Pre-processor for MAC-layer Scheduler to Efficiently Manage Buffer in Modern Wireless Networks

Ankit Bhamri*†, Navid Nikaein*, Florian Kaltenberger*, Jyri Hämäläinen†, Raymond Knopp*
*Eurecom, France (email : firstname.lastname@eurecom.fr)
†Aalto University School of Electrical Engineering, Finland (email : firstname.lastname@aalto.fi)

***Abstract*—Mobile devices have evolved remarkably over the last decade and are now being utilized to access much broader range of internet applications. Moreover, their capability to simultaneously run many applications has significantly transformed the traffic characteristics of mobile networks. Quality of service (QoS) is a fundamental component associated with these applications and network should be able to support multiple QoS requests from the same user at same time. This requires complex buffer management and simultaneous scheduling of resources to multiple users with multiple services. In this paper, we propose a framework with pre-processor for MAC-layer scheduler including two-dimensional buffer management (*users* $\times$ *services*) that enable more efficient allocation of resources to users running multiple internet applications in parallel. The framework will enhance the performance of existing scheduling algorithms by increasing the resolution of scheduling. A comparative analysis of traditional scheduling algorithms is provided to show the gains of proposed framework.**

***Keywords*: Buffer Management, Performance Evaluation, Quality-of-service (QoS), Scheduling Framework**

I. Introduction

Mobile device subscribers have tremendously increased over the years and consequently the wireless network traffic has also significantly increased in volume. All measurements in current mobile networks and all forecasts indicate fast increase in mobile data traffic. For example, widely referenced Cisco VNI forecast reports that mobile data traffic grew 70% in 2012 and global mobile data traffic is expected to increase 13-fold between 2012 and 2017 [1]. In addition, the explosion of internet applications on mobile devices over the last few years has completely transformed the mobile data traffic pattern. Modern mobile devices are capable of simultaneously running multiple internet applications. Consider for example a user having video call on the mobile device. At the same instant of time, there is a possibility of running several other applications such as video buffering, online gaming, voice recognition, and email services. Each of this services has a respective QoS requirement and therefore a dedicated radio bearer is established for each data flow and mapped to a corresponding logical channel [2]. As a result, see Fig. 1, a single user has buffer queued in several logical channels in the same transmission time interval. Furthermore, the queued buffer will expand into two dimensions ($N \times K$) when there are multiple active users in the system requesting for several services. Here$N$ and K refer to total number of active users and total number of logical channels for each user, respectively. Each buffer element (n, k) has a specific QoS requirement and is characterized by several parameters with specific values that are described in Section II. All these parameters are crucial for buffer management and design of scheduling algorithm.

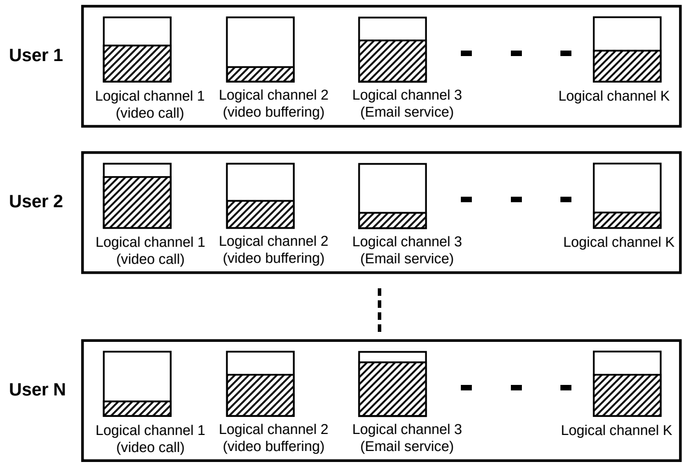
User 1
User 2
User N
Logical channel 1 (video call)
Logical channel 2 (video buffering)
Logical channel 3 (Email service)
Logical channel K

Fig. 1. Example of buffer queue for Mmltiple-service offering

Most of the traditional scheduling frameworks have been designed to deal with users having single service offering, meaning that they classify a user belonging to only one specific QoS class and therefore apply the scheduling algorithm at the user level [3], [4], [5]. Such framework would lead to inefficient buffer management and sub-optimal system performance in multiple-service offering scenario. In the traditional MAC, the scheduling algorithm optimally allocates resources to users based on the assumption of user belonging to single QoS. Through our work, we will get rid of this assumption and as a result improve the performance of MAC schedulers.

In this paper, we propose a scheduling framework with pre-processor for two-dimensional buffer management on per-user per-service basis, meaning that we apply scheduling algorithm at even higher resolution. In contemporary frameworks, most of the schedulers allocate resources on user basis, user belonging to a particular service class. However in modern networks, the user can demand several services belonging to different QoS classes. Therefore we treat allocation in two dimension matrix with each block belonging to a particular service of a user. Essentially, we provide a modular framework to improve the performance of existing scheduling algorithms rather than developing a new algorithm. Although, it is designed for downlink resource allocation, but can easily be extended to uplink as well.

The rest of the paper is organized as follows: Section II defines the required paramters while in Section III, we describe the framework and its interfaces. In Section IV, we do a comparative analysis to demonstrate the gains in terms of total system throughput, percentage of satisfied users and fairness index. The paper is concluded in Section V.

II. Framework Parameters

The scheduler framework with pre-processor is designed for all-IP packet switched networks such as 3GPP LTE. In such networks, the buffer of a logical channel consists of packets and we apply scheduling algorithm to the packets of a logical channel for every user in the system. Henceforth in this section, we describe the subset of parameters utilized for pre-processing buffer and making scheduling decision. We categorize parameters specific to every level i.e. packet $\rightarrow$ logical channel's buffer $\rightarrow$ user $\rightarrow$ system. The parameters are received by MAC through its interface with buffer, physical layer and system, which are then utilized for buffer management and resource allocation schemes. In this work, our system is confined to the scenario of a single base station and downlink scheduling of resources to multiple users within its range. All the parameters described in following sections have a subscript that indicates the interface with MAC. b stands for MAC-buffer interface, p stands for MAC-PHY interface and s represents system-defined parameters.

A. Packet-level parameters

- $\mathrm{APS_b}$: Average packet size (bytes)
- $\mathrm{PIT_b}$: Packets inter-arrival time (ms)
- $\mathrm{PAT_b}$: Packet arrival time (ms)
- $\mathrm{PMD_b}$: Packet maximum-allowable delay (ms)
- $\mathrm{PRT_b}$: Packet remaining time (ms)

B. Logical channel-level parameters

- $\mathrm{NPDU_b}$: Number of protocol data units (PDUs) in a given subframe
- $\mathrm{HOD_b}$: Head-of-line delay for a logical channel (ms)
- $\mathrm{BS_b}$: Buffer size of a logical (channel) (bytes)
- $\mathrm{GBR_b}$: Guaranteed bit-rate for a service (Kbps)
- $\mathrm{TL_b}$: Level of traffic for user in logical channel [0,1], here 0 traffic level means no traffic and 1 is for continuous flow of traffic.

C. User-level parameters

- $\mathrm{TBS_b}$: User total buffer size (bytes)
- $\mathrm{NLC_b}$: Number of logical channels
- $\mathrm{CQI_p}$: Channel quality indicator

D. System-level parameters

- $\mathrm{NU_s}$: Number of active users
- $\mathrm{MAU_s}$: maximum allowed scheduled users
- $\mathrm{CT_s}$: Frame configuration type (FDD or TDD)
- $\mathrm{TTI_s}$: Transmission time interval (ms)
- $\mathrm{MRU_s}$: Minimum resource allocation unit
- $\mathrm{SA_s}$: Scheduler algorithm fixed by system

III. Scheduler Framework with Pre-processor

We model a framework for scheduling users and their logical channels based on the parameters defined in Section II. The primary task of this framework is two-dimensional buffer management at per-user per-service level which results in higher resolution for scheduling algorithms. In addition, the purpose of the framework is to develop a modular approach:

1) Every module has a well-defined specific functionality that will contribute to performance enhancement at the system level.
2) Modular approach adds flexibility to the scheduling framework and it can be integrated conveniently into different standards. Every module can be individually altered depending up on the constraints and system requirements.

Therefore we stress the fact that this framework provides a generic solution to enhance the performance of existing scheduling algorithms for different wireless standards. In the following sections, we describe the framework structure in terms of its interfaces and modules of MAC-layer as shown in Fig. 2.

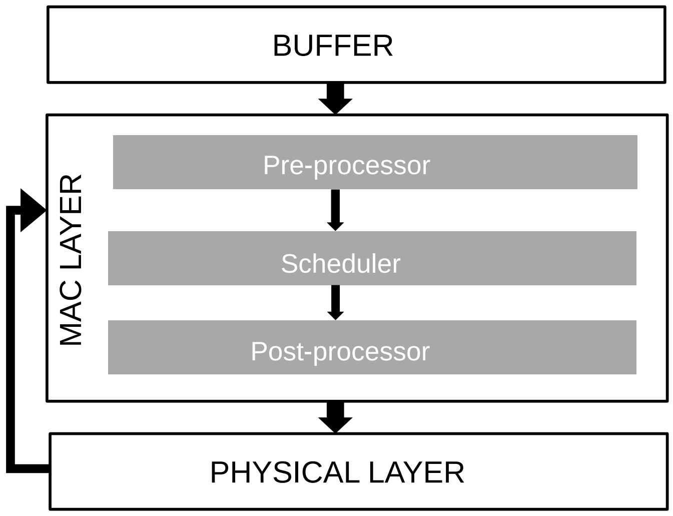

Fig. 2. MAC-layer Scheduling Framework

Algorithm 1 Convert *users* $\times$ *logical channels* into *blocks*

```
$lc\_count = 1$
$block\_count = 1$
**while** $lc\_count \le \mathrm{NLC_b}$ **do**
    $user\_count = 1$
    **while** $user\_count \le \mathrm{NU_s}$ **do**
        **if** $\mathrm{BS_b} > 0$ **then**
            $block\,[block\_count] = [user\_count, lc\_count]$
            $block\_count = block\_count + 1$
        **end if**
        $user\_count = user\_count + 1$
    **end while**
    $lc\_count = lc\_count + 1$
**end while**
```

A. MAC-layer Interfaces

In order to manage buffers, select scheduling algorithm and allocate resources to users, the MAC-layer requires the knowledge of all the parameters defined in Section II. These parameters are received by the MAC-layer from system and interfaces towards other layers.

1) *MAC-Buffer*: This interface shares the packet information of users and logical channels with the MAC-layer. Each logical channel represents a service with specific values to the parameters defined in Section II. Through this interface, we create the two dimensional structure of users and logical channels along with their associated parameters. These parameters informs the scheduler about traffic characteristics of every service in given transmission time interval.
2) *MAC-PHY*: Most of the modern scheduling algorithms are channel-aware, therefore it is crucial for MAC-layer to have a knowledge of the channel quality information of all the active users in the system. Once the channel estimation is done at the user terminals, the result is sent to the base station via feedback channel. The physical layer receives this information and forward through MAC-PHY interface to MAC. Channel quality information is used while calculating the expected throughput for a user and in turn for the entire system.

B. MAC-layer Modules

We have split the framework into three functional modules. Once the MAC has a knowledge of all the necessary parameters through its interfaces, then the following three modules proceed with their defined tasks:

1) *Pre-processor*: This module represents a novel extension to the traditional scheduling framework. Main function of the pre-processor is to convert the two-dimensional buffer of *users* $\times$ *logical channels* into a single dimension as shown in Fig. 3. In the following we refer to each element by term block. For each block there holds

$$\mathrm{BS_b} > 0 \tag{1}$$

The pre-processor's algorithm is given by Algorithm 1. As a result of this pre-processor, the scheduler module receives input in terms of block and its associated parameters. The traditional one-dimensional framework can easily be converted to this two-dimensional framework with the addition of this module and existing scheduling algorithms does not require to change in order to be implemented with this framework.

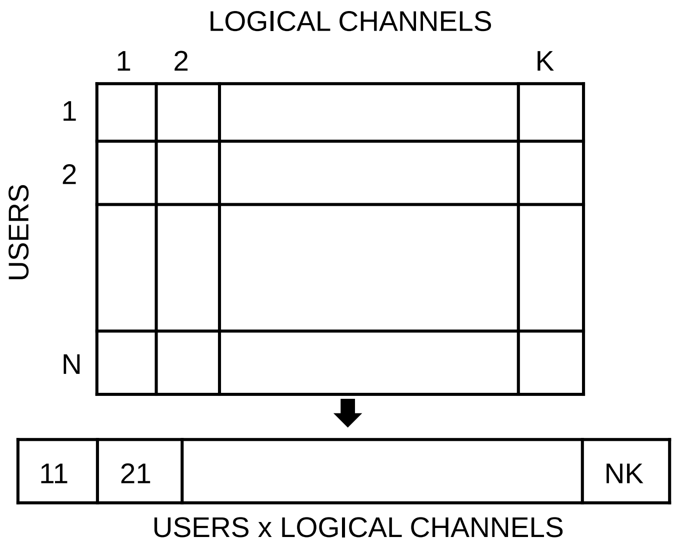

Fig. 3. Transform from 2-dimensional to 1-dimensional system

2) *Scheduler*: Once the conversion to single dimension is done in the pre-processor, the first task of scheduler module is to deal with sorting of blocks based on scheduling requirements. For example, in case of proportional-fair (PF)scheduling algorithm, blocks are sorted in decreasing order wrt. channel quality information. In round robin case, the blocks remain in order of their index. After block sorting, the actual allocation of resources to the sorted blocks is done. As can be seen, the major difference between a traditional framework and our proposed framework is the resolution of scheduling. Since the input to the scheduling module is blocks, the sorting and allocation is done at block level rather than user level. A more detailed comparative analysis is shown in Section IV.
3) *Post-processor* : The post-processor implementation depends on the wireless standard. The mapping of users to resource elements in the frequency domain is applied based on the specifications of the system. For example, in 3GPP LTE, the selection of resource block groups (RBGs) for a particular service of a user is done by post-processor. Number of RBGs depends on the system bandwidth [6]. In LTE, RBGs are the smallest unit of frequency resources that can scheduled for a user.

IV. Comparative Analysis

In this section, we show the gain due to two-dimensional buffer management within the proposed framework. In addition, we show an example of simple scheduling algorithm and explain the implementation differences with our framework. Following that, we validate our claims by simulation results for few traditional scheduling algorithms. The main motive is to show performance gain of scheduling algorithms in terms of system throughput, fairness-index and percentage of satisfied users and services (those achieving GBR). The primary reason for comparing these traditional scheduling algorithms is to demonstrate the effectiveness of the proposed framework even with simple algorithms. However we mathematically show that the performance improvement will happen for most of the scheduling algorithms and therefore it would be applicable for even more complex algorithms. Since we are interested in observing the gains of applying MAC-layer scheduler, we assume perfect decoding in the physical layer. Therefore all the scheduled packets are assumed to be successfully received.

For any scheduling algorithm, sorting of users on the basis of a pre-defined performance metric is a primary step for efficient allocation of users. Sorting of users and allocating resources in the corresponding order contribute to the performance gain of scheduler. For example, suppose there are four users in a system with channel quality (2,1,8,4) and they are required to be sorted in decreasing order of channel quality. Then the sorted list will be (8,4,2,1) and naturally it will provide the best possible performance. It can be seen that as a result of sorting, four users changed their position and each contributed to improved performance. If only one user was sorted, then performance would not be optimal but better than unsorted list. Therefore we can say that with every sorting step, the performance increases till it reaches the best scenario. In order to generalize this explanation, let us define a variable G_1, which is the gain due to the change in position of single user and when x is the number of users that changed position due to sorting, then the total gain would be $x \times G_1$. However, the total gain also depends on the number of users that can be actually scheduled, for that we have introduced Equation 2 which gives the conditional total gain applicable to any traditional scheduling framework

$$\mathrm{TG_t} = \begin{cases} x \times G_1, & \text{if } x \leq N' \\ N' \times G_1, & \text{if } x > N' \end{cases} \tag{2}$$

where N' is the actual number of scheduled users and N is the number of total active users. N' is a subset of N and it can be deduced from Equation 2 that $\mathrm{TG_t} = [0, N \times G_1]$.

With our proposed framework, the scheduler gets an input of blocks from the pre-processor and sorting is done for blocks as depicted in Fig. 3. Consequently the conditional total gain for our proposed framework is

$$\mathrm{TG_p} = \begin{cases} y \times G_2, & \text{if } y \leq B' \\ B' \times G_2, & \text{if } y > B' \end{cases} \tag{3}$$

where G_2 is the gain due to the change in position of single block, y is the number of sorted blocks and B' is the number of scheduled blocks. B' is a subset of $N \times K$ and it can be deduced from Equation 3 that $\mathrm{TG_p} = [0, N \times K \times G_2]$.

By comparing Equation 2 and 3 in a similar scenario, we will show that $\mathrm{TG_p} \geq \mathrm{TG_t}$ since the parameters x, G_1 and N' in Equation 2 always fit inside y, G_2 and B' in Equation 3 respectively.

A. Demonstrating differences between proposed and traditional framework

For the example case, we take the most basic algorithm of round-robin scheduling. All the parameters defined in the algorithm are self-explanatory. First, we show the implementation of round-robin algorithm without our framework. As can be seen from Algorithm 2, scheduling is done only at the user levels. After this step, the resources allocated to each user are used for logical channel in order of their increasing index i.e $alloc\_resources\_per\_lc\_per\_user\,[n]\,[k]$. Then the implementation of round-robin algorithm with our framework is shown in Algorithm 3. We can see that scheduling is done for blocks that have been shown in Fig. 3. This means that the scheduler has a higher resolution of scheduling and is able to consider the constraints and requirements on per-logical channel per-user basis. This leads to more efficient buffer management and optimal allocation of resources.

In terms of complexity, we can see that both the frameworks have the same number of iteration loops i.e. two and the only difference is the number of iteration steps with our proposed framework. The number of iteration steps for traditional algorithm is the number of users, while for the proposed algorithms, it the number of blocks which is greater. Therefore, without any large increase in complexity , we can obtain significant gains for the existing algorithms with our proposed framework. It should also be noted, that this comparison of complexity is between the frameworks and is therefore applicable to any scheduling algorithm.

B. Simulation Setup

For our simulations, we have considered 3GPP LTE setting and used its system parameters [7]. The simulations are done for a system bandwidth of 5MHz with 25 resource blocks and 1000 frames with 10 subframes each. The duration of a frame is 10ms. In addition to the system parameters, we have also used the standard quality of service classs defined in LTE and we support 9 logical channels for different services with varying QoS [8]. In the traffic generator, we utilized the average packet size and average inter-arrival time for each service based on [9] and [10]. The actual inter-arrival time

Algorithm 2 Calculate $alloc\_resources\_to\_user$ (Round-robin without framework)

```
while $number\_of\_resources\_remaining > 0$ do
  while $number\_of\_users\_scheduled \leq max\_allowed\_sched\_user$ do
    $alloc\_resources\_to\_user\,[n] = min\_resource\_alloc\_unit + alloc\_resources\_to\_user\,[n]$
    if $user\_scheduled\,[n] \neq 1$ then
      $user\_scheduled\,[n] = 1$
    end if
    $number\_of\_users\_scheduled = number\_of\_users\_scheduled + 1$
    $number\_of\_resources\_remaining = number\_of\_resources\_remaining\ + min\_resource\_alloc\_unit$
  end while
end while
```

Algorithm 3 Calculate $alloc\_resources\_to\_block$ (Round-robin with proposed framework)

```
while $number\_of\_resources\_remaining > 0$ do
  while $number\_of\_blocks\_scheduled \leq max\_allowed\_blocks$ do
    if $number\_of\_users\_scheduled \leq max\_allowed\_sched\_user$ then
      $allocated\_resources\_to\_block\,[j] = min\_resource\_alloc\_unit + allocated\_resources\_to\_block\,[j]$
      if $user\_scheduled\,[n] \neq 1$ then
        $user\_scheduled\,[n] = 1$
      end if
      if $block\_scheduled\,[j] \neq 1$ then
        $block\_scheduled\,[j] = 1$
      end if
      $number\_of\_users\_scheduled = number\_of\_users\_scheduled + 1$
      $number\_of\_blocks\_scheduled = number\_of\_blocks\_scheduled + 1$
      $number\_of\_resources\_remaining = number\_of\_resources\_remaining\ + min\_resource\_alloc\_unit$
    end if
  end while
end while
```

between two packets for a given service of a user (block) is affected by TLb (ranging on [0,1]) and given as:

$$\mathrm{APIT} = \begin{cases} \frac{\mathrm{PIT_b}}{\mathrm{TL_b}}, & \text{if } \mathrm{TL_b} \neq 0 \\ \text{no traffic}, & \text{otherwise} \end{cases} \tag{4}$$

For our simulation results in Fig. 4, 5 and 6, $\mathrm{TL_b}$ is generated randomly.

C. Results

Performance analysis is done for three traditional scheduling algorithms: round-robin, proportional-fair and maximum-throughput [2]. In order to compare the performance of these scheduling algorithms with and without our proposed framework, we plot system throughput, fairness index and satisfied GBR percentage (percentage of users that are satisfied). All the performance metrics only refer to MAC-layer scheduling performance since we assume that there is a perfect decoding at the physical layer. In Fig. 4, we have compared the system throughput against number of active users in the system. It is evident that the performance of every scheduling method is better in our framework than in traditional framework. We have also plot the theoretical system throughput to set a benchmark.

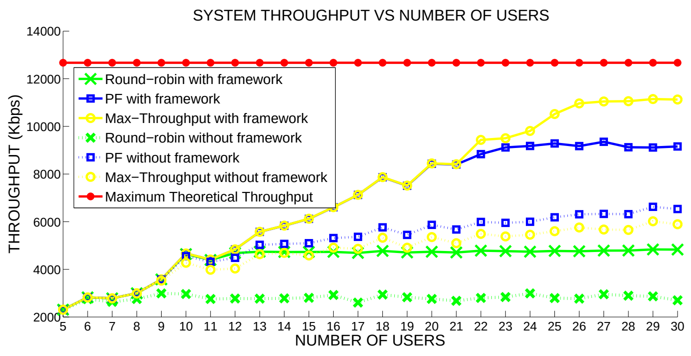

Fig. 4. Throughput Comparison

In order to give a more clear picture, we have plotted the fairness index and the satisfied GBR percentage in Fig. 5 and 6 respectively. The satisfied GBR percentage refers to the percentage of the logical channels (services) that are served with a data rate greater than or equal to GBR. We observe similar pattern as in Fig. 4 that indicates performance enhancement due to our framework. It shows that the framework not only increases the throughput but also results in improved fairness index and provides better quality of service to users with higher percentage of satisfied GBR.

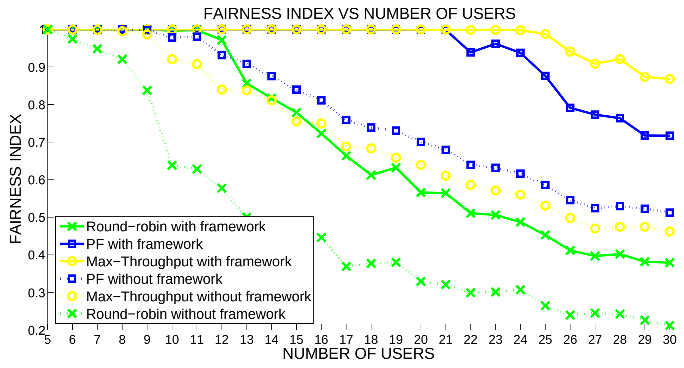

Fig. 5. Fairness Index

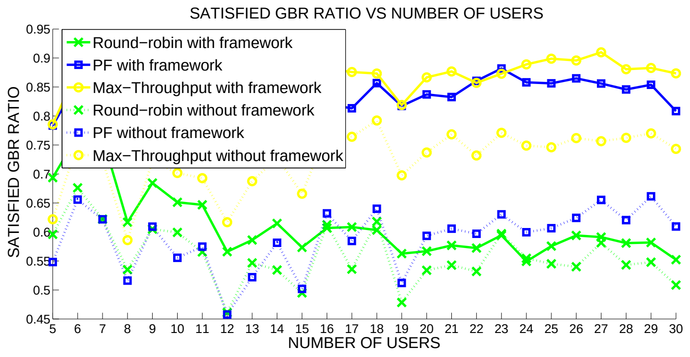

Fig. 6. Satisfied GBR Ratio

In addition, we have presented in Fig. 7 throughput when using different traffic patterns. While, TL_b is randomly generated in case of Fig. 4, 5 and 6, in Fig. 7, we select TL_b such that the traffic pattern is similar to that of a residential area in evening when the services such as streaming, conversational video calls, etc. are requested by large number of users.

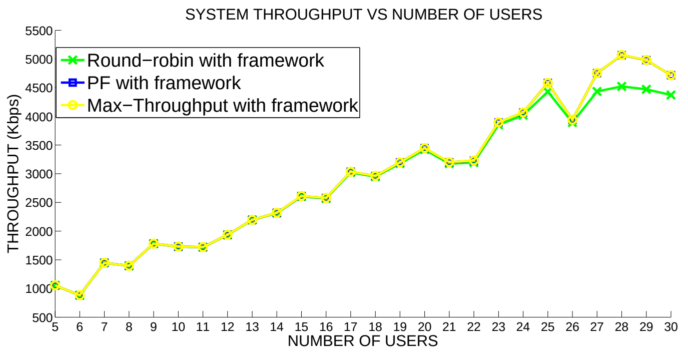

Fig. 7. Throughput Comparison of Scheduling Algorithms in USF for traffic scenario with higher demand for conversational and streaming services

The main purpose of this plot is to demonstrate the performance variation of scheduling algorithms with the change in traffic pattern. It can be concluded that there is no single scheduling algorithm optimal for every scenario. The traffic is continuously evolving in modern wireless networks and therefore static scheduling frameworks are no more an optimal solution. There is a need for more dynamic and adaptive approach. Therefore for our succeeding work, we aim to move forward with our proposed framework in a more dynamic direction. The modular nature of our framework enable us to append any additional module without disrupting the entire framework. Henceforth, an additional module consisting of intelligent APIs could be appended that would act as an interface between MAC-layer and external factors such as operator requirements, network constraints, etc. These intelligent APIs would be capable of dynamically configuring scheduling algorithms depending up on traffic patterns and several other factors.

V. Conclusion

In this paper, we proposed a scheduling framework to improve the performance of any scheduling algorithm by two dimensional buffer for providing a satisfied QoS experience to users with multiple service offerings. We showed clear performance gain in terms of throughput, fairness-index and percentage of satisfied GBR users through our framework and also pointed out that there is necessity for a dynamic re-configurability of scheduling algorithm which will be addressed in our succeeding work.

References

[1] Cisco Visual Networking Index: Forecast and Methodology, 20122017, Retrieved from *http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360.pdf*

[2] Roke Manor Research, LTE MAC Scheduler and Radio Bearer QoS [White paper], Retrieved from *http://www.roke.co.uk/resources/white-papers/0485-LTE-Radio-Bearer-QoS.pdf*

[3] Capozzi, F.; Piro, G.; Grieco, L.A.; Boggia, G.; Camarda, P., Downlink Packet Scheduling in LTE Cellular Networks: Key Design Issues and a Survey, *IEEE Communications Surveys and Tutorials*, vol.15, no.2, pp.678,700, Second Quarter 2013

[4] Zaki, Y.; Weerawardane, T.; Gorg, C.; Timm-Giel, A., Multi-QoS-Aware Fair Scheduling for LTE, *IEEE Vehicular Technology Conference (VTC Spring)*, vol., no., pp.1,5, 15-18 May 2011

[5] Chaudhuri, S.; Das, D.; Bhaskaran, R., Study of advanced-opportunistic proportionate fairness scheduler for LTE medium access control, *IEEE International Conference on Internet Multimedia Systems Architecture and Application (IMSAA)*, vol., no., pp.1,6, 12-13 Dec. 2011

[6] 3GPP TS 36.213. Evolved Universal Terrestrial Radio Access: Physical Layer Procedures. ver. 8.6.0, March 2009. URL http://www.3gpp.org/ftp/specs/html-info/36213.htm

[7] 3GPP TS 36.212. Evolved Universal Terrestrial Radio Access: Multiplexing and Channel Coding. ver. 8.6.0, March 2009. URL http://www.3gpp.org/ftp/specs/html-info/36212.htm

[8] Alasti, M.; Neekzad, B.; Jie Hui; Vannithamby, R., Quality of service in WiMAX and LTE networks [Topics in Wireless Communications], *IEE Communications Magazine*, vol.48, no.5, pp.104,111, May 2010

[9] CISCO, Voice Over IP - Per Call Bandwidth Consumption, Retrieved from *http://www.cisco.com/image/gif/paws/7934/bwidth_consume.pdf*

[10] CISCO, Implementing QoS Solutions for H.323 Video Conferencing over IP, Retrieved from *http://www.cisco.com/image/gif/paws/21662/video-qos.pdf*