Comparison of Scheduling Algorithms for Multiple Mobile Computing Edge Clouds

Article *in* Simulation Modelling Practice and Theory · October 2018

DOI: 10.1016/j.simpat.2018.10.005

6 authors, including:

Tien Van Do
Budapest University of Technology and Economics
128 PUBLICATIONS **701** CITATIONS
SEE PROFILE

Csaba Rotter
Nokia Bell Labs, Hungary, Budapest
15 PUBLICATIONS **47** CITATIONS
SEE PROFILE

Comparison of Scheduling Algorithms for Multiple Mobile Computing Edge Clouds

Tien Van Do[a,b,*], N. H. Do[b], H. T. Nguyen[b], Csaba Rotter[c], Attila Hegyi[c], Peter Hegyi[c]

[a] *Division of Knowledge and System Engineering for ICT (KSE-ICT)*
Ton Duc Thang University, Ho Chi Minh City, Vietnam
[b] *Analysis, Design and Development of ICT systems (AddICT) Laboratory,*
Department of Networked Systems and Services,
Budapest University of Technology and Economics,
H-117, Magyar tudósok körútja 2., Budapest, Hungary
[c] *Nokia Bell Labs Hungary*
Bókay János 36-42, Budapest, Hungary

Abstract

5G networks are anticipated to provide service for vertical industries. Mobile Edge Computing (MEC) framework is a physical infrastructure improvement to host mobile applications for the introduction of new service provisioning in 5G networks. Application Orchestration is responsible for scheduling user sessions to application instances. In this paper, we propose four scheduling algorithms to route and assign sessions in a system with multiple MEC clouds. Results show that scheduling based on the occupancy level of MECs is quite robust in various scenarios and can be applied with the fractional guard channel policy to provide service.

Keywords: Mobile Edge Computing, Mobile Edge Applications, Clouds, Resource allocation, Scheduling algorithm

1. Introduction

5G networks are expected to provide the service of numerous demands with different requirements from vertical industries [1]. For example, vehicular communication needs low latency connections, whereas a live video

*Correspondent author: Tien Van Do, email: dovantien@tdtu.edu.vn,do@hit.bme.hu

service requires the data downlink rate of 300Mbps and the uplink rate of 50Mbps in 200-2500/km^2 connection density [1]. Therefore, the concept of network slicing has been introduced by the third Generation Partnership Project (3GPP) [1]. Multiple on-demand (business-driven, technology-driven) network slices are created through mapping requirements to network slice descriptions and realising network slice descriptions in the shared infrastructure. Specific telecommunication service can be provided by a slice with separate management, service, control and data planes that utilise the resource of network and cloud infrastructure. Each slice instant may include a number of the physical and the virtual network functions, network applications and the underlying cloud infrastructure.

Mobile Edge Computing (MEC) [2] provides a cloud-computing infrastructure and IT service environment at the edge of 5G networks [4]. Network slices and different applications can be constructed with the help of multiple MEC clouds to provide innovative service with low latency and high bandwidth. In MEC, *Application Orchestration* is responsible to deploy, terminate and schedule user sessions to application instances. During the lifetime of a session, the connectivity between the assigned Mobile Edge Application (MEA) instance and the mobile user should be maintained by the appropriate routing when handovers happen. If the latency increases, then a procedure for the application state relocation is initiated by *Application Orchestration*. That is, a scheduling algorithm is executed to find the appropriate MEC cloud for the ongoing user session. In this paper, we propose four scheduling algorithms (the `Random`, the `Index`, the `Smallest-Occupancy-Level` and the `Largest-Free-Slots`) to route and assign sessions to MEC clouds. Results show that scheduling based on the occupancy level of MECs is quite robust in various scenarios and can be applied with the fractional guard channel policy to provide service.

In Section 2 we briefly summarise works related to our research. In Section 3 the problem is introduced in the context of the 5G service provisioning. In Section 4 aspects for the design of resource allocation algorithms in Mobile Edge Computing and a heuristic scheduling algorithm are presented. In Section 5, some numerical results are provided to compare some design alternatives. Finally, Section 6 concludes this paper.

2. Related works

Recently Ahmed et al. [5] provided a brief overview of some solutions and challenges of the introduction of MEC. A critical problem is the optimisation of resource utilisation since inadequate resource allocation may result in overloading MECs unnecessarily. Satria et al. [6] devised a recovery scheme for MEC failures where an overloaded MEC would share its load with other MECs if they are are in the vicinity.

Chen et al. [7] studied the offloading of computation to MEC systems. They set up a model where users had to decide whether offloading should be applied. They approached the problem with game theoretical tools and suggested an algorithm that reached Nash equilibrium. Liu et al. [8] considered the similar problem and formulated it as a Markov decision process for which they proposed an algorithm to minimise the delay of tasks subject to power consumption constraints. Sardellitti et al. [9] also investigated offloading algorithms taking into account the intercell interference. Zhang et al. [10] devised an energy-efficient offloading scheme in MECs. Mao et al. [11] studied offloading procedures with energy harvesting devices.

Wang et al. [13] proposed a partial offloading solution with dynamic voltage scaling where the user device could lower its power consumption and processing power after handing a portion of the task to the MEC. Hegyi et al. [14] studied the orchestration of applications in MECs where a Mobile Edge Orchestrator is responsible for managing the lifecycle of Mobile Edge Applications for efficient operation. Ketykó et al. [15] identified the resource sharing in a MEC system as a multi knapsack problem. For more extensive surveys on MECs see [16, 17, 18].

To the best of our knowledge, this paper is the first work, that discusses user session scheduling problem in the context of 5G service provisioning. Our proposed algorithms can be applied by *Application Orchestration* to enhance the utilisation of the system.

3. The problem of scheduling user sessions in the MEC framework

5G networks is expected to address requirements from the fourth industrial revolution, like a machine to machine communication, massive IoT access and remote robotics. Mobile Edge Computing (MEC) defines the necessary technology to ensure that a mobile edge application can be placed and operated on the network edge. The associated entities are defined by

MEC Framework [2] and can be grouped in three main levels as indicated in Figure 1.

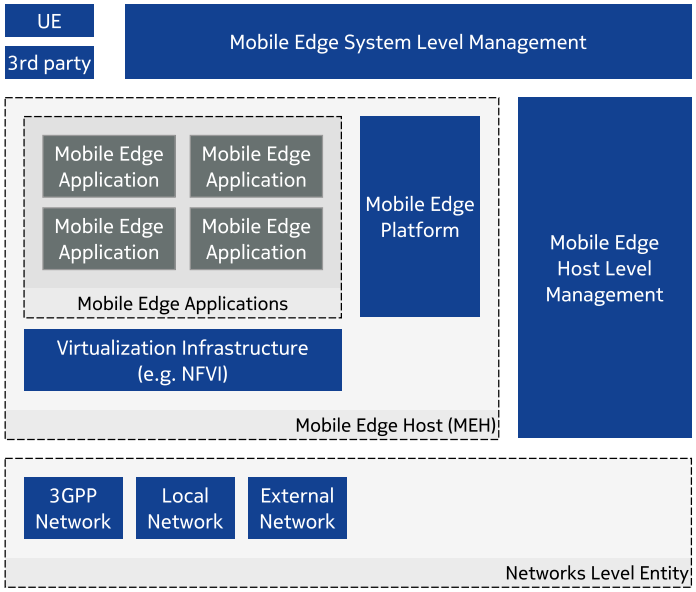


Figure 1: MEC framework

- The mobile edge system level entity defines the Operations Support System, the lifecycle management of user applications and the mobile edge orchestrator. The functions are in charge of the orchestration and management of resources required for the execution of edge applications.

- The mobile edge host level entity is responsible for Mobile Edge host level management and defines the Mobile Edge Host (MEH), which includes the Mobile Edge Platform and the underlying virtualisation infrastructure where mobile edge applications are placed to offer services for edge users [2].

- The network level entities include infrastructure needed to connect towards different networks (the 3GPP cellular network, the local network and the external network).

Mobile Edge Host contains the virtualisation infrastructure, which is responsible for providing resources like CPU, memory, network, storage for the application running on the edge. MEH also includes the essential functionalities to run mobile edge applications in a particular virtual infrastructure.

Mobile edge applications are software components and can be executed inside virtual machines or containers. The resources for mobile edge applications are provided by the infrastructure of physical servers and networks.

The Mobile Edge Orchestrator (MEO) has on overall knowledge and up-to-date information about the network topology, the available and used resources in mobile edge hosts. It also maintains a list of on-boarded applications, enforces the operator policies and also prepares the virtual infrastructure managers to handle the respective applications. MEO is responsible for the instantiation and termination and relocation of the application based on the overall resource availability and the resource requirements for the running applications.

Application Orchestration [14] gives additional zero-touch automation on top of the MEO to support use cases like on-demand scaling or the application mobility since the standard does not consider the automation as part of the MEO's functionality at present. Therefore, the *Application Orchestration* utilises and extends the information that is available on the system level (deployed application instance repository, resource availability, network topology) with the application-specific knowledge. *Application Orchestration* should maintain the mapping of the user session to a specific application instance. We assume an application instance will serve multiple user sessions.

According to the 5G Architecture Working Group of the 5GPPP Initiative [4], functions to realize the 5G architecture are divided into layers (service, management and orchestration, control layer, multi-domain network operating system facilities, and data layer). Practically, *Application Orchestration* executes tasks in the service layer. Based on automation rules *Application Orchestration* makes decisions regarding:

- deployment or termination of new or existing application instances,
- assignment of new or migration of existing user sessions (scheduling) to application instances.

The application deployment level optimises the cloud resource usage via application lifecycle management events. The session assignment includes the routing of a session to the appropriate application instance. During the

lifetime of a session, a user may travel in the area, *Application Orchestration* changes routing to maintain the connectivity between the assigned MEA and the user. If the latency increases, then a scheduling algorithm is executed to find the appropriate MEC cloud for the ongoing user session, and a procedure for the application state relocation is initiated by *Application Orchestration*. Note that software vendors should design MEA to assist the state relocation.

4. Scheduling Algorithms

In this section, we propose scheduling algorithms that can be applied to a problem in Section 3. We consider set $\mathcal{E}$ of evolved NodeBs (eNodeB) that provide users the connectivity to the 5G network through the radio interface. We assume that the mobile network operator deploys MEC clouds to provide cloud computing resources. Let Γ be the set of MEC clouds to host N types of Mobile Edge Applications (MEA). The maximum number of user sessions that request the MEA type-n (for $n = 1, \ldots, N$) service in MEC γ (for $\gamma \in \Gamma$) is $C_{n,\gamma}$.

As a mobile user can freely move inside the area served by the considered 5G network, the mobile user may be attached to different eNodeBs to obtain the connectivity. Furthermore, the mobile user may start a user session to get a service from MEA. If eNodeB $\epsilon_{e(t)}$, ($\epsilon_{e(t)} \in \mathcal{E}$), provides the connectivity to mobile user e at time instant t, it is possible that only a subset of Γ can provide service to the user due to the latency requirement. Let $\Gamma_{n,\epsilon_{e(t)}}$ denote the set of MEC clouds ($\Gamma_{n,\epsilon(t)} \subseteq \Gamma$) where the MEA type-$n$ instances can provide service for mobile users that are connected to eNodeB $\epsilon_{e(t)}$. When a user session is requested by the mobile user, the **scheduling entity** chooses the appropriate MEC from $\Gamma_{n,\epsilon_{e(t)}}$ to assign the user session. Also, the mobile user may leave eNodeB $\epsilon_{e(t^-)}$ and enter another one $\epsilon_{e(t)}$ at time instant t. In the case of a handover the scheduling entity might need to place the session into MEC ($\gamma \in \Gamma_{n,\epsilon_{e(t)}}$) with a session migration if the latency requirement could not be fulfilled with a MEC that serves users attached to eNodeB $\epsilon_{e(t^-)}$.

Algorithm 1 describes a session assignment for a new user session or an ongoing handover session initiated by user e at eNodeB $\epsilon_{e(t)}$. The procedure takes the set of available MECs $\Gamma_{n,\epsilon_{e(t)}}$ and looks for MEC $\gamma_t o$ using `FIND_SUITABLE_MEC`($\Gamma_{n,\epsilon_{e(t)}}$). If the request could not be fulfilled because of the shortage of the capacity, `Nil` value is returned.

In this paper, we consider and compare several alternatives to choose a MEC to fulfil requests (see Algorithm 2).

- The `Random` (RND) rule selects a random MEC with free slots.
- The `Index` (IND) rule chooses the first MEC with at least one free slot.
- In the `Smallest-Occupancy-Level` (SOL) rule, MECs are sorted based on the ratio of occupied slots and the capacity (i.e., $W_\gamma(t)/C_\gamma$ for MEC γ). The MEC with the smallest ranked value is chosen;
- The `Largest-Free-Slots` (LFS) rule selects the MEC with the largest number of free slots.

Note that the complexity of the scheduling algorithms is $O(|\Gamma|)$, where $|\Gamma|$ is the number of MECs.

Algorithm 1 Assignment of a new/ongoing session to an available MEC

Input: $\epsilon_{e(t)}$ the identifier of eNodeB which provides the connectivity of UE e at time epoch t
Output: Routing the session to a MEC or a blocking decision

```
1: procedure ASSIGN_SESSION(ε_e(t))
2:     γ_to ← FIND_SUITABLE_MEC(Γ_{n,ε_e(t)})
3:     if (γ_to ≠ Nil) then
4:         Route the session to MEC γ_to
5:     else
6:         The session is blocked.
7:     end if
8: end procedure
```

5. A numerical study

We investigate a hypothetical case study on the introduction of 5G to the inner part ($9\ km \times 9\ km$) of Budapest (see Figure 2). We assume that eNodeBs are placed to cover a square area of $150\ m \times 150\ m$. Therefore, 3600 eNodeBs are needed. We suppose that 5 MECS are deployed and $\sum_{\gamma\in\Gamma} C_{n,\gamma} = 27120$.

Algorithm 2 Algorithms to choose a MEC to serve a new/ongoing session

Input: Γ_{to}, a set of MECs
Output: return a feasible MEC or `Nil`
Initialization: $\Gamma_{ret} \leftarrow \emptyset$, $\gamma_{to} \leftarrow$ `Nil`, $bestR \leftarrow -1$

`Random` **rule (RND): Choose a random MEC with free slots**

procedure FIND_SUITABLE_MEC(Γ_{to})
  for ($\gamma \in \Gamma_{to}$) **do**
    if ($W_\gamma(t) < C_\gamma$) **then** ▷ Found a MEC with free slots
      $\Gamma_{ret} \leftarrow \Gamma_{ret} \cup \gamma$
    end if
  end for
  if ($\Gamma_{ret} \neq \emptyset$) **then**
    return a random MEC from Γ_{ret}
  else
    return `Nil`
  end if
end procedure

`Index` **rule (IND): Choose a first MEC with free slots**

procedure FIND_SUITABLE_MEC(Γ_{to})
  for ($\gamma \in \Gamma_{to}$) **do**
    if ($W_\gamma(t) < C_\gamma$) **then** ▷ Found a MEC with free slots
      return γ ▷ Return a first MEC with a free slot
    end if
  end for
  return `Nil`
end procedure

`Smallest-Occupancy-Level` **rule (SOL): Find a MEC with the smallest session occupancy ratio**

procedure FIND_SUITABLE_MEC(Γ_{to})
  for ($\gamma \in \Gamma_{to}$) **do**
    if ($W_\gamma(t) < C_\gamma$) **then** ▷ Found a MEC with free slots
      $rank \leftarrow 1 - W_\gamma(t)/C_\gamma$ ▷ Rank based on the occupancy ratio
      if ($rank > bestR$) **then**
        $\gamma_{to} \leftarrow \gamma$, $bestR \leftarrow rank$
      end if
    end if
  end for
  return γ_{to}
end procedure

`Largest-Free-Slots` **rule (LFS): Find a MEC with the largest number of free slots**

procedure FIND_SUITABLE_MEC(Γ_{to})
  for ($\gamma \in \Gamma_{to}$) **do**
    if ($W_\gamma(t) < C_\gamma$) **then** ▷ Found a MEC with free slots
      $rank \leftarrow C_\gamma - W_\gamma(t)$ ▷ Rank by number of free slots
      if ($rank > bestR$) **then**
        $\gamma_{to} \leftarrow \gamma$, $bestR \leftarrow rank$
      end if
    end if
  end for
  return γ_{to}
end procedure

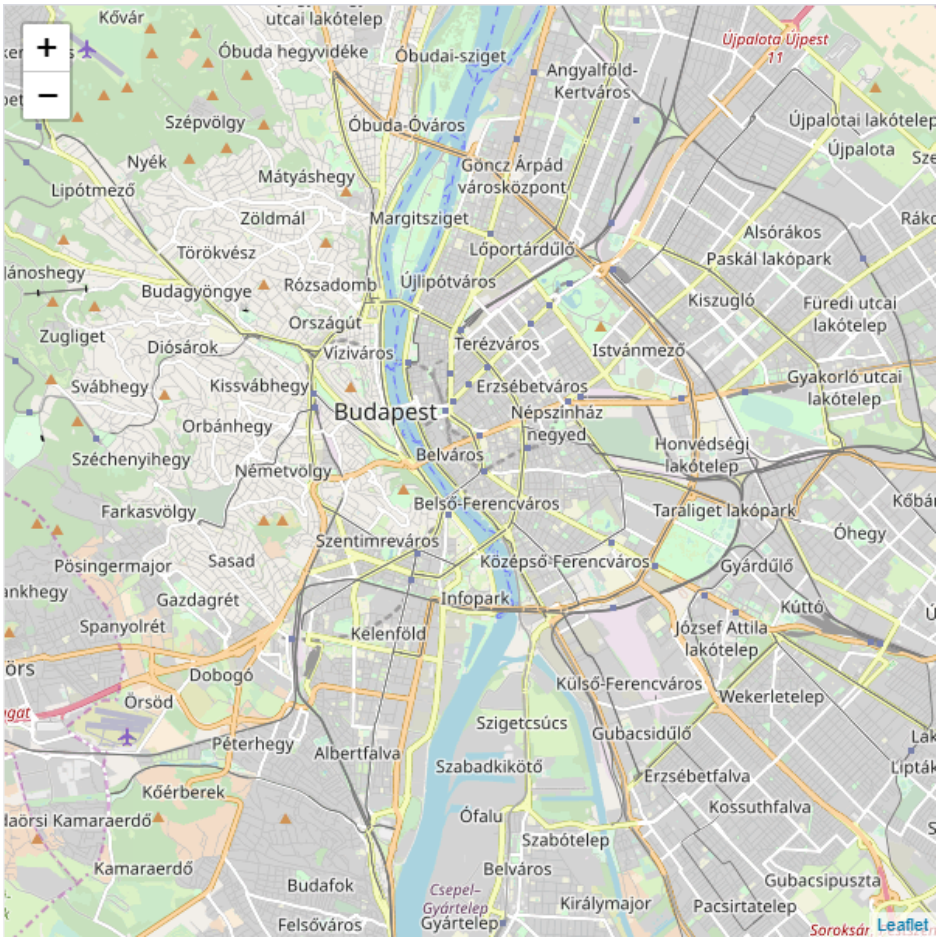
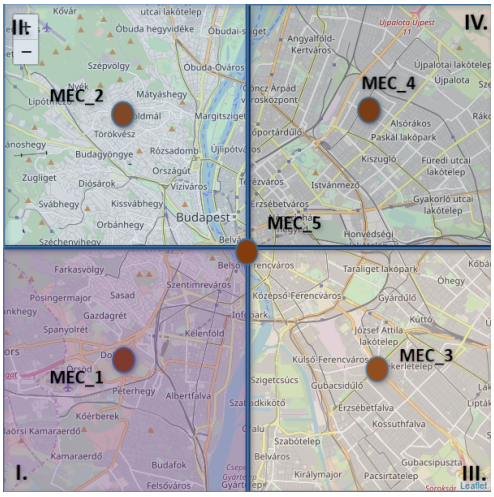


Figure 2: The map of the 9 $km \times 9$ km part of Budapest taken from OpenStreetMap — `https://www.openstreetmap.org/`

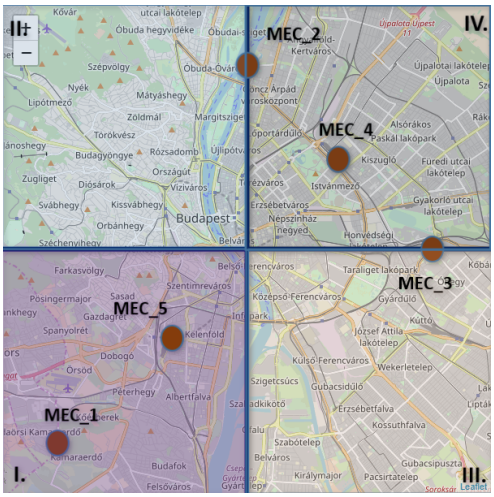
5.1. The stationary performance of scheduling algorithms

In this subsection, we compare the performance of scheduling algorithms, the average duration of user sessions and the average speed of users in the area. Results are obtained with the confidence level of 99%, and the relative precision (i.e., the ratio of the half-width of the confidence interval and the mean of collected observations) of 0.01.

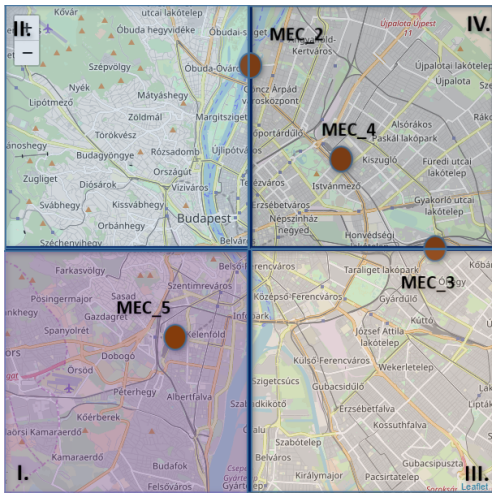
Based on the geography and the population distribution, 4 regions (the south part in the left bank – region I, the north part in the left bank –region II, the south part in the right bank –region III, the north part in the right bank



(a) MEC placement in Scenario 1



(b) MEC placement in Scenario 2



(c) MEC placement in Scenario 3

Figure 3: MEC placements

– region IV, of Danube) are created. Three scenarios are considered in this paper as illustrated in Figure 3. In Table 1, we show matrices that describes which MECs can be used to serve users in a certain region. Such matrices are determined from the latency requirements of mobile edge applications, the traffic condition and the underlying network infrastructure. The capacities

Table 1: Assignments of mobile users in regions to MECs

Assignment:	Scenario 1				Scenario 2				Scenario 3			
Region:	I.	II.	III.	IV.	I.	II.	III.	IV.	I.	II.	III.	IV.
MEC_1	x	x	x		x							
MEC_2	x	x		x		x		x		x		x
MEC_3	x		x	x			x	x			x	x
MEC_4		x	x	x		x	x	x		x	x	x
MEC_5	x	x	x	x	x	x	x		x	x	x	

of MECs in three scenarios are summarised in Table 2.

Table 2: Capacities

	MEC_1	MEC_2	MEC_3	MEC_4	MEC_5
Scenario 1	5424	5424	5424	5424	5424
Scenario 2	5424	5424	5424	5424	5424
Scenario 3	0	4520	4520	6780	11300

In this subsection, we assume that mobile users are evenly distributed in the area and generate sessions according to a Poisson process with rate λ. The holding time of a user session is exponentially distributed with parameter μ. During a session, the Gauss-Markov Mobility Model [19] is applied to drive the mobility of users with $\alpha = 0.7$[1] and with the average travel speed ($\bar{s}$) equal to 1.5 m/s, 5 m/s and 10 m/s, respectively.

The blocking probability of new sessions and the dropping probability of ongoing sessions (P_F and P_H) vs. the average holding time of sessions, the average travelling speed for Scenario 1 are plotted in Figure 4. Also the P_F and the P_H curve versus λ and the average travelling speed at $1/\mu = 600$ s are depicted in Figure 5. It can be observed that four scheduling algorithms result in almost the same performance. The `SOL` and the `LFS` algorithms achieve slightly better performance than other two algorithms.

Figures 6 and 7 plot the blocking probability and the dropping probability

[1] α, where $0 \leq \alpha \leq 1$ is the tuning parameter used to vary the randomness of speed and direction.

vs the average holding time of sessions, the average travelling speed and λ for Scenario 2. It is observed that the `SOL` and the `LFS` algorithms can improve P_F and P_H with one order of magnitude in comparison to the `RND` and the `IND` algorithms.

One can conclude from Figure 8 and Figure 9 that the `SOL` and the `LFS` algorithms also outperform the `RND` and the `IND` algorithms in Scenario 3. Furthermore, the `SOL` algorithm can reduce the dropping probability of ongoing sessions too (see Figure 9).

5.2. The application of Fractional Guard Channel Policy

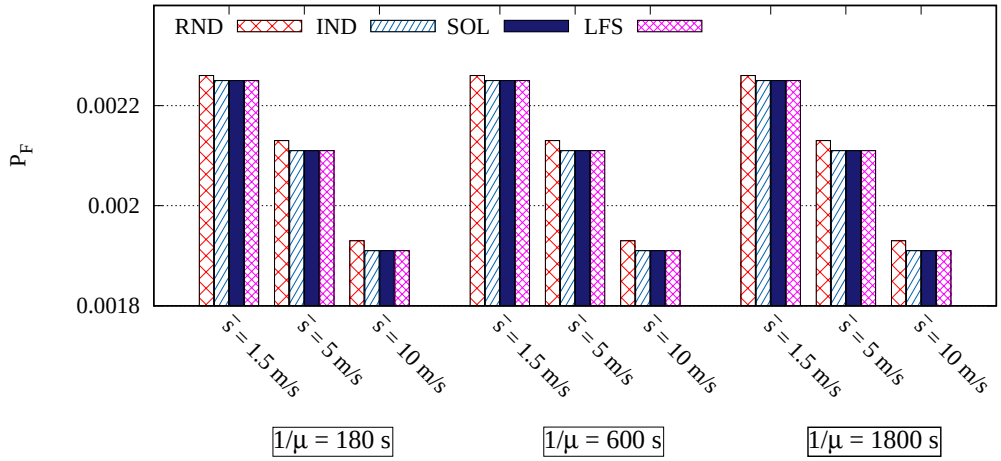
The dropping probability of ongoing user sessions (i.e., P_H) is high if the average holding time of sessions and the travelling speed of users are high (see Figure 8). Now we apply the Fractional Guard Channel policy (FGC) [20, 21] to balance the blocking probability of new sessions and the dropping probability of ongoing sessions.

For a given eNodeB ϵ, $C_\epsilon(t) = \sum_{\gamma \in \Gamma_\epsilon(t)} C_\gamma$ and $W_\epsilon(t) = \sum_{\gamma \in \Gamma_\epsilon(t)} W_\gamma(t)$ are introduced. When a new session request arrives to eNodeB ϵ_{to} at time epoch t, it can be accepted with probability $\beta_{\epsilon_{to}}(t)$ using the FGC policy. Some variants of the FGC policy are defined to handle a new session request arriving to eNodeB ϵ_{to} as follows [21]:

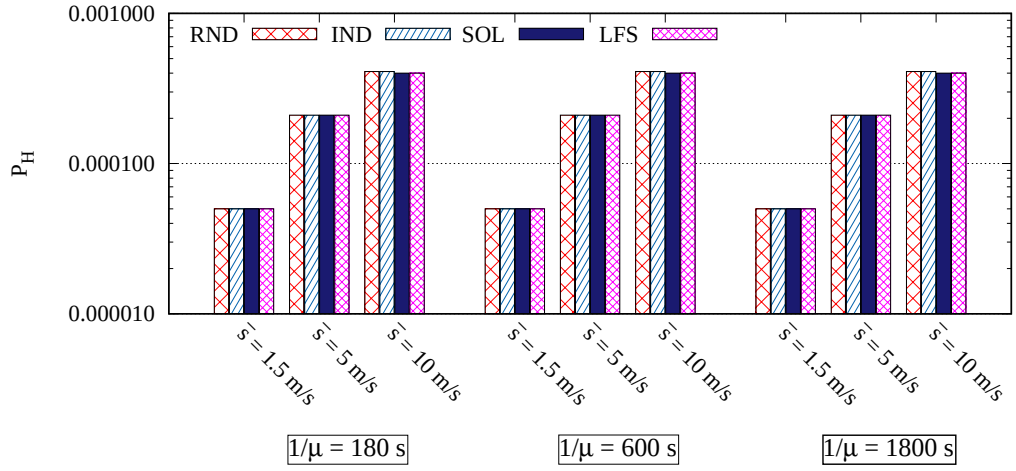
- the Non Prioritization Scheme (NPS): $\beta_{\epsilon_{to}}(t) = 1$ $(0 \leq W_{\epsilon_{to}}(t) < C_{\epsilon_{to}}(t))$. That is, all new session requests are accepted for scheduling with Algorithm 1,
- the Uniform Fractional Guard Channel Policy (UFGC): $\beta_{\epsilon_{to}}(t) = \beta$ $(0 \leq W_{\epsilon_{to}}(t) < C_{\epsilon_{to}}(t))$,
- the Quasi-Uniform FGC (QUFGC):

$$\beta_{\epsilon_{to}}(t) = \begin{cases} 1 & \text{if } 0 \leq W_{\epsilon_{to}}(t) \leq C_{\epsilon_{to}}(t) - \lfloor g_{\epsilon_{to}} \rfloor - 2, \\ 1 - g_{\epsilon_{to}} + \lfloor g_{\epsilon_{to}} \rfloor & \text{if } C_{\epsilon_{to}}(t) - \lfloor g_{\epsilon_{to}} \rfloor - 1 \leq W_{\epsilon_{to}}(t) < C_{\epsilon_{to}}(t), \\ 0 & \text{otherwise,} \end{cases}$$

  where $g_{\epsilon_{to}}$ denotes the real number of reserved channels,



(a) The blocking probability of new sessions (P_F)

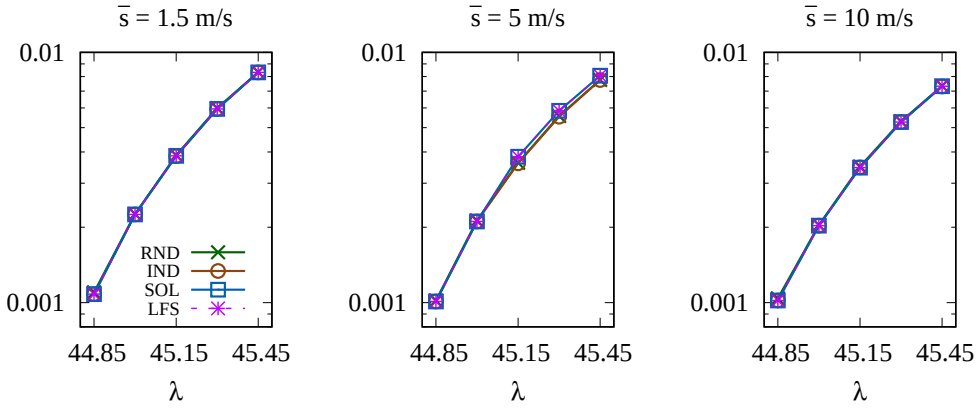


(b) The dropping probability of ongoing sessions (P_H)

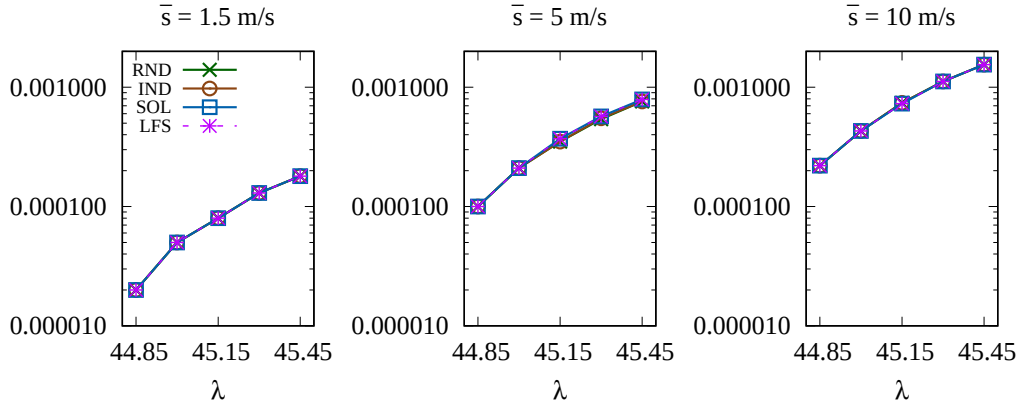
Figure 4: Comparing the Grade of Service (GoS) metrics when applying four scheduling algorithms for $\lambda/\mu = 27000$ and three mean travelling speeds in Scenario 1

- the Limited average Fractional Guard Channel Policy (LFGC):

$$\beta_{\epsilon_{to}}(t) = \begin{cases} 1 & \text{if } 0 \leq W_{\epsilon_{to}}(t) \leq C_{\epsilon_{to}}(t) - \lfloor g_{\epsilon_{to}} \rfloor - 2, \\ 1 - g_{\epsilon_{to}} + \lfloor g_{\epsilon_{to}} \rfloor & \text{if } W_{\epsilon_{to}}(t) = C_{\epsilon_{to}}(t) - \lfloor g_{\epsilon_{to}} \rfloor - 1, \\ 0 & \text{otherwise,} \end{cases}$$



(a) The blocking probability of new sessions (P_F)

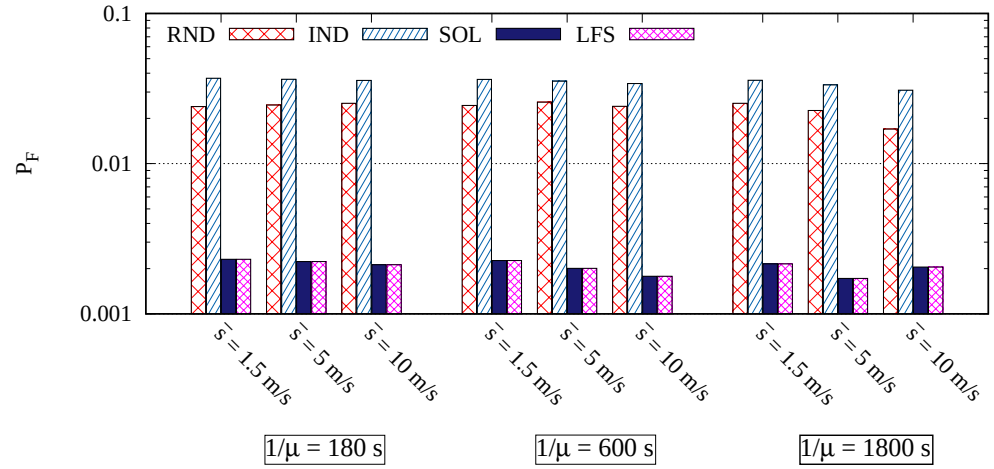


(b) The dropping probability of ongoing sessions (P_H)

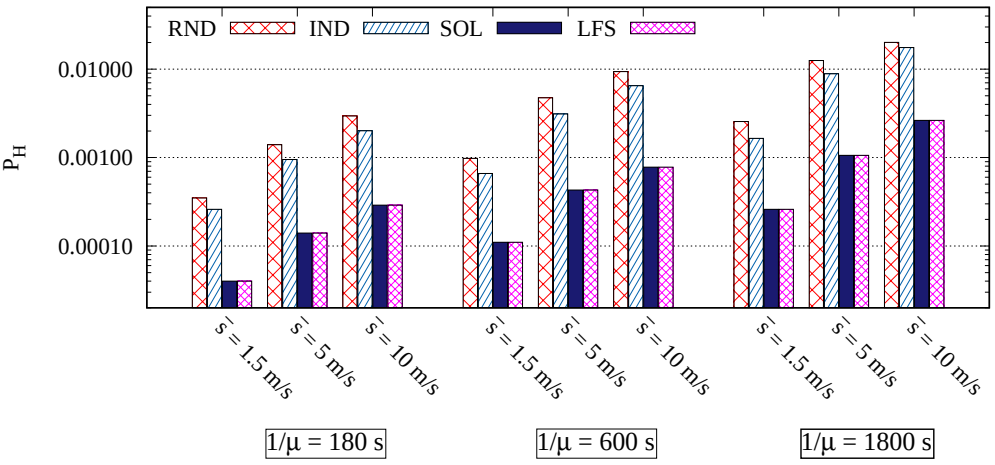
Figure 5: The GoS metrics when applying four scheduling algorithms versus λ for $1/\mu = 600$ s in Scenario 1

where $g_{\epsilon_{to}}$ denotes the real number of reserved channels.

Figure 10 plots the blocking probability of new sessions and the dropping probability of ongoing sessions when the `SOL` algorithm and the Fractional Guard Channel policy are applied in Scenario 3 for $\bar{s} = 5$ m/s and $1/\mu = 1800$ s. From Figure 10, the LFGC policy is the most effective tool to balance the blocking probability and the dropping probability.



(a) The blocking probability of new sessions (P_F)

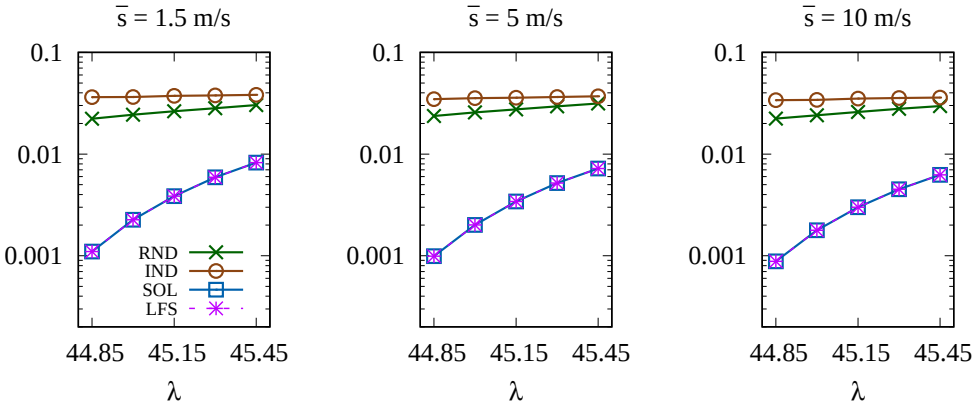


(b) The dropping probability of ongoing sessions (P_H)

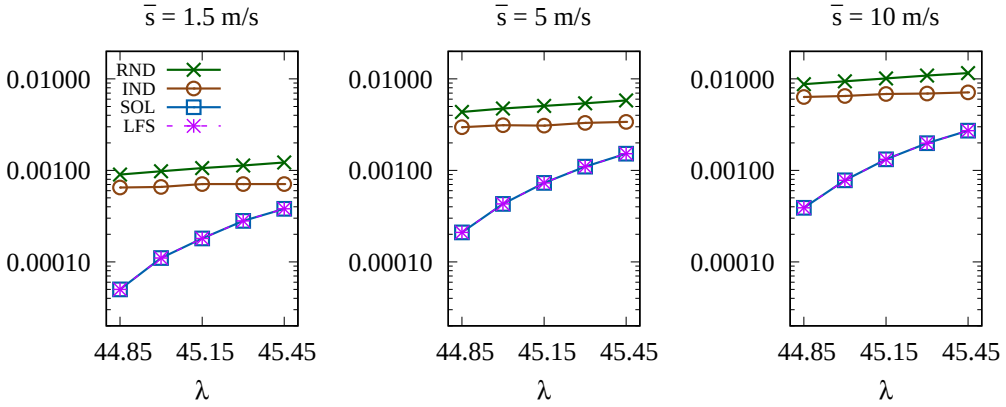
Figure 6: Comparing the GoS metrics when applying four scheduling algorithms for $\lambda/\mu = 27000$ and three mean travelling speeds in Scenario 2

5.3. The impact of users travelling by car in the city

To investigate the impact of users travelling by car in the city, we define office quarters and the sleeping city of Budapest as illustrated in Figure 11.



(a) The blocking probability of new sessions (P_F)

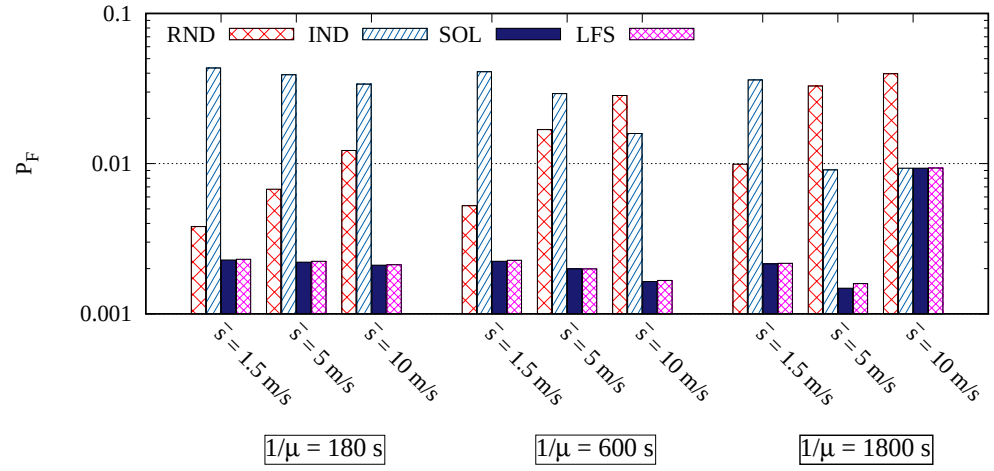


(b) The dropping probability of ongoing sessions (P_H)

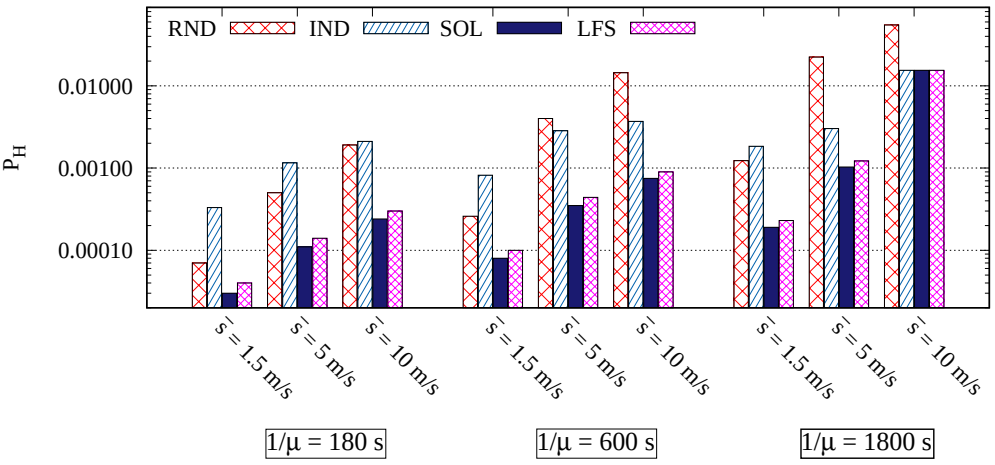
Figure 7: The GoS metrics when applying four scheduling algorithms versus λ for $1/\mu =$ 600 s in Scenario 2

A subscriber participating in a scene called "Office work" typically stays in a building in the office quarter during office hours, browsing the news and getting information for work, but typically does not download movies. A subscriber in a scene named "Park" typically stays in a park in the late afternoon listening to music, but typically does not use real-time applications.

According to the story people set off to work in three waves: 5, 7 and 9 a.m. They finish work also in three waves: 3, 5 and 7 p.m. After work, people go home. There are people (10%) who do not stay at home during sleeping



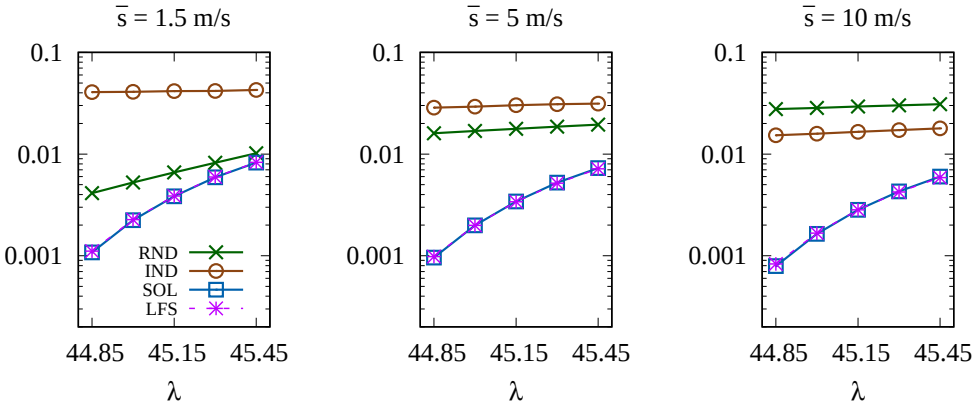
(a) The blocking rate of new sessions (P_F)



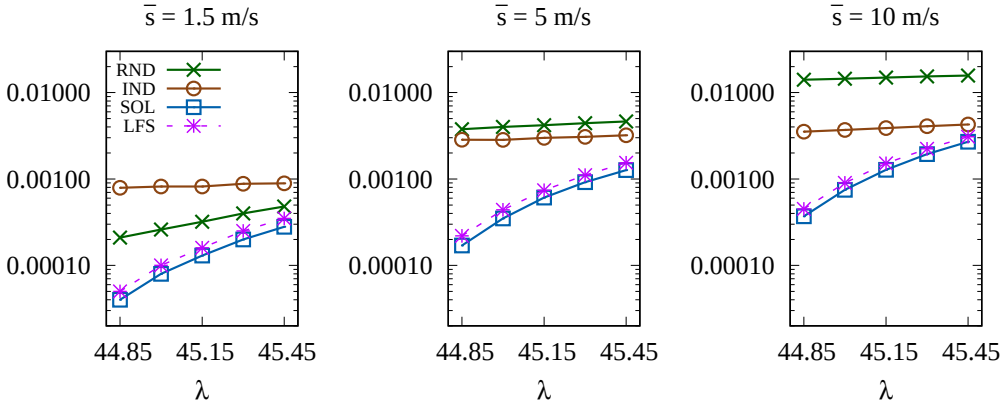
(b) The blocking rate of ongoing sessions (P_H)

Figure 8: Comparing the GoS metrics when applying four scheduling algorithms for $\lambda/\mu = 27000$ and three mean travelling speeds in Case 3

hours (having some night activity), and there are more people (25%) who are constantly moving during office hours (e.g. couriers). Figure 12 shows the percentage distribution of UEs participating in the different scenes. It is well observable that UEs set off to work and finish work in multiple waves



(a) The blocking rate of new sessions (P_F)



(b) The blocking rate of ongoing sessions (P_H)

Figure 9: The GoS metrics when applying four scheduling algorithms versus λ in Case 3 for $1/\mu = 600$ s

as the ratio of moving UEs increases these times. As Figure 13 shows load moves between eNodeBs located in the sleeping city and eNodeBs located in office quarters during the day. We apply Telco Cloud Simulator (TCS) [22] to generate the mobility pattern of 28000 subscribers who travel only on the

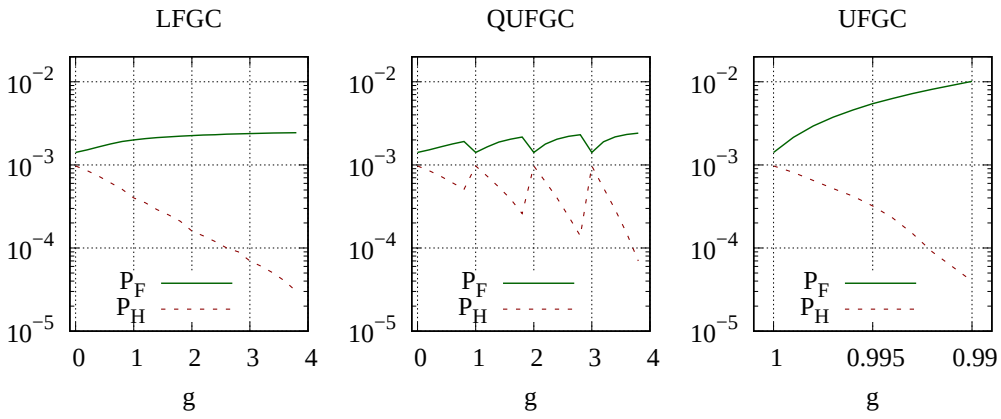
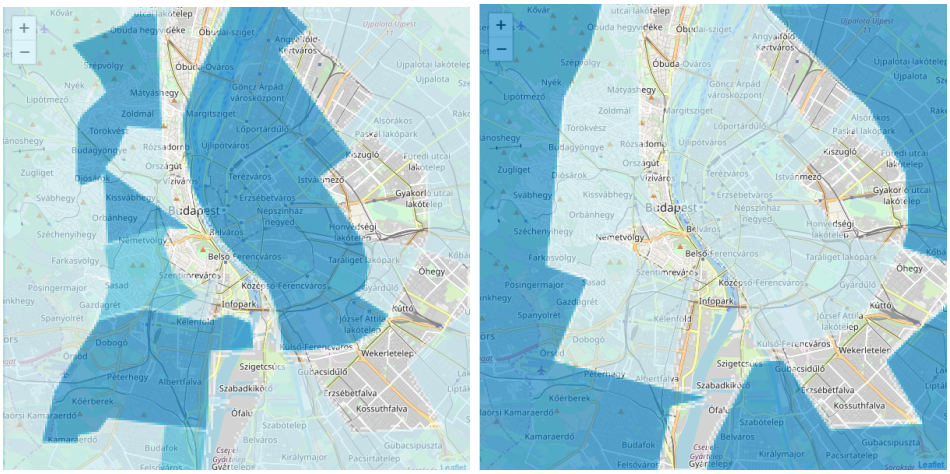


Figure 10: The blocking probability and the dropping probability versus $g_{\epsilon_{to}}$ when applying the `SOL` algorithm with $\bar{s} = 5$ m/s and $1/\mu = 1800$ s in Scenario 3



(a) Office quarter (b) Sleeping city part

Figure 11: Office quarter and sleeping city part

streets of Budapest (see Figure 2).

Figure 14 plots the blocking probability of new sessions and the dropping probability of ongoing sessions vs. time. We can observe a breakpoint in the curves nearly 9 am in Figure 14a. Recall that people set off to work in three waves: at 5, 7 and 9 a.m. People finish work and go home in three waves: at 3, 5 and 7 p.m. As a result, other breakpoints can be observed at

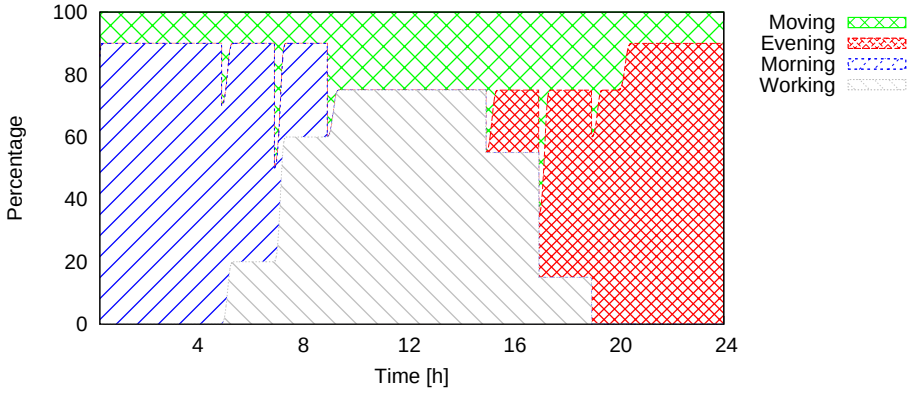
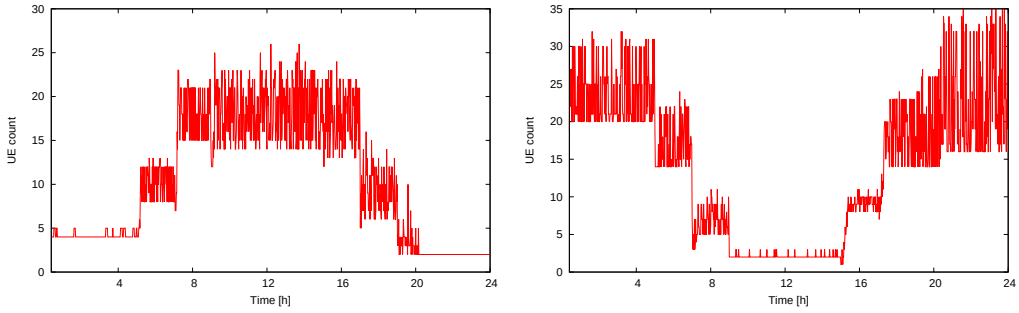


Figure 12: Scenes during the story

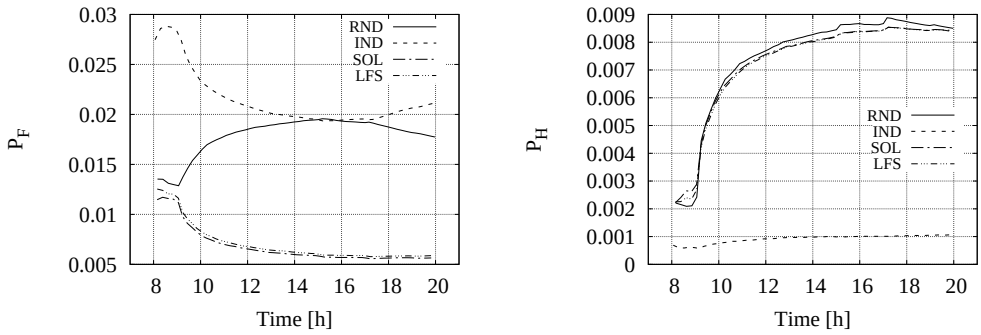


(a) Office quarter

(b) Sleeping city

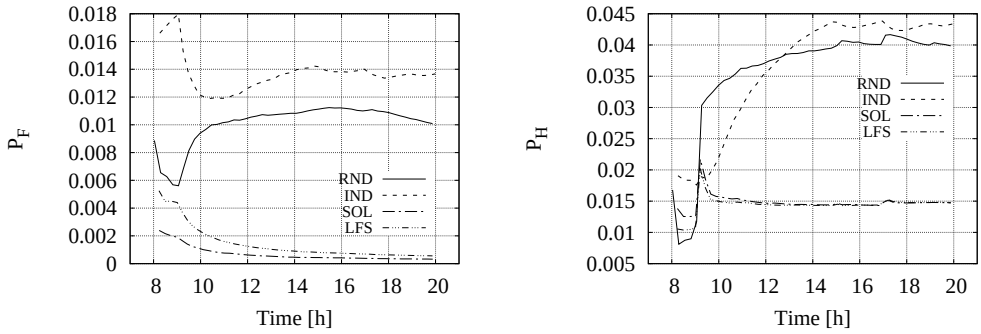
Figure 13: Load of eNodeBs typical for location sleeping city and for location office quarter. Total number of eNodeBs is 3600, while total UE count was 40000

these times in Figure 14b. The blocking probability of new sessions (i.e., P_F) with the `SOL` and the `LFS` algorithms decreases and is always smaller than the blocking probability when the `RND` and the `IND` algorithms are applied



(a) The blocking probability (b) The dropping probability

Figure 14: The trace of the GoS metrics for 27500 users and $1/\mu = 180$ s in Scenario 3



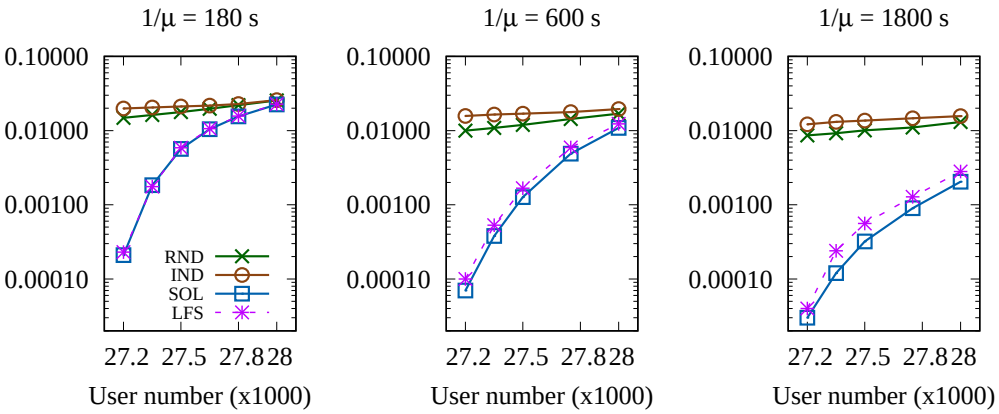
(a) The blocking probability (b) The dropping probability

Figure 15: The trace of the GoS metrics for 27500 users and $1/\mu = 1800$ s in Scenario 3

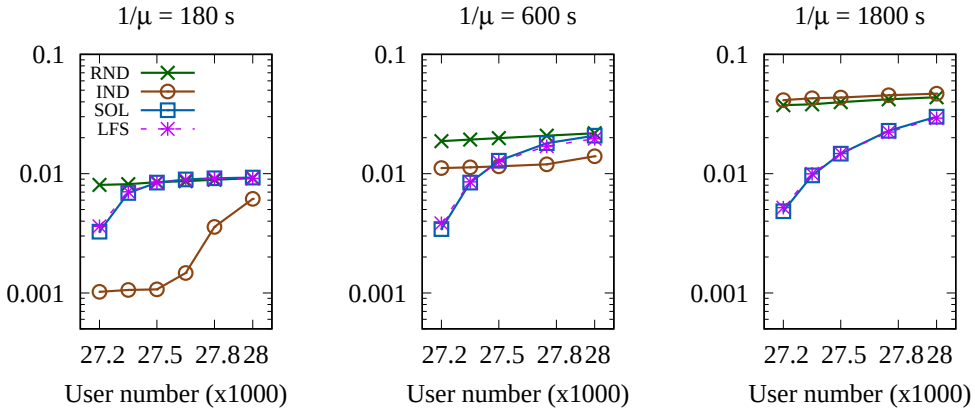
during the investigated duration. The same observation can be obtained from Figure 15 as well.

Figure 16 plots the GoS metrics when the number of considered users increased from 27200 to 28000. In all cases, the `SOL` and the `LFS` algorithms achieve the smallest blocking probability of new sessions (i.e., P_F). They also outperform the `RND` and the `IND` algorithms when $1/\mu$=1800 s.

Figure 16b shows the dropping probability of ongoing sessions when applying the `SOL` algorithm. For example, $P_H = 0.84\%$ and $P_F = 0.57\%$ when $1/\mu = 180$ s for 27500 users. As shown in Figure 17, the FGC policy can



(a) The blocking probability of new sessions (P_F)



(b) The dropping probability of ongoing sessions (P_H)

Figure 16: GoS of four scheduling algorithms versus the number of users in Scenario 3

reduce the dropping probability of ongoing sessions at the expense of increasing the blocking probability. For instance, $P_H = 0.1\%$ and $P_F = 1.38\%$ when applying the LFGC policy with $g_{\epsilon_{to}} = 9$. It is worth emphasizing that this result is better than the result obtained by using the `IND` algorithm (i.e., $P_H = 0.11\%$ and $P_F = 2.12\%$).

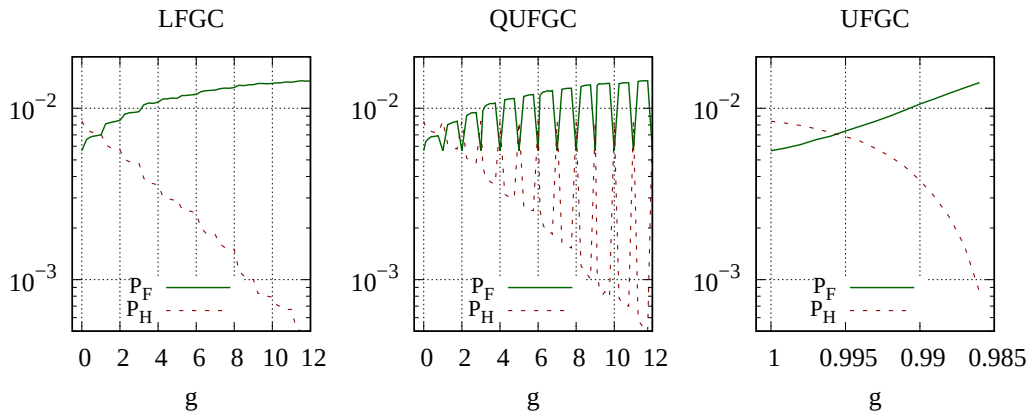


Figure 17: The blocking probability and the dropping probability versus $g_{\epsilon_{to}}$ when applying the `SOL` algorithm for $1/\mu = 180$ s and 27500 users in Scenario 3

6. Conclusion

We have investigated scheduling algorithms for the assignment of user sessions in mobile edge clouds. Four scheduling algorithms (the `Random` (RND), the `Index` (IND), the `Smallest-Occupancy-Level` (SOL) and the `Largest-Free-Slots` (LFS)) have been proposed in the paper. Results have shown that scheduling based on the occupancy level of MECs is quite robust in various scenarios and can be applied with the fractional guard channel policy to provide service for subscribers.

The future work may include the investigation of user scheduling algorithms in the context of the management of multiple slices in 5G systems.

Acknowledgements

The research was partially supported by the Higher Education Excellence Program of the Ministry of Human Capacities in the frame of Artificial Intelligence research area of Budapest University of Technology (BME FIKP-MI/SC). The research of N. H. Do is supported by the OTKA K123914 project. The research of H. T. Nguyen has been partially supported by the European Union, co-financed by the European Social Fund (EFOP-3.6.2-16-2017-00013).

The authors thanks Editors and reviewers for comments that helped to improve the quality of the paper. The authors appreciate the help of Noémi Wagner for her comments on the English usage of the paper.

[1] 3GPP, Feasibility Study on New Services and Markets Technology Enablers: Stage 1, Technical Specification (TS) 22.891, 3rd Generation Partnership Project (3GPP), version 14.2.0 (09 2016).

[2] ETSI Group Specification, ETSI GS MEC 003 V1.1.1: Mobile Edge Computing (MEC); Framework and Reference Architecture, 2016 (2016).

[3] B. Blanco, J. O. Fajardo, I. Giannoulakis, E. Kafetzakis, S. Peng, J. Pérez-Romero, I. Trajkovska, P. S. Khodashenas, L. Goratti, M. Paolino, E. Sfakianakis, F. Liberal, G. Xilouris, Technology pillars in the architecture of future 5G mobile networks: NFV, MEC and SDN, Computer Standards & Interfaces 54 (2017) 216 – 228, sI: Standardization SDN&NFV. doi:https://doi.org/10.1016/j.csi.2016.12.007.

[4] 5GPP, View on 5G Architecture, White paper 22.891, 5GPPP Architecture Working Group, version 14.2.0 (07 2018).

[5] E. Ahmed, M. H. Rehmani, Mobile edge computing: Opportunities, solutions, and challenges, Future Generation Computer Systems 70 (2017) 59 – 63. doi:https://doi.org/10.1016/j.future.2016.09.015.

[6] D. Satria, D. Park, M. Jo, Recovery for overloaded mobile edge computing, Future Generation Computer Systems 70 (2017) 138 – 147. doi:https://doi.org/10.1016/j.future.2016.06.024.

[7] X. Chen, L. Jiao, W. Li, X. Fu, Efficient multi-user computation offloading for mobile-edge cloud computing, IEEE/ACM Transactions on Networking 24 (5) (2016) 2795–2808. doi:10.1109/TNET.2015.2487344.

[8] J. Liu, Y. Mao, J. Zhang, K. B. Letaief, Delay-optimal computation task scheduling for mobile-edge computing systems (2016) 1451–1455doi:10.1109/ISIT.2016.7541539.

[9] S. Sardellitti, G. Scutari, S. Barbarossa, Joint optimization of radio and computational resources for multicell mobile-edge computing, IEEE Transactions on Signal and Information Processing over Networks 1 (2) (2015) 89–103. doi:10.1109/TSIPN.2015.2448520.

[10] K. Zhang, Y. Mao, S. Leng, Q. Zhao, L. Li, X. Peng, L. Pan, S. Maharjan, Y. Zhang, Energy-efficient offloading for mobile edge computing in 5g heterogeneous networks, IEEE Access 4 (2016) 5896–5907. doi:10.1109/ACCESS.2016.2597169.

[11] Y. Mao, J. Zhang, K. B. Letaief, Dynamic computation offloading for mobile-edge computing with energy harvesting devices, IEEE Journal on Selected Areas in Communications 34 (12) (2016) 3590–3605. doi:10.1109/JSAC.2016.2611964.

[12] P. D. Lorenzo, S. Barbarossa, S. Sardellitti, Joint optimization of radio resources and code partitioning in mobile cloud computing, CoRR abs/1307.3835. arXiv:1307.3835.

[13] Y. Wang, M. Sheng, X. Wang, L. Wang, J. Li, Mobile-edge computing: Partial computation offloading using dynamic voltage scaling, IEEE Transactions on Communications 64 (10) (2016) 4268–4282. doi:10.1109/TCOMM.2016.2599530.

[14] A. Hegyi, H. Flinck, I. Ketykó, P. Kuure, C. Nemes, L. Pinter, Application orchestration in mobile edge cloud: Placing of iot applications to the edge, in: S. Elnikety, P. R. Lewis, C. Müller-Schloer (Eds.), 2016 IEEE 1st International Workshops on Foundations and Applications of Self* Systems (FAS*W), Augsburg, Germany, September 12-16, 2016, IEEE, 2016, pp. 230–235. doi:10.1109/FAS-W.2016.56.

[15] I. Ketykó, L. Kecskés, C. Nemes, L. Farkas, Multi-user computation offloading as multiple knapsack problem for 5g mobile edge computing, in: 2016 European Conference on Networks and Communications (EuCNC), 2016, pp. 225–229. doi:10.1109/EuCNC.2016.7561037.

[16] A. Ahmed, E. Ahmed, A survey on mobile edge computing, in: 2016 10th International Conference on Intelligent Systems and Control (ISCO), 2016, pp. 1–8. doi:10.1109/ISCO.2016.7727082.

[17] Y. Mao, C. You, J. Zhang, K. Huang, K. B. Letaief, Mobile edge computing: Survey and research outlook.

[18] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, W. Wang, A survey on mobile edge networks: Convergence of computing, caching and com-

munications, IEEE Access 5 (2017) 6757–6779. doi:10.1109/ACCESS.2017.2685434.

[19] T. Camp, J. Boleng, V. Davies, A Survey of Mobility Models for Ad Hoc Network Research, Wireless Communication & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications 2 (2002) 483–502. doi:10.1002/wcm.72.

[20] T. V. Do, N. H. Do, R. Chakka, A new queueing model for spectrum renting in mobile cellular networks, Computer Communications 35 (10) (2012) 1165 – 1171. doi:https://doi.org/10.1016/j.comcom.2011.12.012.

[21] F. Cruz-Perez, L. Ortigoza-Guerrero, Fractional resource reservation in mobile cellular systems, in: Resource, Mobility, and Security Management in Wireless Networks and Mobile Communications, Auerbach Publications, 2007, Ch. 11, pp. 335–362.

[22] P. Hegyi, N. Varga, L. Bokor, An advanced Telco Cloud Simulator and its usage on modelling multi-cloud and 5G multi-access environments, in: Proceedings of the 21th Innovation in Clouds, Internet and Networks, IEEE, 2018, pp. 264–266, IEEE Catalog Number: CFP1850H-POD. URL http://www.proceedings.com/39375.html