IEEE Open Journal of the
Communications Society

Received XX Month, XXXX; revised XX Month, XXXX; accepted XX Month, XXXX; Date of publication XX Month, XXXX; date of current version XX Month, XXXX.

Digital Object Identifier 10.1109/OJCOMS.2022.1234567

Management Closed Control Loop Automation for Improved RAN Slice Performance by Subslicing

MARIKA KULMAR*, MUHAMMAD MAHTAB ALAM* SENIOR MEMBER, IEEE, AND IVO MÜÜRSEPP*

[1]Thomas Johann Seebeck Department of Electronics, Tallinn University of Technology, Tallinn, Estonia.

CORRESPONDING AUTHOR: Marika Kulmar (e-mail: marika.kulmar@ taltehc.ee).

This work has received funding partly from the European Union's Horizon 2020 Research and Innovation Program under Grant 951867 '5G-ROUTES' and Grant 101058505 '5G-TIMBER'. This work in the project "ICT programme" was supported by the European Union through European Social Fund, and TAR16013 Center of Excellence 'EXCITE IT'.

ABSTRACT Network slicing offers the potential to enhance service satisfaction and optimize resource utilization, particularly in scenarios where radio resources are limited. Subslicing has been shown to improve the slice performance. In this paper, the monitor-analyze-plan-execute-knowledge (MAPE-K)-type management closed control loop (MCCL) is implemented for slice performance improvement by subslicing. The subslicing can improve slice performance if the slice performance depends on the size of slice bandwidth part (BWP). For Plan function, the classifier neural network was trained to decide whether the subslice should be split, merged or not changed by their performance. The training data contains slice performance data of all possible subslice sizes. For Execute function, the subslice splitting algorithm was proposed, which clusters UEs by their block error ratio (BLER) and allocates bandwidth proportionally to group requested sum rate and group BLER. A realistic 5G new radio (NR) band serving a set of user equipments (UE) of different values of their BLER and requested rates was a setup of radio access network (RAN) slice simulated using MATLAB R2021b. Subslicing has reduced bandwidth utilization, and slice BLER while increased slice goodput (application-level throughput). Proposed subslice splitting algorithm when UEs are clustered by their achieved BLER, then the slice BLER reduces additional 20% and slice goodput increases up to additional 9% compared to no subslicing when UEs are clustered by their requested rates. This effect was larger for the uplink. In runtime scenarios for poor-BLER UEs the smaller subslices improve slice utilization and BLER, while larger subslices improve goodput.

INDEX TERMS 5G New Radio simulation, performance management, radio access network subslicing, reconfiguration automation

---

I. INTRODUCTION

Closed-loop network management plays a crucial role in meeting the growing demands for mobile communication, ensuring that cellular networks operate efficiently and effectively to provide a seamless experience for users. This automated management approach is often referred to as self-organizing networks (SON), zero-touch network and service management (ZSM), and management and orchestration (MANO). SON, in particular, relies on closed control loops (CCL) within the domains of self-configuration, self-optimization, and self-healing [1]. This innovation was first introduced for the management of cellular networks starting from 3GPP release 8 [2]. While SON primarily concentrates on automating cellular network-specific functions like configuration, optimization, and healing, ZSM extends these principles to a broader range of network and service management activities. MANO is a framework and set of functions used to manage and orchestrate the various components and services in a virtualized network environment. 3GPP has specified management closed control loop in TS 28.535 [3]. It is an automated process that continuously monitors, analyzes, and adjusts various aspects of a network to maintain or improve its performance, quality, and efficiency. It operates in real-time or near-real-time, allowing for rapid responses to changing network conditions.

3GPP has specified a life cycle of network slice in TS 28.530 [4]. During the slice modification phase, managing slice performance involves two ways: the slice provision system can admit the number of UEs for which the existing resources are sufficient; the other way is to monitor performance and react if slice overload occurs with allocation of more resources or removing UEs.

Subslicing has been shown to improve slice performance in [5] and [6]. If the slice performance depends on slice size, then subslicing a slice into suitable-sized subslices can improve the slice performance. This is applicable in close to slice overload situation when no additional bandwidth available, that is fixed-bandwidth slice or bandwidth constrained slice. In this paper we implement the monitor-analyze-plan-execute-knowledge (MAPE-K) [7] management CCL (MCCL) to introduce subslicing with the aim to improve slice performance on fixed slice bandwidth.

The contributions of this paper are as follows:

- MAPE-K CCL exists, but not for automatic subslicing. We describe it for automatic subslicing with the goal to improve RAN slice performance on fixed slice bandwidth.
- We propose a Plan function that decides based on subslice performance whether the subslice should be split, merged, or not changed. The subslice with the best size has achieved the lowest utilization, highest goodput per allocated RB, and lowest BLER.
- We propose an Execute function, where in the subslice splitting algorithm, the UEs are clustered by their achieved BLER and the slice bandwidth is allocated proportionally to the group BLER and requested sum rate. This improves the slice performance more than if the UEs are clustered by their requested rate.
- The slice is simulated and its performance is evaluated when MCCL-initiated subslicing is performed during the initialization phase and traffic increase and decrease scenarios. Slice performance improvement is shown by a reduction in slice utilization and BLER with an increase in slice goodput per RB.

The remainder of this paper is organized as follows. In Section II, the related work on management CCLs and subslicing described. The performance data at all possible subslice sizes, which is the input data for subslicing decision ML tool, is presented in Section III. The proposed MCCL is described in Section IV and evaluated in Section V. Finally, Section VI concludes the paper.

II. RELATED WORK

For automation in management, the closed control loop (CCL) operates autonomously to attain predefined objectives without human intervention. The loop consists of four functions and a shared database of knowledge. These functions collect and analyze data, make decisions and execute decided actions on the managed entity. Examples of well-known CCLs are MAPE-K (Monitor-Analyze-Plan-Execute, Knowledge) and OODA (Observe, Orient, Decide, Act) [1].

MAPE-K [7] is a management CCL used by autonomic systems for self-management. It functions in a series of steps: the Monitor function gathers performance data, the Analyze function analyzes this data, the Plan function decides changes in configuration, and the Execute function implements these changes to the configuration of the autonomic system. The Knowledge is a database containing information accessible for all other functions.

Vision of closed loop automation in [8] uses MAPE loop on O-RAN architecture for end-to-end (E2E) slice orchestration and network management. The CCL framework described in [9] contains policy-driven CCL and uses intent-based networking (IBN) to enable ZSM in a multidomain environment. It covers service management model, and presents policy generation algorithms for policy generation. [10] presents framework of collaborating MAPE-K closed loops used for end-to-end service management.

CCL types can be categorized as ready-made or made-to-order CCLs. Ready-made CCLs are pre-integrated and made-to-order CCLs are assembled on demand [1]. Our proposed MCCL can be considered as a made-to-order CCL designed to be assembled specifically for managing radio resources in the mIoT slice when a slice is in the close to overload state and no additional bandwidth resources are available.

[11] proposes AI-driven MANO system for massive slicing. They use their own CCL, which contains monitoring, analytics and decision phases, while execution is outside CCL. The monitoring and analytics components predict the RAN resources under service level agreement (SLA) constraints. The federated learning is used to improve the prediction model. The system overhead and computation load reduced, as well as slice SLA violation rate.

While we propose a CCL implementation for subslicing, the full CCL is implemented in [12] to mitigate connection loss for moving UE. UEs are ships in real-world seaport testbed. The movement of UE is predicted using deep neural network (DNN) consisting of 3 hidden layers, and radio link failure predicted. In addition, the power of the beam increased and beam down-tilted to avoid actual radio link failure.

Network slicing monitoring framework in [13] was used to measure the effect of polling interval to consumption of computing and storage resources. The slice-specific data collectors use more resources than if data collection servers used. The smaller polling interval increases resource usage, except storage resource consumption remains the same if slice-specific data collectors used. The polling intervals of 5 s and 1 s were evaluated, however, in [14] the KPIs supposed to be monitored over 30 s period.

Data analytics is discussed in [15]. For RAN the relevant analytics can be used for interference handling, channel quality prediction, control of dynamic radio topology, energy efficient improvements, multi-slice resource management enhancement. Types of analytics are: descriptive, diagnostic, predictive, and prescriptive.

The subslicing in [5] is done to group UEs with similar features into one subslice inside the slice. The purpose of subslicing was to better satisfy the requirements of UEs by clustering UEs by UE features into subslices. They evaluated clustering quality to determine the best number of subslices. They claim that slice performance in terms of throughput, power consumption and energy efficiency have been improved when slice had subslices.

CCL has been used for automatic network configuration adaptation to changes in service requirements or achieved radio signal quality. The slice performance dependence on slice BWP size is not researched. If the performance of the RAN depends on the BWP size of the RAN, the automatic change in the BWP size is not considered. The management CCL has not been used for RAN subslicing with the aim of improving slice performance under bandwidth constraints. In this paper we fill these gaps and contribute to RAN subslicing with slice performance improvement, and propose Plan and Execute functions for management CCL for automatic subslicing on fixed slice bandwidth.

III. SUBSLICE PERFORMANCE DEPENDENCE ON SUBSLICE SIZE (TRAINING DATA)

First, we discover and present the subslice performance dependence from the subslice size by simulating the subslice of all possible sizes and collecting its performance data. This is needed for decisions in which subslice sizes provide better performance and which subslices should be split or merged to improve the performance of their UEs.

The subslice works on 5G band n28 (FDD, UL 730-748 MHz, DL 758-803 MHz) [16], subcarrier spacing is 15 kHz thus maximum subslice bandwidth is 250 RBs. The band n28 can be used for both machine-to-machine (M2M) internet of things (IoT) and vehicle to everything (V2X) service verticals.

For each RB allocated to the subslice, a UE, which is able to consume that RB, is admitted to the subslice. The UE requests rates 200 kbps both in UL and in DL, and packet size used is 40 bytes. The subslice is simulated three times: if it contains all good-BLER, medium-BLER and poor-BLER UEs. To achieve the desired BLER, the UE distance from gNB should be set accordingly. BLER is assumed to increase with distance; thus, UEs that should achieve poor BLER are positioned far from the gNB. The used channel model is clustered delay line C (CDL-C), which is non line of sight (NLOS) model [18]. The smallest subslice to simulate is 4 RBs, because into smaller BWPs the sounding reference signal (SRS) necessary for channel estimation and scheduling, can not be fitted [19]. Other settings are provided in table 1.

TABLE 1. **MATLAB Toolbox settings**

Parameter	**Value**
Band	n28 (FDD, UL 730-748 MHz, DL 758-803 MHz)
Carrier frequency	UL 730 MHz, DL 758 MHz
Channel model (for both UL and DL)	CDL-C
PUSCH preparation time for UEs	200 μs
Logical channels per UE	1
RLC entity type	UM bidirectional
Duplex mode	FDD
Scheduler strategy	Round Robin
Length of scheduling cycle	1 frame
RB allocation limit UL	same as RBs for subslice
RB allocation limit DL	same as RBs for subslice
Simulation time	1 second
Subslice simulation tool from MATLAB 5G Toolbox	NR Cell Performance Evaluation with Physical Layer Integration [17] R2021b

Six key performance indicators (KPI) are measured for each subslice: bandwidth utilization in UL (utilUL) and DL (utilDL), subslice goodput (application-level throughput) in Mbps per one RB in UL (gdp1UL) and DL (gdp1DL), and the subslice average block error ratio (BLER) in UL (blerUL) and DL (blerDL). The total goodput is received, but to compare the results of subslices of different size, the goodput per allocated RB is calculated. One second of working time is simulated using Matlab 5G toolbox tool called NR Cell Performance Evaluation with Physical Layer Integration [17]. Results are shown in figure 1.

In utilUL there are low peaks on size of 4, 11, and 19 RBs. In the range of 37-73 RBs the utilization is lower. Larger subslices have 100% utilization. If subslices are smaller than 37 RBs, the goodput is low. In UL there is the range of 37-73 RBs when utilUL is lower and gdp1UL is better than in smaller subslices. Subslices larger than 37 RBs, downlink utilization and goodput are both high. In UL, both utilization and goodput are high if subslice is larger than 73 RBs.

BLER is initially high, then in subslice size of 10 RBs BLER is low, because there is low goodput. Later, BLER increases with subslice size increase. For small subslices, the BLER is high because of incorrect channel estimation caused by too less reference symbols in the subband.

In UL, the subslice size range 37-73 RBs is the lowest utilUL and low BLER. The highest goodput in DL is in range of subslice size 73-145 RBs and in DL 37-145 RBs.

With medium-BLER and poor-BLER UEs the utilization was higher than with good-BLER UEs. Goodput was the highest when subslice size was smaller than when good-BLER UEs were in the subslice. Because the BLER is high, packet retransmissions are necessary, which increases

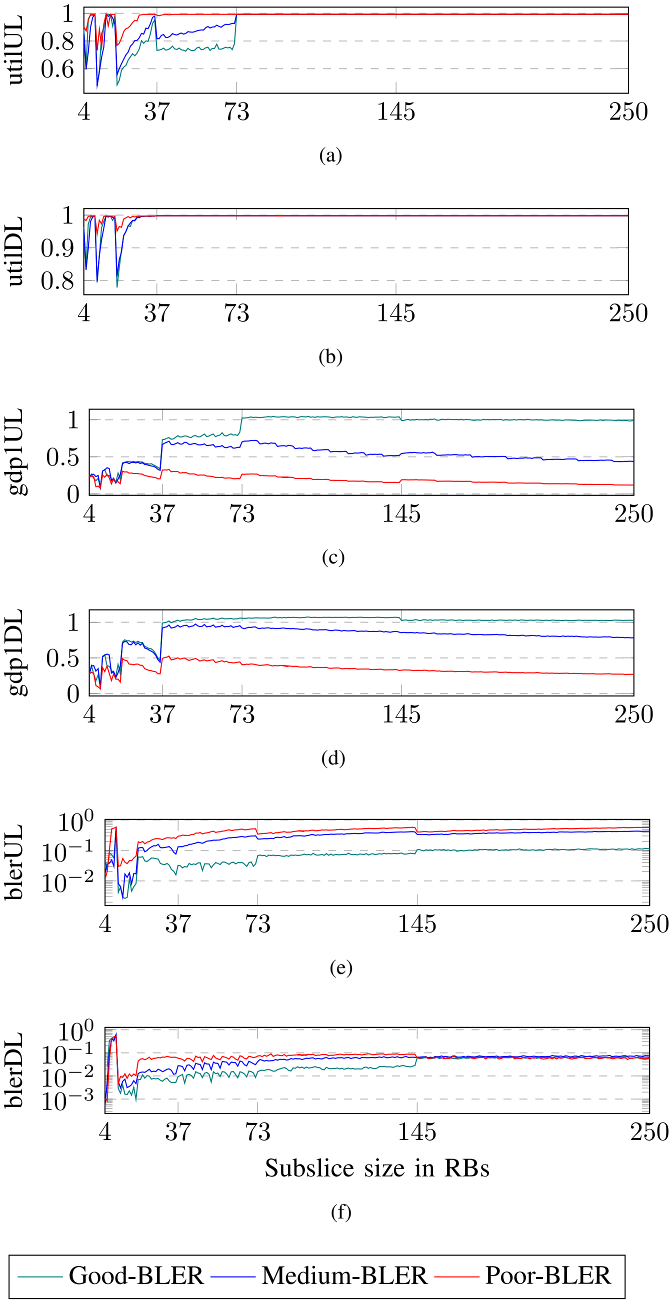


FIGURE 1. **Subslice performance data for subslice size.**

the bandwidth utilization to full, and because of bandwidth shortage, the goodput will be low. Additional bandwidth without additional UEs can improve the performance of the subslice. If no additional bandwidth is available, the selection of the best subslice size can improve the performance achieved by the UE or RB.

IV. PROPOSED MANAGEMENT CCL

A. THE LOOP

The proposed CCL is the MAPE-K type and is similar to the CCL specified in 3GPP TS 28.535 [3]. It consists of four functions. Monitor collects performance KPIs. Analyze function detects if subslicing is necessary. To obtain decisions, the optimization problem is solved to provide training data for neural network (NN). Plan (Decide) function uses this classifier NN to decide whether the subslice should be split, merged or not changed. Execute creates subslices according to planned configuration. The Knowledge is a shared database. Full CCL is shown in figure 2.

Currently, the CCL is reactive type, but it can be proactive, if values of KPIs are predicted.

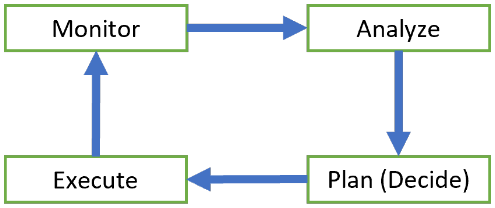


FIGURE 2. **MAPE-K-type management CCL.**

B. MONITOR

The Monitor function collects values of the six KPIs for the slice and each subslice it may have. In addition to slice bandwidth utilization, the goodput per one RB in Mbps and BLER is collected. Mean values of KPIs are available for other functions of the MCCL in the Knowledge database. Now, values are calculated for the time after execute function has changed the slice configuration.

C. ANALYZE

The poor performance of a slice or subslice is indicated by high utilization and low goodput and high BLER. Authors of [14] suggest that the high and low thresholds for bandwidth utilization can be 80% and 20%. For BLER the target value is below 0.1.

The Analyze function provides the pre-trained neural network (NN) for Decide function. Training data is the simulation results of six KPI values presented in section III. The subslice performance at different subslice sizes needs to be learned if subslice contains all good-BLER, medium-BLER and poor-BLER UEs. Then it can work better for slice which contains mixed-BLER UEs. In our previous work [20], the NN was trained using clustering result of a subslice performance data of all possible sizes of a subslice, which contained all good-BLER UEs. This approach does not work correctly in more realistic situation with mixed-BLER UEs in the slice. If poor-BLER UEs are in the slice, the slice utilization is high and goodput is low, then operation ”merge” was decided. However, our other previous work has shown that smaller subslices performed better, if UE BLER was worse [6].

To find the respective output decision (”merge”, ”no change”, ”split”), the optimization problem is solved separately for good-BLER, medium-BLER and poor-BLER simulation results.

1) Optimization problem to find best subslice size

The best size for subslice is with the lowest utilization, highest goodput per allocated RB, and lowest BLER.

Let x be the optimal size of a subslice in RBs.The relevant predictors/KPIs K for good subslice size are utilization $K^{(u)}$ in UL, $K^{(u,UL)}$ and DL $K^{(u,DL)}$, goodput per one RB $K^{(g)}$ in UL $K^{(g,UL)}$ and in DL $K^{(g,DL)}$, block error ratio (BLER) $K^{(b)}$ in UL, $K^{(b,UL)}$ and in DL $K^{(b,DL)}$.

The goal is to find a subslice size where the utilization is the lowest, goodput per one RB is the highest and BLER is the lowest. The multi-objective integer optimization problem can be formulated as:

$$\begin{cases} \min K^{(u)}(x), \\ \max K^{(g)}(x), \\ \min K^{(b)}(x). \end{cases}$$
$$\text{subject to} \quad \begin{cases} K^{(u)}(x) \in [0,1], \\ K^{(g)}(x) > 0, \\ K^{(b)}(x) \in [0,1], \\ x \in \{4, 250\}. \end{cases} \tag{1}$$

This optimization problem needs to be solved for each dataset separately for subslices containing good-BLER, medium-BLER and poor-BLER UEs.

Instead of using multiple functions, the objective function can be formulated as the sum of differences between KPI values and the best (ideal) KPI values of the dataset. The best values of KPIs are calculated for each of 3 datasets separately as follows

$$\begin{aligned} K^{(u,best)} &= \min\nolimits_{i=4,\ldots,250} K_i^{(u)}, \\ K^{(g,best)} &= \max\nolimits_{i=4,\ldots,250} K_i^{(g)}, \\ K^{(b,best)} &= \min\nolimits_{i=4,\ldots,250} K_i^{(b)}. \end{aligned} \tag{2}$$

The difference, ΔK, for each subslice size can be achieved by calculation of root-mean-square error (RMSE) between the current and best values of KPIs using the following formula:

$$\Delta K^{(j)} = \sqrt{(K_i^{(j)} - K^{(j,best)})^2}, i = 4, \ldots, 250, j = 1, \ldots, m. \tag{3}$$

The objective function is to minimize the sum of RMSEs of utilization, goodput and BLER from the best values of utilization, goodput and BLER.

Objective function can be written as:

$$\min_{N^{(RB)}} \sum_{j=1}^{m} \Delta K^{(j)}, \tag{4}$$

where the number of KPIs is $m = 6$.

The objective value a for each subslice i is calculated

$$a_i = \sum_{j=1}^{m} \Delta K^{(j)}. \tag{5}$$

The optimization problem is solved for subslice size performance data presented in section III. The objective values for each subslice size are calculated using equation 5 and results are plotted in figure 3.

The size of a subslice with minimum objective value is the best size for a subslice. Smaller subslices will have decision “merge”. Other subslices with a little greater objective value could also be in a good size and have the decision “no change“. The ε-neighbourhood is considered as percentage from the range of objective value shown on the vertical axis of the plots in figure 3. The “merge” zone is on the left side until the subslice size with the minimum objective value, which is shown in the horizontal axis. The “split” zone has bounds on horizontal axis starting the best subslice size until maximum subslice size, if objective values are higher than a “tolerance” threshold denoted by ε-neighbourhood. The good-BLER subslices have larger ”merge” area, because the best subslice size is 52 RBs (3a), while the poor-BLER subslices have subslices to be merged at sizes below 10 RBs (3c). If ε-neighbourhood is larger, then more subslices are in good size and can have “no change” decisions. Thus, less subslices needed to split. More “no change“ decisions mean less reconfiguration, however it may miss better performance which can be achieved by splitting or merging of subslices.

This formulation of the objective function and calculation of the objective value enables to map the decisions to subslice performance if the subslice size vs subslice performance is different from our dataset. The subslices which have small objective values calculated from their KPIs, have good performance and their size is suitable. The objective value was calculated as a RMSE between actual and best values of KPIs of the dataset which contained the UEs in the same BLER group.

The training data for NN is somewhat different for both ε values and output decisions for subslices at all sizes and all BLER groups are shown in figure 4. NN will get the values of six KPIs of the subslice sizes as the inputs and the output decision for the subslice sizes.

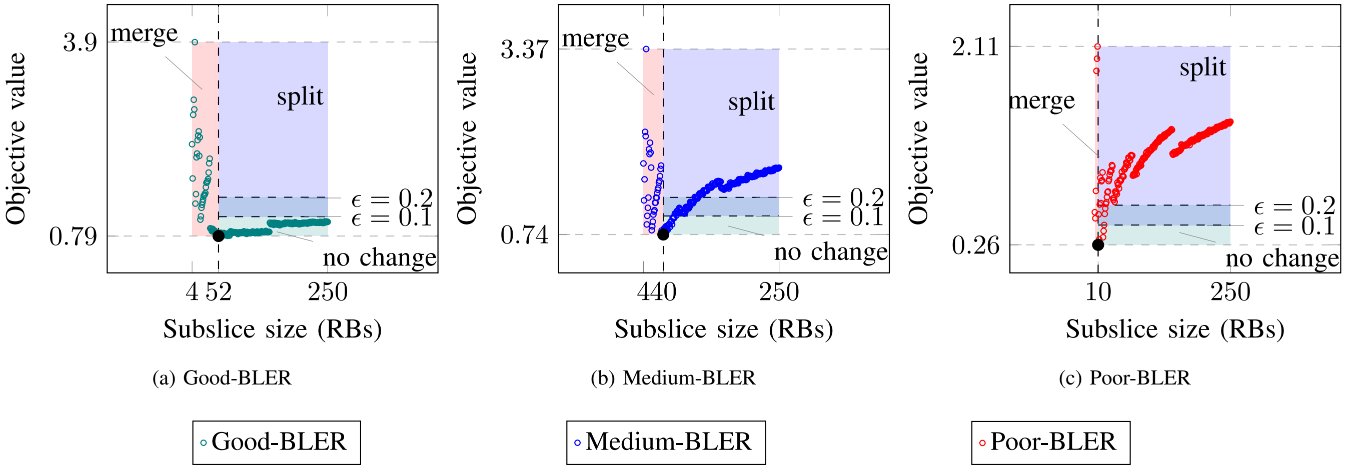
D. PLAN (DECIDE)

The “Decide” function in the MCCL cycle contains a classifier neural network (NN) which decides the slice operation, whether it should be merged, split or not changed. The training data is the simulation results, 6 KPIs, with the output class determined by solving the optimization problems for each BLER group.

The NN is trained using three datasets which have good-BLER, medium-BLER and poor-BLER UEs in the subslice. The NN inputs are values of six KPIs and not the subslice sizes nor UE BLERs. Then we assume that the NN is trained for the decisions, if the slice contains UEs with any BLER.

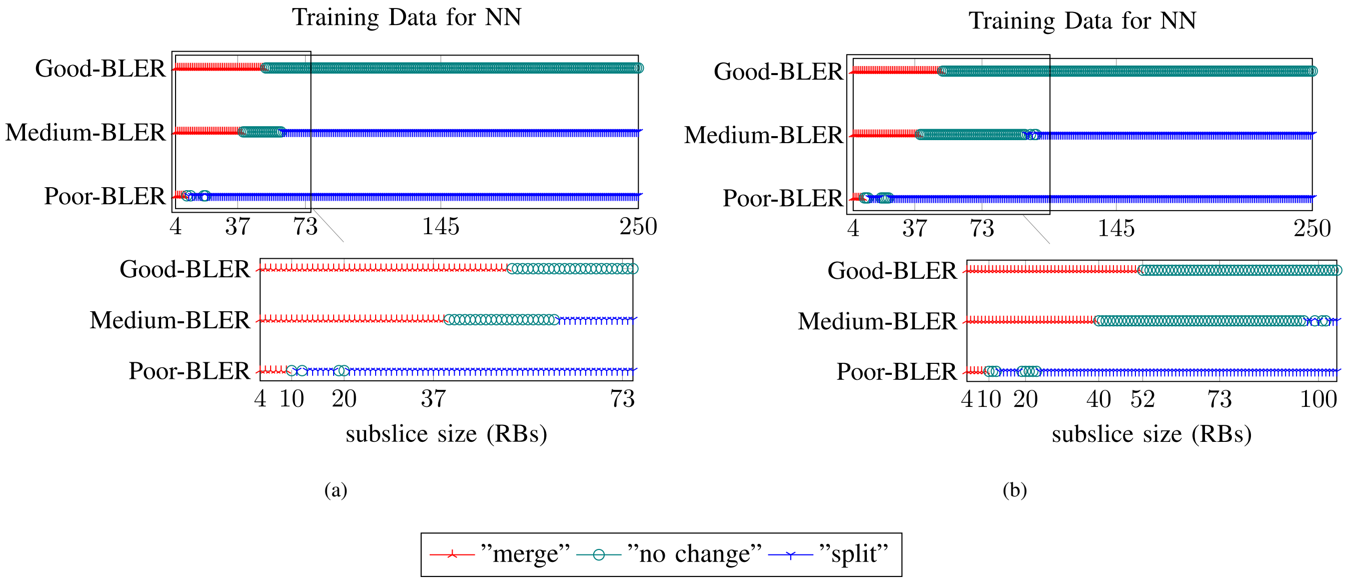
Classifier NN is selected because it can learn any input training data NN settings determined by trials. NN-6-9-3 was selected: 6 inputs (6 KPIs), one hidden layer consisting of 9 neurons, and 3 outputs (one for each decision)

The NN settings of hidden layers are determined by trials. The NNs with 1-2 hidden layers and 2–22 neurons on layer



(a) Good-BLER (b) Medium-BLER (c) Poor-BLER

FIGURE 3. Solution of the optimization problem to find best subslice sizes.



(a) (b)

FIGURE 4. Training data output decisions for NN with (a) $\varepsilon = 0.1$ and (b) $\varepsilon = 0.2$ is used for deciding the change of subslice configuration.

are trained, validated and tested. Then the cross entropy loss is calculated. The least number of layers and neurons where the losses are not decreasing further, is one hidden layer containing 9 neurons, thus the settings are NN-6-9-3. The NN is shown in figure 5.

Each subslice is decided one-by-one by classifier NN. The minimum subslice size is determined to be the best value of the subslice size, and it must not be greater than a half of the bandwidth part of a subslice to be split.

E. EXECUTE

Execute has subslice configuration data: UE data (UE IDs, requested rate and achieved BLER) and subslice data (number of RBs, number of UEs and UE IDs in the subslice) and decided operation for each subslice. The execute algorithm is Alg. 1. First, subslices with decision “no change” are not changed.

Secondly, subslices with decision “merge“ are merged: the smallest is merged with the largest and until all subslices are merged pair-wise. This can avoid merged subslices being too large. If there is an odd number of subslices then 3 smallest are merged. The merged subslices should not become too large, because too large subslice exhibits poor performance again. If just one to merge, then its configuration will not change. This will not change the subslices which should not be changed. Subslice merging combines sets of UEs of subslices to be merged, and combines the number of RBs of subslices to be merged.

Thirdly, subslices with decision “split“ are split: UEs are clustered into two and slice bandwidth is split into two, that is RBs are divided to UE groups proportional to group

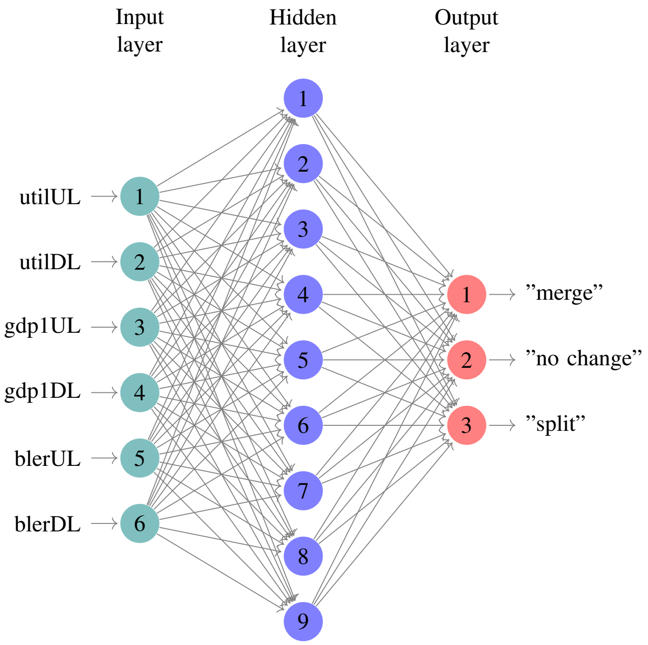


FIGURE 5. **Trained NN.**

Algorithm 1 Execute function

```
Sub 1 ▷ Process subslices with a decision "no change"
The subslice UEs, RBs and $id$ is not changed.
Sub 2 ▷ Process subslices with a decision "merge"
Order subslices ascending
$n \leftarrow$ number of subslices to be merged
if $n \leq 3$ then
    Merge all subslices
else if $n$ is odd then
    Merge 3 smallest subslices
else
    repeat
        Merge the largest subslice with the smallest subslice
    until Done
end if
Sub 3 ▷ Process subslices with a decision "split ($S_{min}$)"
for Each subslice do
    Split subslice into 2 with minimum subslice size constraint
end for
```

Algorithm 2 Split function

```
Convert $S_{min}$ to $R_{min}$
Sub 1 ▷ Cluster UEs: if clustering algorithm is DSbR then by UE Rate, if DSbB then by UE BLER
repeat
    Modified k-means
until All clusters' sum rate$\geq R_{min}$
repeat
    Original k-means
until All UEs assigned to a cluster
Sub 2 ▷ Allocate slice RBs to UE groups
Initial allocation proportional to group sum rate
Second allocation proportional to group BLER
Third allocation
while leftover RBs do
    Allocate leftover RB to the group consisting UE with highest BLER, descending order
end while
```

requested sum rate and group BLER. The subslice splitting algorithm is Alg. 2.

The base algorithm for splitting a subslice is taken from [6] and modified to fit to the UEs which request different rates. First, the minimum subslice size in RBs, S_{min} is converted to minimum subslice sum rate using

$$R_{min} = \left\lfloor S_{min} \cdot \frac{R^{(s)}}{N^{(RB)}} \right\rfloor, \tag{6}$$

where $R^{(s)}$ is the slice sum rate, $N^{(RB)}$ is the number of slice RBs.

The modified k-means stops when the requested minimum sum rate is reached. Sum rate is calculated as a sum of requested rates of the UEs in the cluster. Minimum sum rate is calculated from the sum rate of UEs of the subslice to be split and number of subslice RBs.

V. PERFORMANCE EVALUATION

The aim is to investigate how the proposed MCCL works and whether the slice performance is improved if subslicing is done.

A. SIMULATION SETUP

Slice bandwidth is band n28, 45 MHz (250 RBs, subcarrier spacing is 15 kHz). Slice has a set of 250 UEs which use 40-byte packet sizes. The requested rates are 100, 200 or 400 kbps, symmetrical for UL and DL. The slice sum rate is 50 Mbps. With these UE settings, the slice SLA matches the SLA of 250-RB subslice in section III by number of UEs and requested sum rate. To have the same number of UEs in the slice, and the same requested sum rate of the slice, some UEs request 400 kbps to consume 2 RBs, and some RBs are consumed by 2 UEs, each requesting 100 kbps. The UE distances are such that one third can achieve good BLER, one third can achieve medium BLER and one third can achieve poor BLER. The UE parameters are shown in table 2.

Slice is simulated using MATLAB 5G Toolbox tool called NR Cell Performance Evaluation with Physical Layer Integration [17] R2021b. Settings are shown in table 1. The slice is simulated 10 times in each setting.

TABLE 2. Set of 250 UEs

BLER	Requested rate	Number of UEs
Any	100 kbps	110
Any	200 kbps	85
Any	400 kbps	55
Total	**50 Mbps**	**250**
Good-BLER	Any	83
Medium-BLER	Any	84
Poor-BLER	Any	83
Total	**50 Mbps**	**250**
Good-BLER	100 kbps	36
Good-BLER	200 kbps	28
Good-BLER	400 kbps	19
Medium-BLER	100 kbps	37
Medium-BLER	200 kbps	29
Medium-BLER	400 kbps	18
Poor-BLER	100 kbps	37
Poor-BLER	200 kbps	28
Poor-BLER	400 kbps	18
Total	**50 Mbps**	**250**

The slice performance is compared if the MCCL uses following subslice splitting algorithms:

- "static 3 subslices" (S3S) is proposed in [5], where subslices are created based on UE features. Our set of UEs has 3 different rate requirements, thus the slice is always subsliced into three;
- "dynamic subslicing with UE clustering by requested rate" (DSbR);
- "dynamic subslicing with UE clustering by UE BLER" (DSbB);
- no subslicing, i.e. slice is not split.

Subslice splitting algorithms contain RB allocation proportional to the group sum rate and group BLER.

B. RESULTS OF INITIALIZATION

During the subslicing initialization, MCCL is run as many times as it takes to reach a stable configuration of subslices in the slice, a convergence, that is, when all decisions are "no change".

The initialization of algorithm S3S contains just one iteration to group UEs by requested rate and allocate RBs proportional to group sum rate and group BLER. Slice configuration is shown in the figure 6. For dynamic subslicing algorithms, the MCCL is run and subslicing is done according to thedecisions provided by NN based on six KPIs of the slice/subslice.

The training data for the NN were obtained from the optimization problem. There are different options for mapping the decision to the subslice size based on its objective value. The horizontal boundary between decision "no change" and "split" depends on the selected size of ε-neighbourhood, see figure 3. Two values of ε, 0.1 and 0.2 are selected for subslicing initialization for comparison. The training data for NN of the selected ε values are shown in 4 (a) and (b), respectively.

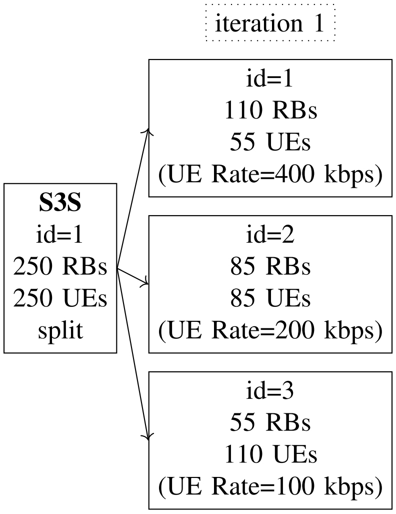


FIGURE 6. Initialization of subslicing algorithm S3S [5] (3 subslices, each for different UE rate).

Next, the slice configuration is compared if dynamic subslicing algorithms were used for initialization. The slice configuration changes during initialization with DSbR splitting algorithm is shown in figure 7.

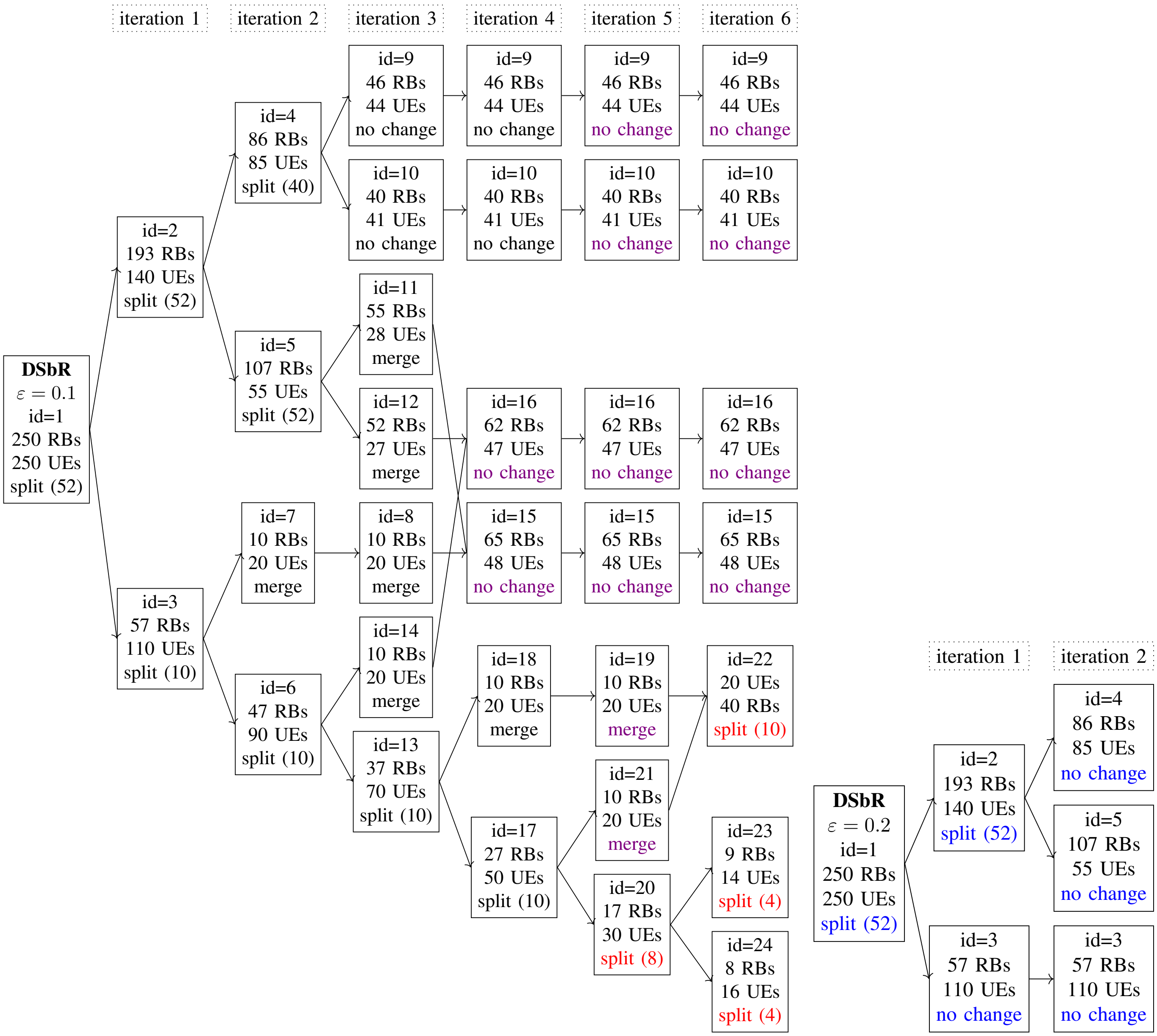
If $\varepsilon = 0.1$ the initialization using DSbR took six iterations (see figure 7a). In the third iteration, it was decided two pairs of subslices to be merged, and this resulted good subslices with $id = 15$ and $id = 16$. Subslice with $id = 13$ was decided to split on iteration 3. Of its subslices in iteration 4, one was decided to be merged, and one was decided to be split. Finally, some subslices decided to be split into sizes of four or five RBs. With greater ε and splitting algorithm DSbR the stable slice configuration was reached faster, within just two iterations, as shown in figure 7b.

The algorithm DSbB needed 4 iterations with both ε values, as shown in figure 8. However, in both values of ε, there is a subslice with split-merge infinite loop in configuration change. The infinite loop is that in one iteration the subslice should be split, and in the next iteration both subslices should be merged.

The initialization of the slice configuration required fewer iterations, and the slice contained fewer subslices with a greater value of ε. If MCCL uses the splitting algorithm DSbR, then the slice contains three subslices, as if the splitting algorithm is S3S. When the splitting algorithm DSbB was used, a split-merge infinite loop appeared for one subslice.

The slice performance results during the initialization are shown in figure 9. Left column contains slice KPI values if $\varepsilon = 0.1$ and right column contains slice KPI values if $\varepsilon = 0.2$.

The static algorithm S3S achieves better slice performance than no subslicing: UL utilization has improved by 1.26%,



(a) (b)

FIGURE 7. **Initialization of scenario DSbR completed. (a)** $\varepsilon = 0.1$ **and (b)** $\varepsilon = 0.2$.

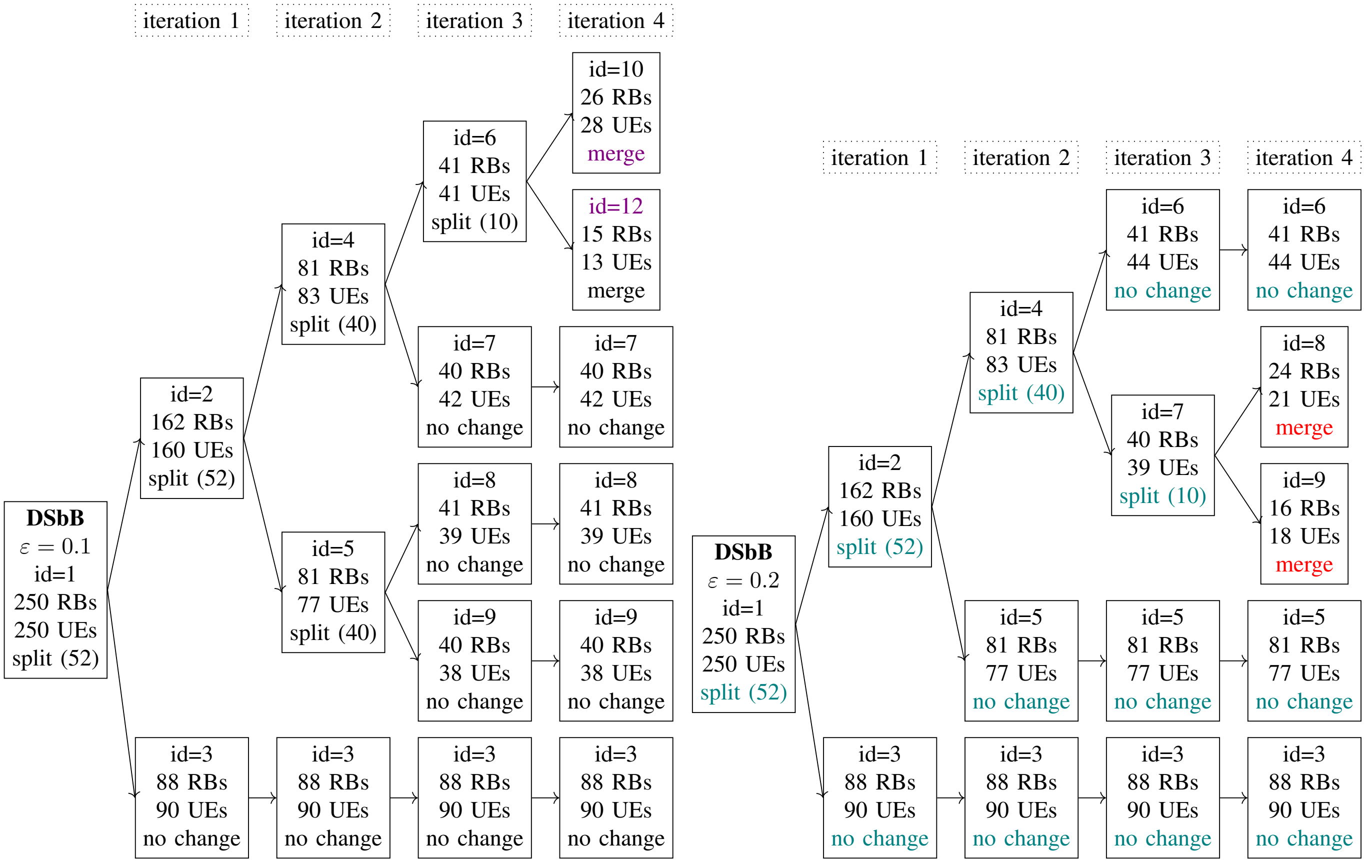
goodput per one RB improved in UL by 28.4% and in DL by 11.5%, and BLER reduced by 31% and 43.7% in UL and DL, respectively.

Dynamic splitting algorithms achieve different performance if different ε values were used. The increase of ε improved slice goodput and BLER for DSbR, however with DSbB the change of KPI values was small. However, despite the split-merge infinite loop, DSbB achieved the best slice performance improvement in reducing slice utilization by 6% in UL, BLER by about 60% in both UL and DL, and improved goodput per one RB in UL the most, 37%. The DSbB had better goodput in iteration 3, but after subslice splitting, the achieved gootput decreased. The dynamic algorithms can miss their peak goodputs if the decisions contain too much splitting of the subslices.

For runtime simulations, the dynamic subslicing algorithms were initialized using training data obtained with $\varepsilon = 0.2$ because initialization requires fewer iterations and slice performance is generally better.

C. RUNTIME SCENARIOS

Two scenarios were used to investigate how the proposed MCCL works in the conditions when slice load increases or decreases. It is expected that subslicing can improve the slice performance on a fixed slice bandwidth. The starting point of the scenarios is the initialized slice from the previous


iteration 1
iteration 2
iteration 3
iteration 4
DSbB $\varepsilon = 0.1$ id=1 250 RBs 250 UEs split (52)
id=2 162 RBs 160 UEs split (52)
id=4 81 RBs 83 UEs split (40)
id=5 81 RBs 77 UEs split (40)
id=3 88 RBs 90 UEs no change
id=6 41 RBs 41 UEs split (10)
id=7 40 RBs 42 UEs no change
id=8 41 RBs 39 UEs no change
id=9 40 RBs 38 UEs no change
id=10 26 RBs 28 UEs merge
id=12 15 RBs 13 UEs merge
iteration 1
iteration 2
iteration 3
iteration 4
DSbB $\varepsilon = 0.2$ id=1 250 RBs 250 UEs split (52)
id=2 162 RBs 160 UEs split (52)
id=4 81 RBs 83 UEs split (40)
id=5 81 RBs 77 UEs no change
id=3 88 RBs 90 UEs no change
id=6 41 RBs 44 UEs no change
id=7 40 RBs 39 UEs split (10)
id=8 24 RBs 21 UEs merge
id=9 16 RBs 18 UEs merge

(a) (b)

FIGURE 8. Initialization of scenario DSbB completed. (a) $\varepsilon = 0.1$ and (b) $\varepsilon = 0.2$.

subsection. The slice load increase is performed in Scenario 1, in which a new UE comes into the subslice with the lowest utilization. The slice load decrease is performed in Scenario 2, where one UE with the highest BLER leaves. The UE that is coming or leaving has high requested rates and is expected to achieve a high (poor) BLER.

The discrete-event simulator shown in figure 10 is created to run the scenarios. Initially, in round 0, the slice has 250 UEs, and the initial configuration of the subslices is based on the splitting algorithms used in the previous subsection. At the end of the round, an event occurs, which means adding or removing one UE, as stated in the scenario. The next round begins with the simulation of a slice to determine the effect of the event. The MCCL then runs and possibly changes the slice configuration for subslicing. The second simulation of the slice in this round shows the effect of the new configuration on slice performance.

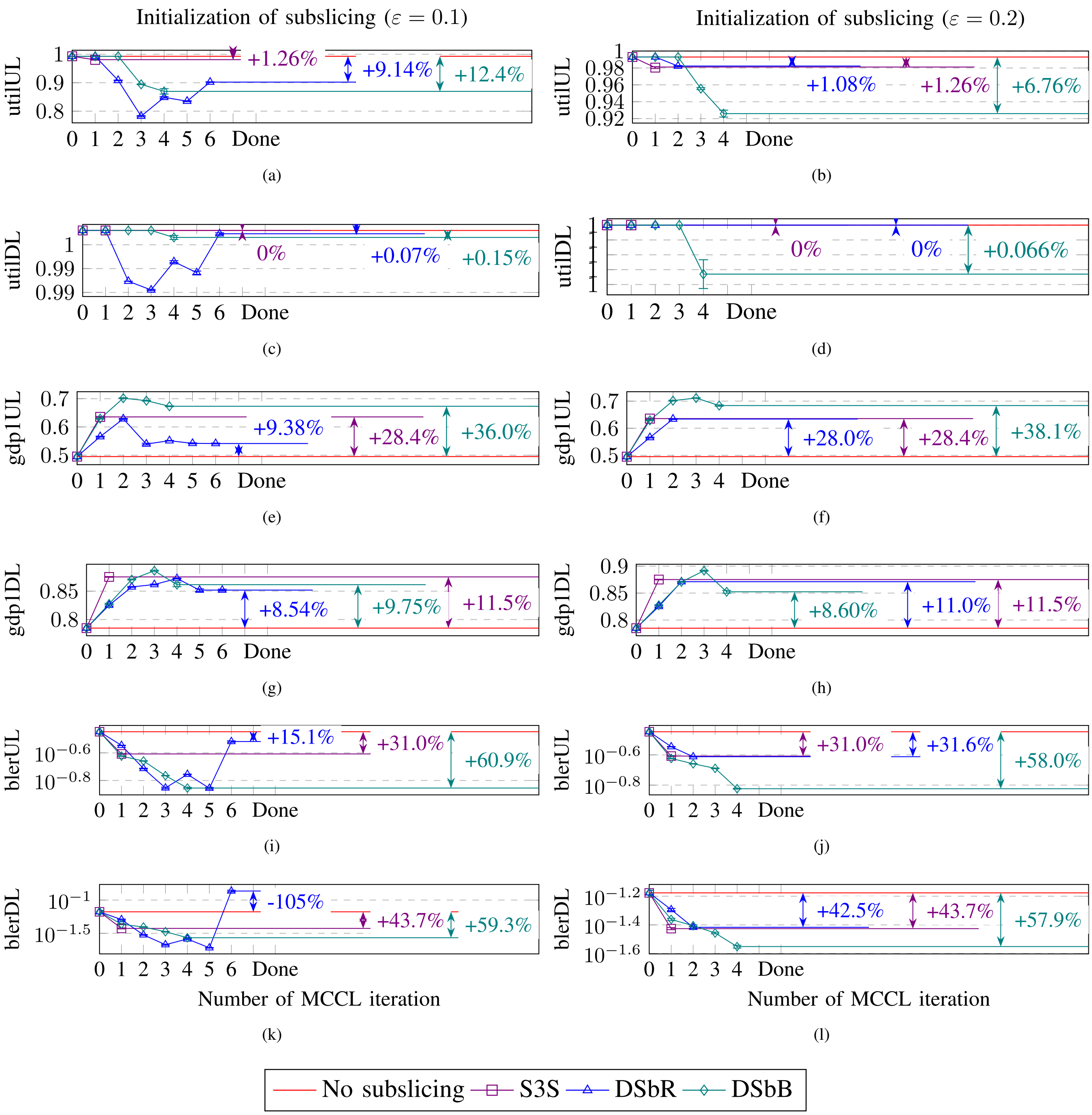
D. RUNTIME RESULTS

The scenarios run seven events and each slice was simulated 10 times. The slice performance changes are shown in figure 11. The runtime simulation results are shown in figure 13.

1) Scenario 1

Let us consider the slice performance data of scenario 1 (figure 11a and 11c). Within 7 events, the slice requested sum rate increases by 5.6% due to new UEs added. When no subslicing done, the slice utilization did not change, goodput per one RB decreased by 2% and 1% in UL and DL, respectively, and BLER increased by 1% in UL and decreased by 2% in DL. If subslice splitting by UE requested rates (algorithms S3S and DSbR) were used, then utilization in UL increased, goodput decreased and BLER increased a few more percent. When subslice splitting by UE BLER (algorithm DSbB) was used, then the change of performance depends on how many subslices are in the slice. After the initialization, the slice contained 5 subslices. After events 5, 6 and 7, the slice contained 4, 5 and 6 subslices, respectively (see table 12. When less subslices, then UL utilization and UL goodput per one RB increased more, up to 5%. In DL, the utilization did not change and goodput increased 4% if 4 subslices, and decreased 2% if 6 subslices. The greatest effect of number of subslices had BLER in both UL and DL. The BLER increased up to 28% if 4 subslices, and decreased up to 41% if 6 subslices were in the slice.

Details on figure 13 left column show dynamics of KPI value changes during the events and MCCL runs. If no



(a) (b) (c) (d) (e) (f) (g) (h) (i) (j) (k) (l)

FIGURE 9. **Slice performance during initialization of subslicing algorithms (a), (c), (e), (g), (i), (k) $\varepsilon = 0.1$ and (b), (d), (f), (h), (j), (l) $\varepsilon = 0.2$. Percentages show how much better values were achieved compared to those without subslicing.**

subslicing the utilization is full already. The UL utilization increases to full utilization, if splitting algorithms S3S and DSbR were used. If DSbB then utilization is lower than others. The utilization in DL does not change, being full. In a few events after MCCL run, the DSbB can achieve slightly lower utilization in DL. The goodput per one RB in UL slowly decreases, but DSbB achieves best values. In DL, the values of goodput per one RB for DSbB changes, being sometimes better and sometimes worse than S3S and DSbR. Splitting algorithms S3S and DSbR can achieve slice BLER similar and not changing. If DSbB splitting algorithm used, the slice BLER values depend on the number of subslices, but still BLER is the smallest in UL and DL.

The slice configurations are shown in figures 14 for SCS, figure 15 for DSbR and figure 16 for DSbB splitting algorithm used in MCCL, respectively.

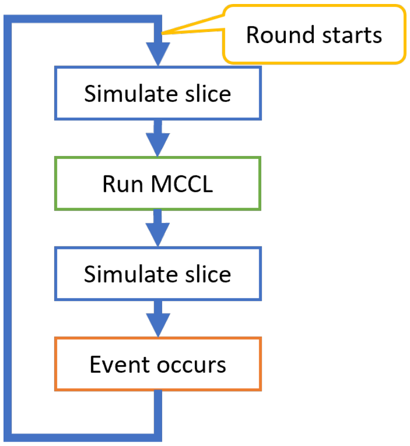


FIGURE 10. **Discrete event simulator.**

The slice configuration, when S3S was used as splitting algorithm in MCCL, had always 3 subslices, each for UEs with different requested rate. UEs are coming in to the largest subslice and resource allocation increases the number of RBs for this subslice by taking RBs away from other subslices. When DSbR was used as splitting algorithm, then slice configuration is stable, consisting of 3 subslices, similarly each for UEs with different requested rate. New UEs come to the subslice with $id = 5$ because this had the lowest bandwidth utilization. The 7th UE comes to the subslice with $id = 3$ because now this had the lowest utilization. The subslice splitting algorithm DSbB had created larger stable subslices for UEs achieving good or medium BLER. The two subslices with $id = 8$ and $id = 9$ contain UEs with poor BLER and are split or merged at each MCCL run. When the 4th and 7th UE added, then there were 6 subslices in the slice, and both goodput and BLER were low. When the slice contained 4 subslices then the goodput achieved the highest values and so did BLER. The best goodput per RB in UL and DL was achieved using splitting algorithm DSbB and 4 subslices.

When slice load increases, the achieved goodput will not increase, however subslice splitting has saved on the initialization few RBs, which will be utilized and a small increase in utilization and goodput is achieved quickly. BLER is lower if poor-BLER UEs are divided into more subslices and this can be done by using subslice splitting algorithm DSbB.

2) Scenario 2

Let us consider slice performance data of scenario 2 (see figures 11b and 11d. Within 7 events, the slice requested sum rate decreases by 5.6% due to UEs removed. When no subslicing used, then after 7 events the slice utilization, goodput and BLER did not change more than 1%, and slice BLER in DL decreased by 4%. Subslice splitting by UE BLER (DSbB), the values of KPIs depend on how many subslices the slice contained. After the initialization, the slice contained 5 subslices. After events 5, 6 and 7, the slice contained 5, 6 and 4 subslices, respectively (see table 12. If less subslices then all KPIs were increased compared to the slice configuration containing 5 subslices at the initialization. The BLER had the greatest change. If 6 subslices then the BLER decreased by more than 40%, while utilization decreased by 7% and 1% in UL and DL, respectively.

After 7 events, the slice utilization in UL decreases by 3-4% if subslice splitting algorithms were used, as shown in figure 11b. It did not affect slice utilization in DL, which did not change. Details in figure 13 right column show that the slice bandwidth was fully utilized, and after 7 events the slice utilization was still full. With subslicing the utilization in UL slowly decreases, but it decreases the most when splitting algorithm DSbB was used. In UL, the best goodput per RB is achieved with splitting algorithm DSbB. In DL, the goodput per RB of DSbB is sometimes the best and sometimes just better than if no subslicing. The subslicing decreases the slice BLER in both UL and DL compared to no subslicing, but the splitting algorithm DSbB achieves the lowest BLER regardless of the number of subslices.

The slice configurations are shown in figure 17 for SCS, figure 18 for DSbR and figure 19 for DSbB splitting algorithm used in MCCL, respectively.

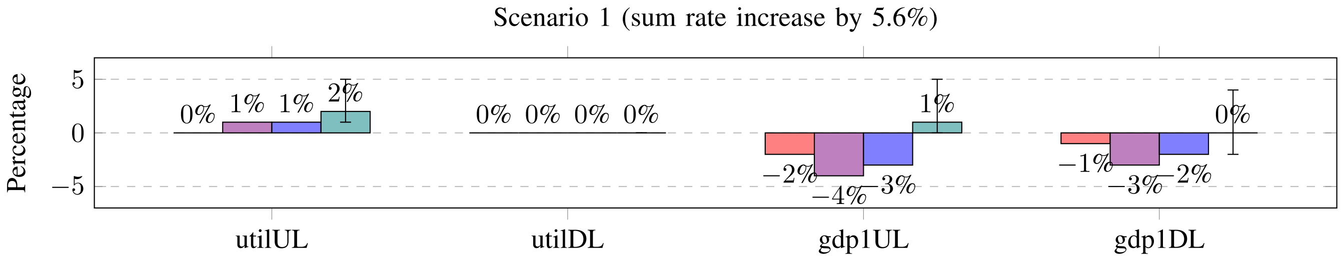
The slice configuration by using S3S splitting algorithm has always 3 subslices where UEs are clustered by requested rate. UE leaves the subslice with $id = 1$.

The scenario with no subslicing has all KPIs unchanged, however with subslicing the goodput per one RB increases and BLER decreases. Resource allocation takes RBs from that subslice and gives those to the others. Similarly, the slice contains 3 subslices by UE rate, if splitting algorithm DSbR was used. The UE always leaves the subslice with $id = 5$ because this contained UEs which request 400 kbps. The subslice splitting algorithm DSbB has stable subslices for good-BLER and medium-BLER UEs. The 2 subslices with $id = 8$ and $id = 9$ are for poor-BLER UEs and their configuration changes on each MCCL run. Similarly, the best goodput per one RB is achieved if these poor-BLER subslices are merged to one, as in events 2, 4 and 7.

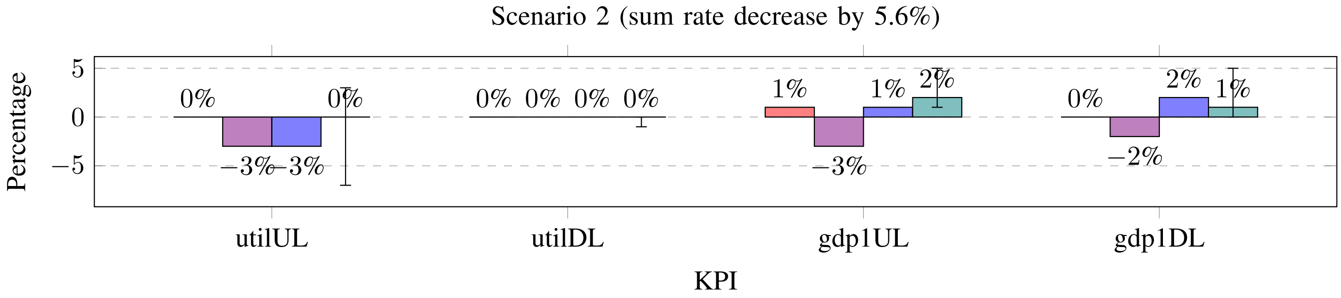
When slice load decreases then the goodput can increase because there are more resources available for packets which needed retransmission before. Similarly, as in slice load increase, the poor-BLER UEs need more subslices and the slice BLER can be decreased significantly.

VI. CONCLUSION

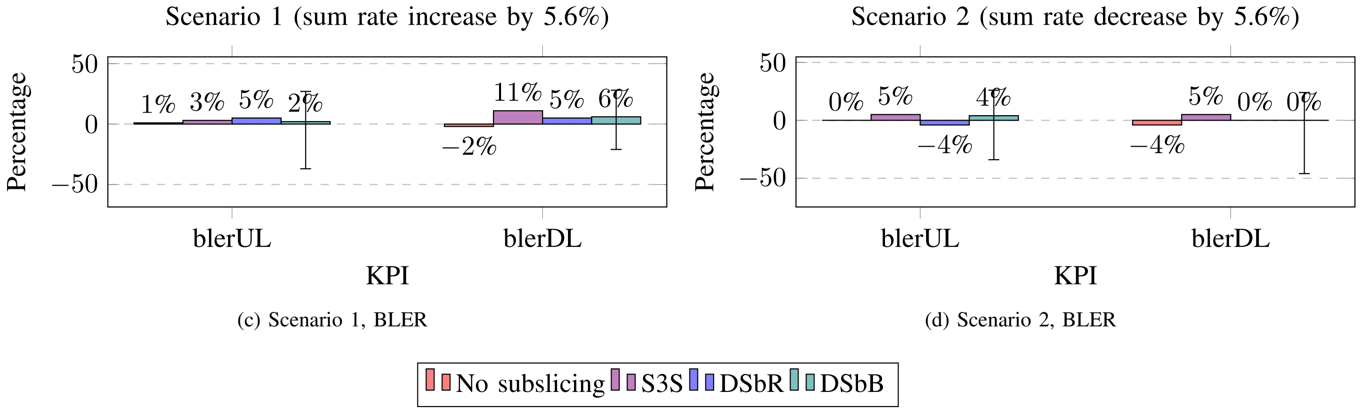
The management closed control loop for subslicing to improve slice performance was implemented in this paper. For slice simulations, the realistic BWP, n28, was used as the fixed slice bandwidth of 50 MHz. The realistic set of 250 V2X or MIoT UEs are set to consume the allocated bandwidth by their requested sum rate of 50 Mbps. The planned slice load was close to slice overload. UEs were placed at different distances from gNB to have UEs with different



(a) Scenario 1, utilization and goodput per one RB



(b) Scenario 2, utilization and goodput per one RB



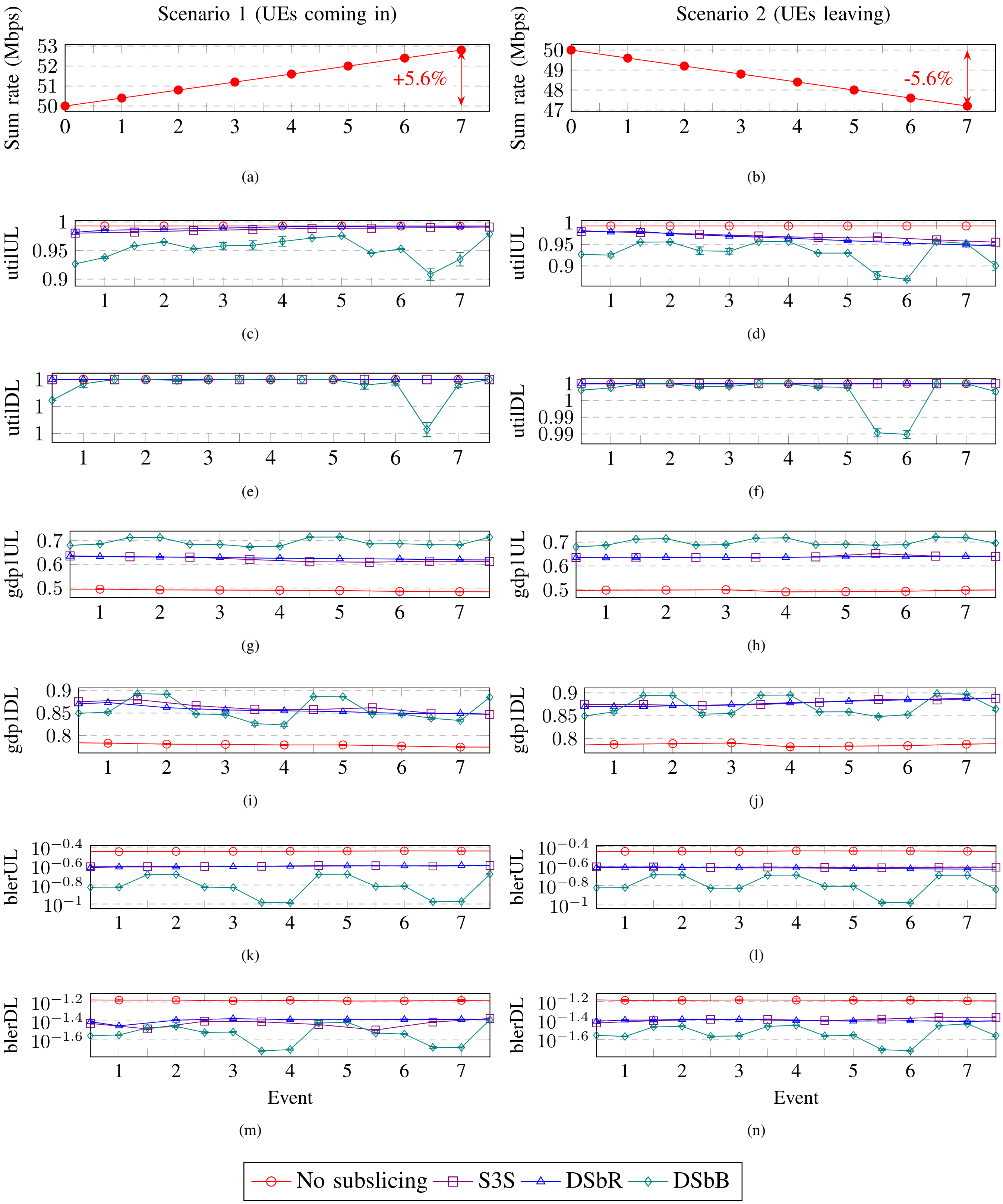
(c) Scenario 1, BLER

(d) Scenario 2, BLER

FIGURE 11. The KPI value changes after 7 events of runtime scenarios. Percentages show difference between KPI value of the initialization and the 7th event if the splitting algorithm was used in runtime scenarios. In DL, the utilization did not change, being full.

Scenario 1 (sum rate increase)							
Number of subslices	**Event number**	**utilUL**	**utilDL**	**gdp1UL**	**gdp1DL**	**blerUL**	**blerDL**
4	5	5%	0%	5%	4%	27%	28%
5	6	2%	0%	1%	0%	2%	6%
6	7	1%	0%	0%	-2%	-41%	-33%
Scenario 2 (sum rate decrease)							
Number of subslices	**Event number**	**utilUL**	**utilDL**	**gdp1UL**	**gdp1DL**	**blerUL**	**blerDL**
4	7	3%	0%	5%	5%	26%	25%
5	5	0%	0%	2%	1%	4%	0%
6	6	-7%	-1%	1%	0%	-42%	-46%

FIGURE 12. The change of slice performance change if the splitting algorithm was DSbB. After initialization, the slice contained 5 subslices. After events number 5, 6 or 7, the slice contained 4, 5 or 6 subslices.



(a) (b) (c) (d) (e) (f) (g) (h) (i) (j) (k) (l) (m) (n)

FIGURE 13. Slice performance on events of scenarios and effect of MCCL.

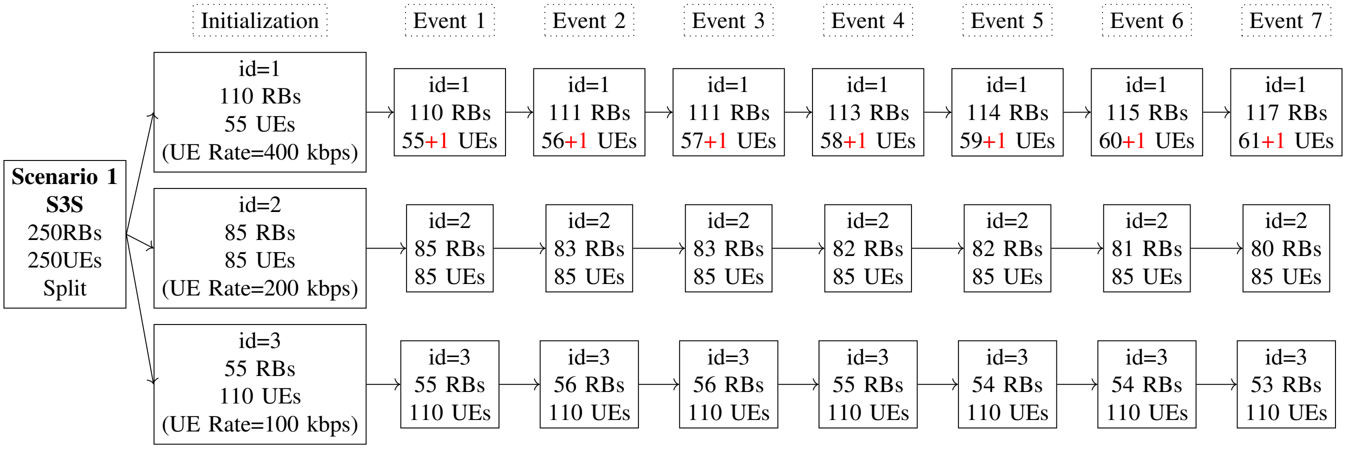


FIGURE 14. Slice configuration, scenario 1, splitting algorithm S3S, adopted from [5], first 7 events.

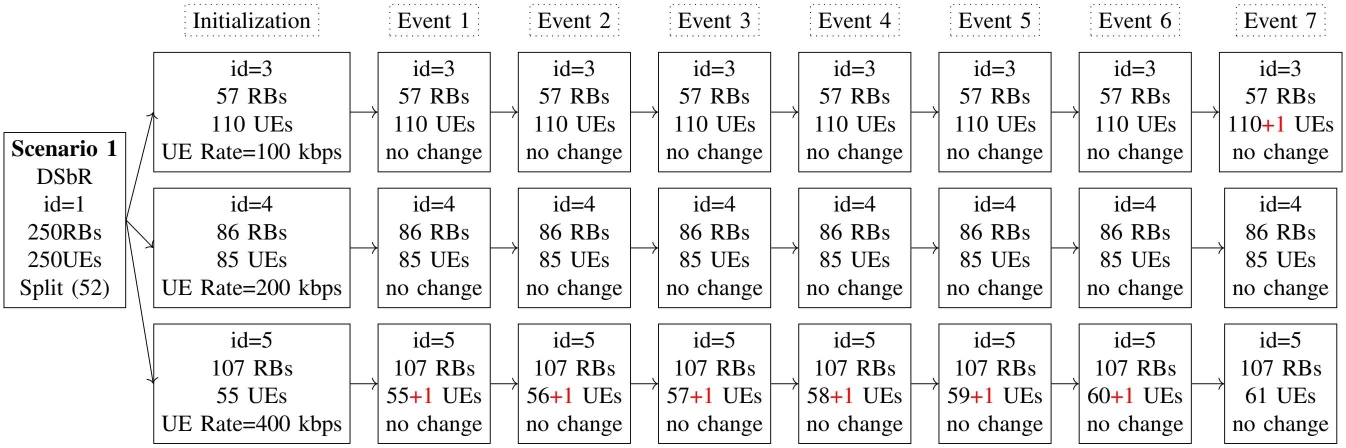

Initialization
Event 1
Event 2
Event 3
Event 4
Event 5
Event 6
Event 7
Scenario 1
DSbR
id=1
250RBs
250UEs
Split (52)
id=3
57 RBs
110 UEs
UE Rate=100 kbps
id=4
86 RBs
85 UEs
UE Rate=200 kbps
id=5
107 RBs
55 UEs
UE Rate=400 kbps
id=3 57 RBs 110 UEs no change
id=3 57 RBs 110 UEs no change
id=3 57 RBs 110 UEs no change
id=3 57 RBs 110 UEs no change
id=3 57 RBs 110 UEs no change
id=3 57 RBs 110 UEs no change
id=3 57 RBs 110+1 UEs no change
id=4 86 RBs 85 UEs no change
id=4 86 RBs 85 UEs no change
id=4 86 RBs 85 UEs no change
id=4 86 RBs 85 UEs no change
id=4 86 RBs 85 UEs no change
id=4 86 RBs 85 UEs no change
id=4 86 RBs 85 UEs no change
id=5 107 RBs 55+1 UEs no change
id=5 107 RBs 56+1 UEs no change
id=5 107 RBs 57+1 UEs no change
id=5 107 RBs 58+1 UEs no change
id=5 107 RBs 59+1 UEs no change
id=5 107 RBs 60+1 UEs no change
id=5 107 RBs 61 UEs no change

FIGURE 15. Slice configuration, scenario 1, splitting algorithm DSbR, first 7 events.

BLERs in the slice. Actually, UEs with mixed BLERs cause bandwidth overutilization due to packet retransmissions. The MAPE-K management CCL was implemented to split or merge subslices if it can improve slice performance on fixed slice bandwidth. Using the discrete event simulator, it is possible to show the dynamics of slice performance and the effect of subslicing.

Slice performance was compared if different subslice splitting algorithms were used in management CCL. For the sake of slice performance, UEs with the similar BLER in one subslice achieve better slice performance than UEs with similar requested rates.

The subslicing enables to increase goodput about 30% in UL and 10% in DL, reduce BLER at least 30% and 40% in UL and DL, respectively. Dynamic subslicing with UE clustering by BLER enables to achieve up to 6% less utilization and up to 10% more goodput, and reduce BLER additional at least 20% compared to other subslice splitting algorithms tried in the implemented management closed control loop.

In runtime scenarios when slice sum rate was increased or decreased, the subslice splitting algorithms where UEs were clustered by UE rate (S3S and DSbR), the slice configuration was stable. If subslice splitting algorithm where UEs were clustered by UE BLER (DSbB) was used, the slice configuration changed on each MCCL cycle for poor-BLER subslices. However, the slice achieved higher goodput per one RB and lower BLER than with other subslice splitting algorithms. For slice utilization and BLER improvement, the splitting algorithm should create more subslices for poor-BLER UEs, and for goodput improvement less subslices for poor-BLER UEs. Slice performance results have shown that BLER is the most sensitive KPI, dependent on the number of poor-BLER subslices. The more slice performance improvement was achieved for UL.

The future work is to find the slice performance thresholds when to start and stop perform subslicing to improve the slice performance on fixed slice bandwidth. Other research direction is to investigate how to decide the subslicing if the

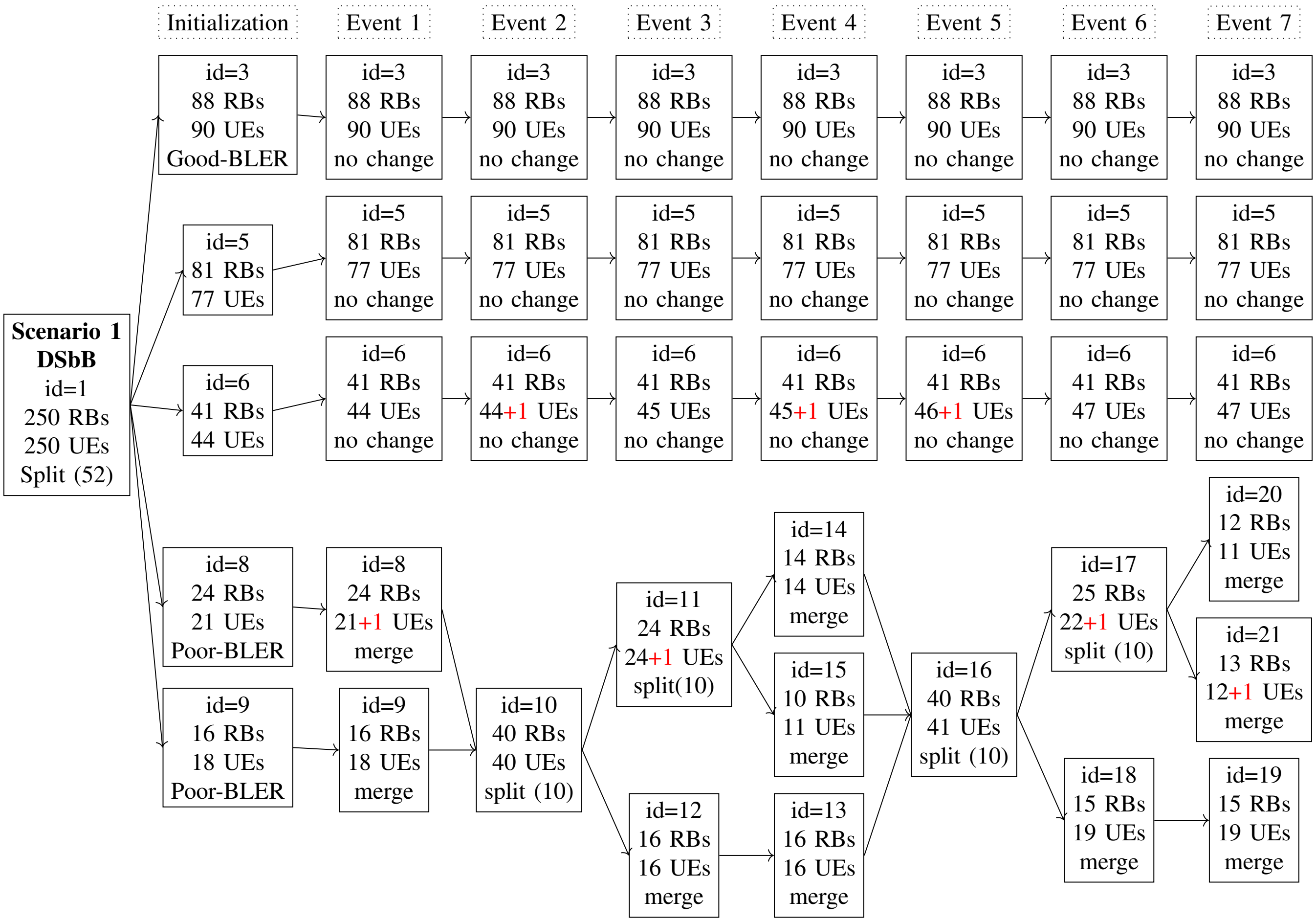

Initialization
Event 1
Event 2
Event 3
Event 4
Event 5
Event 6
Event 7
Scenario 1 DSbB id=1 250 RBs 250 UEs Split (52)
id=3 88 RBs 90 UEs Good-BLER
id=3 88 RBs 90 UEs no change
id=5 81 RBs 77 UEs
id=5 81 RBs 77 UEs no change
id=6 41 RBs 44 UEs
id=6 41 RBs 44 UEs no change
id=6 41 RBs 44+1 UEs no change
id=6 41 RBs 45 UEs no change
id=6 41 RBs 45+1 UEs no change
id=6 41 RBs 46+1 UEs no change
id=6 41 RBs 47 UEs no change
id=8 24 RBs 21 UEs Poor-BLER
id=8 24 RBs 21+1 UEs merge
id=9 16 RBs 18 UEs Poor-BLER
id=9 16 RBs 18 UEs merge
id=10 40 RBs 40 UEs split (10)
id=11 24 RBs 24+1 UEs split(10)
id=12 16 RBs 16 UEs merge
id=14 14 RBs 14 UEs merge
id=15 10 RBs 11 UEs merge
id=13 16 RBs 16 UEs merge
id=16 40 RBs 41 UEs split (10)
id=17 25 RBs 22+1 UEs split (10)
id=18 15 RBs 19 UEs merge
id=20 12 RBs 11 UEs merge
id=21 13 RBs 12+1 UEs merge
id=19 15 RBs 19 UEs merge

FIGURE 16. **Slice configuration, Scenario 1, splitting algorithm DSbB, first 7 events.**

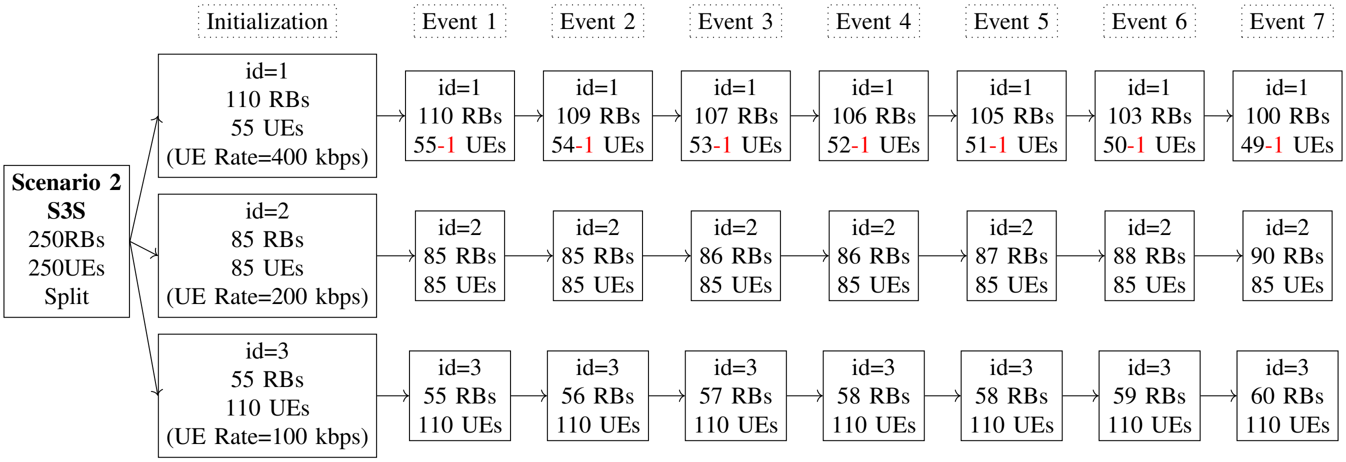


FIGURE 17. **Slice configuration, scenario 2, splitting algorithm S3S, adopted from [5], first 7 events.**

subslice performance dependence of subslice size objective function is not convex or has multiple minimums.

ACKNOWLEDGMENT

During the preparation of this work, the author(s) used Paperpal and LanguageTool in order to improve grammar, spelling and style. After using this tool/service, the author(s)

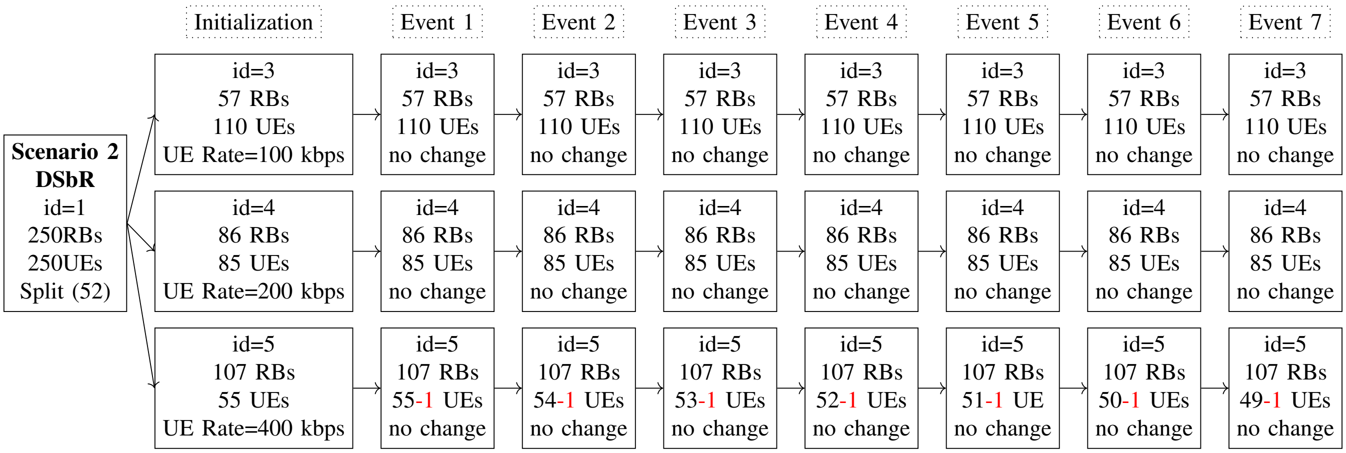

Initialization
Event 1
Event 2
Event 3
Event 4
Event 5
Event 6
Event 7
Scenario 2 DSbR id=1 250RBs 250UEs Split (52)
id=3 57 RBs 110 UEs UE Rate=100 kbps
id=3 57 RBs 110 UEs no change
id=3 57 RBs 110 UEs no change
id=3 57 RBs 110 UEs no change
id=3 57 RBs 110 UEs no change
id=3 57 RBs 110 UEs no change
id=3 57 RBs 110 UEs no change
id=3 57 RBs 110 UEs no change
id=4 86 RBs 85 UEs UE Rate=200 kbps
id=4 86 RBs 85 UEs no change
id=4 86 RBs 85 UEs no change
id=4 86 RBs 85 UEs no change
id=4 86 RBs 85 UEs no change
id=4 86 RBs 85 UEs no change
id=4 86 RBs 85 UEs no change
id=4 86 RBs 85 UEs no change
id=5 107 RBs 55 UEs UE Rate=400 kbps
id=5 107 RBs 55-1 UEs no change
id=5 107 RBs 54-1 UEs no change
id=5 107 RBs 53-1 UEs no change
id=5 107 RBs 52-1 UEs no change
id=5 107 RBs 51-1 UE no change
id=5 107 RBs 50-1 UEs no change
id=5 107 RBs 49-1 UEs no change

FIGURE 18. Slice configuration, scenario 2, splitting algorithm DSbR, first 7 events.

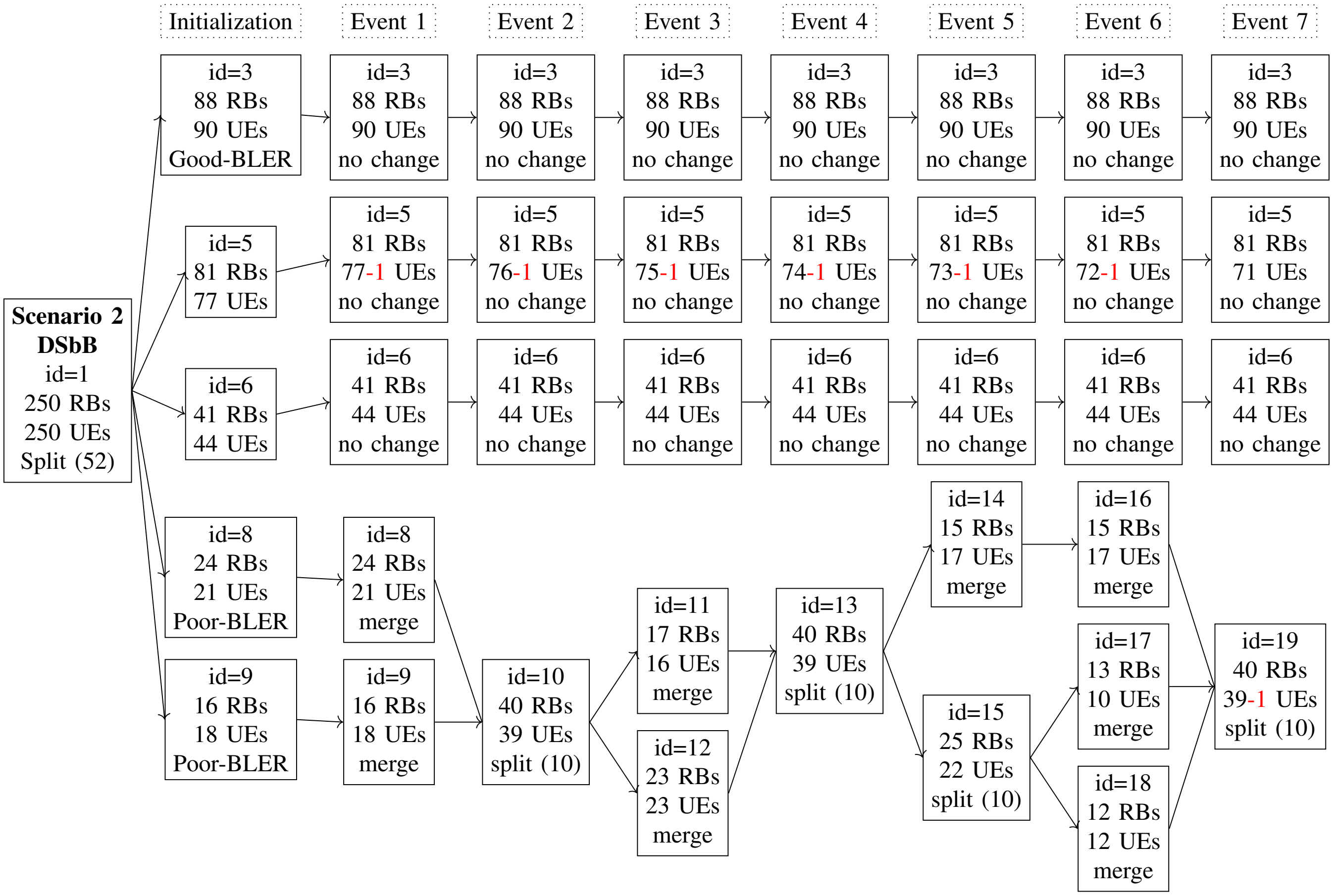

Initialization
Event 1
Event 2
Event 3
Event 4
Event 5
Event 6
Event 7
Scenario 2 DSbB id=1 250 RBs 250 UEs Split (52)
id=3 88 RBs 90 UEs Good-BLER
id=3 88 RBs 90 UEs no change
id=3 88 RBs 90 UEs no change
id=3 88 RBs 90 UEs no change
id=3 88 RBs 90 UEs no change
id=3 88 RBs 90 UEs no change
id=3 88 RBs 90 UEs no change
id=3 88 RBs 90 UEs no change
id=5 81 RBs 77 UEs
id=5 81 RBs 77-1 UEs no change
id=5 81 RBs 76-1 UEs no change
id=5 81 RBs 75-1 UEs no change
id=5 81 RBs 74-1 UEs no change
id=5 81 RBs 73-1 UEs no change
id=5 81 RBs 72-1 UEs no change
id=5 81 RBs 71 UEs no change
id=6 41 RBs 44 UEs
id=6 41 RBs 44 UEs no change
id=6 41 RBs 44 UEs no change
id=6 41 RBs 44 UEs no change
id=6 41 RBs 44 UEs no change
id=6 41 RBs 44 UEs no change
id=6 41 RBs 44 UEs no change
id=6 41 RBs 44 UEs no change
id=8 24 RBs 21 UEs Poor-BLER
id=8 24 RBs 21 UEs merge
id=9 16 RBs 18 UEs Poor-BLER
id=9 16 RBs 18 UEs merge
id=10 40 RBs 39 UEs split (10)
id=11 17 RBs 16 UEs merge
id=12 23 RBs 23 UEs merge
id=13 40 RBs 39 UEs split (10)
id=14 15 RBs 17 UEs merge
id=15 25 RBs 22 UEs split (10)
id=16 15 RBs 17 UEs merge
id=17 13 RBs 10 UEs merge
id=18 12 RBs 12 UEs merge
id=19 40 RBs 39-1 UEs split (10)

FIGURE 19. Slice configuration, scenario 2, splitting algorithm DSbB, first 7 events.

reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

REFERENCES

[1] I. Vaishnavi and L. Ciavaglia, "Challenges Towards Automation of Live Telco Network Management: Closed Control Loops," in *2020 16th Int. Conf. Netw. Serv. Manag.*, IEEE, Nov. 2020, pp. 1–5, ISBN: 978-3-903176-31-7, DOI: 10.23919/CNSM50824.2020.9269048.

[2] J. Moysen and L. Giupponi, "From 4G to 5G: Self-organized network management meets machine learn-

ing," *Comput. Commun.*, vol. 129, pp. 248–268, Sep. 2018, ISSN: 01403664, DOI: 10.1016/j.comcom.2018.07.015.

[3] 3GPP, "TS 28.535 - Management and orchestration; Management services for communication service assurance; Requirements," Tech. Rep., 2020, [Online]. Available: https://www.3gpp.org/DynaReport/28535.htm.

[4] 3GPP, "TS 28.530 - Management and orchestration; Concepts, use cases and requirements," Tech. Rep., 2020, [Online]. Available: https://www.3gpp.org/DynaReport/28530.htm.

[5] S. K. Singh, M. M. Salim, J. Cha, Y. Pan, and J. H. Park, "Machine Learning-Based Network Sub-Slicing Framework in a Sustainable 5G Environment," *Sustainability*, vol. 12, no. 15, p. 6250, Aug. 2020, ISSN: 2071-1050, DOI: 10.3390/su12156250.

[6] M. Kulmar, I. Müürsepp, and M. M. Alam, "Heuristic Radio Access Network Subslicing with User Clustering and Bandwidth Subpartitioning," *Sensors*, vol. 23, no. 10, p. 4613, May 2023, ISSN: 1424-8220, DOI: 10.3390/s23104613.

[7] J. Kephart and D. Chess, "The vision of autonomic computing," *Computer (Long. Beach. Calif).*, vol. 36, no. 1, pp. 41–50, Jan. 2003, ISSN: 0018-9162, DOI: 10.1109/MC.2003.1160055.

[8] R. Boutaba, N. Shahriar, M. A. Salahuddin, S. R. Chowdhury, N. Saha, and A. James, "AI-driven Closed-loop Automation in 5G and beyond Mobile Networks," in *Proc. 4th FlexNets Work. Flex. Networks Artif. Intell. Support. Netw. Flex. Agil.*, New York, NY, USA: ACM, Aug. 2021, pp. 1–6, ISBN: 9781450386340, DOI: 10.1145/3472735.3474458.

[9] N. F. Saraiva de Sousa and C. E. Rothenberg, "CLARA: Closed Loop-based Zero-touch Network Management Framework," in *2021 IEEE Conf. Netw. Funct. Virtualization Softw. Defin. Networks*, IEEE, Nov. 2021, pp. 110–115, ISBN: 978-1-6654-3983-1, DOI: 10.1109/NFV-SDN53031.2021.9665048.

[10] M. Xie, P. H. Gomes, J. Harmatos, and J. Ordonez-Lucena, "Collaborated Closed Loops for Autonomous End-to-End Service Management in 5G," in *2020 IEEE Conf. Netw. Funct. Virtualization Softw. Defin. Networks*, IEEE, Nov. 2020, pp. 64–70, ISBN: 978-1-7281-8159-2, DOI: 10.1109/NFV-SDN50289.2020.9289902.

[11] H. Chergui, A. Ksentini, L. Blanco, and C. Verikoukis, "Toward Zero-Touch Management and Orchestration of Massive Deployment of Network Slices in 6G," *IEEE Wirel. Commun.*, vol. 29, no. 1, pp. 86–93, Feb. 2022, ISSN: 1536-1284, DOI: 10.1109/MWC.009.00366.

[12] M. Gramaglia, M. Kajo, C. Mannweiler, Ö. Bulakci, and Q. Wei, "A unified service-based capability exposure framework for closed-loop network automation," *Trans. Emerg. Telecommun. Technol.*, vol. 33, no. 11, Nov. 2022, ISSN: 2161-3915, DOI: 10.1002/ett.4598.

[13] M. Mekki, S. Arora, and A. Ksentini, "A Scalable Monitoring Framework for Network Slicing in 5G and Beyond Mobile Networks," *IEEE Trans. Netw. Serv. Manag.*, vol. 19, no. 1, pp. 413–423, Mar. 2022, ISSN: 1932-4537, DOI: 10.1109/TNSM.2021.3119433.

[14] S. Kuklinski and L. Tomaszewski, "Key Performance Indicators for 5G network slicing," in *2019 IEEE Conf. Netw. Softwarization*, IEEE, Jun. 2019, pp. 464–471, ISBN: 978-1-5386-9376-6, DOI: 10.1109/NETSOFT.2019.8806692.

[15] E. Pateromichelakis, F. Moggio, C. Mannweiler, *et al.*, "End-to-End Data Analytics Framework for 5G Architecture," *IEEE Access*, vol. 7, pp. 40 295–40 312, 2019, ISSN: 2169-3536, DOI: 10.1109/ACCESS.2019.2902984.

[16] 3GPP, "TS 38.104 - NR; Base Station (BS) radio transmission and reception," 2022, [Online]. Available: https://www.3gpp.org/DynaReport/38104.htm.

[17] MathWorks, *NR Cell Performance Evaluation with Physical Layer Integration*, 2022, [Online]. Available: https://se.mathworks.com/help/5g/ug/nr-cell-performance-evaluation-with-physical-layer-integration.html (visited on 01/19/2022).

[18] 3GPP, "TS 38.901- Radio Access Network; Study on channel model for frequencies from 0.5 to 100 GHz," Tech. Rep., 2019, [Online]. Available: https://www.3gpp.org/DynaReport/38901.htm.

[19] 3GPP, "TS 38.211 - NR; Physical channels and modulation," 2020, [Online]. Available: https://www.3gpp.org/DynaReport/38211.htm.

[20] M. Kulmar, I. Müürsepp, and M. M. Alam, "Enhanced Decision Mechanism for RAN Subslicing in Management Closed Control Loop," in *2023 Eighth Int. Conf. Fog Mob. Edge Comput.*, IEEE, Sep. 2023, pp. 175–181, ISBN: 979-8-3503-1697-1, DOI: 10.1109/FMEC59375.2023.10306223.