Radio Access Network Resource Slicing for Flexible Service Execution

Chia-Yu Chang, Navid Nikaein, Thrasyvoulos Spyropoulos
Communication Systems Department, EURECOM, France
Email: firstname.lastname@eurecom.fr

***Abstract*—Network slicing is a key enabler for the service-oriented 5G vision, that aims to satisfy various per-service requirements. Unlike core network slicing, radio access network (RAN) slicing is still at its infancy, with several works just starting to investigate the challenges and potentials to enable a mutli-tenant, multi-service RAN. One of the major challenges in RAN slicing is to provide different levels of resource isolation, through resource abstraction, virtualization, and splitting among different tenants/services, while at the same time providing multiplexing gains. To this end, in this work we propose a detailed approach for radio resource virtualization, leveraging different resource abstraction types. Based on it, we formulate the problem of inter-slice resource partitioning and allocation, and propose an algorithm to efficiently tackle this problem. Finally, we provide simulation results that support the benefits of resource abstraction and our proposed algorithm.**

I. Introduction

Fifth generation (5G) mobile networks have been associated with a number of paradigm shifts. They are supposed to not only utilize new radio access technologies, wide spectrum, and massive MIMO antenna, but more importantly aim to improve the overall flexibility of optimizing mobile networks per service type and network tenant. Specifically, a key objective is a *service-oriented* architecture that can provide 5G mobile network instances on an *as-a-service* basis. Providing support for multiple services and/or virtual networks on a single physical network requires service definition, agreement, management, and performance guarantees. In this sense, the network infrastructure providers (e.g., operators), service providers (e.g., communication/digital service), and network function providers (e.g., vendors) become decoupled, in order to enable a cost-effective network composition model. A service is thus built through the composition of multi-vendor Physical or Virtual Network Functions (PNFs/VNFs), which shall not only meet the requirements of service providers but also those of network infrastructure providers in terms of PNF/VNF interoperability and compatibility.

Network slicing is one key pillar for providing the required flexibility, as has been highlighted by the Third Generation Partnership Project (3GPP) in TR23.799, and Next Generation Mobile Networks (NGMN) alliance in [1]. A network slice can either be fully isolated from the others, down to different sets of spectrum and/or cell site. It could also share real physical resources with other slices, such as radio spectrum or network functions (e.g., layers of protocol stack), or be customized using virtualized radio resources. To this end, network softwarization and virtualization are key to flexibly customize each slice: they facilitate network function development that can accommodate specific end-to-end service requirements. They also constitute the foundation of a multi-service architecture, and are realized by adapting ideas from Software-Defined Networking (SDN), Network Function Virtualization (NFV), and Cloud Computing [2].

Both the Radio Access Network (RAN) and the Core Network (CN) have been targeted for slicing, and several prototypes have been proposed already (*CN slicing* [3] and *RAN slicing* [4], [5]). The challenges of CN slicing are also addressed by 3GPP in TR23.711, and realized through the evolved Dedicated Core network (eDECOR). Nevertheless, enforcing slices on the RAN still remains challenging, as facilitating a multi-tenant environment with performance isolation and efficient resource sharing is non-trivial. Specifically, two challenges on RAN slicing are crucial: (1) provide different levels of isolation to a slice owner and the ability to customize its service processing chain(s) across different planes, while (2) efficiently utilize the available radio resources to enable the multiplexing gains.

To tackle the aforementioned challenges, we focus on the radio resource slicing problem at the RAN domain utilizing the resource abstraction approach. Such approach serves for two main purposes: (a) isolate resources by presenting a virtual view of the resources that is decoupled from the exact physical locations, and (b) increase the multiplexing gain by adjusting physical allocation types when sharing unused resources. More specifically, we make the following contributions:

- We review the state-of-the-art on RAN resource slicing (Section II);
- We elaborate on our radio resource virtualization approach, and propose several resource abstraction types (Section III);
- We formulate an optimization problem for inter-slice radio resource sharing, propose an efficient algorithm, and perform extensive simulations to investigate its performance (Section IV).

II. Related Works

To enable RAN slicing, several 5G RAN design requirements must be fulfilled, as elaborated in [6]. 3GPP also mentions several realization principles in TR38.801 such as RAN awareness slicing, Quality of Service (QoS) support, resource isolation, Service Level Agreement (SLA) enforcement among the others. These can be enabled through the Software-Defined

978-1-5386-5979-3/18/$31.00 ©2018 IEEE

RAN (SD-RAN) architecture proposed in [7], and later implemented in [8] as the FlexRAN platform. The latter implements the customized Application Programming Interfaces (APIs), through which programmable control logic is enforced with different levels of centralization. To this end, multiple services over the RAN domain can flexibly utilize available radio resources, by means of virtualization [9].

Based on the aforementioned enablers, several RAN slicing studies have been initiated. The NVS approach virtualizes radio resources [10], and enables different resource provisioning approaches to allow several Mobile Virtual Network Operators (MVNOs) to coexist in a single physical RAN; however, the multiplexing gain is not considered. In [11], the radio resource scheduler is separated into intra-slice and inter-slice scheduling. However, no resource abstraction/virtualization is provided. The proposed RAN slicing architecture in [4] proposes the virtualized resource block (vRB) concept, enforcing radio resource management at Physical Resource Block (PRBs) level. Yet, it can not customizedly abstract resource per slice request. The Orion approach groups PRBs into vRB groups through a set of abstractions, and provides only relevant resource information to the corresponding slice [5]. However, it only focuses on resource isolation without investigating any multiplexing opportunities. In summary, no proposed execution environment concurrently supports various slice requirements (e.g., isolation), elasticity to improve multiplexing benefits (through sharing), and radio resource abstractions to serve various flavors of customizable slices [12].

III. Radio Resource Virtualization

Resource virtualization is one key concept to provide the required level of isolation and sharing that is customized for each slice. Namely, it partitions radio resources based on the context and requirements of each slice, abstracts physical resources to/from the virtualized ones, and exposes a virtual view to a slice that is customized and decoupled form the exact physical resources. To this end, we elaborate on three key steps to realize the radio resource virtualization.

A. Inter-slice Resource Partitioning

Radio resource partitioning is a periodic process that happens every allocation window of T, and it distributes radio resources among multiple slices based on their requirements expressed in the slice context. Such requirement contains three elements: (1) *resources type* that defines whether the requested resources are of type physical and/or virtual radio resources in time and frequency domains[1] or capacity in terms of data rate, (2) *resource abstraction type* that specifies how the requested resources are mapped to the physical allocations, namely, fixed position, contiguous, non-contiguous, or minimum Resource Block Groups (RBGs), and (3) *resource structure* that contains the applicable radio frame numerologies in time and frequency domains. More specifically, different numerologies are in terms of Transmission Time Interval (TTI) and Sub-Carrier Spacing (SCS), which can be applied depending on the carrier frequency and/or wireless channel non-idealities due to the user high mobility (i.e., frequency offset due to Doppler shift). For instance, only one type of SCS, i.e., 15 kHZ, exists in LTE system, while there are five applicable SCSs, i.e., 15, 30, 60, 120, and 240 kHz, defined in 3GPP TS38.211 with their corresponding radio frame structures.

[1]It can be extended to other dimensions, e.g., component carrier and space.

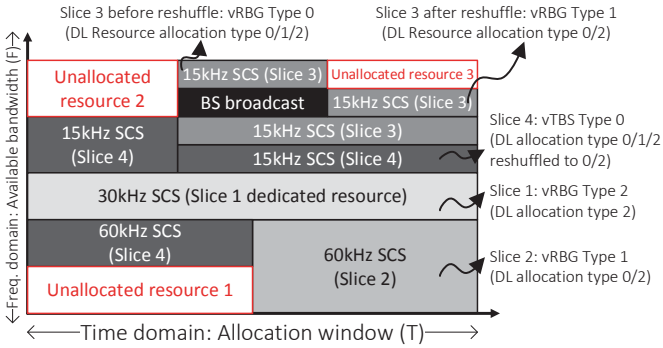


Fig. 1: Resource partition with different abstraction types.

Besides the aforementioned radio resource requirements expressed by the slice owner, the resource partitioning shall also respect the policy defined by the infrastructure provider, e.g., the allowed resource allocation types supported by the underlying Radio Access Technologies (RATs). Take the downlink (DL) resource allocation of LTE system for instance[2], there are three types of resource allocation: (1) Type 0 allocation is based on the RBG as the minimum resource granularity that comprises multiple RBs, (2) Type 1 categorizes RBGs into several subsets and only allocates RBs within the same subset, and (3) Type 2 allocates contiguous virtual RBs (vRBs) that can be physically contiguous (localized vRB) or non-contiguous (distributed vRB). For the uplink (UL) direction, there are two resource allocation types: (a) Type 0 allocates PRBs contiguous, and (b) Type 1 allocates non-contiguous RBGs that are distributed in two clusters.

In summary, four *resource abstraction types* and their respective mapping to the DL/UL resource allocation types are identified in TABLE I. Note that the proposed virtual RBG (vRBG) and virtual transport block size (vTBS) form a superset of legacy resource allocation types and they provide substantial flexibility for both intra-slice resource allocation and inter-slice resource partitioning, which are further elaborated in Section IV. Fig. 1 illustrates an example of resource partitioning among 4 slices over an allocation window T with different abstraction types in DL direction. The resource abstraction allows the infrastructure provider to dynamically reshuffle the resource allocation types. For example, the allocation type can be modified from type 0/1/2 to type 0/2 for slice 3 and 4. Additional flexibility is achieved when unused resources can be shared to increase the resource utilization. For instance, the unallocated resource 1, 2 and 3 can be utilized by slices that request 15 kHz SCS, while the former two can further be leveraged by slices that request 30 kHz SCS.

[2]Even the proposed scheme is based on the LTE system, it can be further applied to more sophisticated allocation types adopted in 5G.

TABLE I: Mapping between resource abstraction and allocation type.

Requested resources	Abstraction types (Resource granularity)	DL Resource allocation type	UL Resource allocation type
Resource block	vRBG Type 0 (Non-contiguous)	Type 0, Type 1, Type 2 distributed	Type 1
	vRBG Type 1 (Contiguous)	Type 0, Type 2 localized	Type 0
	vRBG Type 2 (Fixed position allocation)	Type 2 localized	Type 0
Capacity	vTBS Type 0 (RBGs with min granularity)	All Types	All Types

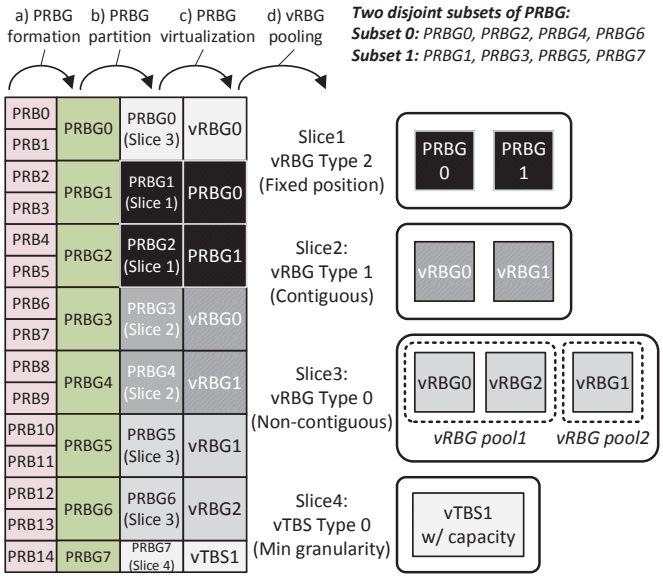
B. Radio Resource Abstraction

As mentioned in the last paragraph of Section I, there are two main purposes for the abstraction of radio resources. Firstly, the resource isolation by presenting a virtual view of the radio resources that is decoupled from the physical resources, and thus preventing other slices to access or even infer the resources allocated to others (benefits the slice owner). Secondly, the resource multiplexing by adjusting the allocation type and scheduling policy, and thus increasing the resource utilization efficiency (benefits the infrastructure provider).

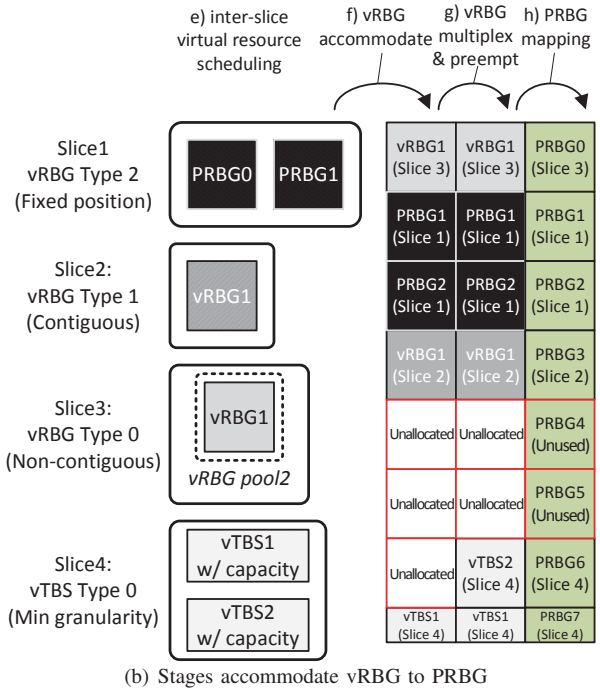
Take the 3 MHz case of LTE system as an example in Fig. 2(a), where there are 15 PRBs and the PRBG granularity is 2 PRBs, i.e., there are 8 Physical RBGs (PRBGs) with the last PRBG only contains 1 PRB. These PRBGs are partitioned for each slice based on the number of required resources and the resource granularity stated in TABLE I. Then, they are virtualized into vRBGs or vTBSs according to the abstraction type. For instance, fixed position resources is requested by slice 1; hence, no virtualization is performed (i.e., PRBG). In contrast, slice 4 requests a capacity value and its PRBGs are abstracted into vTBS with the corresponding capacity. On the other hand, PRBGs of slice 2 and 3 are virtualized into vRBGs via abstracting the exact frequency/time locations and dimensions. Further, these vRBGs are pooled together to maintain their relative frequency dependencies without revealing their absolute physical frequency locations. Take the slice 3 that uses DL resource allocation type 1 as an example, only PRBGs within the same subset can be scheduled at the same time. Hence, vRBGs are pooled to indicate such exclusive condition between vRBG pool 1 (i.e., PRBG0, PRBG6) and pool 2 (i.e., PRBG5), and the intra-slice scheduler of slice 3 will allocate resources to each user from either vRBG pool 1 or pool 2.

C. Resource Accommodation and Multiplexing

After the radio resource partitioning and virtualization, each slice can perform the intra-slice resource scheduling and these scheduling decision will be accommodated into PRBs as shown in Fig. 2(b). Such accommodation do not necessary follow the mapping done in the partitioning stage (cf. Fig. 2(a)) as it allows a better utilization of the radio resources. For instance, vRBG1 of slice 2 and slice 3 are accommodated to their vRBG0 in the partitioning stage respectively to have a larger contiguous unallocated region (i.e., from PRBG4 to PRBG6) that can be further shared to other slices. The unallocated region can be utilized to satisfy some other slices that request more resources, e.g., vTBS2 of slice 4[3]. Moreover, the preemption scheme can also be applied by removing the inter-slice scheduling results of other low-priority slices to boost the perceived performance of high-priority slices[4]. Finally, PRBGs can be mapped and the corresponding Control Information (CI) are formed. Note that the CI is used to indicate the user about the positions of allocated PRBs together with other physical layer information for successful transportation. With limited control region for CI transportation, the unallocated resources (i.e., PRBG4 and PRBG5) can also be utilized to carry CI.



(a) Stages form vRBG and vRBG pool



(b) Stages accommodate vRBG to PRBG

Fig. 2: Different stages for virtualized resources slicing

[3]Such resource multiplexing may not be allowed by slices with fixed position allocation, i.e., vRBG Type 2.

[4]The preemption characteristic shall be described in the slice context.

IV. Resource Partitioning and Accommodation

In this section, we examine the problem of inter-slice virtual resource partitioning and accommodation and then provide the corresponding algorithm and performance evaluation.

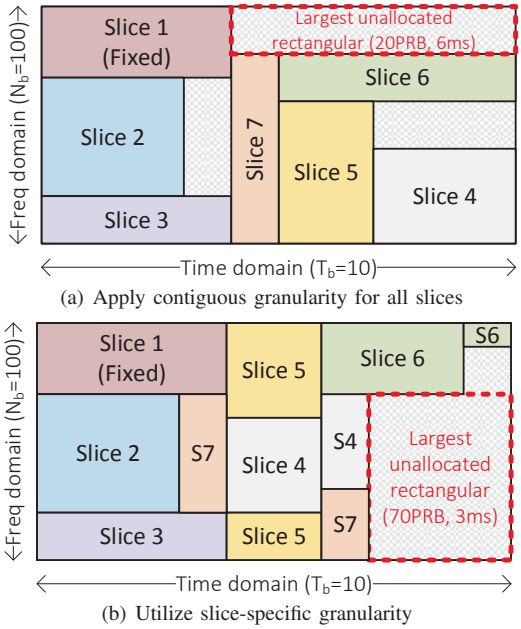
A. Inter-slice Resource Partitioning

As mentioned beforehand, such partitioning is periodic within an allocation window T to partition available resource among the radio bandwidth F. These resources are specifically quantized into a resource grid with T_b TTIs in the time domain and N_b PRBs in the frequency domain with respect to the base SCS (SCS_{base}) used by the infrastructure, e.g., a 20MHz LTE radio bandwidth in a 10ms allocation window is separated into $N_b = 100$ PRBs and $T_b = 10$ TTIs.

Particularly, there are $|S|$ slices that are requesting the radio resources, and they are included in set $\mathcal{S} = \{s_1, \cdots, s_{|s|}\}$. For the k-th slice (i.e., s_k), its radio resource requirements include: (a) SCS_k comprises the applicable SCSs, (b) T_k and N_k is the number of requested resource in time (ms) and frequency domain (Hz) respectively, and (c) g_k is the granularity which can be contiguous, non-contiguous, fixed position (with its fixed staring position as FT_k and FN_k) or minimum granularity (with its requested rate as R_k in bps). The fixed position inherently isolates resources as its partitioned resources are physical without any virtualization. The contiguous one is more suitable for quasi-constant traffic patterns (e.g., streaming) since it can reduce the latency and minimize the CI signaling overhead. The non-contiguous one, on the other hand, accommodates better for variable traffic patterns as it can allocate fragmented resources. The minimum granularity can be utilized by those slice that requests capacity (i.e., vTBS in TABLE I), which allowing for all feasible partitioning.

A resource partitioning example for 7 slices is shown in Fig. 3 with different numbers of requested resource and granularities: $g_1 = \text{F}$ (i.e., Fixed position), $g_2 = g_3 = \text{C}$ (i.e., Contiguous), $g_4 = g_5 = \text{NC}$ (i.e., Non-contiguous), and $g_6 = g_7 = \text{M}$ (i.e., Min granularity). Note that the largest rectangular of unallocated resource is marked which is an important criterion for resource multiplexing. Since such largest rectangular can (1) potentially fit in any numerologies among different SCSs to be shared with other slices, or (2) transport CI signaling and cell information. It can be seen from Fig. 3(a) and 3(b) that although both examples can satisfy all 7 slices, while the latter has a larger unallocated rectangular region via exploiting different resource granularities, e.g., s_4, s_5 are non-contiguous, and s_6, s_7 use minimum granularity.

Based on these observations, the inter-slice resource partition has two complementary goals: (a) satisfy as many slice resource requests as possible, and (b) maximize the size of unallocated rectangular region for further multiplexing. In following, we introduce the control variables for the problem formulation: (1) $m_{i,j,k} \in \mathcal{M}$ is binary, indicating whether the resource substrate $i \in \{1, \cdots, T_b\}$ and $j \in \{1, \cdots, N_b\}$ in time and frequency domain is partitioned for the k-th slice or not, (2) $b_{i,j,k} \in \mathcal{B}$ is binary, representing whether the resource



(a) Apply contiguous granularity for all slices

(b) Utilize slice-specific granularity

Fig. 3: Examples of radio resource partitioning

substrate $i \in \{1, \cdots, T_b\}$ and $j \in \{1, \cdots, N_b\}$ is the unique beginning position for the k-th slice or not, (3) $a_{i,k} \in \mathcal{A}$ is is binary, using by the non-contiguous slices (i.e., $g_k = \text{NC}$) to indicate whether the i-th TTI is partitioned for the k-th slice or not, and (4) $c_k \in \mathcal{C}$ is the applied SCS for the k-th slice. Based on such applied SCS for the k-th slice, the input resource request (i.e., T_k, N_k) is further computed correspondingly as $T_k^{c_k}$ and $N_k^{c_k}$. The overall resource partitioning problem is formulated as follows.

$$\max_{\mathcal{M},\mathcal{B},\mathcal{A},\mathcal{C}} \sum_{i=1}^{T_b}\sum_{j=1}^{N_b}\sum_{k=1}^{|\mathcal{S}|} b_{i,j,k} + w \cdot \text{MaxRec}\left(\mathcal{M}\right) \tag{1a}$$

$$s.t. \sum_{k=1}^{|\mathcal{S}|} m_{i,j,k} = 1, \ \forall i, j \tag{1b}$$

$$\sum_{i=1}^{T_b}\sum_{j=1}^{N_b} b_{i,j,k} \le \begin{cases} b_{FT_k^{c_k}, FN_k^{c_k}, k}, & \text{if } g_k = \text{F} \\ 1, & \text{else} \end{cases}, \forall k \tag{1c}$$

$$\sum_{i=1}^{T_b}\sum_{j=1}^{N_b} \left(m_{i,j,k} - T_k^{c_k} \cdot N_k^{c_k} \cdot b_{i,j,k}\right) = 0, \ \forall k \tag{1d}$$

$$\sum_{p=i}^{i+T_k^{c_k}-1}\sum_{q=j}^{j+N_k^{c_k}-1} m_{p,q,k} \ge b_{i,j,k} \cdot N_k^{c_k} \cdot T_k^{c_k}, \forall i, j, g_k = \text{F}||\text{C} \tag{1e}$$

$$\sum_{j=1}^{N_b} m_{i,j,k} = N_k^{c_k} \cdot a_{i,k}, \ \forall i, g_k = \text{NC} \tag{1f}$$

$$\left(\sum_{j=1}^{N_b} b_{i,j,k}\right)\left(\sum_{p=i}^{T_b} a_{p,k} - T_k^{c_k}\right) = 0, \ \forall i, g_k = \text{NC} \tag{1g}$$

$$[T_k^{c_k}, N_k^{c_k}] = \text{SCSMap}\left(T_k, N_k, R_k, c_k, SCS_{base}\right), \ \forall k \tag{1h}$$

$$[FT_k^{c_k}, FN_k^{c_k}] = \text{SCSMap}(FT_k, FN_k, 0, c_k, SCS_{base}), \forall k \tag{1i}$$

$$m_{i,j,k} \in \{0,1\}, b_{i,j,k} \in \{0,1\}, a_{i,k} \in \{0,1\}, c_k \in \text{SCS}_k \tag{1j}$$

In eq. (1a), two goals are included in the objective function. The first one indicates whether the unique beginning point can be found, while the second relies on the MaxRec $(\cdot)$ function to output the largest unallocated rectangular based on the control variables $\mathcal{M}$. A weight $w > 0$ can balance these two goals. Eq. (1b) guarantees each resource is partitioned only by a single slice, while eq. (1c) ensures at most one beginning point is indicated for each slice. It it noted that the fixed position slices can only set $b_{FT_k,FN_k,k}$ to be 1. Eq. (1d) restricts up to $T_k^{c_k} \times N_k^{c_k}$ resource are partitioned for the kth slice, and eq. (1e) provides the dimensional constraints for fixed-position and contiguous granularity slices. Moreover, eq. (1f) and eq. (1g) are the corresponding dimensional constraints for non-contiguous granularity slices via utilizing $a_{i,k}$ to indicate $T_k^{c_k}$ TTI instances each with $N_k^{c_k}$ PRBs. For minimum granularity slices, there is no dimensional constraint. Finally, eq. (1h) and eq. (1i) use the SCSMap $(\cdot)$ function to compute the quantized resource request (i.e., $T_k^{c_k}$, $N_k^{c_k}$) and fixed position (i.e., $FT_k^{c_k}$, $FN_k^{c_k}$) based on the selected SCS c_k. Note that the quantized resource request for min granularity slice is derived based on the requested rate R_k and channel state information.

We notice that such problem becomes a two-dimensional knapsack problem when all resource granularities are continuous, and it makes the complexity, to find the optimal solution, NP-hard. Some heuristic methods are found in [13], [14] that only deal with the contiguous granularity and focus on a single SCS. In this sense, we provide a unified algorithm in Alg. 1 that sequentially prioritizes each slice (i.e., s_k) based on the priortization policy ($priority$), and partitions resources according to the granularity-specific algorithms. The granularity-specific algorithms aim to greedy map the partitioned resource to the place that can have the largest unallocated rectangular (i.e., MaxRec $(\mathcal{M})$), and we provide more details of them in [15]. The polynomial time complexity is possessed with proportional to the number of SCSs $(\left|\bigcup_{s_k \in \mathcal{S}} SCS_k\right|)$, number of slices $(|\mathcal{S}|)$, and the size of the resource grid $(N_b \times T_b)$.

The proposed resource partitioning algorithm is executed sequentially, and thus high priority slices will impact the available regions for low priority slices. In following, we compare the performance of the proposed algorithm in terms of different slice prioritizations (i.e., $priority$ in Alg. 1):

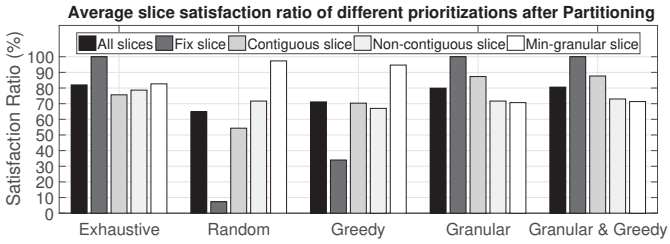
1) *Optimal*: Exhaustive search all possibilities to get the best ordering that matching the objective function in eq. (1a).
2) *Random*: Randomize the slice ordering.
3) *Greedy*: Use the greedy method to prioritize slice that can generate the largest unallocated rectangular region.
4) *Granularity*: Sort and prioritize slices based on their granularities as follows: fixed position, contiguous, non-contiguous, and minimum granularity.
5) *Granular&Greedy*: Use two sequential sorting, firstly based on the granularity and secondly based on the greedy.

The results are shown in Fig. 4 with 7 slices and each slice requests a time-varying uniformly-distributed resources with $N_k \sim$ Uniform $(1.6, 9)$ MHz and $T_k \sim$ Uniform $(1, 10)$ ms under $F = 20$ MHz radio bandwidth. Note that the granularity

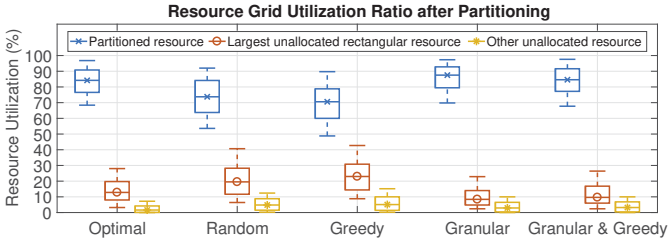
Algorithm 1: Inter-slice Resource Partition Algorithm

Input : T_b and N_b are the size of resource grid in partition window
$\mathcal{S}$ is the set of slices
Output: $\mathcal{M}$ is the resource grid allocation map
$\mathcal{C}$ is the set of applied SCS of each slice
$\mathcal{B}$ is the set of slice satisfaction index

begin
  foreach $s_k \in \mathcal{S}$ **do**
    for $i = 1$ *to* T_b **do**
      for $j = 1$ *to* N_b **do**
        $b_{i,j,k} = 0$; /* Initialize beginning indicator */
        $m_{i,j,k} = 0$; /* Initialize mapping indicator */
    $c_k = 0$; /* Initialize the selected SCS of each slice */
    foreach $scs \in SCS_k$ **do**
      $\left[T_k^{c_k}, N_k^{c_k}\right] =$ SCSMap $(T_k, N_k, R_k, scs, SCS_{base})$;
      $\left[FT_k^{c_k}, FN_k^{c_k}\right] =$ SCSMap $(FT_k, FN_k, 0, scs, SCS_{base})$;
  while *isempty* $(\mathcal{S}) ==$ *false* **do**
    $s_k =$ prioritize $(\mathcal{S}, priority)$; /* Get most prioritized slice */
    switch g_k **do**
      /* Below partitioning are done with largest MaxRec$(\mathcal{M})$ */
      case *Fix* **do**
        $[\mathcal{B}, \mathcal{C}, \mathcal{M}] =$ Fix_RP $(k, \mathcal{S}, \mathcal{M})$; (cf. Alg.2 in [15])
      case *Con* **do**
        $[\mathcal{B}, \mathcal{C}, \mathcal{M}] =$ Con_RP $(k, \mathcal{S}, \mathcal{M})$; (cf. Alg.3 in [15])
      case *NonCon* **do**
        $[\mathcal{B}, \mathcal{C}, \mathcal{M}] =$ NonCon_RP $(k, \mathcal{S}, \mathcal{M})$; (cf. Alg.4 in [15])
      case *Min* **do**
        $[\mathcal{B}, \mathcal{C}, \mathcal{M}] =$ Min_RP $(k, \mathcal{S}, \mathcal{M})$; (cf. Alg.5 in [15])
    if $\sum_{i=1}^{T_b} \sum_{j=1}^{N_b} b_{i,j,k} == 1$ **then**
      $\mathcal{S} =$ SetDiff $(\mathcal{S}, s_k)$; /* Remove satisfied slice from set $\mathcal{S}$*/



(a) Average slice satisfaction ratio



(b) Resource grid occupation ratio

Fig. 4: Different slice priortization in resource partitioning.

of each slices is the same as the ones shown in Fig. 3, and the applicable SCS set is $SCS_k = \{15, 30, 60\}$ kHz. Fig. 4(a) shows the metric of slice satisfaction ratio (i.e., the first objective in eq. (1a)) of all slices or of individual granularity types. The *Optimal* case reaches the highest satisfaction ratio (82% on average for all slices) but with higher time complexity (e.g., 1 day for the considered scenario). Note that the *Granular&Greedy* priortization (81%) outperforms other approaches and is very close to the *Optimal* case as it not only follows the elasticity of resource granularity but also seeking for the largest unallocated region at the same time.

Moreover, the resource grid utilization ratio is shown in Fig. 4(b) comprising the percentage of (a) resources that are allocated, (b) largest unallocated rectangular (i.e., the second objective in eq. (1a)), and (c) other unallocated resource in the box plot. Both random and greedy priortizations have a larger unallocated rectangular (20% and 23% on average) at the cost of a lower partitioning percentage (73% and 71% on average) shown in Fig. 4(a). In contrast, the percentage of the largest unallocated rectangular is close between the *Optimal* (12%) and the *Granular&Greedy* (10%) priortizations, which confirms the efficiency of the proposed algorithm.

B. Inter-slice Resource Accommodation

After inter-slice partitioning and intra-slice scheduling, the resource accommodation is performed by mapping the scheduled virtualized resources into the physical ones. Except for the slices with fixed-position granularity, such accommodation do not necessary map to the same physical resource done in the partitioning stage (cf. Fig. 2(b)). Moreover, our objective function here is the same as the two goals in eq. (1a). In this sense, the accommodation problem is close to the one in eq. (1) but with following differences:

1) SCS_k only contains the the optimal SCS c_k selected in the resource partitioning stage (i.e., output $\mathcal{C}$ from Alg. 1).
2) $T_k^{c_k}$ and $N_k^{c_k}$ are now determined based on the intra-slice scheduling outcomes. For notation clarity, we denote the corresponding results from the partitioning stage of eq. (1) as $T_{k,p}^{c_k}$ and $N_{k,p}^{c_k}$, respectively.
3) $FT_k^{c_k}$ and $FN_k^{c_k}$ are the same as the results from eq. (1).
4) The fixed-position part of eq. (1c) is changed into eq. (2) since its fixed starting point is restricted.

$$\sum_{i=1}^{T_b}\sum_{j=1}^{N_b} b_{i,j,k} \leq \sum_{p=0}^{T_{k,p}^{c_k}-T_k^{c_k}} \sum_{q=0}^{N_{k,p}^{c_k}-N_k^{c_k}} b_{FT_k^{c_k}+p,FN_k^{c_k}+q,k}, \forall g_k = \mathrm{F} \quad (2)$$

With aforementioned changes, we can use the same algorithm in Alg. 1 to accommodate the intra-slice scheduling outcomes. The traffic arrival rate of each slice is assumed to be proportional to the number of requested radio resource that is further multiplied with a time-varying uniformly-distributed *traffic arrival ratio* $p \sim \text{Uniform}\,(0.0, 1.0)$. We then evaluate the performance of two priortizations (i.e., *Optimal*, *Granular&Greedy*) considering two cases: (a) resource abstraction as stated above (labelled A in Fig. 5) and (b) no resource abstraction (labelled NA in Fig. 5). The latter indicates that the accommodation can only be done over the partitioned resources, and thus eq. (3) is further applied where $m_{i,j,k}^p$ is from the results of partitioning stage.

$$m_{i,j,k} \leq m_{i,j,k}^p, \ \forall i,j,k \quad (3)$$

In Fig. 5, the *Optimal* and *Granular&Greedy* priortizations shows similar performance, and the resource abstraction can bring $\sim 12\%$ more gain in the largest unallocated rectangular. Such results confirms the efficiency of the proposed algorithm and the benefits of resource abstraction.

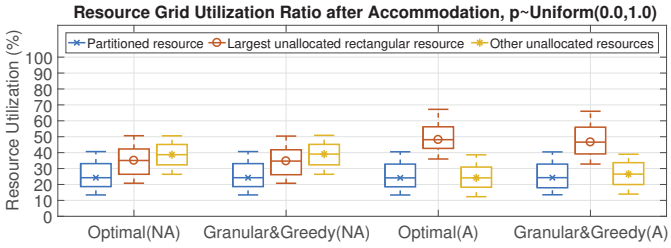


Fig. 5: Slice prioritization and resource abstraction impact.

V. Conclusions

In this work, we examine the RAN resource slicing approach to serve different radio resource requirements. We detail the proposed approach of radio resource virtualization for multiple services leveraging four resource abstraction types. Moreover, we provide the corresponding algorithm to deal with the optimization problem of inter-slice resource partitioning and accommodation. Finally, the simulation results reveals the efficiency of the proposed algorithm and the benefit of resource abstraction.

Acknowledgement

Research and development leading to these results has received funding from the European Union (EU) Framework Programme under Horizon 2020 grant agreement no. 762057 for the 5G-PICTURE project.

References

[1] NGMN, "Description of network slicing concept," Tech. Rep., 2016.
[2] P. Rost *et al.*, "Mobile network architecture evolution toward 5G," *IEEE Communications Magazine*, vol. 54, no. 5, pp. 84–91, 2016.
[3] Z. A. Qazi *et al.*, "A High Performance Packet Core for Next Generation Cellular Networks," in *ACM SIGCOMM*, 2017, pp. 348–361.
[4] A. Ksentini and N. Nikaein, "Toward Enforcing Network Slicing on RAN: Flexibility and Resources Abstraction," *IEEE Communications Magazine*, vol. 55, no. 6, pp. 102–108, 2017.
[5] X. Foukas *et al.*, "Orion: RAN Slicing for a Flexible and Cost-Effective Multi-Service Mobile Network Architecture," in *ACM MobiCom*, 2017.
[6] P. Marsch *et al.*, "5G radio access network architecture: design guidelines and key considerations," *IEEE Communications Magazine*, vol. 54, no. 11, pp. 24–32, 2016.
[7] A. Gudipati *et al.*, "SoftRAN: Software defined radio access network," in *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*. ACM, 2013, pp. 25–30.
[8] X. Foukas *et al.*, "FlexRAN: A Flexible and Programmable Platform for Software-Defined Radio Access Networks." in *ACM CoNEXT*, 2016.
[9] M. Richart *et al.*, "Resource slicing in virtual wireless networks: A survey," *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 462–476, 2016.
[10] R. Kokku *et al.*, "Nvs: A substrate for virtualizing wireless resources in cellular networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 20, no. 5, pp. 1333–1346, 2012.
[11] P. Rost *et al.*, "Network Slicing to Enable Scalability and Flexibility in 5G Mobile Networks," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 72–79, 2017.
[12] N. Nikaein and C.-Y. Chang, "Slicing and orchestration in service-oriented RAN architecture," *IEEE Software Defined Networks Newsletter*, 12 2017.
[13] M. Yang *et al.*, "Karnaugh-map like online embedding algorithm of wireless virtualization," in *IEEE WPMC*, 2012, pp. 594–598.
[14] J. Van De Belt *et al.*, "A dynamic embedding algorithm for wireless network virtualization," in *IEEE VTC Fall*, 2014, pp. 1–6.
[15] C.-Y. Chang and N. Nikaein, "RAN slicing runtime system for flexible and dynamic service execution environment," Eurecom, Tech. Rep., 10 2017.