A Markov Model of Slice Admission Control

Bin Han, *Member, IEEE*, Di Feng, and Hans D. Schotten, *Member, IEEE*

***Abstract*—The emerging feature of network slicing in future fifth generation networks calls for efficient slice management. Recent studies have been focusing on the mechanism of slice admission control, which functions in a manner of state machine. This letter proposes a general state model for synchronous slice admission control, and proves it to be Markovian under a set of weak constraints. An analytical approximation of the state transition matrix to reduce computational complexity in practical applications is also proposed and evaluated.**

***Index Terms*—5G, network slicing, network operations and management, network function virtualization.**

I. Introduction

NETWORK slicing [1] has been considered as an essential feature and one of the most important enablers of the Fifth Generation (5G) cellular communications networks. It allows mobile network operators (MNOs) to manage and utilize their physical and virtual network resources, i.e., the network infrastructure and the capacity of virtualized network functions (VNFs), in the form of logically independent virtual mobile networks, a.k.a. network slices. It provides broad improvements of scalability, flexibility, accountability, shareability and profitability to cellular networks [2], [3].

One emerging challenge brought by network slicing is to efficiently allocate network resources over different slices towards better utility efficiency. More specifically, there are two typical scenarios of such inter-slice resource management. First, when an MNO directly provides services to end users and maintains these services on its own scalable slices, as proposed in [4]. Second, in the resource-sharing use case of so-called *Slice as a Service* (SlaaS), which is discussed in [5], an MNO packs its resources into standardized atomic slices and rents them to external tenants such as virtual MNOs (VMNOs) and service providers for agreement-based periodical revenue. In both scenarios, the MNO aims to maximize the overall network utility rate (e.g., the revenue rate) by adjusting its resource allocation subject to the constraints of resource pool limit and regulation rules.

Manuscript received July 29, 2018; accepted August 30, 2018. Date of publication October 3, 2018; date of current version February 22, 2019. This work was supported in part by the European Union Horizon-2020 Project 5G-MoNArch under Grant 761445, and in part by the Network for the Promotion of Young Scientists, University of Kaiserslautern. The associate editor coordinating the review of this paper and approving it for publication was K. M. Sivalingam. *(Corresponding author: Bin Han.)*

B. Han and H. D. Schotten are with the Institute of Wireless Communication, Department of Electrical and Computer Engineering, University of Kaiserslautern, 67663 Kaiserslautern, Germany (e-mail: binhan@eit.uni-kl.de; schotten@eit.uni-kl.de).

D. Feng is with the Department of Economics and Economic History, Faculty of Economics and Business, Autonomous University of Barcelona, 08193 Bellaterra, Barcelona, Spain (e-mail: di.feng@e-campus.uab.cat).

Digital Object Identifier 10.1109/LNET.2018.2873978

Compared to the case of MNO's own service optimization, the SlaaS problem is more challenging due to the stochastic nature and fluctuating behavior of tenant demand for resources. Recently, multiple studies have been carried out on this topic [6]–[8], applying the similar framework where a binary decision is made by the MNO for every slice admission, i.e., to accept or decline the tenant request for a new slice. Most of these work consider the system as a state machine, where state transitions are triggered by the MNO's responses to randomly arriving tenant requests.

Despite the advantage of simple structure, such state-machine models suffer from two drawbacks. First, tenant requests by nature arrive in an asynchronous manner, leading to an asynchronous state model, while MNOs usually have synchronous frameworks for network controlling and management. Second, the models reported in literature only support simulative or exploitation-based evaluation of MNO's decision strategies. No analytical method of evaluation has been proposed.

In this letter, we focus on an unstudied feature of such state models of slice admission systems: when operated in a synchronous method, i.e., when the MNO makes decision to tenant requests periodically instead of immediately upon request arrivals, is the derived synchronous state-model Markovian? Answering this question will help us to 1) better understand the system behavior towards efficient deployments of advanced techniques to optimize the MNO's decisions; and 2) simplify the analytical evaluation of MNO's decision strategies.

The remainder of this letter is structured as follows: In Section II we formulate the model of asynchronous slice admission systems, defining critical concepts such as resource feasibility and slicing strategy. Then in Section III we consider the synchronous slice admission model, and prove it to be Markovian when the statistics of tenant requests are memoryless. Afterwards we provide the approximate analytical calculations of the state transition probabilities in both single step and long term as well. Section IV numerically evaluates the feasibility of this Markov model and the accuracy of proposed calculation of the state transition matrix. To the end we close this letter with our conclusion and some discussions in Section V.

II. System Model

A. Resource Allocation and Resource Feasibility

A resource pool with M different types of countable resources can be described with a vector $\mathbf{r} = [r_1, r_2, \ldots, r_M]^{\mathrm{T}}$. Consider N different slice types, for every slice type $n \in [1, 2, \ldots, N]$ it costs a resource bundle $\mathbf{c}_n = [c_{1,n}, c_{2,n}, \ldots, c_{M,n}]^{\mathrm{T}}$ to maintain a slice. All slices are atomic and indivisible. Thus, the MNO manages its resources

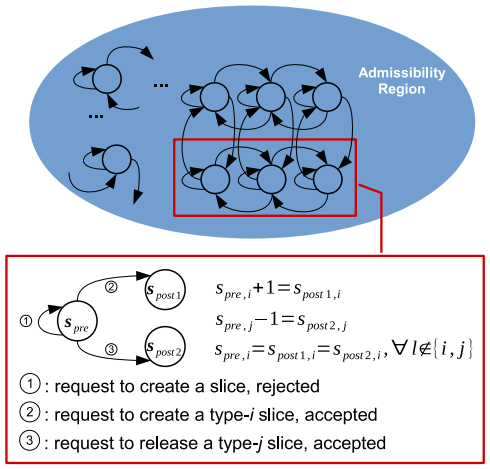

Fig. 1. In asynchronous slice admission, the state is updated at request arrivals.

by adjusting the set of active slices, which can be presented as $\mathbf{s} = [s_1, s_2, \ldots, s_N]^{\mathrm{T}}$, where s_n denotes the number of type-n slices under maintenance. The allocation is subjected to the resource pool size:

$$r_m - \sum_{n=1}^{N} c_{m,n} s_n \geq 0, \quad \forall m \in [1, 2, \ldots, M]. \tag{1}$$

We refer to the set of all allocations $\mathbf{s}$ that fulfill (1) as the *admissibility region* $\mathbb{S}$, which is obviously a finite set.

B. Tenant Requests and Slice Admission Control

In SlaaS, slices are created and released upon requests from tenants. When a tenant requires a new slice to support its service, it sends a request to the MNO, which will be either accepted or declined by the MNO. Upon acceptance, the requested slice will be created and continuously maintained until the tenant requests to release it. Practically, both the requests for creation and for release arrive randomly. It is commonly assumed that:

- For every slice type n, the slice creation requests arrive as Poisson events with a rate of λ_n
- The lifetime of every type-n slice is an independent μ_n-mean exponential random variable.

Thus, a state model can be proposed to describe the process of slice admission control, as illustrated in Fig. 1. Each state represents a resource allocation $\mathbf{s} \in \mathbb{S}$, so that the MNO cannot make any decision that leads to an unfeasible resource allocation conflicting with (1). Besides, note that requests for slice release are always accepted.

Generally, defining the incoming request q and the MNO's binary decision d as

$$q = \begin{cases} n & \text{request to create a type-}n\text{ slice} \\ -n & \text{request to release a type-}n\text{ slice} \end{cases}, \tag{2}$$

$$d = \begin{cases} 1 & \text{request accepted} \\ 0 & \text{request declined,} \end{cases} \tag{3}$$

respectively, the post-transition state $\mathbf{s}_{\mathrm{post}}$ is a function of q, d and the pre-transition state $\mathbf{s}_{\mathrm{pre}}$:

$$\begin{aligned} \mathbf{s}_{\mathrm{post}} &= \mathcal{T}\big(\mathbf{s}_{\mathrm{pre}}, q, d\big) \\ &= \Big[s_{\mathrm{pre},1}, \ldots, s_{\mathrm{pre},|q|} + d \cdot \mathrm{sgn}(q), \ldots, s_{\mathrm{pre},N}\Big]. \end{aligned} \tag{4}$$

C. Slicing Strategy

As aforementioned in Section II-B, when receiving a tenant request q, the MNO makes a binary decision $d \in \{1, 0\}$. If d is only a function of q and the pre-transition network state $\mathbf{s}_{\mathrm{pre}}$, we say that the MNO has a *consistent slicing strategy* $d = D(q, \mathbf{s}_{\mathrm{pre}})$. As the MNO cannot overload the resource pool or decline to release slices, given the admissibility region $\mathbb{S}$, a slicing strategy D is valid only when

$$D(q, \mathbf{s}) = 1, \ \ \forall q \in \{-1, -2, \ldots, -N\}, \forall \mathbf{s} \in \mathbb{S} \tag{5}$$

$$\mathcal{T}(\mathbf{s}, q, D(q, \mathbf{s})) \in \mathbb{S}, \ \ \forall q \in \{1, 2, \ldots, N\}, \forall \mathbf{s} \in \mathbb{S} \tag{6}$$

For any valid slicing strategy D, we have

$$\mathbf{s}_{\mathrm{post}} = \mathcal{T}\big(\mathbf{s}_{\mathrm{pre}}, q, D\big(q, \mathbf{s}_{\mathrm{pre}}\big)\big) = \mathcal{T}\big(\mathbf{s}_{\mathrm{pre}}, q\big). \tag{7}$$

III. Synchronous Slice Admission Control Model

A. Synchronous Slice Admission

In the last section we have built an asynchronous state model of slice admission, where the updates of state, i.e., the responses of MNO, are triggered by requests arriving at random time. Practically, MNOs usually process requests in a framed approach, where all arrived requests, despite of the type, will be buffered in queue and sequentially responded at the end of every *operations period*. This leads to a synchronous state model, where state updates are periodically triggered. The synchronous state model has self-evidently the same admissibility region $\mathbb{S}$ as the corresponding asynchronous state model, but complexer conditions for transitions between states.

Normalizing the operations period length to 1, the MNO makes sequential decisions to buffered requests at discrete time indexes. We use the vector $\mathbf{q}(t) = [q_1(t), q_2(t), \ldots, q_{Q_t}(t)]$ to denote the request queue buffered during the t^{th} operations period, where Q_t is the total number of requests arrived in that period. Given an arbitrary strategy $d = D(q, \mathbf{s})$, the network state at $(t+1)$ is

$$\begin{aligned} \mathbf{s}(t+1) &= \mathcal{T}\big(\ldots \mathcal{T}(\mathcal{T}(\mathbf{s}(t), q_1(t)), q_2(t)), \ldots, q_{Q_t}(t)\big) \\ &\triangleq \tilde{\mathcal{T}}(\mathbf{s}(t), \mathbf{q}(t)), \end{aligned} \tag{8}$$

where $\mathcal{T}(\cdot)$ is implemented according to (4) w.r.t. $D(\cdot)$.

B. Equivalent Markov Model

The most important corollary of (8) is that: *with memoryless distributions of request arrivals and a consistent slicing strategy, the synchronous state model of slice admission is Markovian*, as this subsection will show.

As suggested in Section II-B we assume that type-n slice creation requests arrive as Poisson events in rate of λ_n, and that the lifetime of every type-n slice is an exponential random variable with mean of μ_n. Thus, the probability mass function (PMF) that $k>0$ requests for type-n slice creation $q = n>0$ arrive during one operations period is

$$\mathrm{Prob}(k_n) = \frac{\lambda_n^{k_n} e^{-\lambda_n}}{k_n!}, \quad n \in [1, 2, \ldots, N]. \tag{9}$$

Given s_n as the number of type-n slices under maintenance, the PMF that $0 < k_{-n} \leq s_n$ requests for type-n slice release

$q = -n<0$ in the same operations period is

$$\mathrm{Prob}(k_{-n} \mid s_n) = \frac{s_n!\left(1 - e^{-\frac{1}{\mu_n}}\right)^{k_{-n}}}{k_{-n}!(s_n - k_{-n})!e^{\frac{s_n - k_{-n}}{\mu_n}}}. \quad (10)$$

We can merge them as

$$p_{\mathrm{A}}(k_q, q \mid s_{|q|}) = \begin{cases} \frac{\lambda_q^{k_q} e^{-\lambda_q}}{k_q!} & q > 0 \\ \frac{s_{|q|}!\left(1-e^{-\frac{1}{\mu_{|q|}}}\right)^{k_q}}{k_q!(s_{|q|}-k_q)!e^{\frac{s_{|q|}-k_q}{\mu_{|q|}}}}, & q < 0 \end{cases} \quad (11)$$

For convenience of reference, we define $\hat{\mathbf{q}}$ to denote the elements in a request sequence $\mathbf{q}$ *regardless the order*. Because the arrival processes of different requests are independent from each other, the conditional probability that $\hat{\mathbf{q}}$ arrives during one operation period in the current network state $\mathbf{s}$ is

$$\mathrm{Prob}(\hat{\mathbf{q}} \mid \mathbf{s}) = \prod_{q \in \{\pm 1, \ldots, \pm N\}} p_{\mathrm{A}}\left(\sharp^q_{\hat{\mathbf{q}}}, q \mid s_{|q|}\right), \quad (12)$$

where $\sharp^x_{\mathbf{x}}$ denotes the occurrence times of x in $\mathbf{x}$. Note that every request arrival is independent from the others. Furthermore, the memoryless distributions guarantee that the arrival of every individual request remains consistent over the entire operations period. Thus, the arrival probability of a request sequence is obviously independent of its order (proven in [9] as a feature of *dependent trials*), i.e.,

$$\mathrm{Prob}(\mathbf{q}_1 \mid \mathbf{s}) = \mathrm{Prob}(\mathbf{q}_2 \mid \mathbf{s}), \quad \forall\{\mathbf{q}_1, \mathbf{q}_2\} : \hat{\mathbf{q}}_1 = \hat{\mathbf{q}}_2. \quad (13)$$

So we have

$$\mathrm{Prob}(\hat{\mathbf{q}} \mid \mathbf{s}) = \sum_{i:\hat{\mathbf{q}}_i = \hat{\mathbf{q}}} \mathrm{Prob}(\mathbf{q}_i \mid \mathbf{s}) = Q! \times \mathrm{Prob}(\mathbf{q} \mid \mathbf{s}), \quad (14)$$

where Q is the length of $\mathbf{q}$. This yields that

$$\mathrm{Prob}(\mathbf{q} \mid \mathbf{s}) = \frac{\prod_{q \in \{\pm 1, \ldots, \pm N\}} p_{\mathrm{A}}\left(\sharp^q_{\hat{\mathbf{q}}}, q \;\middle|\; s_{|q|}\right)}{Q!}. \quad (15)$$

Now calling back (8), we are able to obtain the synchronous transition probability from state $\mathbf{s}(t)$ to $\mathbf{s}(t+1)$ as

$$\mathrm{Prob}(\mathbf{s}(t+1) \mid \mathbf{s}(t)) = \sum_{\mathbf{q}:\tilde{\mathcal{T}}(\mathbf{s}(t),\mathbf{q}) = \mathbf{s}(t+1)} \mathrm{Prob}(\mathbf{q} \mid \mathbf{s}(t)), \quad (16)$$

which depends only on $\mathbf{s}(t)$. *The synchronous slice management process is therefore Markovian*. As any other finite state Markov process, it can be characterized by an enumeration

$$f : \{1, 2, \ldots, |\mathbb{S}|\} \to \mathbb{S} \quad (17)$$

and a corresponding transition probability matrix

$$\mathbf{P} = \begin{bmatrix} P_{1,1} & P_{1,2} & \cdots & P_{1,|\mathbb{S}|} \\ P_{2,1} & P_{2,2} & \cdots & P_{2,|\mathbb{S}|} \\ \vdots & \vdots & \ddots & \vdots \\ P_{|\mathbb{S}|,1} & P_{|\mathbb{S}|,2} & \cdots & P_{|\mathbb{S}|,|\mathbb{S}|} \end{bmatrix}, \quad (18)$$

where $P_{i,j} = \mathrm{Prob}(f(j) \mid f(i))$ as defined by (16).

TABLE I
THREE DIFFERENT SERVICE DEMAND SCENARIOS

Scenario	A	B	C
Request arriving rate for new slices (λ_1)	1	0.8	0.5
Average slice lifetime (μ_1)	4	4	4

C. Approximation in Practical Applications

There can be an infinite number of different request sequences $\mathbf{q}$ that fulfill $\tilde{\mathcal{T}}(\mathbf{s}(t+1), \mathbf{q}) = \mathbf{s}(t+1)$, i.e., the calculation of $\mathbf{P}$, according to (16), contains a traversal over an infinite domain of $\mathbf{q}$, which is computationally impossible.

However, practically, if we set the operation period to a sufficiently short duration, after normalization the values of λ_n will be small while μ_n being large for all n. In this case, the value of $p_{\mathrm{A}}(k, q, \mid s_{|q|})$ fades out rapidly with increasing k for all q, and Prob($\mathbf{q} \mid \mathbf{s}$) will therefore have non-negligible values *only* for short request sequences $\mathbf{q}$. Thus, we can ignore all the cases with long sequences of arriving requests, and thereby limit the traversal operation in (16) to a limited domain of $\mathbf{q}$, making it computationally feasible to solve $\mathbf{P}$.

D. Probability of Staying in a Specific State

Given a finite-state Markov chain which initiates from the state $\mathbf{s}(0)$ at $t = 0$, with its transition probability matrix $\mathbf{P}$, the probability that it stays in a state $\mathbf{s}(T)$ after T periods is

$$\mathrm{Prob}(\mathbf{s}(T) \mid \mathbf{s}(0)) = Q_{i,j}(T), \quad (19)$$

where $i = f^{-1}(\mathbf{s}(0))$, $j = f^{-1}(\mathbf{s}(T))$ and

$$\mathbf{Q}(T) \triangleq \mathbf{P}^T = \begin{bmatrix} Q_{1,1}(T) & Q_{1,2}(T) & \cdots & Q_{1,|\mathbb{S}|}(T) \\ Q_{2,1}(T) & Q_{2,2}(T) & \cdots & Q_{2,|\mathbb{S}|}(T) \\ \vdots & \vdots & \ddots & \vdots \\ Q_{|\mathbb{S}|,1}(T) & Q_{|\mathbb{S}|,2}(T) & \cdots & Q_{|\mathbb{S}|,|\mathbb{S}|}(T) \end{bmatrix}. \quad (20)$$

IV. NUMERICAL VERIFICATION

A. Environment Setup

To verify the feasibility of our proposed model, we setup a simplified slicing scenario with a normalized one-dimensional resource pool, i.e., $\mathbf{r} = [r_1] = [1]$. It supports to implement slices of an only type that each of them costs a resource of $c_{1,1} = 0.3$. Under this specification, only 2 types of request and 4 states are available:

$$q = \pm 1, \quad (21)$$

$$\mathbb{S} = \{\mathbf{s}_1, \ldots \mathbf{s}_4\}, \quad \mathbf{s}_i = [i - 1]. \quad (22)$$

There are in total $2^{2\times 4} = 256$ different constructions of D under such specifications. Nevertheless, according to (5) and (6), only $2^3 = 8$ out of them are valid slicing strategies. To study the relation between the precision of our model and the request arrival rate, we designed three different slice service demand scenarios, as listed in Table I.

B. Simulation Procedure and Results

Given a certain scenario and a certain strategy, the state transition probability matrix $\mathbf{P}$ can be estimated as described in Section III-B. As discussed in Section III-C, we must set

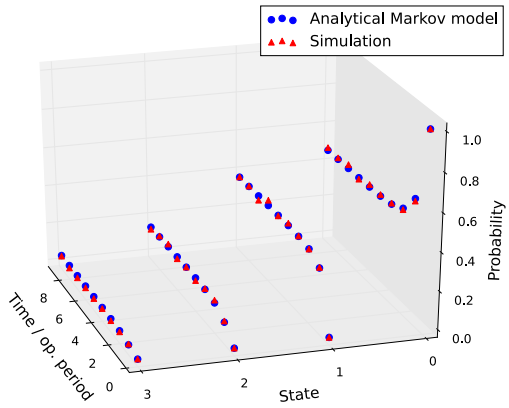

Fig. 2. The estimated and simulated PMFs of network state in the first 10 operation periods. The resource pool was initialized fully idle, an "always accept" strategy was taken in the reference scenario C, $Q^{+}_{\max} = 4$ was considered.

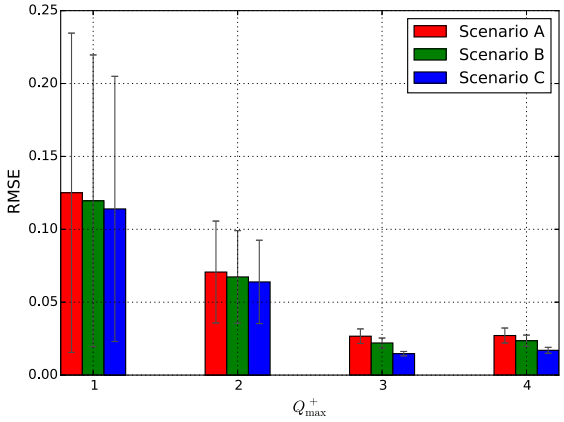

Fig. 3. The estimation error of state transition probability matrix P with respect to the $Q^{+}_{\max}$ configuration, the strategy and the service demand scenario.

an upper-bound $Q^{+}_{\max}$ to the possible arrival number of slice creation requests, so that $\mathbf{P}$ could be approximated by an estimation $\hat{\mathbf{P}}$ with a limited effort of computation. Fig. 2 shows an example result in the first ten operations periods with fixed initial state, service demand scenario and decision strategy. An excellent match between the analytical estimation based on our proposed Markov model and the simulation results, verifying the feasibility of our proposed approximate estimator $\hat{\mathbf{P}}$.

Then, to evaluate the estimation accuracy, we obtained the empirical transition probability matrix through simulations. For every specification set of scenario, strategy and $Q^{+}_{\max}$, 1000 independent runs of Monte-Carlo test were carried out. In each individual run, the network was first initialized to a random state, then operated with the specified slicing strategy for 100 operations periods. We recorded the system state in every operation period, and thereby obtained the empirical transition probability matrix $\tilde{\mathbf{P}}$. The root of mean square error (RMSE) of the estimation $\hat{\mathbf{P}}$ is thus evaluated as:

$$\epsilon = \sqrt{\frac{1}{|\mathbb{S}|^2}\sum_{i=1}^{|\mathbb{S}|}\sum_{j=1}^{|\mathbb{S}|}\left[\frac{2(\hat{P}_{i,j}-\tilde{P_{i,j}})}{\hat{P}_{i,j}+\tilde{P_{i,j}}}\right]^2} \tag{23}$$

Note that the value of ϵ depends on the decision strategy $D(\cdot)$, the preset upper-bound $Q^{+}_{\max}$ and the request statistics. So we repeated the simulation 4 times with $1 \leq Q^{+}_{\max} \leq 4$, respectively. For every value of $Q^{+}_{\max}$, we evaluated ϵ for all 8 valid strategies in all 3 aforementioned reference service demand scenarios. The results are depicted in Fig. 3. It can be seen that independent of the scenario, both the mean and the variance of estimation error sink quickly to a negligible level as $Q^{+}_{\max}$ increases, which significantly supports our analysis on the Markov model in Section III. We can also observe a dependency of the RMSE on the scenarios, that the error decreases along with the arrival rate of requests for new slices, which can be easily explained by the point that a higher request arrival rate leads to higher probability of long request sequences, which indicates a degrade of approximation accuracy with the same configuration of $Q^{+}_{\max}$. This also suggests to select a shorter operations period in practical application, in order to limit the required computational effort.

V. Conclusion and Discussion

In this letter, we have proposed a state model of synchronous slice admission in 5G mobile communications networks. We have proved that when the statistics of request arrivals are memoryless, e.g., when they are Poisson / exponential processes, and when the MNO takes a consistent slicing strategy, the state model is Markovian. An approximate analytical calculation of the state transition matrix is then proposed and verified by numerical simulations.

We would like to highlight one outcome of this letter that under certain service demand statistics, the Markov model of the network state is individually determined by the applied slicing strategy. This guarantees that the optimization problems of slice admission can be equivalently transformed into the problem of searching the best slicing strategy.

It is also worth to note that the space of valid slicing strategy can dramatically grow to a huge size as the slice pool and number of slice types increase, and thus calls for fast and efficient optimizing algorithms, which deserves further study in the future.

References

[1] "NGMN 5G white paper," Frankfurt, Germany, Next Gener. Mobile Netw., White Paper, 2015.
[2] P. Rost *et al.*, "Mobile network architecture evolution toward 5G," *IEEE Commun. Mag.*, vol. 54, no. 5, pp. 84–91, May 2016.
[3] P. Rost *et al.*, "Network slicing to enable scalability and flexibility in 5G mobile networks," *IEEE Commun. Mag.*, vol. 55, no. 5, pp. 72–79, May 2017.
[4] B. Han, S. Tayade, and H. D. Schotten, "Modeling profit of sliced 5G networks for advanced network resource management and slice implementation," in *Proc. 22nd IEEE Symp. Comput. Commun. (ISCC)*, Jul. 2017, pp. 576–581.
[5] V. Sciancalepore, F. Cirillo, and X. Costa-Perez, "Slice as a service (SlaaS) optimal IoT slice resources orchestration," in *Proc. GLOBECOM IEEE Glob. Commun. Conf.*, Dec. 2017, pp. 1–7.
[6] D. Bega *et al.*, "Optimising 5G infrastructure markets: The business of network slicing," in *Proc. 36th IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Atlanta, GA, USA, 2017, pp. 1–9.
[7] P. Caballero, A. Banchs, G. de Veciana, and X. Costa-Pérez, "Network slicing games: Enabling customization in multi-tenant networks," in *Proc. 36th IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Atlanta, GA, USA, 2017, pp. 1–9.
[8] 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, pp. 33137–33147, 2018.
[9] L. H. Y. Chen, "Poisson approximation for dependent trials," *Ann. Probab.*, vol 3, no. 3, pp. 534–545, 1975.