DREAMIN: Channel-Aware Inter-Slices Radio Resource Scheduling for Efficient SLA Assurance

Daniel Campos[1], Gabriel M. Almeida[1], Mohammad J. Abdel-Rahman[2], Kleber V. Cardoso[1]

[1]Institute of Informatics (INF), Universidade Federal de Goiás (UFG), Goiânia, Brazil,

[2]School of Computing Sciences (SCC), Princess Sumaya University for Technology (PSUT), Amman, Jordan.

E-mail: {danielcampossilva, gabrielmatheus, kleber}@inf.ufg.br[1] E-mail: m.AbdelRahman@psut.edu.Jo[2]

Agenda

1. Introduction and related work
2. System model and problem formulation
3. DREAMIN scheduler
4. Evaluation
5. Conclusion and future work

Agenda

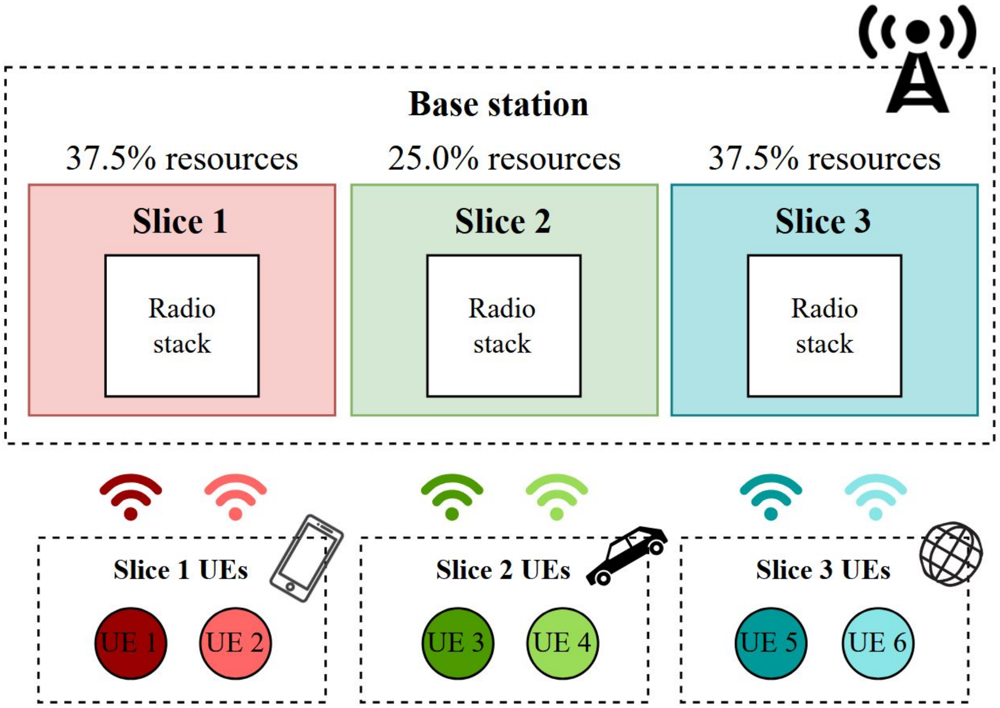
1. Introduction and related work
2. System model and problem formulation
3. DREAMIN scheduler
4. Evaluation
5. Conclusion and future work

Network slicing

Network slicing isolates resources among groups of User Equipments (UEs) who have similar service requirements

Radio resource scheduling determines the UE data transmission directly

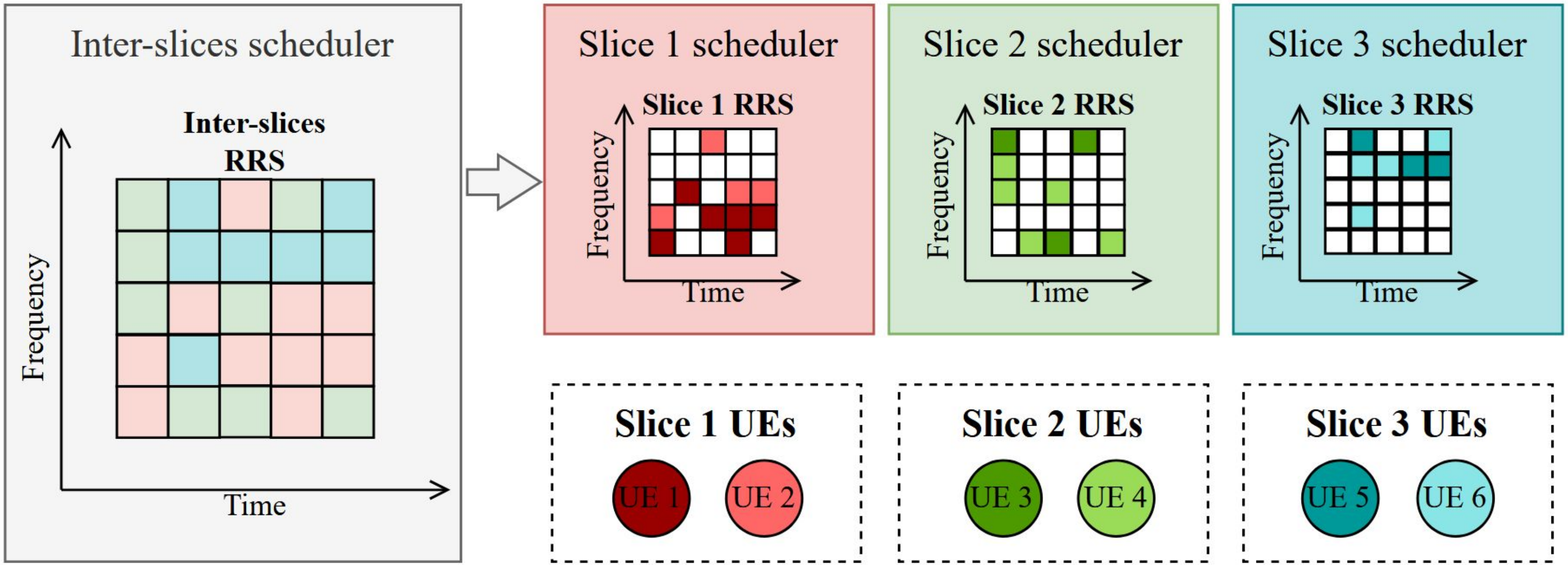
We schedule **resource block groups** (RBGs) at every **transfer time interval** (TTI)



Radio Resource Scheduling (RRS) with network slicing

Each slice has a **intra-slice scheduler** to distribute resources among its UEs

The inter-slice scheduler can only **schedule resources among slices**, not UEs



Service Level Agreement (SLA)

The scheduler main goal is to **ensure SLA**

Each slice specifies its own SLA: a set of required values for network metrics

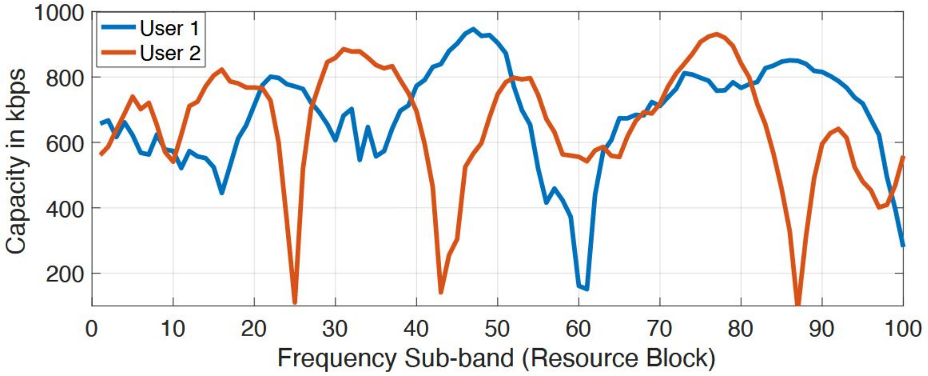
- **Latency**: 15 ms at maximum
- **Instantaneous capacity**: 1 Mbps at minimum
- **Long-term capacity**: 8 Mbps at minimum

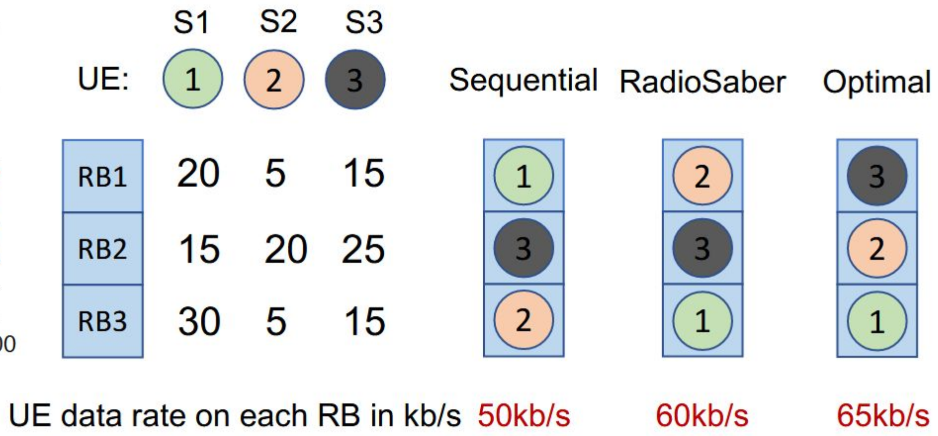
Channel-awareness

The capacity reached by a user may vary depending on **which RBG** it receives

The scheduler **chooses** not only how many RBGs, but **which RBGs** are allocated

An inter-slice scheduler must **predict the intra-slice scheduler** to be channel-aware





Source: [1] Y. Chen, R. Yao, H. Hassanieh, e R. Mittal, “Channel-Aware 5G RAN Slicing with Customizable Schedulers”.

Related work

[1] RadioSaber scheduler

- Has a **static quota of RBGs** for each slice
- Always allocates the RBG-Slice with higher capacity - as **maximum-throughput**
- Evaluates the scheduler in a **trace-driven simulation**
  - Leverages a Channel Quality Indicator **(CQI) dataset**

[2] Intent-aware Deep Reinforcement Learning (DRL) scheduler

- Based on **intent-drift**: normalized distance between network metric and SLA requirement
  - **Zero if meets** the requirement
  - We call it **SLA-drift (SLAd)** to broaden the term
- Develops a DRL agent whose **action is the ratio of resources** for each slice
  - **Non-channel-aware**
  - If changed the number of slices, needs to be **retrained**

Related work

- Most related work **allocate 100% resources** and **maximize/minimize network metrics**
  - This leads to **overprovisioning UEs**
  - **Minimizing allocated RBGs** in low-demand scenarios can reduce power consumption and **improve energy efficiency**

The problem we are solving

Inter-slice Radio Resource Scheduling

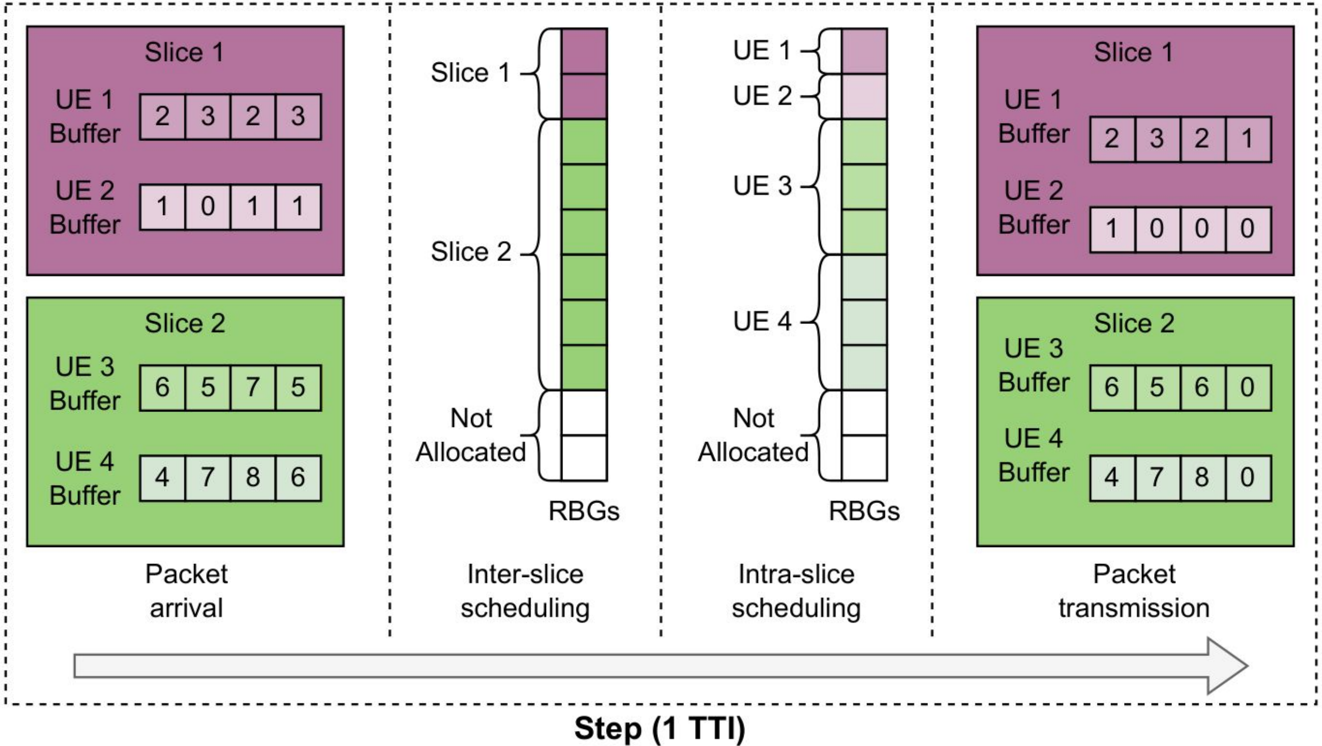
- Channel-aware
- Minimize SLAd
- Minimize resource usage

Agenda

1. Introduction and related work
2. System model and problem formulation
3. DREAMIN scheduler
4. Evaluation
5. Conclusion

System model

The system is represented as a **sequence of TTIs**



Sets, decision variables, and capacity

Sets

TTIs $\mathcal{T} = \{0, \ldots, |\mathcal{T}| - 1\}$

RBGs $\mathcal{R} = \{r_1, \ldots, r_{|\mathcal{R}|}\}$

Slices $\mathcal{S} = \{s_1, \ldots, s_{|\mathcal{S}|}\}$

Users (per-slice) $\mathcal{U}_s = \{u_j, \ldots, u_{j+|\mathcal{S}|-1}\}$

Users $\mathcal{U} = \bigcup_{s \in \mathcal{S}} \mathcal{U}_s$

SLA metrics (per-slice) $\mathcal{M}_s \subseteq \{CAP, LTC, LAT\}$

Decision variables

$$\rho^u_{r,t} \in \{0, 1\}$$

Capacity

$$c^u_t = \sum_{r \in \mathcal{R}} \rho^u_{r,t} \cdot C^u_{r,t}$$

Inputs

Arrived packets A_t^u

Buffer size (packets) Z_s

TTI length (seconds) I

Packet size (bits) P_s

Maximum latency (TTIs) L_s

Achievable capacity (bits/s) $C_{r,t}^u$

Time window (TTIs) for historical metrics TW

Adjusted time window (TTIs) for the first TTIs in the model

$$AW_t = \begin{cases} t+1, & \text{if } t < TW \\ TW, & \text{if } t \geq TW \end{cases}$$

Inputs

Initial historical capacity (bits/s)	y_u
SLA-required instantaneous capacity (bits/s)	Q_s^{CAP}
SLA-required long-term capacity (bits/s)	Q_s^{LTC}
SLA-required latency (TTIs)	Q_s^{LAT}
Weight of metric m in the slice's SLA	W_s^m
Weight of the slice	W_s

Buffer modeling

Packets in the buffer at the beginning of the TTI

$$b_t^u = \begin{cases} 0, & \text{if } t = 0 \\ b_{t-1}^u + A_{t-1}^u - k_{t-1}^u - df_{t-1}^u - dl_{t-1}^u, & \text{if } t > 0 \end{cases}$$

Packets dropped due to buffer full

$$df_t^u = \max(0, b_t^u + A_t^u - Z_u)$$

Fully sent packets

$$k_t^u = \min\left(\left\lfloor \frac{c_t^u \cdot I}{P_u} + \bar{k}_{t-1}^u \right\rfloor, b_t^u + A_t^u - df_t^u\right)$$

Partially sent packets

$$\bar{k}_t^u = \begin{cases} 0, & \text{if } b_{t+1}^u = 0 \text{ or } dl_t^u > 0 \\ \frac{c_t^u \cdot I}{P_u} - k_t^u, & \text{otherwise} \end{cases}$$

Buffer modeling

Packets with latency l in buffer at the end of TTI t

$$p_{t,l}^{u} = \max\left(0, b_{t-l}^{u} + A_{t-l}^{u} - df_{t-l}^{u} - \sum_{t'=t-l}^{t} k_{t'}^{u} - \sum_{t'=t-l}^{t-1} dl_{t'}^{u}\right)$$

Packets dropped due to maximum latency achieved

$$dl_{t}^{u} = p_{t,L_s}^{u}$$

SLA metrics

Instantaneous capacity (CAP)

$$CAP_t^u = c_t^u$$

Long-term capacity (LTC)

$$LTC_t^u = \frac{\sum_{t'=t-AW_t+1}^{t} c_{t'}^u}{AW_t}$$

Latency (LAT)

$$LAT_{t,s}^u = \max_{l \in 0,\ldots,L_s} (l \cdot (p_{t,l}^u > 0))$$

Metric SLAd

CAP SLAd

$$f_{t,s}^{CAP,u} = \begin{cases} \frac{Q_s^{CAP} - CAP_t^u}{Q_s^{CAP}}, & \text{if } CAP_t^u < Q_s^{CAP} \\ 0, & \text{if } CAP_t^u \geq Q_s^{CAP} \end{cases}$$

LTC SLAd

$$f_{t,s}^{LTC,u} = \begin{cases} \frac{Q_s^{LTC} - LTC_t^u}{Q_s^{LTC}}, & \text{if } LTC_t^u < Q_s^{LTC} \\ 0, & \text{if } LTC_t^u \geq Q_s^{LTC} \end{cases}$$

LAT SLAd

$$f_{t,s}^{LAT,u} = \begin{cases} \frac{LAT_{t,s}^u - Q_s^{LAT}}{L_s - Q_s^{LAT}}, & \text{if } LAT_t^u > Q_s^{LAT} \\ 0, & \text{if } LAT_t^u \leq Q_s^{LAT} \end{cases}$$

Aggregated SLAd

User SLAd

$$f^u_{t,s} = \sum_{m \in \mathcal{M}_s} f^{m,u}_{t,s} \cdot W^m_s$$

Slice SLAd

$$f_{t,s} = \frac{1}{|\mathcal{U}_s|} \sum_{u \in \mathcal{U}_s} f^u_{t,s}$$

Overall SLAd at base station

$$f_t = \sum_{s \in \mathcal{S}} f_{t,s} \cdot W_s$$

Objective function

Scenario indicator - scarce or plentiful

$$a_t = \begin{cases} 0, & \text{if } f_t = 0 \\ 1, & \text{if } f_t > 0 \end{cases}$$

Primary objective: minimize SLAd

Secondary : minimize resource usage

$$\underset{\rho^u_{r,t}}{\text{minimize}} \quad \sum_{t \in \mathcal{T}} \left(a_t(1 + f_t) + (1 - a_t)\frac{1}{|\mathcal{R}|} \sum_{u \in \mathcal{U}} \sum_{r \in \mathcal{R}} \rho^u_{r,t} \right)$$

Constraints

Allocate each RBG to at most 1 UE at the same TTI

$$\sum_{u \in \mathcal{U}} \rho^{u}_{r,t} \leq 1, \qquad \forall r \in \mathcal{R}, t \in \mathcal{T}$$

Historical capacity (calculated as an EWMA)

$$h^{u}_{t} = \begin{cases} y_u, & \text{if } t = 0 \\ (1 - \frac{1}{TW})h^{u}_{t-1} + \frac{1}{TW} c^{u}_{t-1}, & \text{if } t > 0 \end{cases}$$

Proportional Fairness model: always choose the UE with highest coefficient

$$\sum_{u' \in \mathcal{U}_s} \frac{C^{u'}_{r,t}}{h^{u'}_{t}} \cdot \rho^{u'}_{t,r} \geq \frac{C^{u}_{r,t}}{h^{u}_{t}} \cdot \sum_{u' \in \mathcal{U}_s} \rho^{u'}_{t,r},$$

$$\forall s \in \mathcal{S}, u \in \mathcal{U}_s, r \in \mathcal{R}, t \in \mathcal{T}$$

Agenda

1. Introduction and related work
2. System model and problem formulation
3. DREAMIN scheduler
4. Evaluation
5. Conclusion and future work

Proposed heuristic

Drift and REsource Allocation MINimization (DREAMIN) scheduler

- Fast **greedy** algorithm **approximating the optimal** solution in polynomial time
- As RadioSaber [1], checks through **every possible RBG-slice** allocation
- Chooses the one **most reducing the overall SLAd**
- **Stops** when
  - The overall SLAd is zero - **plentiful scenario**
  - All RBGs are allocated - **scarce cenario**
- Has the same complexity of RadioSaber [1]: $O(|\mathcal{R}|^2 \cdot |\mathcal{S}|)$
  - Assuming SLAd calculation in constant time - with the use of data structures

DREAMIN scheduler

Algorithm 1: DREAMIN allocation process.

Data: $\mathcal{R}$, $\mathcal{S}$, $\mathcal{U}$, $C^{u}_{r,t}$, t
Result: RBGs allocated for each slice $s \in \mathcal{S}$

for $s \in \mathcal{S}$, **do**
  allocation[s] $\leftarrow \emptyset$
for $u \in \mathcal{U}$ **do**
  cap[u] $\leftarrow 0$
  slad[u] $\leftarrow$ slad_for_cap(u, 0)
available_rbgs $\leftarrow R$
while |available_rbgs| > 0 **do**
  if $\sum_{u \in \mathcal{U}}$ slad[u] = 0 **then**
    return allocation
  best_reduction $\leftarrow 0$
  for $r \in$ available_rbgs **do**
    for $s \in \mathcal{S}$ **do**
      $u \leftarrow$ intra_sched(s, r)
      reduction $\leftarrow$ slad[u] - slad_for_cap(u, cap[u] + $C^{u}_{r,t}$)
      if best_reduction $<$ reduction / $|\mathcal{U}_s|$ * W_s **then**
        best_reduction $\leftarrow$ reduction / $|\mathcal{U}_s|$ * W_s
        $u^* \leftarrow u$
        $r^* \leftarrow r$
        $s^* \leftarrow s$
  cap[u^*] $\leftarrow$ cap[u^*] + $C^{u^*}_{r^*,t}$
  slad[u^*] $\leftarrow$ slad_for_cap(u^*, cap[u^*])
  available_rbgs $\leftarrow$ available_rbgs $\setminus \{r^*\}$
  allocation[s^*] $\leftarrow$ allocation[s^*] $\bigcup \{r^*\}$
return allocation

Initializing data structures

Plentiful scenario

Selecting the allocation that most reduces SLAd

Updating data structures

Scarce scenario

Agenda

1. Introduction and related work
2. System model and problem formulation
3. DREAMIN scheduler
4. Evaluation
5. Conclusion and future work

Simulation and parameters

Channel data: RadioSaber traces dataset [1]

- 475 TTIs
- 512 PRBs (100 MHz)

3GPP TS 38.214 Table 5.2.2.1-2 maps CQI to spectral efficiency

Slice	GBR	Non-GBR	DC-GBR
5QI	2	80	86
Service	Conversational video	Augmented reality	V2X messages
Demand/UE	12 Mbps[5]	50 Mbps[6]	10 Mbps[7]
Packet size P_u	256 bytes	1024 bytes	128 bytes
Max. latency L_u	200 ms	100 ms	3 ms
Weight W_s	0.333	0.333	0.333
Q_s^{CAP} (W_s^{CAP})	12 Mbps (0.5)	-	10 Mbps (1.0)
Q_s^{LTC} (W_s^{LTC})	-	50 Mbps (0.5)	-
Q_s^{LAT} (W_s^{LAT})	130 ms (0.5)	8 ms (0.5)	-

$Z_s = 256$ Kbits $\quad TW = 10$ TTIs

Each experiment is executed 20 times (with different randomness seeds)

Baselines

- **RadioSaber (RS)** [1] - channel-aware, maximizing throughput, fixed slice quotas

- **Weighted Round-Robin (RS)** - non-channel-aware, “random”, fixed slice quotas

- **Approximated solution (APPR)**: solved with IBM’s Constraint Programming Optimizer
  - Only evaluated in very small scenarios
  - Even in small scenarios, we can not obtain the optimality certificate for the solution
  - Limited to 1 hour running the solver

Scenarios

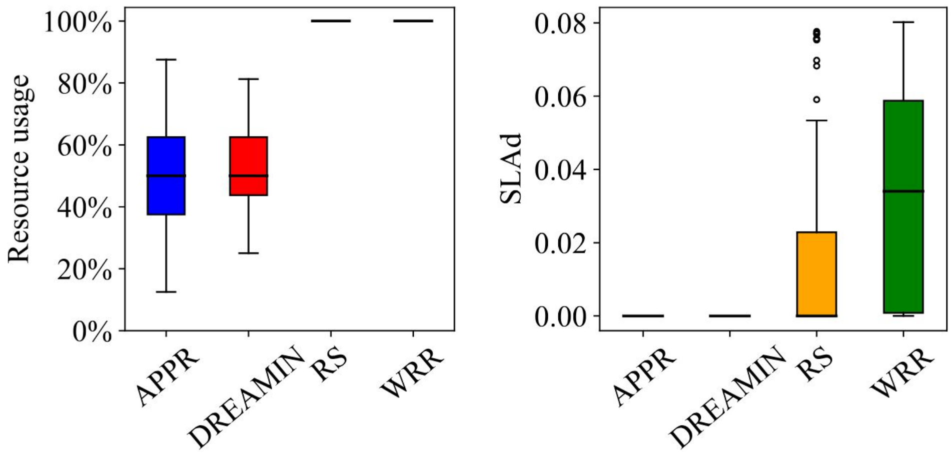
- **Small–scale**
  - **10 TTIs**
  - **8 RBGs with 32 PRBs/RBG** (same bandwidth, large granulosity on allocation)
  - **1 UE/slice**
  - **Plentiful x Scarce scenarios: requirements 3x more restrict**

- **Large–scale**
  - **475 TTIs**
  - **128 RBGs with 4 PRBs/RBG** (same bandwidth, large granulosity on allocation)
  - **Plentiful x Scarce scenarios:** 1 UE/slice for plentiful, 3 UEs/slice for scarce

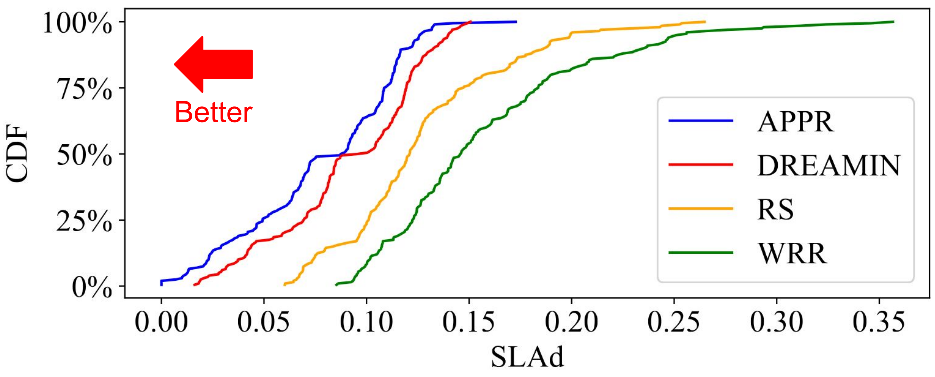
Small-scale scenario

DREAMIN approximates well the optimization model's solution

RS and WRR have similar performances - **fixed slice resource proportions**



Plentiful scenario

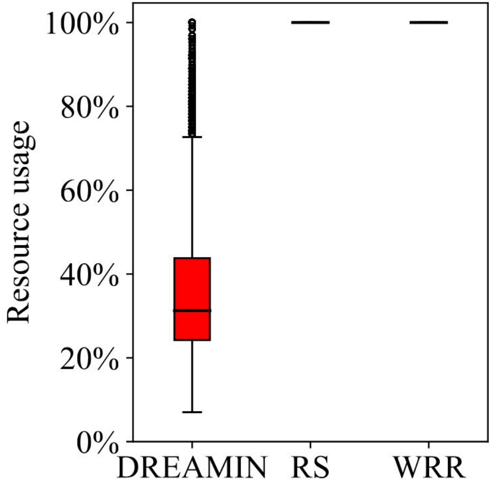


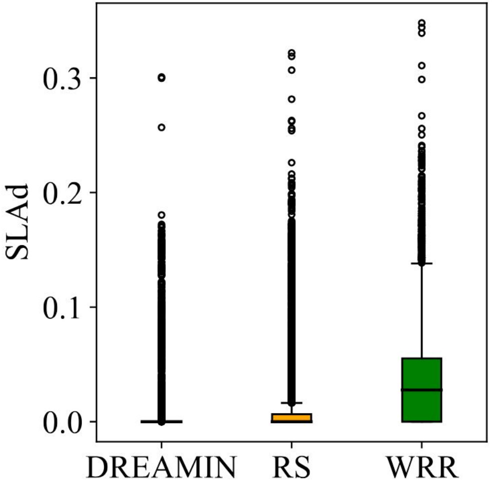
Scarce scenario

Large-scale plentiful scenario

The lower granularity (more RBGs) leads to **more efficient allocations** (DREAMIN's average resource usage drops **from 52% to 38%**)

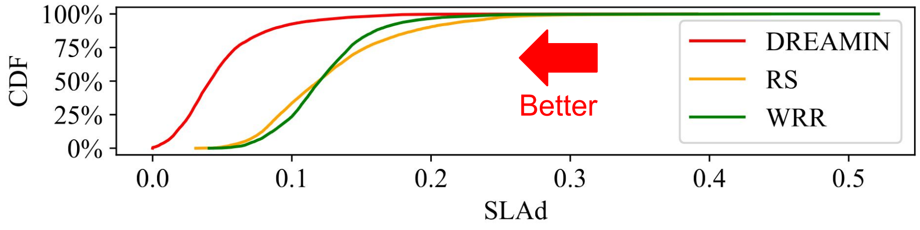
With more TTIs, there are **outlier** moments where **channel quality is broadly poor** and the **SLAd is greater than 0**

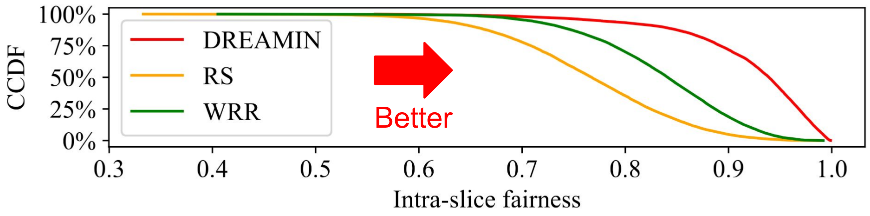




Large-scale scarce scenario

- DREAMIN's average **SLAd is 62% lower than RS's** despite also using 100%
- **Intra-slice fairness**: high values (>0.9) occur for DREAMIN in 72% of the TTIs, **14 times greater** than RS's 5%
    - **RS manipulates the proportional fair scheduler** (by predicting its allocations) to **impose unfair allocations** that maximize throughput, overprovisioning UEs with good channel qualities and starving UEs in worse conditions, which generates SLAd worse than WRR
    - The **SLAd is a metric correlated to fairness** since the slice SLAd is the average SLAd of its UEs
    - Such behaviour could only appear in a **scenario with more UEs/slice**





Agenda

Conclusion

This work approached the **channel-aware inter-slice radio resource scheduling problem oriented to minimizing SLA-drift and resource allocation**

- **Formulation** - **constraint programming** problem
- **Heuristic** - **DREAMIN approximates** well the **optimal** solution
- **RadioSaber** - **overperformed by DREAMIN**, which reduces allocation in 62% (plentiful scenario) and average SLAd in 62% (scarce scenario)
- **Fairness** - **SLAd-oriented** schedulers have **higher intra-slice fairness** indexes (DREAMIN has 14 times more occurrences of high values than RadioSaber)

Future work

- **Integrate DREAMIN with an xApp** (Open-RAN architecture) defining RRMPolicies to evaluate a **joint data-driven + channel-aware scheduling**

- **Increase problem scalability** by using other optimization techniques

- Evaluate more **complex 3GPP-compliant scenarios**
  - Multi-slice user association
  - MIMO channel simulation
  - Mixed numerology
  - Stochastic traffic models

References used in the presentation

- **[1] CHEN, Yongzhou et al. Channel-Aware 5G RAN slicing with customizable schedulers. In: 20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23). 2023. p. 1767-1782.**

- **[2] NAHUM, Cleverson Veloso et al. Intent-aware radio resource scheduling in a ran slicing scenario using reinforcement learning. IEEE Transactions on Wireless Communications, v. 23, n. 3, p. 2253-2267, 2023.**

Thank you!

https://github.com/LABORA-INF-UFG/paper-DGMK-2024/

dcamposfsa@vt.edu

- Supporting agencies
    - Ministério da Ciência, Tecnologia, Inovações e Comunicações (MCTIC)
    - Comitê Gestor da Internet no Brasil (CGI.br)
    - Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP)
    - Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPQ)
    - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES)