gamma_n/(\gamma_n + j), \tag{17}$$

$$P(A_n, J_n) = \sum_{j=1}^{+\infty} p_n(j)\delta_n^j \gamma_n/(\gamma_n + j), \tag{18}$$

$$P(A_n|J_n) = P(A_n, J_n)/P(J_n). \tag{19}$$

It can be obtained that

$$f_\mathrm{a}(W_n) = \frac{p_n(0)\alpha_n e^{-(\mu_n+\alpha_n)W_n}}{P(A_n, J_n)} \sum_{l=1}^{+\infty} \frac{\left(1 - e^{-\alpha_n W_n}\right)^l \delta_n^{\frac{l(l+1)}{2}}}{l!(l-1)!} \tag{20}$$

$$f_\mathrm{r}(W_n) = \alpha_n e^{-\alpha_n W_n} \frac{1 - P(A_n|J_n)g(W_n)}{1 - P(A_n|J_n)}, \tag{21}$$

$$f_\mathrm{q}(W_n) = P(A_n|J_n)\left[f_\mathrm{a}(W_n) - \alpha_n e^{-\alpha_n W_n} g(W_n)\right] + \alpha_n e^{-\alpha_n W_n}, \tag{22}$$

where $g(W_n) = \int_0^{W_n} e^{\alpha_n \xi} f_\mathrm{a}(\xi)\mathrm{d}\xi$.

The expectations of waiting times are therefore

$$\overline{W}_{\mathrm{a},n} = \int_0^{+\infty} f_\mathrm{a}(W_n)\mathrm{d}W_n, \tag{23}$$

$$\overline{W}_{\mathrm{r},n} = 1/\alpha_n - P(A_n|J_n)\overline{W}_{\mathrm{a},n}/[1 - P(A_n|J_n)], \tag{24}$$

$$\overline{W}_{\mathrm{q},n} = [1 - P(A_n|J_n)]/\alpha_n. \tag{25}$$

VI. Decision analysis of impatient tenants

To decide which models shall be used to describe the behavior of impatient tenants in the SlaaS scenario, we take the tenant's point of view and consider the business model of every individual tenant service instance through its life cycle. A rational strategy of impatience shall be obtained from the perspective of decision making by tenants.

A. Tenant business model

Generally, the motivation of tenants to request network slices is to fulfill the end-user service demands from their own customers. For simplification, here we consider w.l.o.g. that for every certain tenant, every service demand can be supported by one slice of the same type. Once a tenant is granted with the requested network slice, it launches its business session to deliver service to the end-users. The duration of a business session, i.e. the lifetime of the corresponding network slice, is a random variable, of which the expectation can be estimated by the tenant before issuing the slice request.

It can take the tenant a one-time cost u_0 to issue the service request to the MNO, which is used to issue the request and prepare the end-user service. Besides, this cost may also covers part (or even the whole lump) of the rent for requested network slice, which the MNO may also requires the tenants to prepay as deposit in advance. Additionally, when a request waits in queue, it can consistently generate a periodical waiting cost u for the tenant, which is used to keep the tenant standby for launching the business session. The value of u_0 and u shall be carefully tuned by the MNO with respect to its network slice service capacity, selection policies and the tenants' demand, which might be mapped onto an optimal pricing problem, out of scope of this study.

Upon an admission, a new network slice will be created and granted to the corresponding tenant to support the desired end-user service. This service is supposed to generate a periodic revenue *Rev* that we assume as known or well predictable by the tenant. Meanwhile, the tenant has to pay a periodical expenditure *Exp* that is composed by the operations cost to maintain the service, and the residential rent for the network slice in case that the rent is not completely prepaid to the MNO in the request-issuing phase. We can assume w.l.o.g. that the periodical profit $\zeta = Rev - Exp$ is always positive – as the tenant will never issue such a request otherwise.

B. Rational balking & reneging strategies

A request can be characterized by a vector $[n, u_0, u, \zeta, \tau]$, where n is the slice type, τ is the expected slice life time upon acceptance, u_0, u and ζ are the business model parameters discussed in Section VI-A. Meanwhile, a queue can be characterized by $[l, \lambda, \mu]$ where l is its current queue length, λ and μ are the arrival rate and serving rate, respectively.

When the business demand arises, i.e. request is generated (not issued yet) by the tenant, the expected total profit that this business session can generate is

$$\mathrm{E}\{\mathrm{Profit}\} = \zeta\tau. \tag{26}$$

Meanwhile, the expected cost of issuing the request and waiting in the queue till acceptance is

$$\mathrm{E}\{\mathrm{Cost}\} = u_0 + u\mathrm{E}\{w_l\}, \tag{27}$$

where w_l is the time a request must wait in queue until being accepted, when there are $l-1$ other requests ahead of it.

Assume that the tenants can obtain the a priori knowledge about w_l, self-evidently, a rational tenant will issue the request if and only if $\mathrm{E}\{\mathrm{Profit} - \mathrm{Cost}\} \geq 0$, which can be described as a binary decision model:

$$D_\mathrm{b} = \begin{cases} 1 & \zeta\tau - u_0 - u\mathrm{E}\{w_l\} \geq 0; \\ 0 & \text{otherwise,} \end{cases} \tag{28}$$

where $D_\mathrm{b} = 1$ stands for issuing and $D_\mathrm{b} = 0$ for balking.

The rational strategy for reneging mostly follows the same principle, but slightly differs from the balking case by concentrating on future cost and neglecting the sunken costs, i.e. the issuing cost and the waiting cost already generated. For an arbitrary request at time t, denote with $k(t)$ its current position in queue, and with $w(k)$ its remaining waiting time at position k, the tenant is able to rationally choose whether to renege according to:

$$D_\mathrm{r}(t) = \begin{cases} 1 & \zeta\tau - u\mathrm{E}\{w_k(t)\} \geq 0; \\ 0 & \text{otherwise,} \end{cases} \tag{29}$$

where $D_\mathrm{r}(t) = 1$ indicates waiting and $D_\mathrm{r}(t) = 0$ for reneging.

C. Rational balking without reneging

For simplification, we first ignore reneging, so that

$$\mathrm{E}\{w_l\} = \frac{l}{\mu}, \tag{30}$$

$$D_\mathrm{b} = \begin{cases} 1 & \tau \geq \frac{u_0\mu + ul}{\mu\zeta} \\ 0 & \text{otherwise} \end{cases}. \tag{31}$$

Hence, given certain ζ, u_0 and u, it yields that $D_\mathrm{b} = D_\mathrm{b}(l, \tau)$ if the tenant is able to observe l before pushing its request into the queue. The balking chance of such a tenant is therefore a function of l under any certain distribution of τ:

$$\begin{aligned} b(l) = &\int_0^{+\infty} D_\mathrm{b}(l, t) f_\tau(t)\mathrm{d}t = \int_{\frac{u_0\mu + ul}{\mu\zeta}}^{+\infty} f_\tau(t)\mathrm{d}t \\ =&1 - F_\tau\left(\frac{u_0\mu + ul}{\mu\zeta}\right) \end{aligned} \tag{32}$$

Particularly, when $u_0 = 0$ and $\tau \sim \mathcal{U}(0, \tau_\mathrm{max})$:

$$b_\mathrm{uni}(l) = \begin{cases} 1 - \frac{ul}{\mu\zeta\tau_\mathrm{max}} & 0 \leq l \leq \frac{\mu\zeta\tau_\mathrm{max}}{u}; \\ 0 & \text{otherwise,} \end{cases} \tag{33}$$

which is the linear balking model in [17]. Note that there is an implicit limit for the queue length $l_\mathrm{max} = \frac{\mu\zeta\tau_\mathrm{max}}{u}$, even if no such limit is explicitly set by the MNO.

When $u_0 = 0$ and $f_\tau(t) = \frac{1}{(t+1)^2}$ (rational distribution):

$$b_\mathrm{rat}(l) = 1 - \left(-\frac{1}{t+1}\right)\Big|_{t=0}^{\frac{ul}{\mu\zeta}} = \frac{\mu\zeta}{ul + \mu\zeta} = \frac{\frac{\mu\zeta}{u}}{l + \frac{\mu\zeta}{u}}. \tag{34}$$

When $u_0 = 0$ and $\tau \sim \mathrm{Par}(1, 1)$:

$$b_\mathrm{par}(l) = \begin{cases} 1 & l = 0; \\ 1 - \left(-\frac{1}{t}\right)\Big|_{t=1}^{\frac{ul}{\mu\zeta}} = \frac{\frac{\mu\zeta}{u}}{l} & \text{otherwise,} \end{cases} \tag{35}$$

which is the hyperbolic balking model in [18] with the factor of patience $\beta = \frac{\mu\zeta}{u}$. Note that this model assumes every slice remains active for at least an unit time period.

When $u_0 = 0$ and $\tau \sim \mathrm{Exp}(\eta)$:

$$b_\mathrm{exp}(l) = 1 - \left(-e^{-\eta t}\right)\Big|_{t=0}^{\frac{ul}{\mu\zeta}} = e^{-\frac{\eta u}{\zeta}\frac{l}{\mu}}, \tag{36}$$

which is the exponential balking model in [14] with the exponent of impatience $\beta = \frac{\eta u}{\zeta} > 0$.

D. Rational reneging

Now we consider the reneging behavior. As we will derive below, the decision of reneging highly relies on the tenant's knowledge of the queue. So we discuss this problem separately in different cases.

1) Full knowledge: First, we assume that every tenant is not only able to observe the position k of its request in the queue in real time, but also informed by the MNO about $\mathrm{E}\{w_k\}$. In this case, Eq. (29) can be directly applied to determine the maximal waiting time $t_{\max}$:

$$\zeta\tau - u\mathrm{E}\{w_{k(t_{\max})}\} = 0 \Rightarrow \mathrm{E}\{w_k\} = \sum_{i=0}^{k-1} \frac{1}{\mu + \sum_{j=0}^{i}\omega_j}, \quad (37)$$

where $\omega_i \geq 0$ is the reneging rate at the queue position i for $i > 0$ and $\omega_0 = 0$. Therefore, this case has an equivalent form where the MNO informs the tenant that issues the k^{th} request in queue about μ and ω_i for all $i < k$. The values of μ and ω_i converge to stable levels in long term when the business scenario remains consistent, therefore we consider them here as constants that are known by the MNO.

2) Knowledge of serving rate: In this case, we assume that every tenant is informed by the MNO about μ, and keeps observing the position k of its request in the queue, but has no knowledge about ω_i. As a tenant generally lacks knowledge of requests issued by other tenants, i.e. the statistics of their parameters $[u_0, u, p, \tau]$. So no tenant is able to estimate the values of ω_i in this case, which disables the estimation of $\mathrm{E}\{w_k\}$ according to Eq. (37). However, knowing that $\omega_i \geq 0$ for all i, the tenants can make conservative estimations based on

$$\mathrm{E}\{w_k\} = \sum_{i=0}^{k-1} \frac{1}{\mu + \sum_{j=0}^{i}\omega_j} \geq \frac{k}{\mu}, \quad (38)$$

and therefore Eq. (29) becomes

$$D_{\mathrm{r}}(t) = \begin{cases} 1 & k(t) \leq \frac{\mu\zeta\tau}{u}; \\ 0 & \text{otherwise.} \end{cases} \quad (39)$$

3) Knowledge of position: In this case, every tenant is able to track its request's current position k in queue, but has no a priori knowledge about μ. Thus, the tenant has to estimate μ through an online learning process while waiting in queue:

$$\hat{\mu}(k) = \frac{l-k}{T_k}, \quad (40)$$

where T_k is the time that the request took to arrive the k^{th} position since its entrance to the queue. Thus, Eqs. (38) and (39) become respectively

$$\mathrm{E}\{\hat{w}_k\} = \sum_{i=0}^{k-1} \frac{1}{\hat{\mu}(k) + \sum_{j=0}^{i}\omega_j} \geq \frac{k}{\hat{\mu}(k)} = \frac{kT_k}{l-k}, \quad (41)$$

$$D_{\mathrm{r}}(t) = \begin{cases} 1 & k(t) \leq \frac{l\zeta\tau}{uT_{k(t)}+\zeta\tau}; \\ 0 & \text{otherwise.} \end{cases} \quad (42)$$

It has to be noted here that the estimation in Eq. (41) is only valid when $k < l$, and the estimation error ϵ_{μ}^{2} decreases as k increases. Therefore a threshold ΔK should be set whereas $\hat{\mu}(t)$ is estimated only when $l - k \geq \Delta K$. In summary:

$$D_r(t) = \begin{cases} 1 & k(t) < l - \Delta K, \\ 1 & l - \Delta K \leq k(t) \leq \frac{l(\zeta\tau}{uT_{k(t)}+\zeta\tau}, \\ 0 & \text{otherwise.} \end{cases} \quad (43)$$

4) Knowledge of average waiting time: In this case, all tenants are only informed about the average waiting time $\overline{w}$ since joining the queue till being served, in which the waiting time requests that balk and renege do not count. Meanwhile, the current position in queue k is unobservable for a request unless $k = 0$ (i.e. when the request gets served).

In this case, a request can only roughly consider all other requests ahead of it in queue as a batch. Since the service to every single request is a Poisson event, we approximately consider the complete service to this batch (of unknown length) as a Poisson event with arriving rate of $1/\overline{w}$. Thus, the reneging decision can be made as

$$D_{\mathrm{r}}(t) = \begin{cases} 1 & \zeta\tau - u\overline{w} \geq 0; \\ 0 & \text{otherwise.} \end{cases} \quad (44)$$

Note that Eq. (44) is independent of t or k so that it always returns the same decision.

5) Blindness: If the tenant possesses neither the position k of its waiting request in the queue, nor any knowledge about the dynamics of queue, it can only make a blind reneging, where a maximal waiting time is predetermined at the queue entrance. A straightforward solution is to set a maximal cost proportional to the total profit that can be generated by the requested slice upon admission: $u_{\max} = u_0 + ut_{\max} = \backepsilon\zeta\tau$, where $\backepsilon \geq 0$ is the factor of risking that indicates the tenant's intension of waiting in the queue. This yields that

$$D_{\mathrm{r}}(t) = \begin{cases} 1 & t < \frac{\backepsilon\zeta\tau - u_0}{u}, \\ 0 & \text{otherwise.} \end{cases} \quad (45)$$

It is worth to note that when $u_0 = 0$ and $\tau \sim \mathrm{Exp}(\eta)$, the blind reneging model becomes the classic reneging model [18] where the maximal waiting time is exponentially distributed. Furthermore, when $\backepsilon \longrightarrow +\infty$ the tenants will never renege and therefore become *patient*.

E. Balking with renaging

When tenants are able to renege on their requests and the issuing cost $u_0 = 0$, the balking behavior can be considered as a special case of reneging at the queue entrance ($t = 0, k = l$). Clearly, this implies to disable balking in the aforementioned cases VI-D3 and VI-D5 where no a priori estimation of $\mathrm{E}\{w\}$ is available for tenants.

VII. Slice admission strategy optimization

In slice admission control, there are various performance metrics that may include: the overall network utility, the admission rate, and the average request waiting time, as discussed below.

The network utility rate of a slice can be differently defined, such as the periodical payment u that the MNO receives from the tenant, or the generated network throughput, etc. It is common to consider the utility rate of a slice as determined by the slice type, and the overall network utility rate at any time instant t as the sum of utility rates of all active slices:

$$u_\Sigma(t) = \sum_{n=1}^{N} s_n(t)u_n, \tag{46}$$

where $s_n(t)$ is the number of type-n active slices at time t, and u_n is the utility rate of every type-n slice. In long term, the average overall network utility rate can be estimated from the acceptance and releasing rates of different slice types:

$$\overline{u}_\Sigma = \sum_{n=1}^{N} \frac{\mu_n u_n}{\eta_n}. \tag{47}$$

The average waiting time of all requests in queues is

$$\overline{W}_\mathrm{q} = \frac{\sum_{n=1}^{N} \overline{W}_{\mathrm{q},n} L_n}{\sum_{n=1}^{N} L_n}. \tag{48}$$

The overall admission rate is the following

$$\overline{P}(A) = \frac{\sum_{n=1}^{N} \lambda_n P(A_n)}{\sum_{n=1}^{N} \lambda_n}. \tag{49}$$

All three criteria are determined by the request behavior parameters $\alpha_n, \beta_n, \lambda_n$ and the acceptance rate μ_n. Given a certain combination of $[\alpha_n, \beta_n, \lambda_n, \eta_n]$, μ_n is uniquely determined by the MNO's strategy, i.e. by the preference matrix $\mathbf{\Phi}$. Hence, with consistent behaviors of request arrival and slice releasing, we can optimize either of them by selecting the best $\mathbf{\Phi}$.

A major challenge for analysis exists in the complex relation between the acceptance rates $[\mu_1, \mu_2, \dots, \mu_N]$ and the strategy $\mathbf{\Phi}$, as $\mathbf{\Phi}$ does not directly imply the MNO's action or statistics, but only its preference.

Nevertheless, if the steady-state probability of queue lengths $p_n(l)$, as defined in Eq. (8), is known or measurable for all $n \in \mathcal{N}$, we can estimate μ_n for all n with respect to $\mathbf{\Phi}$ and the initial state $\mathbf{s}_\mathrm{init}$ as follows.

First, define a bijection $\mathbb{S} \leftrightarrow \{1, 2, \dots, |\mathbb{S}|\}$ as $J = J_\mathbb{S}(\mathbf{s})$ where $J_\mathbb{S}(\mathbf{s}) = I_\mathbb{A}(\mathbf{s})$ for all $\mathbf{s} \in \mathbb{A}$. Then extend the definitions in Eqs. (4), (6) and (8) with

$$\Delta\mathbf{s}_0 = \underbrace{[0, 0, \dots, 0]}_{N}, \tag{50}$$

$$\tilde{\phi}_{i,j} = \begin{cases} 0 & j > |\mathbb{A}| \\ \phi_{i,j} & j \leq |\mathbb{A}| \end{cases}, \forall i \in \{1, 2, \dots, N+1\}, \tag{51}$$

$$p_0(0) = 0, \tag{52}$$

respectively. The probability of state transition from $\mathbf{s} \in \mathbb{S}$ to $\mathbf{s} + \Delta\mathbf{s}$ can be then calculated as

$$\mathrm{Prob}(\mathbf{s} \longrightarrow \mathbf{s} + \Delta\mathbf{s}_n) = \prod_{k=1}^{n-1} p_{\tilde{\phi}_{k,J}}(0)(1 - p_{\tilde{\phi}_{n,J}}(0)). \tag{53}$$

Thus, when the initial state $\mathbf{s}_\mathrm{init}$ is known, we can obtain the long-term probability distribution of system state $\mathbf{s}$ as

$$\mathrm{Prob}(\mathbf{s}_j \mid \mathbf{s}_\mathrm{init} = \mathbf{s}_i) = \lim_{K\to\infty} \frac{1}{K} \sum_{k=0}^{K} [\mathbf{\Psi}^k]_{i,j}, \tag{54}$$

where $\mathbf{\Psi}$ is the transition matrix where $\Psi_{i,j} = \mathrm{Prob}(\mathbf{s}_i \longrightarrow \mathbf{s}_j)$:

$$\mathbf{\Psi} = \begin{bmatrix} \Psi_{1,1} & \Psi_{1,2} & \dots & \Psi_{1,|\mathbb{S}|} \\ \Psi_{2,1} & \Psi_{2,2} & \dots & \Psi_{2,|\mathbb{S}|} \\ \vdots & \vdots & \ddots & \vdots \\ \Psi_{|\mathbb{S}|,1} & \Psi_{|\mathbb{S}|,2} & \dots & \Psi_{|\mathbb{S}|,|\mathbb{S}|} \end{bmatrix}. \tag{55}$$

More generally, if not the exact value but the probability distribution of the initial state is available as $P_\mathrm{init} = [p_\mathrm{init}(\mathbf{s}_1), p_\mathrm{init}(\mathbf{s}_2), \dots, p_\mathrm{init}(\mathbf{s}_{|\mathbb{S}|})]$, the long-term probability distribution $\mathbf{s}$ is the following

$$\mathrm{Prob}(\mathbf{s}_j \mid P_\mathrm{init}) = \lim_{K\to\infty} \frac{1}{K} \sum_{k=0}^{K} \sum_{i=1}^{|\mathbb{S}|} p_\mathrm{init}(\mathbf{s}_j)[\mathbf{\Psi}^k]_{i,j}. \tag{56}$$

We can obtain the expected active slice number $\overline{s}_n$ of every slice type n as a function of $\mathbf{\Psi}$ and thus, as a function of $\mathbf{\Phi}$. Now, recalling Eqs. (46–47) it yields that

$$\overline{s}_n = \frac{\mu_n}{\eta_n}, \tag{57}$$

and then we can write the following

$$\begin{aligned} \mu_n &= \frac{\overline{s}_n}{\eta_n} = \frac{\sum_{\mathbf{s}\in\mathbb{S}} \mathrm{Prob}(\mathbf{s} \mid P_\mathrm{init}) s_n}{\eta_n} \\ &= \frac{1}{\eta_n} \sum_{\mathbf{s}\in\mathbb{S}} \lim_{K\to\infty} \frac{1}{K} \sum_{k=0}^{K} \sum_{i=1}^{|\mathbb{S}|} p_\mathrm{init}(\mathbf{s}_j)[\mathbf{\Psi}^k]_{i,j}. \end{aligned} \tag{58}$$

Based on this analytical expression, we are able to optimize $[\mu_1, \mu_2, \dots, \mu_n]$ with respect to $\mathbf{\Phi}$. However, it is evident that Eq. (58) is non-convex w.r.t. $\mathbf{\Phi}$, which prohibits analytical solution of the global optimum. On the other hand, the overall domain size of $\mathbf{\Phi}$ is $2^{(N+1)|\mathbb{A}|}$, which can assume unaffordable high values for any realistic dimension of $|\mathbb{A}|$ in practical networks, making the exhaustive search impossible. This is an integer linear programming (ILP) problem that is proven to be NP-Hard. Advanced machine learning and heuristic search methods might be explored to solve it at an affordable computational load. Due to space constraints this is left as future work.

VIII. Numerical simulations

A. Simulating the decisions of impatient tenants

1) Simulation setup: In the numerical simulations we define a simplified scenario, where the MNO holds a normalized two-dimensional ($M = 2$) resource pool $\mathbf{r} = [1, 1]$ and $N = 2$ different slice types are defined and specified with the parameters listed in Table II. Note that we assume the lifetime of every type-n slice is an exponentially distributed random variable $\tau_n \sim \mathrm{Exp}(\eta_n)$, where $1/\eta_n$ is the average lifetime of type-n slices.

TABLE II: Specifications of the defined slice types (all parameters normalized to dimensionless quantity).

Slice type (n)	$\mathbf{u}_n$	λ_n	$1/\eta_n$	$u_{0,n}$	u_n	ζ_n
1	$[0.01, 0.05]$	6	5	0	1	8
2	$[0.05, 0.01]$	10	3	0	1.5	12

Under these specifications, the admissibility region $\mathbb{A}$ is composed of 341 different values of $\mathbf{s}$. For the sake of simplicity, we do not consider the option of "reserve" in the MNO's slicing strategy: in this way for every state $\mathbf{s}$ only two different preference vectors, i.e. $\varphi_1 = [1, 2, 0]$ and $\varphi_2 = [2, 1, 0]$, are available. Therefore, there are in total 2^{341} different slicing strategies applicable for the MNO. We randomly select 1000 from these valid slicing strategies, and with every MNO slicing strategy, we evaluate the rational balking/reneging strategies of impatient tenants with different information available. For every individual evaluation, we simulate the arrivals of tenant requests and the MNO's operations for 1000 periods. To mitigate divergences of queue lengths, especially for the case of patient tenants, an upper bound of length is set to 100 for every queue.

2) Evaluation results: During the simulation, we track the end-profit of every issued slice request defined as follows

$$\zeta_\mathrm{e} = \begin{cases} \zeta\tau - u_0 - uw & \text{accepted;} \\ -u_0 - uw & \text{reneged,} \end{cases} \tag{59}$$

where w is the total waiting time from queue entrance to admission/reneging. Then we evaluate the balking/reneging strategies of tenants with three different metrics:

- *Total profit*: the sum of end-profits obtained by all issued slice requests;
- *Mean profit*: the average end-profit obtained by all issued slice requests;
- *Profiting chance*: the ratio of slice requests that lead to positive end-profits in all issued requests.

The simulation results are listed in Table III.

It can be easily observed from the results that, given the knowledge about position of its request in queue and the queue's serving rate, a tenant has a high chance to make correct decisions of balking and reneging. Thereby it is able to mitigate most losses caused by excessive waiting in case of request congestion, and thus obtain a positive profit.

The information about reneging rates provides a further improvement in addition, but only by an insignificant degree. One reason of this phenomena could be that, after a rational balking with sufficient knowledge, the reneging rate of requests generally remains limited, and therefore it exhibits a little impact on the waiting time in queue.

In contrast, when provided with only insufficient information, tenants are likely to benefit less from their impatience. An impatient tenant knowing only the mean waiting time in queue can reasonably avoid most extreme long waitings by balking, and therefore has more chance to achieve a positive profit in comparison to patient tenants, yet significantly lower than the tenants with full knowledge. The knowledge about current position of request in queue alone fail to assist tenants with their decisions, resulting to a similar performance as that of the patient tenants – which is the bound provided by the artificial queue truncation. Unwise decisions are also made by the blind tenants that renege after predetermined waiting time, whose performance strongly depends on the patience factor $\ni$, and in the worse case even outperformed in profiting chance by patient tenants.

In summary, network slice tenants need information about queue dynamics from the MNO—at least the minimum information to enable balking—so that they can benefit from impatience in case of slice requests congestion.

3) Distribution of reneging time: In Section VI-C, we have analytically proven the applicability of various classical models of balking statistics in the slice admission control scenario upon different distributions of the slice lifetime τ. The distribution of reneging time in SAC, however, is relatively challenging to derive in such way.

To evaluate the applicability of existing reneging models, we execute additional numerical simulations. The environment is configured to the same specifications listed in Table II, and the tenants possess full knowledge of the queuing system. First we randomly generate 1000 different admission strategies, for every strategy we carry out 25 rounds of Monte-Carlo test, in each round the MNO operates 40 periods. Then we fix the MNO to a static admission strategy that $\varphi = [2, 1, 0]$ for all $\mathbf{s} \in \mathbb{A}$, and repeat this test 1000 times, also with 25 rounds per time and 40 operations periods per round. We observe the waiting time of all reneged requests and illustrate the obtained results in Fig. 4. It can be observed that the exponential distribution generally provides a satisfactory fit to the reneging time in most cases, which supports applying the classical model proposed in [18] to simplify queue models from the MNO's perspective. However, it shall be remarked that in case of strong congestion where a queue can be extremely long, e.g. Queue 1 in the fixed strategy test, the reneging time may become fat-tail distributed and no more exponential.

4) Impatience model selection: Certainly, when fed with the knowledge of the current active queues, tenants may be more encouraged to balk or renege from densely congested queues of slice requests, which in turn leads to a decreased number of awaiting slice requests. Nevertheless, it should be noted that the phenomena of balking and reneging are only significant when the queues are considerably long. In this case, the MNO's resources are already sufficiently utilized, and the utilization rate is hardly impacted by the impatience of tenants. On the other hand, if there is a lack of information about the queues, as demonstrated in Section VIII-A2, tenants can suffer from high probability of business loss. This will, self-evidently, suppress the tenants' interest for the MNO's slice service on long-term windows, leading to a consistent loss of customers from the MNO's perspective. In summary, we can argue that it is a *win-win option for the MNO to share full knowledge of the queues*, or at least the request's current position in queue and the serving rate of queue, to every awaiting tenant. In such context, we assert it to be reasonable and rational to apply the models of exponential balking and reneging, as we did in Section V-D.

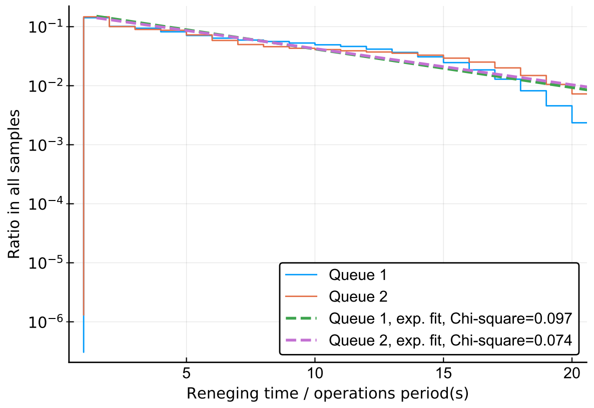
B. Simulating the heterogeneous multi-queue slice controller

To carry out simulations in a consistently specified environment, we consider the same scenario as defined in Table II.

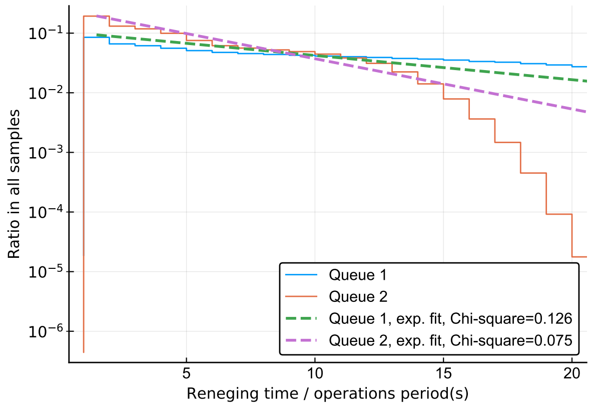
1) Verification of geometric IAT distribution: In case of patient tenants, Remark 1 can also be verified through numerical simulations. We consider all tenants as patient, and set an upper bound of queue length to 100 for all queues to mitigate queue divergence. Then we randomly generate

TABLE III: Tenant profits in 1000 operation periods under different balking/reneging strategies, "Patience" is the benchmark strategy where no balking or reneging takes place.

Case		**Total profit** ($\times 10^3$)		**Mean profit**		**Profiting chance**	
		Type 1	Type 2	Type 1	Type 2	Type 1	Type 2
Patience		36.06	16.88	10.74	2.86	49.21%	40.48%
Blindness	$\ni= 1$	33.62	24.08	7.93	3.54	46.27%	43.47%
	$\ni= 0.1$	110.89	139.23	18.45	13.90	25.42%	22.93%
	$\ni= 0.01$	129.45	153.98	21.53	15.40	46.29%	43.39%
Knowledge of position ($\Delta K = 2$)		36.03	16.86	10.72	2.79	49.23%	40.47%
Knowledge of average waiting time		50.88	60.11	32.97	25.36	85.65%	79.18%
Knowledge of serving rate		93.12	100.07	57.17	43.25	94.80%	83.59%
Full knowledge		92.68	101.53	57.66	44.34	95.57%	83.87%



(a) Random admission strategy



(b) Fixed admission strategy (prefer type 2)

Fig. 4: The distribution of tenant requests' reneging time in multi-queue slice admission control with fitting results.

1000 slicing strategies, in which the resource reservation ($n = 0$) is always assigned with the least preference. For each strategy, 25 rounds of Monte-Carlo tests are executed. In each testing round, an MNO with a 2-queue slice admission controller is initialized to a random but fully resource-utilized state, and then operates under the consistent strategy for 40 operations periods. Then we investigate the distribution of inter-acceptance time (IAT) for each queue, and attempt to fit the measurements with geometric distributions, which is the discrete-time version of exponential distribution. Out of all the 25 000 tests, 99.948% IAT records were successfully fitted by a Maximum-Likelihood-Estimator (MLE). A sample result is shown in Fig. 5(a), where a good fitting performance can be observed.

To evaluate the fitting for every strategy we use the Kullback-Leibler divergence (KLD) [19]:

$$D_{\mathrm{KL}}(P_{\mathrm{IAT}} \mid \mathrm{Geom.}) = \sum_{k=0}^{\infty} p_{\mathrm{IAT}}(k) \log \frac{p_{\mathrm{IAT}}(k)}{(1-\hat{p})^k \hat{p}}, \qquad (60)$$

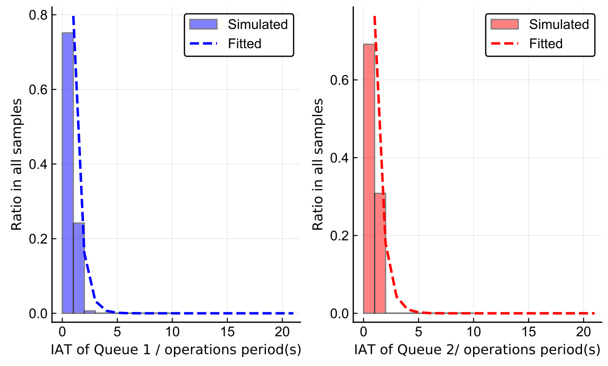
where $p_{\mathrm{IAT}}(k)$ is the empirical probability mess function (PMF) of the measured IAT, and $(1-\hat{p})^k\hat{p}$ is the geometric PMF with fitted parameter $\hat{p}$. KLD is an indicator of fitness between two distributions, which equals 0 for two identical distributions and approaches towards $+\infty$ for two completely irrelevant ones. The KLD distribution over all 25 000 tests is depicted in the left part of Fig. 5(b), which shows a satisfactory fitness for both queues (slice types).

Furthermore, to verify the impact of impatient tenants' behavior, we grant all tenants with full knowledge about the queues to activate balking and reneging, and then repeat the aforementioned simulation procedure. Only 20.568% of the measured IAT tracks can be successfully fitted with the geometric distribution this time (on the rest measurements, the MLE fails to converge). The KLD distribution of successfully fitted IAT tracks is illustrated in the right part of Fig. 5(b). Compared to the case of patient tenants, we can observe a significant increase of KLD here, confirming our assertion that the behaviors of balking and reneging will remove the Markovian feature of the system. Remark that when the balking and reneging rates are low, such impact can be slight enough to be neglected.

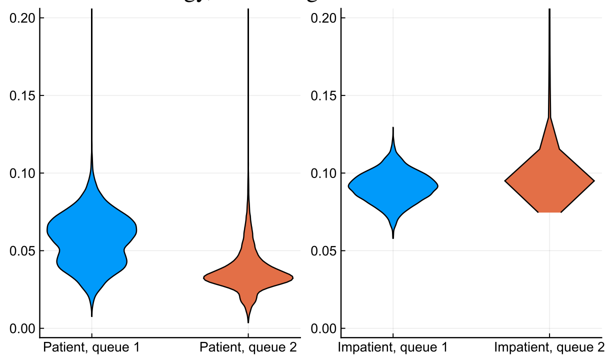
2) Evaluation of the proposed controller: To verify the effectiveness and potential in optimization of the proposed multi-queue slice admission controlling mechanism, we generate 10 000 random strategies, and measure all three above-mentioned performances metrics $\overline{u}_\Sigma$, $\overline{W}_\mathrm{q}$ and $\overline{P}(A)$ for every strategy in both reference scenarios 1 and 2. Similar to the last tests, every strategy is evaluated through a 25-round Monte-Carlo test where each round begins with a random initial state and lasts 40 operations periods. Impatient tenants are considered.

To provide benchmarks, we test the controller with two specific "naïve" strategies: *Prefer type 1*: the preference vector is $[1, 2, 0]$ at all system states; *Prefer type 2*: the preference vector is $[2, 1, 0]$ at all system states. Moreover, we implement and test a simple "greedy" single-queue slice admission controller that always accepts the first request in its queue regardless of type, as long as the resource pool supports.

The results are illustrated in Fig. 6. It can be observed



(a) The distribution of inter-acceptance time in two different queues under a random strategy, fitted as geometric distribution.



(b) The KLD of fitting IAT with geometric distribution, 1000 random strategies tested, 25 Monte Carlo tests for each. Fitting success rate is 99.948% for patient tenants and 20.568% for impatient ones.

Fig. 5: The IAT of individual queue under an arbitrary strategy is geometrically distributed *iff* tenants are patient.

that the multi-queuing controller, when specified with an appropriate strategy, outperforms the greedy single-queue solution, especially when under dense demand and queue are congestions. Note that, however, the performance highly relies on the strategy selection, leading to a critical necessity of strategy optimization.

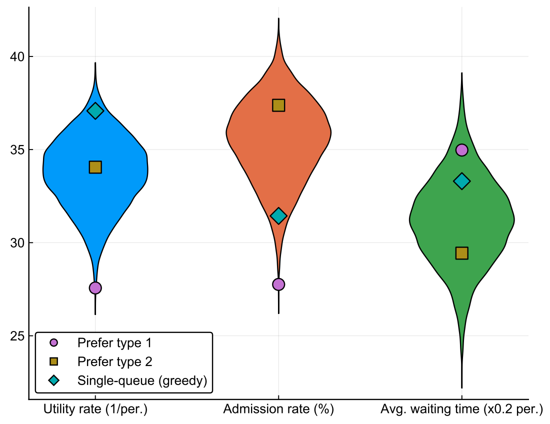


Fig. 6: Performance distribution of the proposed multi-queue slice admission controller with 10 000 random strategies, in comparison to selected benchmarks.

IX. Further discussion

In practical wireless networks, both the dynamics of resource availability (e.g. channel fading) and the resource elasticity of active slices must be taken into account. The model in this paper is an approximation with a static resource pool **r** and rigid slices, which holds in long-term with appropriate dynamic scheduling to multiplex slices. Note that such a slice multiplexing implicitly enables slice *overbooking* with a risk to break SLAs [20], [21]. The challenge of balancing the multiplexing gain and the overbooking risk in heterogeneous multi-queue admission control settings deserves future study.

It shall also be noticed that the assumptions of Poisson arrivals/releases may not hold in some practical service scenarios. In this case, the queues are not $M/M/1$ systems and cannot be considered as continuous-time Markov systems. Nevertheless, as pointed out in [14], many such continuous-time non-Markov processes can be easily transformed into discrete-time Markov chains by observing only the state transitions. Therefore, the analyses given above also apply to most scenarios with non-Poisson request arrivals/releases.

X. Related work

We summarize the main research effort in the literature on different topics, such as Slice-as-a-Service, queuing theory for cloud services and network slicing admission control.

An overview on multi-tenancy service and 5G network slicing is given in [3] from perspectives of architecture and standardization, introducing the novel concept of *network slice broker*, which executes the admission control. Different attempts have been made in [5], [8], [22] and [23] to demonstrate how the admission control can benefit the overall network resource utilization. In [24], a robust network slicing mechanism by addressing the slice recovery and reconfiguration in a unified framework Additionally, in [13] a multi-queuing system for heterogeneous tenant requests is modelled to derive statistical behavior models showing how this can be approximated to a Markovian system. However, none of the above-mentioned works have addressed the option of allowing infrastructure provider to share information with upcoming tenants so as to improve the overall system performance while, at the same time, formulating a network slice admission optimization problem based on a novel utility model.

While we have considered network slicing in a generic and abstracted view, which is generally applicable in both radio access network (RAN) and core network (CN) domains, recently there has been a dense specific research interest for RAN slicing and its impact on radio resource management (RRM). On that [25] and [26] provide interesting solutions for efficient resource management and orchestration. From the perspective of slicing admission strategy optimization, the methods reported in [5], [7], [8] can be worthwhile to refer. A dynamic resource controller for vRANs based on deep reinforcement learning is presented in [27], where the authors also showed a real implementation over different platforms. The authors of [28] introduces a novel framework for RAN slicing by showing performance requirements in terms of the required number of resources per deadline interval. Although all these works only consider a binary decision mechanism where declined requests simply vanish instead of being served after a delay, the algorithms deployed by them to solve

ILP problems will inspire future development of model-less heuristic strategy optimizers for the proposed multi-queue slice admission controller.

SlaaS shall be considered as a specific type of public cloud environment, where service sessions can be categorized into multiple types with significantly heterogeneous resource demands. Queuing theory has been widely applied for cloud computing services to model the statistics of service demand and delivered quality of service (QoS), such as [29] and [30]. Especially, service schedulers with heterogeneous queues for different service types are discussed in [31] and [32]. In addition, [33] relates to multi-resource sharing between flows with heterogeneous requirements providing a convergence proof. These models provide valuable reference views in addition to the model proposed in this paper. Finally, balking and reneging behavior of impatient clients in queuing systems are extensively studied in [34], [35].

Differing from the aforementioned works wherein a "strategy" usually represents the decision as a function of the system state, our study proposes a novel mechanism of multi-queuing slice admission control where the slicing strategy represents the MNO's preference of slice types in different system states. Besides, out paper also considers impatient tenants, which, from the best of our knowledge, has never been investigated in SlaaS environments.

XI. Conclusion

The network slicing paradigm is expected to play a key-role in next generation networks design. However, devising an admission control solution that takes into account complex network tenants behaviors involves a large number of challenges.

In this paper, we have proposed a multiservice-based network slicing controller that automatically accounts for tenants waiting to get their network slices request granted given certain request frequency and patience characteristics. Our results show that *i*) unexpected tenants behaviors may be modeled with advanced admission control policies, *ii*) the decisions of rational impatient tenants can be mapped onto classical queuing-theoretic models comprising balk and renege parameters and *iii*) numerical simulations closely follow the exponential balking/reneging models derived.

Appendix
The steady state probability of queue with exponential balking and reneging

Consider the queue of type-n requests where the request arriving rate is λ_n, the request serving rate is μ_n, the reneging factor is α_n and the balking factor is $1-e^{-\beta_n l/\mu_n}$. Let $p_n(l,t)$ denote the transient probability the queue contains l requests at the time instant t, we can write the transition equations of the dynamic queue state:

$$\frac{\partial p_n'(0,t)}{\partial t} = -\lambda_n p_n(0,t) + \mu_n p_n(1,t), \tag{61}$$

$$\begin{aligned}\frac{\partial p_n'(0,t)}{\partial t} =& -\left(\lambda_n e^{-\beta_n/\mu_n} + \mu_n\right) p_n(1,t) + \lambda_n p_n(0,t) \\ &+ (\mu_n+\alpha_n)p_n(2,t),\end{aligned} \tag{62}$$

$$\begin{aligned}\frac{\partial p_n'(0,t)}{\partial t} =& -\left(\lambda_n e^{-\beta_n l/\mu_n} + \mu_n + (l-1)\alpha_n\right) p_n(l,t) \\ &+ \lambda_n e^{-\beta_n (l-1)/\mu_n} p_n(l-1,t) \\ &+ (\mu_n + l\alpha_n)p_n(l+1,t), \quad l\in\{3,4,5,\dots\}.\end{aligned} \tag{63}$$

Let $\gamma_n = \mu_n/\alpha_n$, $\kappa_n(l) = \lambda_n e^{-\gamma_n l/\mu_n}$, the steady-state equations are therefore

$$0 = -\kappa_n(0)p_n(0) + \gamma_n p_n(1) \tag{64}$$

$$0 = -(\kappa_n(1)+\gamma_n)p_n(1) + \kappa_n(0)p_n(0) + (\gamma_n+1)p_n(2) \tag{65}$$

$$\begin{aligned}0 =& -(\kappa_n(l)+\gamma_n+l-1)\,p_n(l) + \kappa_n(l-1)p_n(l-1) \\ &+ (\gamma_n+l)p_n(l+1), l\in\{3,4,5,\dots\}\end{aligned} \tag{66}$$

From the steady-state equations we have

$$\begin{aligned}p_n(l) =& \frac{\kappa_n(l-1)}{\gamma_n+l-1}p_n(l-1) = p_n(0)\prod\nolimits_{i=1}^{l}\frac{\kappa_n(i)}{\gamma_n+i} \\ =& p_n(0)\frac{\lambda_n e^{-i\beta_n/\mu_n}}{\mu_n+i\alpha_n}, \qquad l\in\mathbb{N}^+.\end{aligned} \tag{67}$$

Knowing that $\sum_{l=0}^{+\infty} p_n(l) = 1$, $p_n(0)$ can be calculated as

$$p_n(0) = 1 \bigg/ \left(1 + \sum\nolimits_{l=1}^{+\infty}\prod\nolimits_{i=1}^{l}\frac{\lambda_n e^{-i\beta_n/\mu_n}}{\mu_n+i\alpha_n}\right). \tag{68}$$

Acknowledgments

The work of B. Han and H. D. Schotten was supported in part by the European Union Horizon-2020 Project 5G-MoNArch under Grant Agreement 761445, in part by the Network for the Promotion of Young Scientists (TU-Nachwuchsring), Technische Universität Kaiserslautern with individual funding. The work of V. Sciancalepore and X. Costa-Perez was supported by the European Union Horizon-2020 Project 5G-Transformer under Grant Agreement 761536.

References

[1] GSMA, "An introduction to network slicing," 2017.

[2] "5G network slicing for cross industry digitization: Position paper," https://www.fokus.fraunhofer.de/download.5G-Network-Slicing_whitepaper.pdf.

[3] K. Samdanis, X. Costa-Perez, and V. Sciancalepore, "From network sharing to multi-tenancy: The 5G network slice broker," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 32–39, 2016.

[4] C. Marquez, M. Gramaglia, M. Fiore et al., "How should I slice my network? A multi-service empirical evaluation of resource sharing efficiency," in *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking (Mobicom)*, 2018.

[5] V. Sciancalepore et al., "Slice as a service (SlaaS): Optimal IoT slice resources orchestration," in *IEEE Global Communications Conference (GLOBECOM)*, Dec 2017.

[6] A. E. Kalor, R. Guillaume, J. J. Nielsen, A. Mueller, and P. Popovski, "Network slicing in industry 4.0 applications: Abstraction methods and end-to-end analysis," *IEEE Transactions on Industrial Informatics*, 2018.

[7] B. Han, L. Ji, and H. D. Schotten, "Slice as an evolutionary service: Genetic optimization for inter-slice resource management in 5G networks," *IEEE Access*, vol. 6, no. 1, pp. 33 137–33 147, 2018.

[8] D. Bega, M. Gramaglia, A. Banchs, V. Sciancalepore, and X. Costa-Perez, "A machine learning approach to 5G infrastructure market optimization," *IEEE Transactions on Mobile Computing*, 2019.

[9] R. D. Yates and S. K. Kaul, "The age of information: Real-time status updating by multiple sources," *IEEE Transactions on Information Theory*, vol. 65, no. 3, pp. 1807–1827, 2019.

[10] H. Khojasteh, J. Mišić, and V. B. Mišić, "Prioritization of overflow tasks to improve performance of mobile cloud," *IEEE Transactions on Cloud Computing*, vol. 7, no. 1, pp. 287–297, 2019.

[11] B. Han, D. Feng, and H. D. Schotten, "A Markov model of slice admission control," *IEEE Networking Letters*, vol. 1, no. 1, pp. 2–5, March 2019.

[12] D. Bega, M. Gramaglia *et al.*, "Optimising 5G infrastructure markets: The business of network slicing," in *IEEE International Conference on Computer Communications (INFOCOM)*, May 2017.

[13] B. Han, V. Sciancalepore, D. Feng, X. Costa-Perez, and H. D. Schotten, "A utility-driven multi-queue admission control solution for network slicing," in *IEEE International Conference on Computer Communications (INFOCOM)*, April 2019.

[14] J. F. Shortle, J. M. Thompson, D. Gross, and C. M. Harris, *Fundamentals of Queueing Theory*. John Wiley& Sons, 2018.

[15] J. D. Little, "A proof for the queuing formula: $L = \lambda W$," *Operations Research*, vol. 9, no. 3, pp. 383–387, 1961.

[16] D. G. Kendall, "Some problems in the theory of queues," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 151–185, 1951.

[17] C. Ancker Jr and A. Gafarian, "Some queuing problems with balking and reneging–I," *Operations Research*, vol. 11, no. 1, pp. 88–100, 1963.

[18] ——, "Some queuing problems with balking and reneging–II," *Operations Research*, vol. 11, no. 6, pp. 928–937, 1963.

[19] S. Kullback, *Information Theory and Statistics*. Courier Corp., 1997.

[20] L. Zanzi, V. Sciancalepore et al., "OVNES: Demonstrating 5G network slicing overbooking on real deployments," in *IEEE Conference on Computer Communications Workshops (INFOCOM DEMO)*, April 2018.

[21] J. X. Salvat, L. Zanzi, et al., "Overbooking Network Slices through Yield-driven End-to-End Orchestration," in *ACM Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, December 2018.

[22] R. Addad, M. Bagaa, T. Taleb, D. L. Cadette Dutra, and H. Flinck, "Optimization model for Cross-Domain Network Slices in 5G Networks," *IEEE Transactions on Mobile Computing*, 2019.

[23] V. Sciancalepore, X. Costa-Perez, and A. Banchs, "RL-NSB: Reinforcement Learning-Based 5G Network Slice Broker," *IEEE/ACM Transactions on Networking*, vol. 27, no. 4, pp. 1543–1557, Aug 2019.

[24] R. Wen, G. Feng, J. Tang, T. Q. S. Quek, G. Wang, W. Tan, and S. Qin, "On robustness of network slicing for next-generation mobile networks," *IEEE Transactions on Communications*, vol. 67, no. 1, pp. 430–444, Jan 2019.

[25] O. Sallent, J. Perez-Romero, R. Ferrus, and R. Agusti, "On radio access network slicing from a radio resource management perspective," *IEEE Wireless Communications*, vol. 24, no. 5, pp. 166–174, 2017.

[26] P. L. Vo, M. N. Nguyen, T. A. Le, and N. H. Tran, "Slicing the edge: Resource allocation for RAN network slicing," *IEEE Wireless Communications Letters*, 2018.

[27] J. J. Ayala, A. Garcia-Saavedra, M. Gramaglia, X. Costa-Perez, A. Banchs, and J. J. Alcaraz, "vrain: A deep learning approach tailoring computing and radio resources in virtualized rans," in *Proceedings of the 25th Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2019.

[28] T. Guo and A. Suarez, "Enabling 5G RAN Slicing With EDF Slice Scheduling," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 3, pp. 2865–2877, March 2019.

[29] J. Vilaplana, F. Solsona, I. Teixidó, J. Mateo, F. Abella, and J. Rius, "A queuing theory model for cloud computing," *The Journal of Supercomputing*, vol. 69, no. 1, pp. 492–507, 2014.

[30] X. Chang, B. Wang, J. K. Muppala, and J. Liu, "Modeling active virtual machines on IaaS clouds using an M/G/m/m+ K queue," *IEEE Transactions on Services Computing*, vol. 9, no. 3, pp. 408–420, 2016.

[31] F. Li, J. Cao, X. Wang, and Y. Sun, "A QoS guaranteed technique for cloud applications based on software defined networking," *IEEE Access*, vol. 5, pp. 21 229–21 241, 2017.

[32] M. Guo, Q. Guan, and W. Ke, "Optimal scheduling of VMs in queueing cloud computing systems with a heterogeneous workload," *IEEE Access*, vol. 6, pp. 15 178–15 191, 2018.

[33] T. Bonald, J. Roberts, and C. Vitale, "Convergence to multi-resource fairness under end-to-end window control," in *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, May 2017.

[34] S. Bocquet, "Queueing theory with reneging," Defence Science and Technology Organisation, Australia, Tech. Rep., 2005.

[35] De-quan Yue and Yan-ping Sun, "Waiting time of M/M/c/N queuing system with balking, reneging, and multiple synchronous vacations of partial servers," *Systems Engineering-Theory & Practice*, vol. 28, no. 2, pp. 89–97, 2008.

Bin Han (M'15) received in 2009 his B.E. degree from Shanghai Jiao Tong University, M.Sc. in 2012 from Technische Universität Darmstadt, and in 2016 the Ph.D. (Dr.-Ing.) degree from Kalsruher Institut für Technologie. Since July 2016 he has been with Technische Universität Kaiserslautern, researching in the broad area of wireless networks and signal processing, with recent special focus on network slicing and timely information delivery.

Vincenzo Sciancalepore (S'11–M'15–SM'19) received his M.Sc. degree in Telecommunications Engineering and Telematics Engineering in 2011 and 2012, respectively, whereas in 2015, he received a double Ph.D. degree. Currently, he is a senior 5G researcher at NEC Laboratories Europe GmbH in Heidelberg, focusing his activity on network virtualization and network slicing challenges. He is currently involved in the IEEE Emerging Technologies Committee leading the initiatives on SDN and NFV. He was also the recipient of the national award for the best Ph.D. thesis in the area of communication technologies (Wireless and Networking) issued by GTTI in 2015.

Xavier Costa-Pérez (M'06–SM'18) is Head of 5G Networks R&D and Deputy General Manager of the Security & Networking Research Division at NEC Laboratories Europe. His team contributes to products roadmap evolution as well as to European Commission R&D collaborative projects and received several awards for successful technology transfers. He received both his M.Sc. and Ph.D. degrees in Telecommunications from the Polytechnic University of Catalonia (UPC) in Barcelona and was the recipient of a national award for his Ph.D. thesis.

Di Feng received the M.A. in Economics from Yokohama National University and M.Res. in Economics from Universitat Autònoma de Barcelona in 2016 and 2018, respectively. Currently, he is working for his Ph.D. degree in Economics at HEC - University of Lausanne. His research field covers decision theory, game theory and operations research, with a particular interest in market/mechanism design.

Hans D. Schotten (S'93–M'97) received the M.Sc. (Dipl.-Ing.) and Ph.D. (Dr.-Ing.) degrees in Electrical Engineering from the Aachen University of Technology RWTH, Germany in 1990 and 1997, respectively. Since August 2007, he has been full professor and head of the Institute of Wireless Communication at the University of Kaiserslautern. Since 2012, he has additionally been Scientific Director at the German Research Center for Artificial Intelligence heading the "Intelligent Networks" department.