# A QoS-Aware Uplink Scheduling Paradigm for LTE Networks

Haidar Safa, Wassim El-Hajj, and Kamal Tohme
Department of Computer Science
American University of Beirut
Beirut, Lebanon
{hs33, we07, kgt02}@aub.edu.lb

*Abstract*—**LTE uplink frequency scheduling algorithms have neglected the *user equipment*'s (UE) QoS requirements, relying only on the time domain to provide such requirements when creating the allocation matrix for the next *transmission time interval*. Two time domain paradigms exist for creating the resource allocation matrix: channel-dependent and proportional fairness. The channel dependent paradigm considers mainly the channel quality of UEs, allowing for users with high channel quality to get assigned most resources. The proportional fairness paradigm allocates resources to users based on the ratio of their channel condition over their lifelong service rate, allowing for users with low channel conditions to get some resources, but fewer than those with better channel conditions. Even though the proportional fairness paradigm's main focus is to achieve high system throughput without starving any user, it does not account for QoS requirements in many scenarios especially when UEs with high priority data pending for transmission have worst channel conditions than those with lower priority data. In this paper we propose a QoS-aware resource allocation paradigm for LTE uplink scheduling that gives more advantage to UEs having high priority data, while not starving other users. The proposed approach is scalable and mobility aware where the dynamic nature of the network is taken into account while devising the algorithm. When simulated using NS3, the proposed algorithm produced very promising results and outperformed the state-of-the-art approaches presented in literature.**

***Keywords-LTE; scheduling; QoS; resource allocation;***

## I. INTRODUCTION

L*ong term evolution* (LTE) was proposed by 3GPP [3, 5] to replace the *universal mobile telecommunications system* (UMTS) architecture by the *system architecture evolution* (SAE) [4]. SAE is characterized by having an all-IP-network architecture that is flatter than that of UMTS with several functions moved from the core of the network to its edge allowing for latency reduction and faster data routing. It consists of a user-plane called *evolved UMTS terrestrial radio access network* (eUTRAN), and a control-plane called *evolved packet core* (EPC) as shown in Fig. 1. The only entity in the eUTRAN is the *evolved NodeB* (eNodeB), which handles *radio resource management* (RRM). The EPC consists of five nodes: the *mobility management entity* (MME), the *serving gateway* (SGW), the *packet data network gateway* (PGW), the *policy and charging rules function* (PCRF), and the *home subscriber server* (HSS) [1].

The *orthogonal frequency division multiplex* (OFDM) is the radio access technology used by the eUTRAN [6, 8]. It offers high spectral efficiency and reduces bit errors greatly. OFDM's main disadvantage is that it has a high *peak to average power ratio* (PAPR), which makes it unsuitable for the uplink. For this reason, the *single carrier-frequency division multiple access* (SC-FDMA), a variation of OFDM, that incorporates the advantages of OFDM with the low PAPR trait of single carrier systems, was adopted by 3GPP [7, 11]. To achieve low PAPR in SC-FDMA, resources assigned to the same UE must be contiguous in the frequency domain, making packet scheduling for the uplink an unprecedented problem. The resources to be assigned to users are called *resource blocks* (RBs) with 180 KHz each constituting the entire bandwidth.

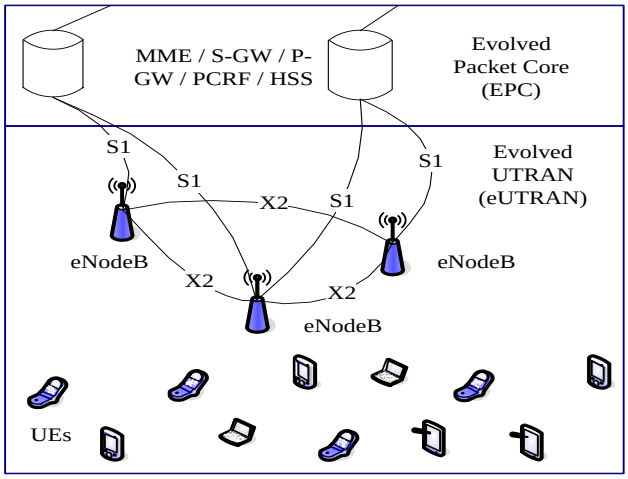


Figure 1. System Architecture Evolution.

The *packet scheduler* (PS) [9] is the controlling entity of the eNodeB's MAC layer, and deals with allocating RBs to UEs every *transmission time interval* (TTI) of 1ms. Scheduling decisions are carried out on a per-user basis, even though each user may have several data flows. Fig. 2 shows how the PS interacts with several RRM functionalities which provide, for instance, the channel quality of UEs on every RB frequency and the UE's QoS requirements. The PS interacts closely with the *hybrid ARQ* (HARQ) manager, which is responsible for scheduling retransmissions. The *link adaptation* (LA) unit provides information to the PS about the supported modulation and coding schemes for a user.

LTE uplink scheduling algorithms for SC-FDMA were left undefined by 3GPP giving vendors the flexibility to design and implement the scheduling algorithm they see appropriate. Various algorithms were proposed in the literature [10-14].

These algorithms require a resource allocation matrix as input. Two paradigms are mainly used to compute the resource allocation matrix, the *channel-dependent* (CD) paradigm and the *proportional fairness* (PF) paradigm. The CD paradigm aims to maximize total throughput by allocating resources to users having the best channel conditions. In this approach, fairness is not achieved and users with bad channel conditions will get starved. The PF paradigm allocates resources to users based on the ratio of their channel condition over their lifelong service rate. As a result, users with low channel conditions will get some resources but fewer than those with better channel conditions. Both paradigms do not consider UE QoS requirements (such as maximum delay) when forming the allocation matrix and still suffer from several problems related to fairness. Moreover, UE mobility might lead to unfair resource allocation where fairness is not provided when dealing with UEs with high ping-pong rate (high rate of joining and leaving the network). In this paper, we propose a new QoS-aware resource allocation paradigm that addresses these issues and improves the number of UEs served efficiently (i.e., those whose QoS requirements are met). The rest of the paper is organized as follows. In section II, we present the LTE packet scheduling approaches that are used to construct a resources-to-users allocation matrix along with the existing LTE uplink scheduling algorithms. In section III, we present our proposed QoS-aware approach. In section IV, we evaluate the performance of the proposed approach against both CD and PF using two different scheduling algorithms. In section V, we conclude the paper and present future work.

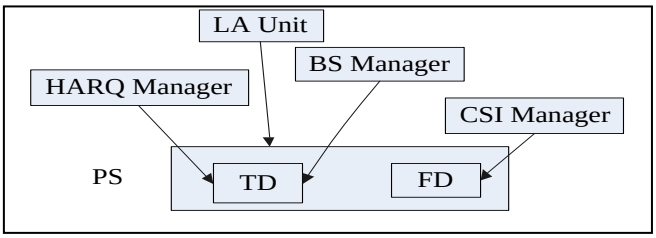


Figure 2. Interaction between the PS and other RRM functionalities.

## II. LTE PACKET SCHEDULING

### A. Channel Allocation Matrix and Signaling Messages for LTE Scheduling

To provide the channel quality of UEs on every RB frequency and the amount of data pending for transmission and their priorities to the serving eNodeB, LTE uses mainly two signaling messages, the *sounding reference signal* (SRS) and *buffer status report* (BSR). LTE uses a channel sounding technique that allows the eNodeB to monitor the channel condition of every UE over the entire bandwidth, i.e. over all RBs [8]. Each UE sends, every 1 ms, a SRS to its serving eNodeB. The latter then extracts *channel state information* (CSI) and passes it to the CSI manager. The CSI manager, in its turn, generates a metric value for each RB for each UE, creating a matrix called *channel conditions matrix* (i.e., resource allocation matrix as shown in the left side of Fig. 3) that is used by the PS for an efficient resource allocation. To request resources from the eNodeB, LTE defined an enhanced scheduling request, the BSR, which also allows the UE to inform the eNodeB about the amount of buffered data and their priority. In LTE, data are classified into four groups called *radio bearer groups* (RBGs), each of which has a different priority level. Two formats are defined for the BSR: a short one if only one RBG is reported, and a long one that allows four RBGs to be reported. Each RBG holds data from applications of the same priority. The BS manager stores these BSRs, allowing the eNodeB to know exactly how much data each user has in each of its four transmission queues as shown in the right side of Fig. 3.

| | RB1 | RB2 | ... | RBm |
|---|---|---|---|---|
| UE1 | M1,1 | M1,2 | ... | M1,m |
| UE2 | M2,1 | M2,2 | ... | M2,m |
| ... | ... | ... | ... | ... |
| UEn | Mn,1 | Mn,2 | ... | Mn,m |

| | RBG1 | RBG2 | RBG3 | RBG4 |
|---|---|---|---|---|
| UE1 | | | | |
| UE2 | | | | |
| ... | | | | |
| UEn | | | | |

Figure 3. (left) Channel Conditions Matrix, (right) Buffer Status.

### B. LTE Scheduler and Resource Allocation Paradigms

LTE Packet Scheduler is divided into two algorithms. The first algorithm is that of the *time domain* (TD), in which *N* users are selected for potential scheduling based on their session's required QoS, which are handled by the *buffer status* (BS) manager, and such that users having pending data for transmission are configured for scheduling in the next TTI. These selected users are passed to the *frequency domain* (FD) scheduling algorithm, which allocates RBs to them ensuring that each RB is allocated to at most one UE and all RBs allocated to a single UE are contiguous in frequency. The FD algorithm uses the channel conditions matrix available from the CSI manager to schedule UEs on RBs with high channel quality. When creating the allocation matrix, the CD paradigm [12] considers the channel quality of UEs on every RB frequency, while respecting the contiguity constraint. As a result, users with high channel quality get assigned most resources. The scheduling algorithm would take as input a matrix consisting of metric values $r_c^i(t)$, which represent the instantaneous channel rate for user *i* on RB *c* at time *t.* The CD approach suffers from the starvation problem as users with low quality channels are being assigned the least resources, or even may not be assigned any resources at all. This disadvantage is partially solved with the PF paradigm [13, 14]. PF allocates resources to UEs based on the ratio of the UE's channel condition over its lifelong service. As a result, users with low channel conditions will get some resources, but fewer than those with better channel conditions. Hence, fairness is proportional to the channel conditions. PF does that by constructing the matrix $\lambda_i^c(t) = \frac{r_i^c(t)}{R_i(t)}$, where $\lambda_i^c(t)$ is the PF metric value that user *i* has on RB *c* at time slot *t.*and $R_i(t)$ is the long term service rate of user *i* till time *t.* The matrix is then used as input to the scheduling algorithm.

The time domain PF paradigm aims at maximizing the logarithmic utility function $\sum_i \log R_i$ where $R_i$ is the long term service rate of user *i*. To maximize this function, $\sum_i \frac{d_i(t)}{R_i(t)}$ should be maximized, where $d_i(t)$ is the total data transmitted to user *i* at time *t*, and $R_i(t)$ is $R_i$ up till time *t* [14]. To adapt this algorithm to frequency domain scheduling algorithms, let $x_i^c(t)$

denote that RB *c* is assigned to user *i* at time slot *t*; then the goal of the FD algorithm would be to maximize $\sum_i \sum_c x_i^c(t)\, \lambda_i^c(t)$ , while respecting the RB contiguity constraint. However, incorporating the RB contiguity constraint into scheduling algorithms was proven to be NP-hard [14], thus exhaustive search is impractical. For this reason, all proposed FD scheduling algorithms are greedy heuristic ones gaining computational performance and conceptual simplicity at the expense of accuracy.

### C. Uplink Scheduling Algorithms

Most LTE frequency domain uplink scheduling algorithms can be applied to both channel dependent matrix and proportional fairness matrix even though they were originally proposed as CD algorithms [10, 11, 12] or PF ones [13, 14]. This is true because CD and PF contribute only to the way the allocation matrix is formed. The *first maximum expansion* (FME) algorithm was proposed in [12] and was used with the CD matrix. It first assigns RB *j* to user *i* such that M[*i, j*] is the highest metric value, where M is the CSI matrix. Then it expands the allocation on either column *j+1* or *j-1*of M, depending on which has the higher metric value. If that metric value belongs to UE *i*, the RB is allocated to it. If it is not, then that RB is allocated to the new UE and the allocation continues with it. RBs keep getting allocated to the same UE as long as that UE has the highest metric value for the current RB, or higher metric values belong to already served UEs, which would break the contiguity constraint if the RB gets assigned to them. Each UE is considered served whenever another UE having better metric is found. When the expansion on one side reaches the end of the bandwidth, the allocation is spread out on the other side.

The *carrier-by-carrier in turn* was proposed in [14] and used with a PF matrix. It starts assigning resources from left to right to active users having the highest $\lambda^c_i$ on the RB that the iteration in progress is working on, deactivating users that cannot get any more RBs due to the contiguity constraint along the way. This iterative approach has the drawback of not producing a "good enough" schedule. A greedy strategy "take the largest first" is applied in the *largest-metric-value-RB first* algorithm [14]. This algorithm allocates RB *i* to UE *j* in one iteration if there are no RBs assigned to another user between RB *i* and the RBs already allocated to UE *j*, such that $\lambda^i_j$ is the highest remaining metric value. Otherwise, it picks the second highest metric and so on. When a RB is allocated to an UE, all RBs between that RB and RBs already allocated to that UE are assigned to it as well. The Riding Peaks algorithm [14] tries to use each user's highest valued RBs as much as possible. It relies on the *doppler effect*; i.e., in multi-carrier systems, the channel states of a user are correlated in both time and frequency. So if UE *i* has a good channel quality on RB *c*, then it is highly probable that he will also have a good channel quality on RB *c-1* and RB *c+1*. The main idea of this algorithm is to ride user's peaks in the FD. This algorithm assigns a RB to a user that already has allocated RBs if they are neighbors.

### D. Limitations

Channel dependent algorithms try to maximize throughput by exploiting their knowledge of the channel conditions for all users across the entire bandwidth, which may starve distant nodes. PF algorithms, on the other hand, make a compromise between fairness and throughput, in such a way that fairness is proportional to the channel condition; i.e., distant nodes would not starve but would get fewer resources than the ones close to the eNodeB. However, the PF paradigm still suffers from several drawbacks: 1) QoS might not be provided, 2) mobility of UEs might jeopardize fairness, 3) dynamic nature of cellular networks are not fully considered as explained later.

First of all, both FD PF and FD CD algorithms rely on the TD algorithm to provide QoS to users. However, what if distant nodes are the ones that have the highest data priority? On one hand, if a CD paradigm is being used, RBs will be allocated to UEs that have the highest metric values on them, which results in the starvation of distant UEs that have the highest data priority. On the other hand, if a PF paradigm is used, distant nodes would not starve anymore, but since fairness is proportional to channel quality, these UEs would get assigned the least resources, while they have the highest data priority. Moreover, what if packets belonging to distant nodes are getting close to their maximum allowed delay, while those belonging to closer nodes can be delayed without affecting their application's QoS requirements? Consequently, QoS is definitely going to be jeopardized unless the priorities of the UE's applications and their packets' waiting time are taken into consideration in the FD PS.

Furthermore, UE mobility might lead to an unfair treatment and disruption of service. Indeed, PF depends on the long term service rate to provide fairness to UEs. However, consider a user with high channel condition who moves away from the eNodeB at a high velocity or who suddenly goes in a tunnel. As a result, the channel quality of this UE will decrease swiftly, but its long term service rate remains high since it used to have a favorable channel quality. With this combination of channel condition and long term service rate, the UE's $\lambda$ on all RBs would be very low, denying it from getting resources for some time until the system stabilizes.

Finally, the existing works in literature assume that there are *n* UEs, and solve the problem of allocating RBs to them. But one of the fundamental properties of cellular networks is their dynamic nature. For example, UEs close to the eNodeB may keep getting admitted at a high frequency, run a service for a short period of time and disconnect. These UEs will have favorable conditions, and a long term service rate starting at 0 and finishing at a very low value. The other UEs in the system can then be divided into 2 groups: those with low channel quality and those with favorable channel quality but a higher long term service rate. Both groups will get starved, since their instantaneous channel rate over their long term service rate ratio will be lower than the UEs getting repeatedly admitted. Consequently, PF paradigm has a starvation problem when the dynamic nature of cellular networks is taken into account in the resource allocation paradigm.

## III. THE QOS-AWARE RESOURCE ALLOCATION PARADIGM

### A. Basic concepts

The proposed QoS-aware allocation paradigm builds on top of the PF approach, since it aims, in addition to providing QoS to UEs, to guarantee fairness while ensuring a high total throughput. It gives more advantage to UEs having applications of high priority, while not starving UEs whose data is close to the maximum allowed delay. By doing so, QoS would become integrated into the LTE frequency domain packet scheduler. Moreover, UEs whose channel quality drops suddenly as a result of mobility, would not wait too long to receive resources. Indeed, they would start receiving them as soon as their data gets close to its maximum allowed delay. Similarly, UEs with low channel conditions that would get starved if new UEs closer to the eNodeB are getting constantly connected will receive the resources they need. To achieve these objectives, the proposed approach integrates, for each UE, 4 metrics in the creation of the allocation matrix: 1) its channel quality to keep the system throughput high, 2) its long term service rate for fairness purpose, 3) its applications' highest priority with the aim of differentiating between different priorities and providing better service for the highest priority applications, and 4) the time its packets have been waiting for transmission with the intention of providing users with their requested QoS.

In the PF paradigm, each metric value in the allocation matrix is represented by $\lambda_i^c(t) = \frac{r_i^c(t)}{R_i(t)}$, where $r_i^c(t)$ is the instantaneous channel rate for user *i* on RB *c* at time *t*, and $R_i(t)$ is the long term service rate of user *i* from when it began transmitting till time *t*. The proposed approach consists of transforming each metric value $\lambda_i^c(t)$ into $\gamma_i^c(t)$, where $\gamma_i^c(t)$is a function of $\lambda_i^c(t)$, the data of user *i* pending for transmission, their priorities, the delays of every "burst of packets" till time *t*, and the maximum allowed delay for each priority. In the proposed approach, the BS manager should interact with the CSI manager to provide it with the buffer statuses of all UEs as shown in Figure 4.

The burst of packets concept can be explained as follows. When packets are served, the *buffer status* of the corresponding UEs will decrease at the eNodeB. When new packets arrive at the UE's buffers, its *buffer status* will increase when the eNodeB receives a new BSR. The packets might belong to one or more RBGs. Every time a RBG of a certain UE increases at the eNodeB, the added quantity is what we dub a burst of packets. For instance, assume that the buffer status of $UE_i$ at time *t* is 0, 15, 23, and 0 for RBGs 1 through 4 respectively. Now assume that at time *t*, $UE_i$ transmitted 8 data units, 5 from $RBG_2$ and 3 from $RBG_3$, making its buffer status 0, 10, 20 and 0. At *t+1*, a new BSR was received by the eNodeB with values 0, 13, 26, and 0. Two bursts of packets will be recorded, the first is 3, and belongs to $UE_i$ for $RBG_2$ with start time *t+1*, and the second is 6, and belongs to $UE_i$ for $RBG_3$ with start time *t+1*.

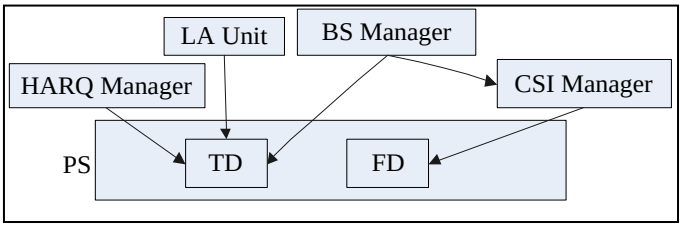

Figure 4. Interaction between the PS and other RRM functionalities.

### B. Computation of the QoS-aware resource allocation matrix

LTE specifications define only 4 RBGs as mentioned earlier. A UE sends a short BSR if it has only 1 RBG and a long one if it has more. That being said, we now define the metric value, $\gamma_i^c(t)$, in the QoS-aware allocation matrix, *γ*, as :

$$\gamma_i^c(t) = \frac{\lambda_i^c(t)}{\alpha_i(t)}$$

where $\alpha_i(t)$ is the QoS introducer of UE *i* at time *t* .

To describe how $\alpha_i(t)$ is calculated, we define first the following notations:

- $D_k$: Let it be 90% of the minimum of the maximum allowed delay of all applications belonging to RBG *k; $D_k$ = min (max delay of application A, max delay of application B, ..., max delay of application Z)*, such that applications A through Z are all applications that belong to RBG *k*. The reasoning behind the 90% will be discussed later.
- $d_i^k(t)$: Let it be the minimum of $D_k$ and the time from when a burst of packets arrived to the RBG *k* of user *i* till time *t*; i.e., $d_i^k(t) = \min\left(D_k, t - startTime(burst_i)\right)$.
- $\tau_i(t)$*:* Let it be the delay weight; $\tau_i(t) = \max\left(\frac{d_i^k(t)}{D_k}\right)$ for $1 \le k \le 4$ and for all packet bursts of user *i*. It starts at 0, since $d_i^k(t)$will be 0 at the first TTI, and as data of user *i* approaches 90% of their maximum allowed delay, it gets closer to its upper limit 1, since $d_i^k(t)$ will be equal to $D_k$ when that value is reached.
- $\delta_i^k(t)$: Let it be the data pending for transmission of user *i* at time *t* in the RBG *k*.

Then for every $\delta_i^k(t) > 0$ and for every user *i*, the QoS introducer, $\alpha_i(t)$ is given as:

$$\alpha_i(t) = \min(k) - \min(k) * \tau_i(t) + \log(\min(k)) + \varepsilon.$$

The $\alpha_i(t)$ equation is divided into four parts that can be described as following:

1) $\min(k)$ which allows $\alpha_i(t)$ to be smaller for users having higher priority data. For example, if $\delta_i^2(t)$ and $\delta_i^3(t)$ are not 0 (i.e., UE *i* has data pending for transmission in $RBG_2$ and $RBG_3$ at time *t*), *min(k)* would be 2.

2) $-\min(k) * \tau_i(t)$ which starts at 0 and reaches $-\min(k)$ when data have been pending for transmission for 90% of their maximum allowed delay, since $\tau_i(t)$goes from 0 to 1.

3) $\log(\min(k))$ which allows the differentiation between users having different priorities if 90% of their maximum allowed delay is reached at the same time, thus giving an advantage to the highest priority users.

4) $\varepsilon$ which is used to never allow $\alpha$ to be 0.

The reason why $D_k$ was chosen to be 90% of the minimum of the maximum allowed delays of applications whose packets belong to RBG $k$ is to make $\tau_i(t)$ reaches 1 and consequently $\min(k) - \min(k) * \tau_i(t)$ reaches 0, which gives the user the opportunity to be allocated resources before the maximum allowed delay is reached.

Figure 5 illustrates how the proposed approach calculates the resource allocation matrix.

```
1.  Let M_{i,c} be the channel quality indicator of UE i on RB c
2.  Let R_i be the long term service rate of UE i
3.  Let BS_{i,k} be the buffer status of UE i in RBG k
4.  for UE i = 1 to n do
5.      Update R_i
6.      max = 0
7.      min = 0
8.      for RBG k = 4 to 1 do
9.          Update BS_{i,k}
10.         If δ_i^k > 0 then
11.             d_i^k ++
12.             min = k
13.             If max < d_i^k / D_k then
14.                 max = d_i^k / D_k
15.             end if
16.         end if
17.     end for
18.     τ_i = max
19.     α_i = min - min * τ_i + log(min)
20.     for RB c = 1 to m do
21.         λ_i^c = M_{i,c} / R_i
22.         γ_i^c(t) = λ_i^c / α_i
23.     end for
24. end for
```

Figure 5. Creating a QoS-aware allocation Matrix.

Table I illustrates the differences between the PF matrix and the one of the proposed QoS-aware approach. In this table, we calculate the expected values of $\alpha$ and $\gamma$ taking the different RBGs into account. As an example, for users with data belonging to $RBG_1$, when packets just hit the transmission queues (i.e., $d_i^1 = 0 \rightarrow \tau_i = 0$), then $\alpha_i(t) = \min(k) - \min(k) * \tau_i(t) + \log(\min(k)) + \varepsilon = 1 - 0 - 0 + \varepsilon = 1$ and $\gamma_i^c(t) = \frac{\lambda_i^c(t)}{\alpha_i(t)} = \lambda_i^c(t)$ ; but when packets have been delayed for $D_1$ (i.e., $d_i^1 = D_1 \rightarrow \tau_i = 1$), then $\alpha_i(t) = 1 - 1 - 0 + \varepsilon = \varepsilon$ and $\gamma_i^c(t) = \frac{\lambda_i^c(t)}{\alpha_i(t)} = \frac{\lambda_i^c(t)}{\varepsilon} = \infty$. For users whose highest priority data belongs to $RBG_2$, when packets just hit the transmission queues, then $\alpha_i(t) = 2 - 2 * 0 + 0.3 + \varepsilon = 2.3$ and $\gamma_i^c(t) = \frac{\lambda_i^c(t)}{2.3} = 0.435\lambda_i^c(t)$ ; but when packets have been delayed for $D_2$ (i.e., $\tau_i = 1$) then $\alpha_i(t) = 2 - 2 * 1 + 0.3 + \varepsilon = 0.3$ *and* $\gamma_i^c(t) = 3.33\lambda_i^c(t)$ and so on. The table illustrates how the value of $\gamma_i^c(t)$ (which represents the allocation matrix metric in the QoS-aware approach) is varying compared to $\lambda_i^c(t)$ (which represents the allocation matrix metric in the PF approach) and how the proposed QoS-aware paradigm is differentiating between the different RBGs and giving higher priority to RBG 1 over RBG 2, which, is given, in its turn a higher priority over RBG 3, and so on.

TABLE I. ILLUSTARTION PF VS. QOS AWARE

| | $\gamma_i^c(t) = \frac{\lambda_i^c(t)}{\alpha_i(t)}$ **as $\tau_i$ goes from *0* to *1*** |
|---|---|
| **$RBG_1$** | $\lambda_i^c(t) \rightarrow \infty$ |
| **$RBG_2$** | $0.435\lambda_i^c(t) \rightarrow 3.33\lambda_i^c(t)$ |
| **$RBG_3$** | $0.287\lambda_i^c(t) \rightarrow 2.08\lambda_i^c(t)$ |
| **$RBG_4$** | $0.217\lambda_i^c(t) \rightarrow 1.66\lambda_i^c(t)$ |

## IV. PERFORMANCE EVALUATION

We have evaluated the performance of the proposed approach and compared it with CD and PF paradigms using NS3 [2]. In this evaluation, we have implemented and integrated into NS-3 the three paradigms using two scheduling algorithms: the FME algorithm [12] and the Riding Peaks algorithm [14]. Each of these algorithms was implemented three times, each time with a different resource allocation matrix (i.e., created either by the proposed QoS-aware, CD, or PF). The simulation parameters are summarized in Table II.

TABLE II. SIMULATION PARAMETERS

| Parameter | Value |
|---|---|
| System Type | Single Cell |
| Channel Model | Urban |
| # of Active Users in Cell | 8, 24, 48, 96, 144, 192, 240 |
| Users Distribution | Random |
| Traffic Model | Infinitely Backlogged |
| User Transmit Power | 125 mW |
| System Bandwidth | 5 MHz |
| # of RBs | 25 |
| # of Subcarriers per RB | 12 |
| RB Bandwidth | 180 KHz |
| Transmission Time Interval (TTI) | 1 ms |
| Maximum Delays | 10 ms, 40 ms, 90 ms, 150 ms |
| Mobility | Random < 30 Km/h |
| Simulation Time | 10000 TTIs |

### A. System Throughput

We first study the total throughput achieved by the three approaches. The system throughput is defined as the overall amount of user data carried by the system and calculated as $Th = \frac{\sum_{i=0}^{n} PacketSize_i}{SimTime}$ where $PacketSize_i$ is the size of packet $i$ in Mbits, $n$ is the number of packets transmitted throughout the whole simulation, and *SimTime* is the simulation time in seconds. Results are shown in Fig. 6(a). We can observe that CD paradigm provides the best system throughput. This is normal because CD algorithms allocate RBs to UEs with the highest channel conditions, and the transmission rate and the

channel condition are proportional to each other. The difference between PF and QoS-aware can be justified as follows. Both resource allocation schemes include fairness, but QoS-aware paradigm distinguishes between UEs and allocates less resources to low priority applications in favor of giving more resources for higher priority applications. Therefore, when users having the highest priority applications reside close to the eNodeB, their throughput in a particular TTI would be the same as in PF. On the other hand, when they reside far from the eNodeB, their throughput would be less than the throughput of users with lower priority applications residing closer to the eNodeB, which are granted more RBs if PF was used. We also observe that Riding Peaks scheduling algorithm has a better performance than FME. This is normal since FME finds the first maximum and then works iteratively on the RBs, while Riding Peaks recursively find the maximums, hence providing better performance.

### B. *Fairness Index*

We then study the fairness of the approaches using Jain's Fairness Index. The formula is given as $f(x_1, x_2, \dots, x_n) = \frac{(\sum_{i=1}^{n} x_i)^2}{n \times \sum_{i=1}^{n} x_i^2}$, where there are $n$ UEs in the system and $x_i$ is the number of resources given to UE $i$. Jain's Index returns a number between 0 and 1, with 1 being perfectly fair. Results presented in Fig. 6(b) show that CD resource allocation scheme has a very low fairness index no matter how many users are in the system, which is understandable, since the only UEs that will receive resources are those with the best channel conditions, and all those not having the highest CQI are subject to be starved. Both PF and QoS produce a relatively fair resource allocation scheme, which slowly decreases as the number of users increases. This slow decrease is reasonable since fairness is proportional to the channel conditions; hence with more users in the system, more users have low and high channel conditions. As a result, more users are getting additional resources than others, thus decreasing the fairness in the system. We also observe that Riding Peaks outperforms FME for the same reasons mentioned previously.

### C. *Users Served*

With this very low fairness index when a CD algorithm is used, one has to wonder about the number of users that are being served. Fig. 7(a) shows the number of users that are being granted resources, with respect to the total number of users in the system, which is calculated as $\sum UE_i$ such that $UE_i$ has been assigned at least 1 RB throughout the whole simulation. The figure shows how unfair CD paradigm is since out of 240 users in the system, less than 20 were being granted resources. PF and QoS allocate resources to all users, as expected. We then study the number of users being served efficiently; i.e., those whose QoS requirements (such as maximum delay, jitter, throughput, or/and error rate) are met. Fig. 7(b) shows that CD paradigm provides a very small number of users with their required QoS. This is justified since only a small number of users are being granted resources in the first place. QoS and PF paradigms, on the other hand, give far better results, but as the number of users increases in the system, less users get served efficiently. With PF-FME the rate of users who are getting efficiently served drops fast as the number of UEs increases in the system. With 240 users in the network, PF-FME provides the QoS requirements of only 117 on average, that's 47%. QoS-FME and QoS-Riding Peaks, alternatively, serves most users efficiently, with a very slow decrease in the rate of users efficiently served as their number grows large. By seeing this decreasing number of users whose QoS requirements are being met, we can observe how CD and PF paradigms jeopardize the UE QoS requirements. So even though users are being selected for potential scheduling in both paradigms, a large number of those users may not get the required resources.

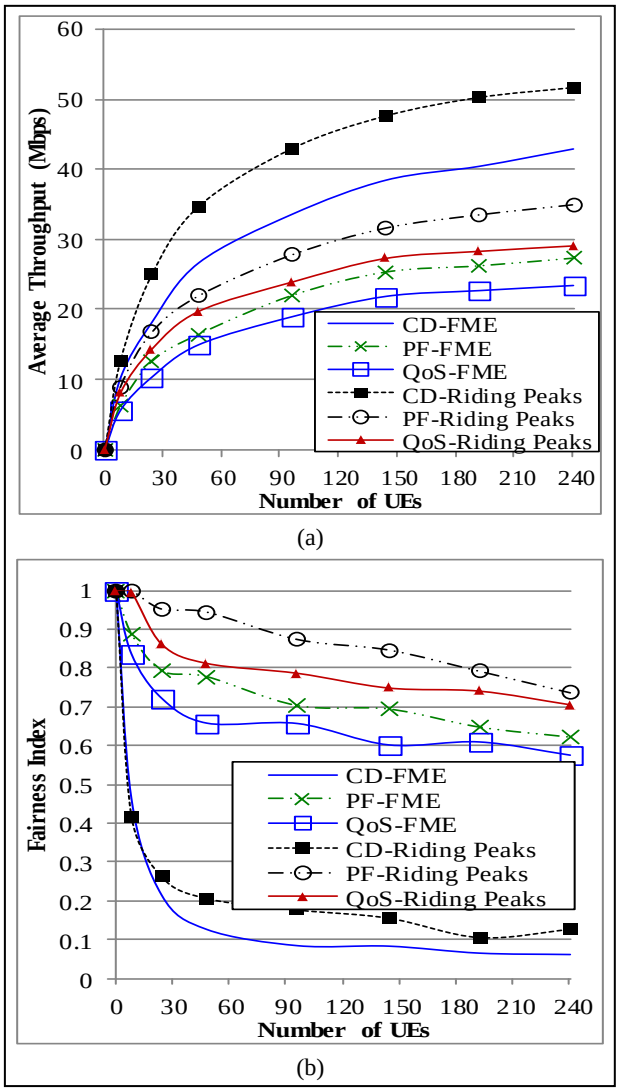

Figure 6. (a) System throughput, (b) Fairness index .

### D. *Packets Arriving after their Maximum Allowed Delay*

We next study the rate of delayed packets while taking RBGs into account. Figure 8 shows that with both CD and PF allocation paradigms the packets that are being delayed the most belong to RBG1, which are the packets having the highest priority. Then, less packets are being delayed working our way to RBG4, which has a 0% rate of delayed packets. These results explain also why the number of users getting served efficiently decreases fast as the number of users increases in the system. Indeed, by not taking RBGs into account, all packets are being treated equally. But lower priority data are

more delay tolerant than higher priority data. The figure shows that with QoS paradigm, the scheduler delays lower priority packets to deliver those with higher priority without violating their delay requirements, thus the high delay for low priority packets (i.e., RBG 4). With QoS-aware paradigm, RBG 1 packets are never delayed more than 10 ms before they get transmitted even when the number of operational UEs reaches 240. RBG 2 packets start getting delayed only with a relatively high number of UEs. With 240 active users, only 3% of them get delayed. RBG 4 packets, on the other hand, have a high average delay of 150 ms when 240 UEs are in the system.

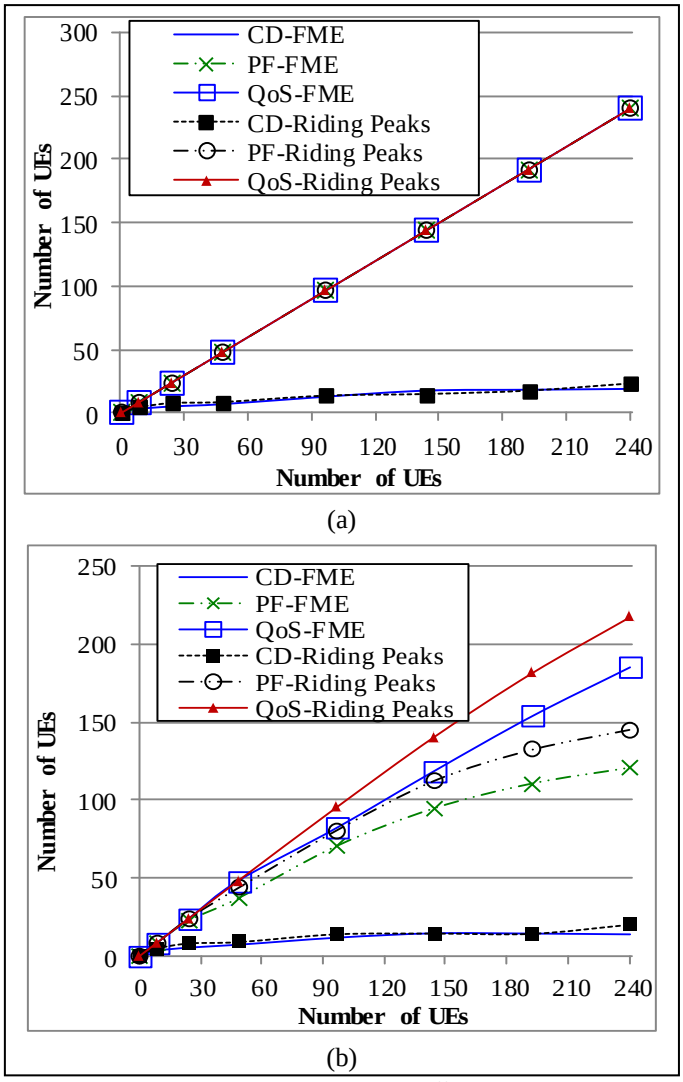


Figure 7. (a)User Served, (b) Users Efficiently Served

## E. *Mobility*

This experiment shows that if a UE with good channel conditions moves fast away of the eNodeB or its channel condition degrades suddenly for any reason, it will definitely be denied resources for the services it is running with CD paradigm, and will most probably have its services interrupted with PF paradigm, which is not the case in our approach. The simulations involved 47 UEs randomly distributed around the eNodeB, and 1 user close to the eNodeB, who goes underground (in a tunnel for instance) after transmitting for 1 minute, i.e. his CQI decreases drastically instantaneously. We monitored this user's throughput over the simulation time. Fig. 9(a) shows that with CD paradigm, the user was denied service for good as his CQI dropped. With PF algorithms, on the other hand, that UE was denied resources for around 400 ms, and it then continued to be served with a lower throughput. These 400 ms were the time needed for the system to stabilize (i.e., user's $\lambda$), On the other hand, when QoS-aware paradigm is used, the user kept getting resources without any disruption. Fig. 9(b) shows a clearer picture of what went on in the first 700 ms after the CQI decreased.

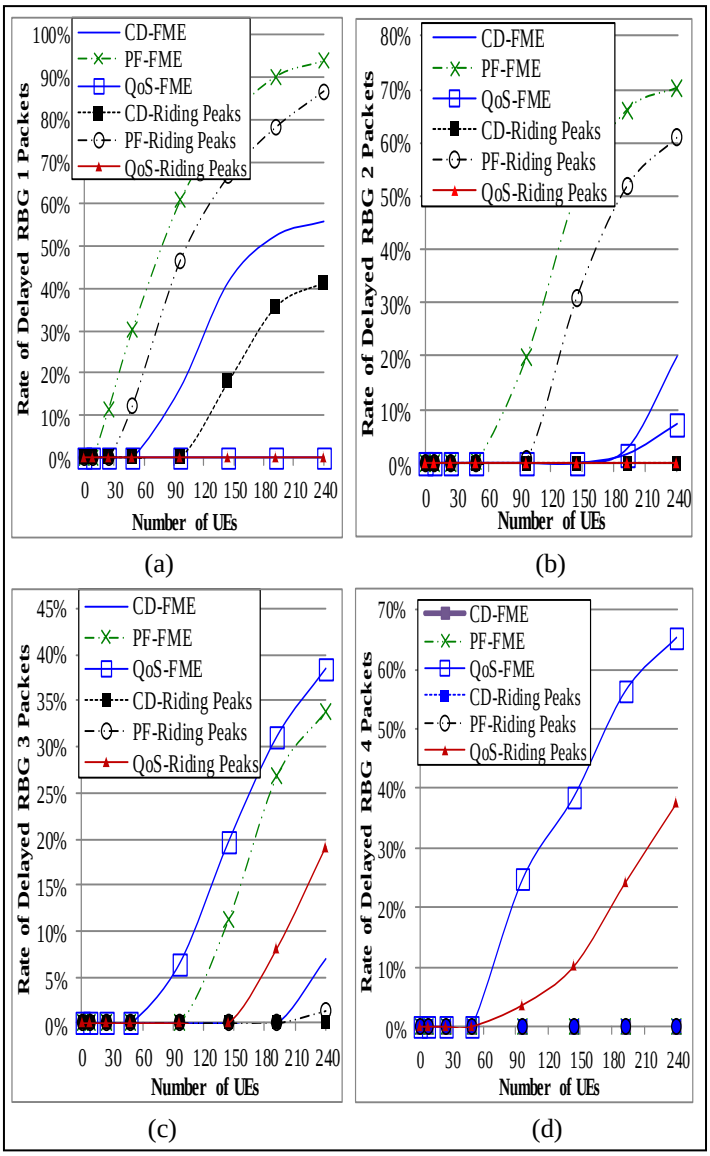


Figure 8. Delayed Packets Rate of (a) RBG1, (b) RBG2, (c) RBG3, (d) RBG4.

## V. CONCLUSION

In this paper, we have proposed a QoS-aware resource allocation paradigm for frequency domain scheduling algorithms in LTE networks. Unlike the existing CD and PF

paradigms, the proposed approach takes into consideration the UE's QoS requirements when creating the resource allocation matrix that is passed the frequency scheduling algorithm. It favors UEs with high priority data, while not starving other UEs. It also handles the mobility related problems such as UEs whose CSI drops suddenly and UEs with high ping-pong rate (high rate of joining and leaving the network). We have implemented the proposed approach, CD, and PF paradigms and integrated them into NS 3 using two scheduling algorithms (FME and Riding Peaks). Simulation results show that the proposed approach provides better QoS to UEs, is more scalable, solves the problems related to mobility and to the dynamic nature of networks. However, it does have a lower system throughput. But by weighing the advantages with the disadvantages of the three approaches, it is definite that a QoS-aware technique for uplink frequency scheduling in LTE provides is indeed a need.

## ACKNOWLEDGMENT

This work was supported in part by a grant from the Lebanese National Council For Scientific Research (no. 01-13-12, LNCSR-2012/13).

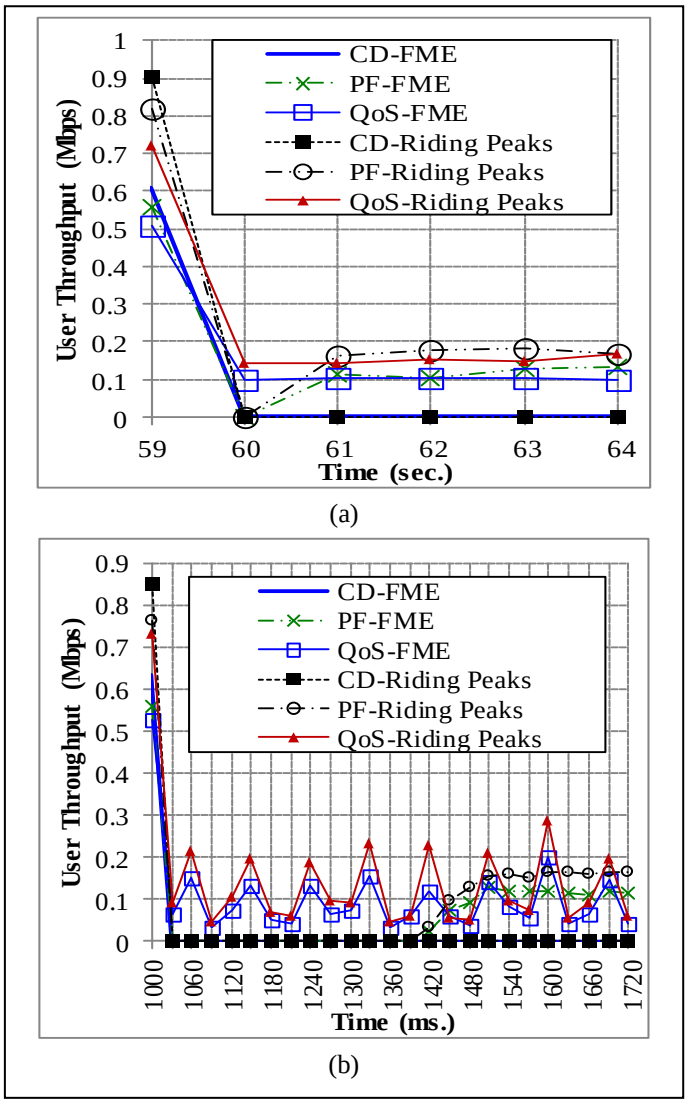


Figure 9. Mobiliy problem

## REFERENCES

[1] "LTE," 3GPP. Available from http://www.3gpp.org/LTE; accessed 14 October 2010.
[2] NS-3, Available from http://www.nsnam.org/; accessed 16 January 2011
[3] M. Rinne, and O. Tirkkonen, "LTE, the Radio Technology Path toward 4G," in Computer Communication, Volume 33, Issue 16, pp. 1894—1906, October 2010.
[4] "TR 23.882, System Architecture Evolution (SAE)," 3GPP.
[5] "3GPP". Available from http://www.3gpp.org/; accessed 14 October 2010
[6] A.J. Paulraj, D.A. Gore, R.U. Nabar, and H. Bolcskei, "An Overview of MIMO Communications – Key to Gigabit Wireless," in Proceedings of the IEEE, Volume 92, Issue 2, pp. 198—128, February 2004.
[7] M. Rumney, "3GPP LTE: Introducing Single-Carrier FDMA," in Agilent Measurement Journal, Issue 4, Second Quarter 2008, pp. 18--27, 2008.
[8] "TS 36.321, E-UTRAN, MAC protocol specification," 3GPP.
[9] "TS 23.203, RAN; E-UTRA; E-UTRAN; Overall Description, Stage 2," 3GPP.
[10] J. Lim, H.G. Myung, K. Oh, and D.J. Goodman, "Channel-Dependent Scheduling of Uplink Single Carrier FDMA Systems," in Vehicular Technology Conference, 2006, VTC Fall 2006, IEEE, Montreal, QC, Canada, September 2006.
[11] J. Lim, H.G. Myung, and D.J. Goodman, "Single Carrier FDMA for Uplink Wireless Transmission," in IEEE Vehicular Technology Magazine, Volume 1, Issue 3, pp 30—39, 2007.
[12] L. R. de Temino, G. Berardinelli, S. Frattasi, and S. Mogensen, "Channel-aware scheduling algorithms for SC-FDMA in LTE uplink," in Personal, Indoor and Mobile Radio Communications, 2008. IEEE 19th International Symposium, Cannes, France, December 2008.
[13] F. Calabrese, P. Michaelsen, C. Rosa, M. Anas, C. Castellanos, D. Villa, K. Pedersen, and P. Mogensen, "Search-tree based uplink channel aware packet scheduling for UTRAN LTE," in Vehicular Technology Conference, 2008, VTC Spring 2008, IEEE, Singapore, pp. 1949--1953, May 2008.
[14] L. Suk-Bok, I. Pefkianakis, A. Meyerson, X. Shugong, and L. Songwu, "Proportional Fair Frequency-Domain Packet Scheduling for 3GPP LTE Uplink," in IEEE Proceedings of the 28th Conference on Information Communications (INFOCOM) 2009, Rio De Janeiro, Brazil, pp. 2611—2616, June 2009.