A Crowding Game for Dynamic RAN Slicing Algorithm in 5G Networks

Karen Boulos†, Kinda Khawam*, Mohamad Yassin† and Salvatore Costanzo†
†Orange-Labs, 44, Avenue de la République, 92320 Châtillon, France
*University of Versailles, 55 Avenue de Paris, 78000 Versailles, France

***Abstract*—Fifth generation mobile networks (*i.e*, 5G) are not only characterized by higher capacity and throughput, but also by the support of heterogeneous services, characterized by different Quality of Service (QoS) requirements in terms of throughput, delay or energy consumption. Such multi-service networks require the partitioning of the available radio resources into mutliple slices, where each service occupies one slice in an isolated manner. However, a static isolation scheme among slices might have a negative impact on radio resource efficiency. For example, when one slice is highly loaded, while the second is lowly loaded, the resources of the first slice are over-utilized, whereas the resources of the second are under-utilized. Consequently, a Dynamic Radio Access Networks (RAN) slicing algorithm is paramount so as to adapt to the different load conditions of services. In that way, the radio resource utilization efficiency is increased, while the different QoS requirements of services are satisfied. In this paper, we tackle this problem by proposing a Dynamic RAN Slicing algorithm based on a crowding game. The algorithm efficiently allocates the resources among heterogeneous RAN slices in a way to satisfy the different QoS requirements of users belonging to different services while increasing the resource utilization efficiency. The results show that our proposed solution overcomes the Static Scheme in terms of QoS guarantee and utilization efficiency.**

***Index Terms*—5G, Dynamic RAN Slicing, Crowding Game, Best Response, Pure Nash Equilibrium**

I. Introduction

The support of a simultaneous range of business models, such as Internet of Things (IoT) technologies, self-driven cars and smart cities is a main target of the Fifth Generation (5G) mobile networks. Such heterogeneity among different applications, having each its own QoS requirements in terms of high throughput, strict delay and low energy consumption, requires an efficient partitioning of the scarce Radio Access Network (RAN) resources. In this context, RAN slicing has been recently proposed and consists of partitioning the RAN resources into multiple logical networks (*i.e*, slice), where each slice would be customized to support one service type having its own QoS requirements. In this way, multiple services co-exist over one single infrastructure and commonly share the RAN resources in terms of hardware equipment and spectral resources. Capital Expenditures (CAPEX) are consequently reduced, while RAN resources are efficiently utilized.

In order to perform slicing, the third Generation Partnership Project (3GPP) suggests an isolation among the different slices sharing the spectral resources of one base station [1]. In fact, a minimum isolation among slices is crucial in order to prevent one service from affecting the performance of the other services. However, given the fluctuations and the non-uniform space-time distribution of the mobile traffic over time, the resource utilization efficiency might be compromised if a strict isolation is applied. In fact, a Dynamic RAN Slicing is paramount in order to achieve adaptability to different demands and mobile traffic variations. At one hand a Dynamic RAN Slicing scheme allows for an efficient use of the radio resources, on the other hand, this might compromise the isolation among slices. Thus, a challenge would be to design a Dynamic RAN Slicing algorithm which adapts to different load conditions while satisfying the various QoS requirements for all services.

In this paper, we address the Dynamic RAN Slicing problem. In particular, we design an efficient slice resource allocation algorithm based on a crowding game [2]. In our proposed scheme, users having any service type (*i.e*, S-NSSAI) select their own resources according to predefined cost functions. The cost functions are carefully defined and tuned so as to fit the QoS of each service. For the sake of simplicity, we compare the results of our proposition with a static isolation scheme where a subset of the spectral resources are exclusively dedicated to each slice. Additionally, we analyze the time complexity of our algorithm and discuss a scenario of implementation in the case of the O-RAN architecture (Open RAN) [3].

II. Related Work

Network slicing has taken much attention in the recent studies. Many Standard Developing Organizations (SDOs), such as 3GPP, O-RAN alliance, ETSI, ITU and IETF are working to concretize this concept and make it feasible. Network slicing involves core slicing and RAN slicing [4]. For example, the work in [5] proposes a core slicing solution enabled by the Software Defined Networks (SDN), while [6] discusses many challenges related to RAN Slicing. In fact, many studies (*e.g*, [7], [8]) consider the Dynamic RAN Slicing problem among mutliple operators (*i.e*, tenants), sharing one single infrastructure and having each their own Service Level Agreement (SLA) to satisfy. Those works rely solely on an elastic traffic model which is not realistic in presence with other types of services. Contrarily, [9] tackles the Dynamic RAN Slicing problem and considers two different types of services (*i.e*, IoT and video streaming). Additionally, [10] proposes a game theoretical framework for the Dynamic RAN Slicing considering only the inelastic services with minimum rates to

978-1-6654-2854-5/21/$31.00 ©2021 IEEE

guarantee. While all those works tackle the Dynamic RAN Slicing problem, they do not consider the effect of inter-slice interference due to inter-cell interference in the case of a multi-cellular network. Nonetheless, few studies such as [11] and [12] address the problem of inter-slice interference that might affect the isolation among slices. However, those works do not focus on the resource allocation problem among slices. In other terms, the Dynamic RAN Slicing problem is not treated. Conversely, our work tackles the Dynamic RAN Slicing problem which is formulated as a crowding game. Unlike previous studies, our work considers two different types of services with different QoS requirements to satisfy. Particularly, we consider Ultra-Reliable and Low Latency Communications (URLLC) service which is characterized by a high reliability requirement and strict delay, along with enhanced Mobile Broadband (eMBB) service which requires high throughput. Further, different cost functions per service type are considered and well tuned so as to improve radio resource utilization efficiency while meeting the different QoS constraints. An important practical aspect is also highlighted and consists of proposing a scenario of implementation in the case of O-RAN architecture.

The rest of the paper is organized as follows:
In section III the system model is discussed. Section IV presents the crowding game framework for the resource allocation algorithm and discusses the cost functions definition and tuning. Section V analyses the time complexity of our proposed solution and discusses a scenario of implementation in the case of O-RAN architecture. Section VI evaluates the performance of our proposed solution in comparison with the Static Scheme. Section VII concludes the paper.

III. System Model

Our system model consists of a set of N users having each one communication service type. Within a practical scenario, one user could have eight different slices. Consequently, there is a one-to-M mapping between one user and the different slices, where $M_{max} = 8$ [13]. However, for the sake of simplicity, we consider a one-to-one mapping between one user and one slice. In other terms, a user has only one communication service type, and consequently is served by one slice only. As previously mentioned, two different types of communication service are present:

- *Ultra-Reliable Low-Latency Communication*: The URLLC is characterized by high reliability and stringent delay requirements. Usually such type of communication concern real-type applications, such as connected cars, emergency services, and connected medicine applications.
- *enhanced Mobile Broadband*: The eMBB is known for its extremely high demand in terms of throughput and a satisfaction that increases with the amount of granted resources. Such type of communication includes applications like augmented reality, Ultra High Definition (UHD) video streaming, and online gaming.

We consider the smallest spectral resource unit to be one Resource Block (RB). Therefore, the total bandwidth of a base station would be divided into N_{tot} RBs. The total resources are divided into two subsets, where each subset is allocated to one slice. We consider that n_{min} resources (*i.e*, RBs) are initially allocated for the eMBB slice which is identified by a S-NSSAI (*i.e*, slice identifier according to 3GPP terminology) value of '1', while the remaining part (*i.e*, $N_{RB} = N_{tot} - n_{min}$) are allocated for URLLC slice which is identified by a S-NSSAI value of '2'.

A. Mean Performance Indicator

Within the Static Scheme, the allocated subset of resources for each slice remains fixed. However, our model has more flexibility as a user of type URLLC can select the subset of resources initially allocated for the eMBB slice and vice versa if a different resource subset (*i.e*, slice) endows it with lower cost than the one allocated to its proper slice. It is worth noting that the URLLC communication service is characterized by high network reliability. In other terms, such service requires a 100% guarantee of minimum delay. Consequently, we assume that a user of URLLC type must have a delay equivalent to one RB. Any additional allocated RB will not increase its satisfaction, as such type of service does not require high throughput, as for the eMBB service.
Before expressing a user rate, let us define the different abbreviations for the sake of clarity:

- N_{URLLC}: is the number of users having S-NSSAI $= 2$
- N_{eMBB}: is the number of users having S-NSSAI $= 1$
- N: is the total number of users within the system. It is equal to $N_{URLLC} + N_{eMBB}$
- N_U: is the number of users that URLLC slice resources
- N_M: is the number of users that share the eMBB slice resources
- N_{RB}: is the number of physical RBs allocated for the URLLC slice
- n_{min}: is the number of physical RBs allocated for the eMBB slice
- N_{tot}: is the total number of RBs and is equal to $n_{min} + N_{RB}$

While the algorithm runs, N_U and N_M could be users having different S-NSSAI (*i.e*, 1 or 2). In other terms, they could be users having different service types. When the algorithm outputs a solution, a different number of RBs would be allocated for each slice. Thus, N_{RB} and n_{min} change.
In our system model resources are differently allocated over the bandwidth. In particular, N_U users that select the URLLC slice resources are granted 1 RB each. This case applies only when $N_U \leq N_{RB}$. Otherwise, URLLC resources are shared using a fair resource sharing scheme. For the N_M users that select the eMBB slice resources, a fair resource sharing scheme is always applied. As URLLC type users are interested with a prompt service but limited amount of resources (*i.e*, 1 RB) our model allocates the remaining RBs of the URLLC slice (*i.e*, $N_{RB} - N_U$) for eMBB slice whenever $N_U \leq N_{RB}$. Thus, an eMBB type user is motivated to select its natural slice (*i.e*,

eMBB slice) when the latter grants it high throughput (cf. subsection IV-B).

In our studies, we consider average radio conditions. In fact, the Dynamic RAN Slicing algorithm is not to be executed at the smallest time scale. Such problem concerns the bandwidth partitioning, rather than the RB allocation. Thus, we consider average radio condition metrics per RB. We also denote by C_{ap}^{k} the average rate achieved by one physical RB when average radio conditions are considered.

If a user k selects the eMBB slice, the rate Thr_k is calculated as follows:

$$Thr_k^{eMBB} = \begin{cases} C_{ap}^{k} \cdot \big(n_{min} + (N_{RB} - N + N_M)\big)/N_M \\ \text{if } N_U \leq N_{RB} \\ C_{ap}^{k} \cdot \big(n_{min}/N_M\big) \\ \text{Otherwhise} \end{cases}$$

Otherwise, if user k selects the URLLC slice, Thr_k is calculated as follows:

$$Thr_k^{URLLC} = \begin{cases} C_{ap}^{k} & \text{if } N_U \leq N_{RB} \\ C_{ap}^{k} \cdot \big(N_{RB}/N_U\big) & \text{Otherwhise} \end{cases}$$

Note that the performance indicator of an eMBB type user is expressed in terms of throughput, whereas the performance indicator of an URLLC type user is expressed in terms of delay. A delay Del_k of a user k is calculated as: $Del_k = 1/Thr_k$

IV. Crowding Game Framework for Resource Selection

A crowding game consists of a non-cooperative game, where players compete over a common resource. Hence, it is well suited for an efficient resource selection algorithm. In our case, the game is modeled as a multi-player game $\mathcal{G}$, where users are the players, whereas the subsets of allocated RBs for each slice are the resources. The framework of the game is presented as follows:

- The set $\mathcal{K}$ as the set of players (*i.e*, users)
- The set $\mathcal{S} = \{e, u\}$ as the set of strategies, where e designates the eMBB slice, whereas u coins the URLLC slice. An action of a user k consists of selecting one subset $s \in \mathcal{S}$
- The strategy of a user k is denoted as y_k and whose components are the binary variables $y_{k,s}$ which equate to 1 when user k chooses the slice s. Hence $\mathbf{y} = (y_k)_{k \in \mathcal{K}} \in \mathcal{S}$ is a pure strategy profile, and $\mathcal{S} = S_1 \times S_2 ... \times S_K$ is the space of all profiles
- A set of cost functions $\{C_1, C_2, ..., C_K\}$ that quantify the players' preferences over the possible outcomes of the game.

A. Cost Function

Each user seeks to minimize its own cost function by selecting an adequate strategy s (*i.e*, slice s). Note that a cost function is defined according to a user type (*i.e*, S-NSSAI). Particularly, the cost function of an eMBB type user is expressed in terms of throughput, while the cost of an URLLC type user is expressed in terms of delay (cf. subsection III-A).

When a user k selects the eMBB slice its cost function is expressed as follows:

$$C_k^{eMBB} = \begin{cases} \big(D_k/Thr_k^{eMBB}\big) + C_{eMBB} \\ \text{if user } k \text{ is of type eMBB} \\ \big(Del_k^{eMBB}/T_K\big) + C_{eMBB} \\ \text{if user } k \text{ is of type URLLC} \end{cases}$$

Otherwise, if user k selects the URLLC slice, its cost function would be expressed as:

$$C_k^{URLLC} = \begin{cases} (D_k/Thr_k^{URLLC}) + C_{URLLC} \\ \text{if user } k \text{ is of type eMBB} \\ \big(Del_k^{URLLC}/T_K\big) + C_{URLLC} \\ \text{if user } k \text{ is of type URLLC} \end{cases}$$

D_k expresses the comfort rate of an eMBB type user, while T_k expresses the comfort delay of an URLLC type user. C_{eMBB} expresses the cost of selecting the eMBB slices, while C_{URLLC} expresses the cost of selecting the URLLC slice. To enhance the model efficiency, the cost functions parameters must be well tuned in order to push the selfish users towards their most adequate strategies which satisfy their QoS requirement. Within the following, we detail the parameters choice to achieve such goal.

B. Tuning the Cost Function Parameters

D_k verifies $D_k \geq n.C_{ap}^{k}$, where n is an integer greater than 1. The reason of such a value is to push an eMBB type user to favor the slice that gives it more than one RBs (*i.e*, slice e) and thus, increases its throughput. Recall that C_{ap}^{k} is the rate achieved by one RB. We set T_k to $1/C_{ap}^{k}$ to give incentive to an URLLC type user to choose the strategy that endows it with 1 RB. Both C_{eMBB} and C_{URLLC} are greater than 0 and verify $C_{eMBB} \geq C_{URLLC}$. The reason of such constraint is to push the URLLC type users to select the slice u when the latter grants them one physical RB each, or in other terms, when $N_U \leq N_{RB}$. In fact, we want to grant an URLLC type user no more than 1 RB as giving away more RBs does not increase its satisfaction. In that way, more RBs would be available for eMBB type users that are in need of high throughput. However, when $N_U > N_{RB}$, Del_k/T_K would have a high value, which increases the cost of an URLLC type user in case the latter selects slice u. Thus, an URLLC user must endeavor to satisfy its reliability requirement by selecting the slice that grants it at least one physical RB (in such as case, slice e).

To summarize, when $N_U \leq N_{RB}$, Del_k/T_k would be small enough in slice u. In that way, an URLLC type user favors more slice u since $C_{eMBB} \geq C_{URLLC}$. On the other hand, when $N_U > N_{RB}$, Del_k/T_k significantly increases in slice u. Consequently, an URLLC type user favors the slice e that grants it at least one physical RB.

After the algorithm converges, N_{RB} and n_{min} are adjusted for each slice. In fact, the convergence of the algorithm is attained when the Pure Nash Equilibrium (PNE) is reached. Within the following subsection IV-C we describe how the PNE is reached and proof of its existence.

C. Attaining the Pure Nash Equilibrium

In a non-cooperative game, a solution is obtained when all players adhere to a Nash Equilibrium (NE). A NE is a profile of strategies in which no player will profit from deviating its strategy unilaterally. Hence, it is a strategy profile where each player's strategy is an optimal response to the other players' strategies.
Our game $\mathcal{G}$ is a finite game and in general such games are not guaranteed to have PNEs . Nevertheless, they possess a mixed NE where each player has to continually change its slice selection according to a distribution probability over the strategy set. However, our game has the Finite Improvement Path (FIP) property which is a valuable property. In fact, for such games, simple dynamics converge to PNE [14]. The common feature of such games is the existence of a potential function which represents commonly the quality of the various strategy profiles for all players [15]. The unilateral change of one user strategy results in a change of its cost function that is equal to the change of the so-called potential function.

Proposition 1 *The game $\mathcal{G}$ has the FIP property.*

PROOF The game $\mathcal{G}$ is an unweighted crowding game, as it is a normal-form game in which users share a common set of actions and the cost function of user k for choosing a particular action is player specific and a non-increasing function of the total number of users choosing that same action. Unweighted crowding games have PNE. Furthermore, when players have only two strategies (*i.e*, e and u), the game has the Finite Improvement Path (FIP) property and hence a Best Response [16] algorithm permits attaining the PNE of the game [17].

The Best Response algorithm is presented as follows:

1) Best Response algorithm: : At each round of the Best Response, a user k goes through all available strategies and chooses the strategy (*i.e*, $s \in \mathcal{S}$) that provides it the lowest cost. Convergence is attained when all users choose the same strategies as in the previous round. We denote by $a_{k\in\mathcal{K}}(i)$ and $y_{k\in\mathcal{K}}(i)$ the action and the strategy of user k at round i respectively. The behavior of the Best Response is described in algorithm 1:

Initialize all vectors $y_{k\in\mathcal{K}}(0)$ of pure strategies;
repeat
  Each user goes through all strategies of set $\mathcal{S}$ and chooses a startegy s^* that respects $s^* = \text{argmin}_{s\in\mathcal{S}} C_k$;
  Actions $a_{k\in\mathcal{K}}(i)$ of each user are updated;
until *Attaining PNE*;

Algorithm 1: Stage 1: Best Response

Within the following, we analyze the amount of time the Best Response needs to converge. Additionally, we discuss a scenario for implementing the algorithm in the O-RAN architecture.

V. Algorithm Implementation in O-RAN Architecture

The O-RAN architecture involves several entities. We describe hereafter two main ones as we consider that our algorithm would be implemented at their level:

- The non-Real time RAN Intelligent Controller (non-Real-Time RIC): The non-real-Time RIC consists of a virtual entity offering the possibility to manage the radio resources in non-real-time (*i.e*, within a time scale larger than 1 s). The non-Real-Time RIC bases on Artificial Intelligence and machine learning algorithms to set RAN optimization policies.
- The near-real time RAN Intelligent Controller (near-Real-Time RIC): The near-Real-Time RIC consists of a virtual entity that offers the possibility to manage the radio resources in near-real-time (*i.e*, within an interval of 10 ms to 1 s). The near-real-time RIC operates on user and cell-specific metrics.

In fact, non-real-time policy sets are executed at the level of the near-real-time RIC which enriches the non-real-time RIC by specific KPIs used to feed the machine learning algorithms so as to output efficient optimization policies based on long-term measurements [18]. In the case of our proposed solution, the Best Response algorithm is to be executed at the level of the near-real-time RIC since convergence is attained fastly (cf. figure 4). In that case, a user selects a slice within one slot (0.5 ms in the case of LTE) and reports its KPI in terms of throughput or delay to the near-real-time RIC which is in charge of performing the cost calculation for all users and registering their adequate strategies in each round. One user needs S slots to select all strategies, where S is the total number of strategies. Consequently, all users need $N \times S$ to finish one round of the Best Response algorithm. Thus, a solution is obtained in maximum $N \times S \times R_{max}$ slots, where R_{max} is the maximum number of rounds for convergence. Our algorithm presents a very low value of R_{max} (cf. figure 4). Consequently, when N and S have reasonable values, the algorithm converges within a time interval of 10 ms to 1 s, which corresponds to the interval time of the near-real time RIC. However, when a reinforcement learning algorithm is employed to reach convergence, the algorithm converges at a time scale larger than 1 s, since R_{max} would be significantly high. In that case, KPIs have to be reported to the non-real-time RIC, where cost calculations and decisions are performed and transmitted over the A1 interface connecting the non-real-time RIC to the near-real-time RIC which has the charge of decisions execution.

VI. Performance Evaluation

We use Matlab for simulations and evaluate the different metrics with respect to the number of URLLC type users denoted as N_{URLLC}. The different metrics used to evaluate the results are the following:

- The URLLC Service Reliability which is equal to 100% if all URLLC users are served by their minimum delay

requirement, in other terms when all URLLC type users are guaranteed to have at least 1 RB in any scheduling epoch (*i.e*, $N_{URLLC} \leq N_{RB}$). Otherwise, the Service Reliability would be expressed as follows:

$$\frac{\text{Number of RBs allocated for URLLC slice}}{N_{URLLC}} \times 100$$

In that case (*i.e*, $N_{URLLC} > N_{RB}$), the above measurement reflects the percentage of time a user has its required delay as a fair Resource time sharing is applied.

- The Total Throughput of eMBB type Users
- The Resource Utilization Efficiency of URLLC Slice
- The Total Number of Rounds of the Best Response algorithm

The simulation parameters are shown in table I.

TABLE I
SIMULATION PARAMETERS

Parameter	Description	Value
C_{ap}^{k}	Average radio conditions per RB	1 Mb/s
N_{tot}	Total number of RBs	50 RB
N	Total number of users	40 users
D_k	The comfort rate of a eMBB type user	$2 \cdot C_{ap}^{k}$
T_k	The comfort delay of a URLLC type user	$1/C_{ap}^{k}$
C_{eMBB}	The cost of selecting the eMBB slice	1.375
C_{URLLC}	The cost of selecting the URLLC slice	1

$N_{RB} = 20$ RB are initially allocated for the URLLC slice and $n_{min} = 30$ RB for the eMBB slice. Within the Static Scheme N_{RB} and n_{min}, are fixed and do not change.

We start by displaying the URLLC Service Reliability with respect to N_{URLLC}. Recall that $N_{eMBB} = N - N_{URLLC}$. We notice that the Dynamic RAN Slicing Scheme is showing a 100% of service reliability for any number of URLLC type users (*i.e*, N_{URLLC}). This is due to the adaptability of the slice resource allocation when service load variation occurs. It also highlights the efficiency of the proposed algorithm as it is able to offer a 100% reliability for any URLLC load conditions, while having a high resource utilization rate (cf. figure 3). However, the Static Scheme shows a 100% of reliability only when $N_{URLLC} \leq 20$. In fact, the initial subset of resources allocated for the URLLC slice consists of 20 RBs. Consequently, the Static Scheme guarantees the minimum required delay for an URLLC type user since the minimum delay requirement for one user is equivalent to 1 RB. When $N_{URLLC} > 20$, the Static Scheme shows a degradation in this metric as there is not enough resources to satisfy the required QoS. When N_{URLLC} reaches 40, we notice a reliability value that plummets to 50%. In this case 40 URLLC users are sharing 20 RBs. Consequently, each user is guaranteed its minimum required delay half of the time.

Figure 2 shows the Total Throughput of eMBB type Users with respect to N_{URLLC}. When $N_{URLLC} \leq 20$, the total throughput achieved by the Dynamic RAN Slicing Scheme is higher than the one achieved by the Static Scheme. The reason is that all of the spared resources of the URLLC slice are allocated for the eMBB slice. However, when $N_{URLLC} > 20$, the Static Scheme realizes more throughput. In that case, a part

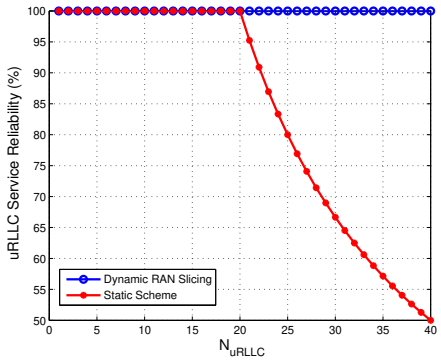

Fig. 1. URLLC Service Reliability (%)

of the eMBB slice resources would be allocated for URLLC type users. Recall that an URLLC cost function is tuned so as to push users to select the slice that grants them at least one physical RB in case of shortage (cf. subsection IV-B). Consequently, the resource allocation is adjusted in a way to satisfy the increased demand of URLLC service. Contrarily, within the Static Scheme the eMBB resources are only occupied by eMBB type users regardless of the value of N_{URLLC}. This comes at the detriment of the URLLC Service Reliability (cf. figure 1). We also notice that the throughput linearly decreases for the Dynamic RAN Slicing Scheme. In fact, when N_{URLLC} increases, one additional RB would be occupied by a user of type URLLC, which decrements the number of available RBs for the eMBB slice. Again, Such behavior highlights the adaptability of our proposed scheme, which achieves an efficient compromise between two different service types. For both schemes, the total throughput is 0 when N_{URLLC} reaches 40. In this case, N_{eMBB} would be null.

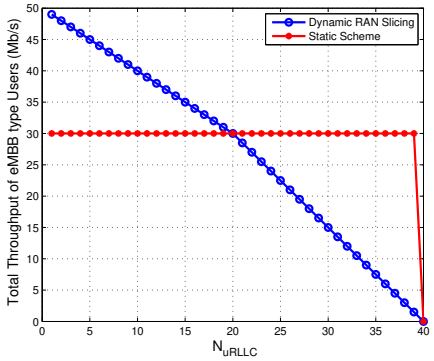

Fig. 2. Total Throughput of eMBB type Users (Mb/s)

Figure 3 displays the Resource Utilization efficiency of the URLLC Slice with respect to N_{URLLC}. Within the Dynamic RAN Slicing Scheme, the utilization efficiency of the URLLC resources is 100% for any value of N_{URLLC}. The reason is that all users of type URLLC are occupying all RBs allocated for the slice. This highlights the efficiency of our proposition that allocates the right amount of URLLC resources to satisfy the

QoS requirements. However, for the Static Scheme, we notice an under-utilization of the URLLC resources when $N_{URLLC} \leq 20$ and an over-utilization when $N_{URLLC} > 20$. Recall that a fixed number of RBs is allocated per service in the case of Static Scheme. Consequently, resources would be over-provisioned in case of low N_{URLLC} values (*i.e*, $N_{URLLC} \leq 20$), and under-provisioned for high N_{URLLC} values (*i.e*, $N_{URLLC} > 20$).

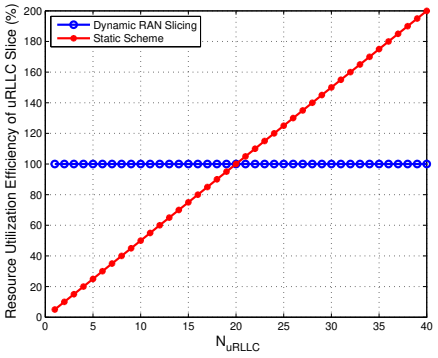

Fig. 3. Resource Utilization Efficiency of URLLC Slice (%)

Figure 4 shows the Total Number of Rounds of the Best Response Scheme to converge to PNE with respect to N_{URLLC}. For $N_{URLLC} \leq 20$, one round is enough to attain the convergence. In fact, the initial conditions of the Best Response algorithm are set in a way to allocate any spared resource from URLLC slice to eMBB users. Owing to such reasonable initial setting, the PNE is attained in only one round. When $N_{URLLC} > 20$, the PNE is attained in two rounds. In fact, URLLC users type need an additional round to select the eMBB resources when their minimum delay is not guaranteed. Note that many PNEs exist. In fact, when the initial resource distribution is different, a different solution is obtained. However, the initial conditions are set in a way to achieve good results with very small amount of time.

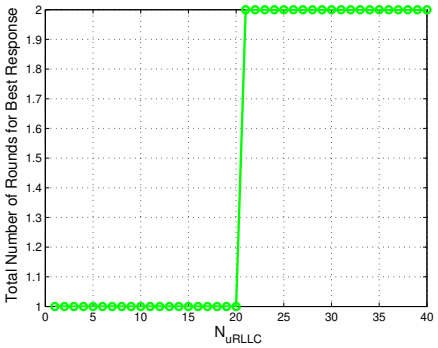

Fig. 4. Number of Rounds of Best Response

VII. Conclusion

In this paper, the Dynamic RAN Slicing problem is addressed and formulated as a crowding game. The objective is to let different type users select their most convenient resources in a way to achieve their different QoS requirements while improving the utilization efficiency of spectral resources. Cost functions for different service types are proposed and well tuned in order to push users to select their most appropriate strategies in terms of QoS. Results are compared with the Static Scheme where the bandwidth is statically partitioned among the slices, and show that our proposed solution overcomes the Static Scheme in terms of QoS and utilization efficiency. Additionally, the time convergence of our algorithm is analyzed and assessed which allows us to propose a practical scenario of algorithm implementation in the case of O-RAN. For future work, we aim to implement our proposed algorithm in a real O-RAN platform. We also aim to study the problem in a multi-cellular context, where inter-slice interference are in present and might violate the isolation.

References

[1] "Study on radio access network (ran) sharing enhancements," 3GPP, Tech. Rep. 22.852, Jun 2017.
[2] I. Milchtaich, "Crowding games are sequentially solvable," *Int. J. Game Theory*, 1998.
[3] "O-ran architecture description," O-RAN Alliance, Tech. Rep. v02.00, 2020.
[4] H.-T. Chien, Y.-D. Lin, C.-L. Lai, and C.-T. Wang, "End-to-end slicing as a service with computing and communication resource allocation for multi-tenant 5g systems," *IEEE Wireless Communications*, 2019.
[5] D. Chekired, M. A. Togou, L. Khoukhi, and A. Ksentini, "5g-slicing-enabled scalable sdn core network: Toward an ultra-low latency of autonomous driving service," *IEEE Journal on Selected Areas in Communications*, 2019.
[6] S. E. Elayoubi, S. B. Jemaa, Z. Altman, and A. Galindo-Serrano, "5g ran slicing for verticals: Enablers and challenges," *Comm. Mag.*, Jan. 2019.
[7] R. Mahindra, M. A. Khojastepour, H. Zhang, and S. Rangarajan, "Radio access network sharing in cellular networks," in *2013 21st IEEE International Conference on Network Protocols (ICNP)*, 2013.
[8] P. Caballero, A. Banchs, G. de Veciana, and X. Costa-Pérez, "Multi-tenant radio access network slicing: Statistical multiplexing of spatial loads," *IEEE/ACM Transactions on Networking*, 2017.
[9] J. Kwak, J. Moon, H.-W. Lee, and L. B. Le, "Dynamic network slicing and resource allocation for heterogeneous wireless services," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 2017.
[10] P. Caballero, A. Banchs, G. de Veciana, X. Costa-Pérez, and A. Azcorra, "Network slicing for guaranteed rate services: Admission control and resource allocation games," *IEEE Transactions on Wireless Communications*, 2018.
[11] S. D'Oro, F. Restuccia, A. Talamonti, and T. Melodia, "The slice is served: Enforcing radio access network slicing in virtualized 5g systems," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019.
[12] M. Zambianco and G. Verticale, "Interference minimization in 5g physical-layer network slicing," *IEEE Transactions on Communications*, 2020.
[13] "Study on enhancement of radio access network (ran) slicing for nr," 3GPP, Tech. Rep. 38.832.
[14] D. Monderer and L. S. Shapley, "Potential games," *Games and Economic Behavior*, 1996.
[15] R. W. Rosenthal, "A class of games possessing pure-strategy nash equilibria," 1973.
[16] M. Feldman, Y. Snappir, and T. Tamir, "The efficiency of best-response dynamics," in *Algorithmic Game Theory*, V. Bilò and M. Flammini, Eds. Springer International Publishing, 2017.
[17] I. Milchtaich, "Congestion games with player-specific payoff functions," *Games and Economic Behavior*, 1996.
[18] "O-ran working group 1 use cases analysis report," O-RAN Alliance, Tech. Rep. v05.00.03, 2021.