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 (5G) mobile communication 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 paper 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 is also proposed and evaluated.

Index Terms—5G, network slicing, network resource management, network function virtualization

I. INTRODUCTION

Network slicing has been considered as an essential feature and one of the most 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 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.

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 a 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], a MNO packs its resources into standardized atomic slices and rent 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 constrains of resource pool limit and regulation rules.

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 paper, 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) enable and simplify long-term evaluation of MNO’s decision strategy in analytical approach.

The remainder of this paper is structured as follows: In Sec. II we formulate the model of asynchronous slice admission systems, defining critical concepts such as resource feasibility and slicing strategy. Then in Sec. 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. Sec. 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 paper with our conclusion and some discussions in Sec. 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 $r = [r_1, r_2, \ldots, r_M]^T$. Consider $N$ different slice types, for every slice type $n \in [1, 2, \ldots, N]$ it costs a resource bundle $c_n = [c_{n,1}, c_{n,2}, \ldots, c_{M,n}]^T$ to maintain a slice. All slices are atomic and indivisible. Thus, the MNO manages its resources by adjusting the set of active slices, which can be presented as $s = [s_1, s_2, \ldots, s_N]^T$, where $s_n$ denotes the number of type-$n$ slices under maintenance. The allocation is subject to the resource pool size:

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

We refer to the set of all allocations $s$ that fulfill (1) as the admissibility region $\mathcal{S}$, 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 \( \lambda_n \).
• The lifetime of every type-\( n \) slice is an independent \( \mu_n \)-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 \( s \in S \), so the MNO cannot make any decision that leads to an unfeasible resource allocation conflicting with \( S \). Besides, note that all requests for slice release cannot be declined but only accepted.

![Fig. 1: In asynchronous slice admission, the state is updated at request arrivals.](image)

Generally, defining the incoming request \( q \) and the MNO’s binary decision \( d \) as

\[
q = \begin{cases} 
  n & \text{request to create a type-} n \text{ slice} \\
  -n & \text{request to release a type-} n \text{ slice} 
\end{cases}, \quad d = \begin{cases} 
  1 & \text{request accepted} \\
  0 & \text{request declined} 
\end{cases}.
\]

respectively, the post-transition state \( s_{\text{post}} \) is a function of \( q, d \) and the pre-transition state \( s_{\text{pre}} \):

\[
s_{\text{post}} = T(s_{\text{pre}}, q, d) = [s_{\text{pre}, 1}, \ldots, s_{\text{pre}, |q|} + d \cdot \text{sgn}(q), \ldots, s_{\text{pre}, N}] .
\]

C. Slicing Strategy

As aforementioned in Sec. 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 \( s_{\text{pre}} \), we say that the MNO has a consistent slicing strategy.

\[
d = D(q, s_{\text{pre}}). \quad \text{As the MNO cannot overload the resource pool or decline to release slices, given the admissibility region} \ S, \text{ a slicing strategy} \ d \text{ is valid only when}
\]

\[
D(q, s) = 1, \quad \forall q \in \{-1, -2, \ldots, -N\}, \forall s \in S (5)
\]

\[
T(s, q, D(q, s)) \in S, \quad \forall q \in \{1, 2, \ldots, N\}, \forall s \in S (6)
\]

For any valid slicing strategy \( D \), we have

\[
s_{\text{post}} = T(s_{\text{pre}}, q, D(q, s_{\text{pre}})) = T(s_{\text{pre}}, q). \quad (7)
\]

III. SYNCHRONOUS SLICE ADMISSION CONTROL MODEL

A. Synchronous Slice Admission

So far, 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 \( S \) 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 \( q(t) = [q_1(t), q_2(t), \ldots, q_{Q}(t)] \) 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 \( d = D(q, s) \), the network state at \( (t + 1) \) is

\[
s(t + 1) = T \ldots T(T(s(t), q_1(t)), q_2(t)), \ldots, q_{Q}(t)) = \hat{T} (s(t), q(t)), \quad (8)
\]

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

B. Equivalent Markov Model

The most important corollary of \( 3 \) 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 Sec. II-B we assume that type-\( n \) slice creation requests arrive Poissonianly in rate of \( \lambda_n \), and that the lifetime of every type-\( n \) slice is an exponential random variable with mean of \( \mu_n \). Thus, the probability mass function (PMF) that \( k > 0 \) requests for type-\( n \) slice creation \( q = n > 0 \) arrive during one operations period is

\[
\Pr(k_n) = \frac{\lambda_n^k e^{-\lambda_n}}{k_n!}. \quad (9)
\]

And 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

\[
\Pr(k_n | s_n) = \frac{s_n! \left(1 - e^{-\lambda_n/s_n}\right)^{k_n}}{k_n!(s_n - k_n)! e^{\lambda_n/s_n}}. \quad (10)
\]
Furthermore, the memoryless distributions guarantee that the arrival probability of a period in the current network state \( \hat{q} \) each other, the conditional probability that \( \hat{q} \) arises during one operation period in the current network state.

\[
\begin{align*}
\text{We can merge them as:} \\
p_A(k,q | s_{[q]}) &= \begin{cases} \\
\frac{k^q e^{-kq}}{k_q} & q > 0 \\
\frac{s_{[q]}(1-e^{-kq})}{k_q} & q < 0 
\end{cases}
\end{align*}
\]

(11)

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

\[
\text{Prob}(\hat{q} | s) = \prod_{q \in \{s_1, \ldots, s_N\}} p_A \left( \hat{q}_q | q | s_{[q]} \right),
\]

(12)

where \( \hat{q}_q \) denotes the occurrence times of \( x \) in \( x \). 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 operation periods. Thus, the arrival probability of a request sequence is obviously independent of its order (proven in \( \text{[9]} \) as a feature of dependent trials), i.e.

\[
\text{Prob}(q_1 | s) = \text{Prob}(q_2 | s), \quad \forall \{q_1, q_2\} : q_1 = q_2.
\]

(13)

So we have

\[
\text{Prob}(\hat{q} | s) = \sum_{i,q} \text{Prob}(q_i | s) = Q! \times \text{Prob}(\hat{q} | s),
\]

(14)

where \( Q \) is the length of \( \hat{q} \). Therefore it turns out that

\[
\text{Prob}(q | s) = \frac{\prod_{q \in \{s_1, \ldots, s_N\}} p_A \left( \hat{q}_q | q | s_{[q]} \right)}{Q!}.
\]

(15)

Now calling back \( \text{[9]} \), we are able to obtain the synchronous transition probability from state \( s(t) \) to \( s(t+1) \) as

\[
\text{Prob}(s(t+1) | s(t)) = \sum_{\hat{q} : \hat{t}(s(t), \hat{q}) = s(t+1)} \text{Prob}(q | s(t)),
\]

(16)

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

\[
f : \{1, 2, \ldots, |S|\} \rightarrow \mathbb{S}
\]

(17)

and a corresponding transition probability matrix

\[
P = \begin{bmatrix}
P_{1,1} & P_{1,2} & \cdots & P_{1,|S|} \\
P_{2,1} & P_{2,2} & \cdots & P_{2,|S|} \\
\vdots & \vdots & \ddots & \vdots \\
P_{|S|,1} & P_{|S|,2} & \cdots & P_{|S|,|S|}
\end{bmatrix},
\]

(18)

where \( P_{i,j} = \text{Prob}(f(j) | f(i)) \) as defined by (16).

**C. Approximation in Practical Applications**

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

However, practically, if we set the operation period to a sufficiently short duration, the values of \( \lambda_n \) will be small while the values of \( \mu_n \) will be large for all \( n \). In this case, the value of \( p_A(k, q | s_{[q]}) \) fades out rapidly with increasing \( k \) for all \( q \), and \( \text{Prob}(q | s) \) will therefore have non-negligible values only for short request sequences \( q \). 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 \( q \), making it computationally feasible to solve \( P \).

**D. Probability of Staying in a Specific State**

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

\[
\text{Prob}(s(T) | s(0)) = Q_{i,j}(T),
\]

(19)

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

\[
Q(T) = \Delta P^T = \begin{bmatrix}
Q_{1,1}(T) & Q_{1,2}(T) & \cdots & Q_{1,|S|}(T) \\
Q_{2,1}(T) & Q_{2,2}(T) & \cdots & Q_{2,|S|}(T) \\
\vdots & \vdots & \ddots & \vdots \\
Q_{|S|,1}(T) & Q_{|S|,2}(T) & \cdots & Q_{|S|,|S|}(T)
\end{bmatrix},
\]

(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. \( r = [r_1] = [1] \). 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:

\[
q = \pm 1,
\]

(21)

\[
S = \{s_1, \ldots, s_4\}, \quad s_i = [i-1].
\]

(22)

There are in total \( 2^{2^{16}} = 256 \) 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 Tab. 1.

| Scenario | A | B | C |
|----------|---|---|---|
| Request arriving rate for new slices (\( \lambda_1 \)) | 0.8 | 0.5 |
| Average slice lifetime (\( \mu_1 \)) | 4 | 4 |

**TABLE I: Three different service demand scenarios**

**B. Simulation Procedure and Results**

Given a certain scenario and a certain strategy, the state transition probability matrix \( P \) can be estimated as described in Sec. III-B. As discussed in Sec. III-C, we must set an upper-bound \( Q_{\text{max}} \) to the possible arrival number of slice creation requests, so that \( P \) could be approximated by an estimation \( \hat{P} \) 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 \( \hat{P} \).
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 work 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 also worths 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.

V. CONCLUSION AND DISCUSSIONS

So far, in this paper 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 work 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 also worths 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,” Next Generation Mobile Networks, White Paper, 2015.
[2] P. Rost, A. Banchs et al., “Mobile Network Architecture Evolution toward 5G,” IEEE Communications Magazine, vol. 54, no. 5, pp. 84–91, 2016.
[3] P. Rost, C. Mannweiler et al., “Network Slicing to Enable Scalability and Flexibility in 5G Mobile Networks,” IEEE Communications Magazine, vol. 55, no. 5, pp. 72–79, 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 The 22nd IEEE Symposium on Computers and Communications (ISCC), July 2017.
[5] V. Sciancalepore, F. Cirillo, and X. Costa-Pérez, “Slice as a Service (SlaaS) Optimal IoT Slice Resources Orchestration,” in GLOBECOM 2017 - 2017 IEEE Global Communications Conference, Dec 2017, pp. 1–7.
[6] D. Bega, M. Gramaglia et al., “Optimising 5G Infrastructure Markets: The Business of Network Slicing,” in The 36th IEEE International Conference on Computer Communications (INFOCOM), 2017.
[7] P. Caballero, A. Banchs, G. de Veciana, and X. Costa-Pérez, “Network Slicing Games: Enabling Customization in Multi-Tenant Networks,” in The 36th IEEE International Conference on Computer Communications (INFOCOM), IEEE, 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, no. 1, pp. 33 137–33 147, 2018.
[9] L. H. Chen, “Poisson approximation for dependent trials,” The Annals of Probability, pp. 534–545, 1975.