Received: March 17, 2021. Revised: April 7, 2021.

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

A New Processing Approach for Scheduling Time Minimization in 5G-IoT Networks

Ali Majeed Mahmood[1] **Azad Raheem Kareem[1]***

[1]*Control and Systems Engineering Department, University of Technology-Iraq, Baghdad, Iraq*
* Corresponding author's Email: Azad.R.Kareem@uotechnology.edu.iq

Abstract: Increasing the number of mobile network connections represented by humans and machines leads to a higher demand for resources, which have to be allocated for all User Equipment (UEs) with an acceptable level of fairness. However, the cost of allocating the required resources for a great number of connections may result in growing the computational complexity of scheduling algorithm and can delay the response of the Base Station (BS). In this paper, a new processing approach is proposed for Fifth Generation - Internet of Things (5G-IoT) networks. The computational complexity of the scheduling process is minimized in two folds. First, the Spark framework is used as a distributed and parallel processing paradigm, which can reduce the computational complexity from $O(N^2)$ to $O(N/M)^2$. Then, to minimize the scheduling latency of the scheduler itself. The second level of complexity minimization is presented by using a low computational scheduling algorithm via modifying the traditional proportional fair scheduling algorithm to further decrease the complexity from $O(N/M)^2$ to $O(N/M)$. Simulation results demonstrate the efficacy of the proposed scheduling approach by applying the concept of divide and conquer in achieving the scheduling process. Hence, the effectiveness of the new scheduling approach represented by scaling down the time of scheduling large number of UEs to the time of small sub-clusters of UEs. The desired percentage of gain in Scheduling Time (ST) depends on the size of sub-clusters such as deploying 250 UEs in form of clusters of (50, 30, or 10) UEs will reduce the ST compared to the classical Proportional Fair (PF) with a percentage of 85%, 89%, and 97% respectively. This property can be exploited to support network scalability at low scheduling latency in next-generation mobile networks.

Keywords: 5G, IoT, Scheduling algorithms, Proportional fair algorithm, Spark framework, Scalability, Parallel processing.

1. Introduction

Mobile networks have been evolved dramatically in recent years. Hence, one million connections for each square kilometre at the one-millisecond end-to-end latency is anticipated with the deployment of the Fifth Generation - Internet of Things (5G-IoT) networks [1, 2]. This is along with higher data rates mobile applications that are working on smartphones, for instance, media streaming, gaming applications, video conferencing, and other online applications. This is coming side by side with the massive number of sensor nodes in the IoT networks for the technologies of a smart city including smart homes, metering, security and surveillance, self-driving cars, eHealth, and smart grids, and other uncountable numbers of future applications. the point of interest here is that all of these applications would share the same infrastructure and always in high demand for resources for their operation [3]. All of the former aspects reveal how cellular networks are continually expanded exponentially. On the other hand, the available bandwidth is limited [4, 5], which requires to be partitioned and managed by efficient scheduling algorithms between the active users to ensure fairness and high spectral efficiency. In the next-generation cellular networks (5G), the network includes several characteristics such as scalability with reliable connectivity to the world, along with the high data-rate and low-latency. Fig. 1 shows the amount of reduction in the latency of 5G is almost 10 times

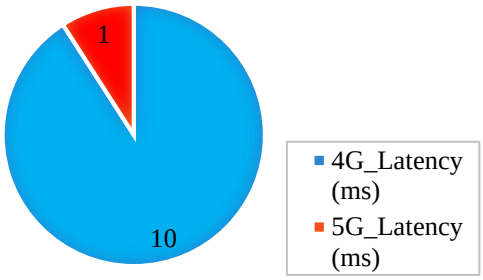

Figure. 1 The desired theoretical latencies in 4G and 5G

smaller compared to the Fourth Generation (4G) network. All of these new superior features of 5G cellular networks can be utilized for fast real-time transfer of data [6]. Consequently, this enables many innovative applications and technologies to be deployed that have not been possible before 5G such as IoT technology. Hence, the burden of computation overhead in the networks will be increased accordingly. It is worth stating that 5G-IoT networks will not only include the general-purpose smartphones, tablets, or computers, but also, there are different other types of low power and cost User Equipment (UEs) connected to the network with almost fixed functionality, such as thermostats, security cameras, door locks, and even connected kitchen appliances. According to the most recent Ericsson mobility report in June 2020 [7], by the end of 2025, the massive IoT connections are expected to be 52 percent of all cellular connections.

In this paper, the main contributions are presented as follows:

1) Application of Spark framework as a distributed and parallel processing approach in the scheduling algorithm of cellular networks to minimize the scheduling delay of serving a dense number of contended UEs and then maintaining the network scalability.

2) A modified low computational Proportional Fair (PF) scheduling algorithm is proposed.

The rest of the paper is structured as follows: In section 2, the related literature is discussed. Section 3 presents a general explanation for the main theoretical components of this research as well as the problem of the research is defined and investigated from different aspects. In section 4, the proposed modified PF is explained. In section 5, the proposed Spark framework is illustrated. The performance evaluation and discussion are shown in section 6. Section 7 wraps up the paper with the concluding remarks.

2. Related works

Since the types of scheduling algorithms have not been specified in the standards of the Third Generation Partnership Project (3GPP), the design of the optimal scheduler for meeting the intended objectives is still an open research problem [8]. Hence, several research studies have been conducted to focus on developing an efficient scheduling algorithm. In general, after reviewing the existing contributions in this field, several routes have been followed by the researchers. The first one is to develop resource scheduling algorithms to improve spectral efficiency [9, 10]. An additional research field has been followed to design a scheduler for minimizing the inter-cell interference of cell edge UEs [11]. The third trend of studies has been mainly conducted to maintain the fairness requirements between UEs [12], while the fourth area of research has focused on the trade-off between the fairness and the spectral efficiency between UEs [13]. Furthermore, one of the key research directions that have been conducted is to design a low computation complexity scheduling algorithm or to minimize the latency that may be generated in the scheduling process where this route represents the point of interest in this paper. It is worth mentioning that the presence of latency in allocating resources can lead finally to degrade the performance of the entire network. Hence, the number of contributions within the field of computational complexity minimization for the resource allocation algorithms is discussed broadly as follows. The authors in [14] have designed a linearized model to simplify the optimization of a nonlinear problem into a simpler linear program for the problem of resource allocation to maximize the data rate. Likewise, in [15], based on the concepts of matching theory, the authors propose a schedule-oriented optimization technique in mm-wave mobile networks. The technique is capable of solving the problem of maximum throughput-fair scheduling via a linear program. The authors in [16] have designed a suboptimal association approach for resource allocation between the network flying platforms such as unmanned aerial vehicles and small cells to maximize the data rate subject to the quality of service and the available bandwidth, where the complexity is minimized through the application of centralized and distributed algorithms. The near-optimal resource management method is also investigated in [17] for the Multiple-Input and Multiple-Output (MIMO) systems, which has been focused on the sum rate and the fairness between users. In [18], a cross-layer algorithm is proposed to

attain high performance on lower feedback with low algorithmic complexity. In [19], The authors have developed an algorithm that can exploit the gains of multi-user scheduling frequency-selective whereas avoiding unnecessary segmentation of ultra-reliable low latency payloads for various transmissions. In [20], a resource allocation method in the Long-Term Evolution (LTE) network is used to maximize the data throughput via solving a low complexity optimization problem of a frame level. The authors in [21] suggest a low complexity scheduling algorithm using Federated machine learning via the so-called age of update as a metric for this purpose. In [22], a resource allocation approach has been developed, which considers a nature-inspired algorithm using slice characteristics for low computational complexity and resource utilization. The authors in [23], have designed an approach for assigning a sub-channels bandwidth to several Device to Device (D2D) communication pairs and remote radio unit. In this way, pre-allocated sub-channels can be reused at low computational complexity. The resource allocation that has been proposed in [24] deployed an outer approximation algorithm to minimize the computational complexity that may be generated from the full search, which can be enlarged with the increasing number of users. In [25], a low complexity scheduling algorithm has been proposed based on a concept of delay outage minimization to support real-time traffic with varied traffic classes in LTE/ LTE-Advanced (LTE-A) networks. In the delay constraint solutions, the optimization problem becomes unfeasible in case that the number of UEs exceeds the existing resource units. Conversely, multi-objective optimization is computationally expensive. The authors in [26] have suggested a scheduling algorithm with two stages to meet the delay time limit, which is the time-domain packet and frequency-domain packet scheduling. However, this can lead finally to a suboptimal solution [27]. The authors in [28], design a channel allocation technique named hyper-fraction for a vehicle-to-vehicle communication system. The method splits the road into several zones to allocate a channel to vehicles located within the zone for latency minimization. The Genetic Algorithm (GA) is used to solve optimization problems of resource allocation. The authors of this study are appreciated since they attempt to decrease latency between the Base Station (BS) and the UE by using the concept of splitting the task of resource allocation into small zones. However, they use an approach to attain this purpose, which inherently entails high computational complexity. Hence, the advantage of minimizing the latency may be degraded due to the complication of the suggested algorithm. The authors additionally do not examine how the latency is increased when the size of the network is scaled up.

In this paper, one of the most important scenarios of the 5G-IoT network that has been rarely studied is the impact of the computation overhead or the latency of the scheduling algorithm, particularly when scaling up the network to accommodate a huge number of UEs in the future networks. In other words, each UE will suffer from the burden of increasing the network size. Therefore, the Spark framework is proposed to serve the UEs but in the form of groups by shafting the Scheduling Time (ST) from the entire network to the specified size of a small group of UEs, which significantly improves the network scalability. Adopting of Spark framework in the mobile network can be considered as the key contribution in this paper since it can extend the scalability of the network to a massive number of UEs via its processing capabilities. To the best of the authors' knowledge, there is no former research that has used or proposed the Spark framework in the scheduling algorithm of the cellular networks. For further reduction in the computation complexity, a modified two-step pre-calculation proportional fair scheduling algorithm is also developed in this work.

3. Preliminaries

In this section, the essential components that are related to the proposed approach of this paper are discussed and explained briefly, including the most common scheduling algorithms and the Spark framework architecture. Also, the problem definition is illustrated concisely.

3.1 Scheduling algorithms

Three of the most common scheduling algorithms are explained briefly as follows:

Round Robin (RR) scheduling is a channel unaware algorithm. RR is working based on the idea of allocating Physical Resource Blocks (PRBs) cyclically for all UEs without considering the channel conditions, which can be realized by the report of the Channel Quality Indicator (CQI) of the connected UE. In RR, same resources will be allocated for each user. Hence, the UEs are equally scheduled [29]. Although RR has low complexity, its major drawback is that it works without taking into account the CQI of UEs, which reflects the current quality of the link and this may lead to lower data throughput due to fact that the UE may not be served based on its real demand of resources.

Proportional Fair (PF) scheduling is another common scheduling algorithm. Unlike the operation of the RR, in the PF scheduling algorithm, the UEs are assigned the PRBs based on the consideration of the channel quality as well as the fairness level. The key objective of the PF algorithm is to reach a balance between the data rate and the fairness to prevent UEs from starvation. In the PF scheduling algorithm, the method of resource allocation can be determined according to the following expression [30]:

$$Q(i) = arg\,max_i \left[\frac{R_i(t)}{\bar{R}_i(t)}\right] \quad (1)$$

where, $R_i(t)$ is the current transmission rate for UE i; $\bar{R}_i(t)$ is the average data transmission rate for user i, which represents the history of all previous data rates of ith UE; Q is the priority of scheduling; $i = 1, 2, \ldots N_{UEs}$; N_{UEs} is no. of active UEs. Based on Eq. (1), the PF algorithm is examined the average UE throughput to maintain the fairness level between the UEs throughout the process of scheduling [31].

The third most comment scheduling algorithm is the Best-Channel Quality Indicator (BCQI). The basic operation concept of the BCQI scheduling algorithm is to allocate the available resource to the UEs that are experienced good channel conditions [31]. The UE that sends the best CQI value to the BS has the highest scheduling priority in this scheduler compared to other UEs. Hence, finding the BCQI is periodically determined in the BCQI scheduling algorithm as shown in Eq. (2).

$$Q(i) = arg\,max_i\, R_i \quad (2)$$

where, R_{i} is the recent transmission throughput for UE_{i}; Q is the priority of scheduling.

3.2 Background to spark framework

Spark represents an open-source framework that has been used to process efficiently a great amount of, structured, semi-structured, and unstructured data [32]. The architecture of Spark is considered as an alternative to Hadoop and map-reduce architecture for the processing of big data due to its superior features. Spark architecture is associated with Resilient Distributed Datasets (RDD) besides the Directed Acyclic Graph (DAG) for data storage and processing [33]. It consists of four components that are part of the architecture as shown in Fig. 2 including i) Spark driver (master node), which is in charge of translating the input code into real spark tasks to be run in the cluster. The driver program is also negotiated with the cluster manager during the operation; ii) Executors are distributed agents in control of the execution of the tasks in the Spark, perform all the data processing, and saves the produced results in the storage unit. Every spark application has its executor process; iii) Cluster manager can be considered as a bridge that acquires resources on the Spark clusters then allocates them to the spark job, the cluster manager manages and allocates the required system resources to the Spark jobs. Furthermore, it manages and follows the track to the live/dead nodes in a cluster. It enables the execution of jobs submitted by the driver on the worker nodes (also called Spark workers) and finally tracks and shows the status of various jobs running by the worker nodes; iv) Worker nodes are executing the business logic submitted by the Spark driver. Spark workers are abstracted and are allocated dynamically by the cluster manager to the Spark driver for the execution of submitted jobs. Spark utilizes the dataset and data frames as the primary data storage component that helps to optimize the Spark process and the big data computation. The common features [34] of the Spark framework are illustrated in Fig. 3.

3.3 Problem definition

In fact, there is no ideal scheduling algorithm that can maintain all UEs' requirements particularly in networks like the 5G-IoT. This means that improving

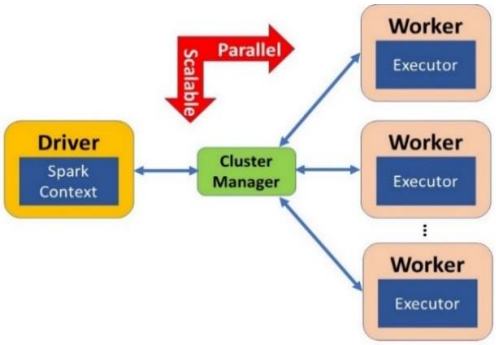

Figure. 2 General diagram of Spark architecture

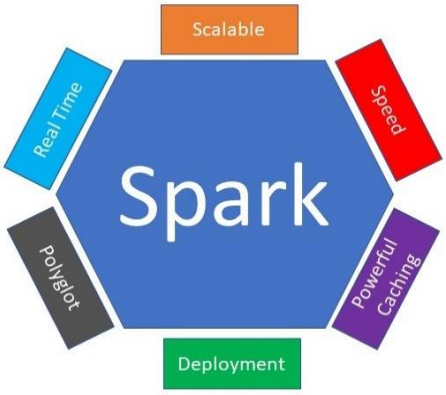

Figure. 3 Common features of Spark framework

data flow will be at the expense of fairness between UEs and vice versa. Another aspect is the amount of computation overhead of the scheduling algorithms when the number of connections increased in the network. Hence, to explore the characteristics of the RR, PF, and BCQI algorithm, simple simulation tests are conducted. The results depicted in Fig. 4 demonstrate the overall cell throughput of the scheduling algorithms with 5 UEs at 1 km/hr. The best CQI scheduler as illustrated has the highest performance in terms of data rate whereas the RR has the lowest performance and the PF is located between the BCQI and RR. This is because BCQI is a channel-aware scheduler which means it responds to the real conditions of the UEs, unlike the RR algorithm.

Fig.5 illustrates the fairness measurement, which is quantified using Jain's Fairness Index (JFI) as expressed in Eq. (3) of each scheduling algorithm. The results show that the RR has the best fairness then followed by PF and finally the worst fairness is presented by the BCQI due to allocating resources to UEs of the highest channel conditions without considering the other low channel condition UEs. Hence, BCQI is better in providing data throughput but it has a problem in fairness provision and vice versa for the RR scheduling algorithm, whereas PF performance is in between the characteristics of both the BCQI and the RR algorithms.

$$JFI(tp) = \frac{(\sum tp_i)^2}{N_{UEs} \cdot \sum {tp_i}^2} \quad (3)$$

where, JFI is the fairness index, tp is a vector of UE measured throughputs. The average throughput of user i, $N_{UEs}$is no. of active contending users.

The test in Fig. 6 illustrates the ST of RR, PF, and BCQI for 1UE, where, the RR has the lowest scheduling time compared to PF and BCQI since RR is an unaware channel scheduling algorithm as mentioned earlier. On the contrary, BCQI has the largest time due to the extra processing procedure represented by finding the UE of the best CQI among all active UEs.

Hence, based on the previous results, the PF scheduler can be nominated to be used with the proposed Spark framework since its performance represents a compromise between fairness, data throughput, and the computation overhead of the RR and BCQI schedulers. The next test shown in Fig.7 is focused on the latency that can be added to the system by scaling up the number of UEs in the network by using only the PF scheduling algorithm. The result shows that the latency of the scheduling task begins

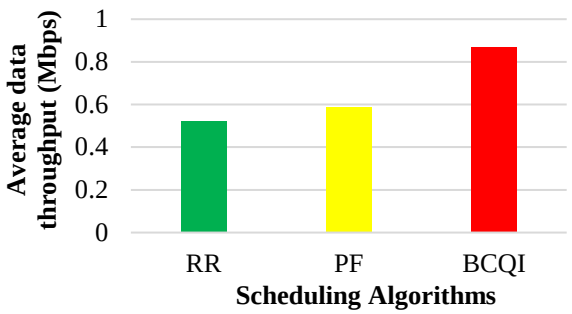

Figure. 4 The cell throughput with the RR, Best CQI, and PF scheduling algorithms (5 UEs, 1.4 MHz ***BW***, 1.9 GHz, and 1x1 antenna system)

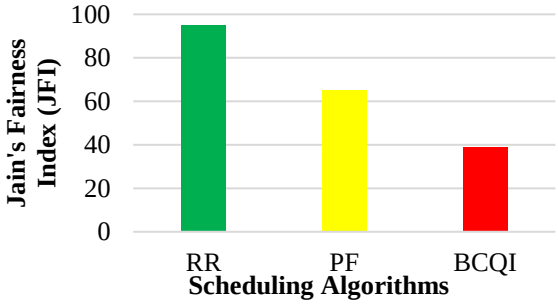

Figure. 5 The fairness of RR, Best CQI, and PF algorithms (5 UEs, 1.4 MHz ***BW***,1.9 GHz, and 1x1 antenna system)

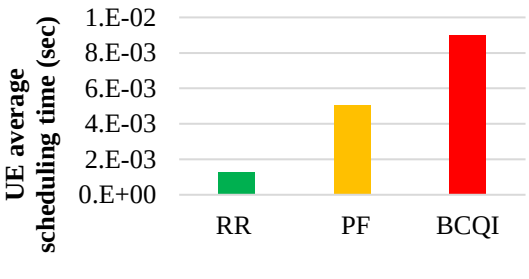

Figure. 6 Comparison of the applied scheduling algorithms in terms of average ST per UE

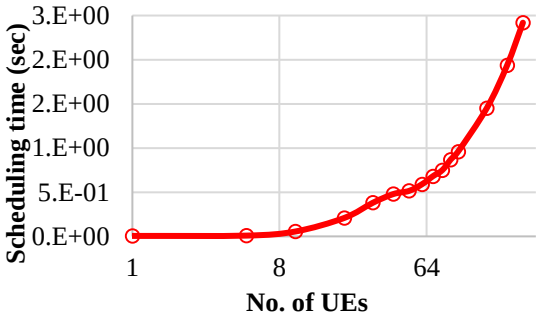

Figure. 7 Growing scheduling time with increasing number of UEs using PF

to rise with increasing the UEs' number. It can be logically noticed that the produced latency will finally limit the scalability of the network particularly the latency is one of the key metrics and essential requirements of the 5G-IoT networks.

4. The proposed modified proportional fair scheduling algorithm

The complexity of PF algorithm is generated from extra calculation for maintaining the fairness between the contenders UEs. As shown in Eq. (1), the pervious history of all data transmission rates of certain UE has to be determined simultaneously at the time of calculating the priority Q for that UE. Then based on the value of Q, the UE will be either scheduled and received resources or not. The burden of computation time for finding the priority can be lessen or relaxed in the PF scheduling algorithm by using two steps pre-calculation for the average data rate per UE as shown in Fig. 8.

The basis of priority calculation in the proposed Modified-PF (MPF) algorithm depends on the equation of a straight-line $y = mx + c$ and this is explained in the following equations. To scale down data rates (x-axis) of UEs to normalized values between 0 to 1 as illustrated in Fig. 8, the following expression is used:

$$R_{ni} = \frac{R_i - Rmin_i}{Rmax_i - Rmin_i} \tag{4}$$

$Rmax_i$ is maximum expected data rate per UE, $Rmin_i$ is minimum data rate per UE with 0 initial value, R_i is the current data rate of the UE, R_{ni} is the normalized value of data rate between 0 to 1.

Now calculating the priority is achieved by determining the mapping value of data rate on the y-axis for two successive values as illustrated in Eq. (5), where this calculation is similar or mimic the way of finding the degree of membership between [0,1] of fuzzy logic set system, which is described as follows.

$$\mu_{(Rn_{i,j})} = \frac{Rn_{i,j} - Rn_{i\,min}}{Rn_{i\,max} - Rn_{i\,min}},$$

$$\mu_{(Rn_{i,j-1})} = \frac{Rn_{i,j-1} - Rn_{i\,min}}{Rn_{i\,max} - Rn_{i\,min}} \tag{5}$$

where, $\mu_{(Rn_{i,j})}$ refers to the mapping value of the current data rate of the i^{th} UE on the y-axis, which is added to the mapping value $\mu_{(Rn_{i,j-1})}$ of the former data throughput of the same UE, $i = 1,2, \ldots, N_{UEs}$, $j = 2$.

Then, to attain an accurate decision to the priority of scheduling and to check the amount of fairness between the i^{th} UE and others, the average of two mapping points (current and last data rates) μ_{AV} of the UE can be calculated as follows:

$$\mu_{AVi} = \frac{\mu_{(Rn_{i,j})} + \mu_{(Rn_{i,j-1})}}{2} \tag{6}$$

Now the priority of scheduling i^{th} UE $Q(i)$ can be determined as follows:

$$Q(i) = arg(1 - \mu_{AVi}) \tag{7}$$

Unlike the traditional PF algorithm, in the MPF as demonstrated in above, the saving in the computational complexity is represented by reducing steps of calculating the average data rates of UE from $j = K$ to $j = 2$, where K is all the previous data rates of a particular UE. It is worth stating that the saving in the computational latency will be clearly observable when there are massive number of active connected UEs similar to the IoT scenario as will be shown in the section of results and discussion. The complexity analysis of the modified PF is as follows: In the traditional PF there are two nested loops to find the scheduling priority. The outer loop executes N times (no. of UEs) and at each time the outer loop executes, the inner loop executes K times (no. of previous data rates). As a result, the statements in the inner loop execute a total of $N\ x\ K$ times. Hence, the total complexity for the two loops is $O(N^2)$ in the traditional. Whereas with the Modified-PF the value of K is constant equals 2. Hence the total complexity is $2N$ times, therefore the total complexity is $O(N)$.

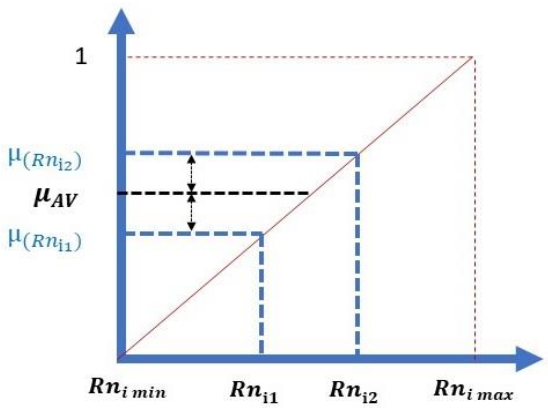

Figure. 8 Priority calculation based on the straight-line equation for the modified PF

5. The proposed spark framework for scalable parallel processing

Spark framework is proposed to keep the computing complexity of the scheduling algorithm dominated by the size of the clusters of the set of UEs instead of the entire active UEs in the mobile network. The burden of the scheduling task will be minimized via allocating a predefined cluster of UEs instead of the entire number of UEs. Application of the proposed Spark framework necessitates two parts, the first part is to group the connected UEs into clusters of pre-defined numbers before assigning them into the scheduling algorithm, which is set inside the Spark framework technology as explained and demonstrated in Algorithms (1 and 2). The second part of the proposed approach is represented by applying the parallel processing to these small groups of UEs to minimize the computational overhead that could be generated from increasing the number of UEs. Also, to avoid the potential queueing delay that can be produced from serving a large number of UEs in the traditional system before the application of the Spark framework. Algorithm 3 illustrates the pseudo-code of the operation in the Spark framework with the MPF of the scheduling algorithm. Fig. 9 depicts the structure of the proposed Spark framework with the MPF scheduling algorithm.

A prerequisite requirement to the operation of the Spark framework is to prepare groups of the input data to be processed in the form of batches, where, batch processing refers to the process of the request of sets of UEs over some time. Hence, the scheduling of a large number of UEs can be completed by handing out a sparsified set of UEs instead of the full number of UEs. Through the proposed Spark framework, the computational complexity is further minimized from $O(N)$ to $O(N/M)$, where, M is the number of UEs per cluster in the Spark framework as shown in Algorithm 1.

It is worth stating that the active connected UEs in the cell can be calculated according to [35], where the number of active UEs per cell follows the Poisson distribution and can be determined by Eq. (8).

$$M_{UEs} \sim Poisson(\varphi_M) \tag{8}$$

where, $\emptyset_M = 0.0031\, e^{1.095\, log_{10}(\beta_{cell})}$, β_{cell} is the mean of cell throughput (bits/s), which can be obtained by the summation of overall UEs throughput as follows:

$$\beta_{cell} = Sum\,(\beta_{UE}) \tag{9}$$

The throughput per UE (β_{UE}) can be found using as follows:

$$\beta_{UE} = BW\, log(1 + SNR) \tag{10}$$

where BW is the system bandwidth; SNR is the signal-to-noise ratio.

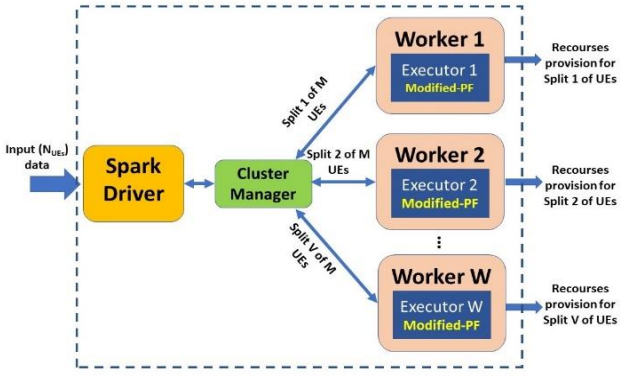

Figure. 9 The structure of the proposed Spark framework

Algorithm 1: UEs_Clustering

Input:
- UEs-Data
- Desired-Cluster-size of UEs (M)

Output:
- Set of UEs_clusters (V)
- No. of required workers (W)

1: **Function UEs_Clustering** (M, UEs)
2: { N_{UEs} = Count_Active_UEs ( )
3: UEs_Array [N_{UEs}] // an array of all active UEs
4: V [r, s] = 0 // r is no. of clusters; s= no. of UEs per cluster
5: Set G = [N_{UEs} /M]
6: F=1
7: For i=1: N_{UEs} do
8: V [f, i] = UEs_Array [N_{UEs} (i)]
9: If (i % G= =0)
10: F = F+1 // moving to next cluster
11: End if
12: End for
13: W=V } // no. of clusters equals to no. of workers
14: End function

Algorithm 2: Count_Active_UEs

Input:
- Mean the desired throughput per cell in bit/s (β_{cell})
- Poisson parameter (Φ_M)

Output: Number of active UEs

1: **Function Count_Active_UEs** ( )
2: $\beta_{UE} = BW * log(1 + SNR)$
3: $\beta_{cell} = Sum(\beta_{UE})$
4: $\phi_M = 0.0031 . e^{1.095 . \log_{10}(\beta_{cell})}$

5: $M_{UEs} \sim Poisson(\varphi_M)$ // Eq. (8)
6: Return (M_{UEs}) }
7: End function

Algorithm 3: Scalable Scheduling Using Spark Framework

Input: UEs-Data
Output:
- Resources allocation for clusters of UEs

1: Initialization
2: M = desired cluster size
3: (Wr, M) = **UEs_Clustering** (M, UEs-Data)
4: - Set up Cluster Manager for monitoring the Worker nodes
- Perform copies of the user program (the PF scheduler) for (Wr) Workers / Executors;
5: **For** worker_id = 1 to W // W: no. of workers= no. of Executors
6: Worker (worker_id). Read (data) // call for read
7: **End for**
8: result = Spark (Read, @worker, @ MPF); //Call Spark function
9: Repeat from step 1
10: End
:
11: //Call function of workers (1) to the worker (W) simultaneously for parallel processing
:
12 object worker (data)
:
13 { MPF scheduling // Eq. (7)
:
14 assign resources
15 Return (resource blocks)
:
16 }
:
17 End
:

6. Results and discussion

In this section, the performance of the proposed Spark framework and the MPF algorithm are evaluated and discussed. The simulation tests are conducted using a laptop with Intel(R) Core i7-4500U CPU / RAM 8 GB. The setting of parameters in the simulation is shown in Table 1. The channel estimation is assumed to be perfectly measured with errorless channel feedback information. Another aspect is that the architecture of 5G virtualized Fog cloud radio access networks is considered [36], which relies on the cloud computing capabilities for baseband resources provision.

Table 1. Simulation parameters

Transmission	Uplink
Transmission mode	CLSM
Network size	1 BS and variable numbers of UEs per BS
Carrier frequency	1.9 GHz
Bandwidth	1.4 MHz
UE mobility speed	1 Km/hr
Estimation	Perfect estimation
Scheduling Algorithm	Proposed MPF
Antenna's system	1x1

First of all, the amount of saving in the ST with the modified-PF is examined. The result in Fig.10 illustrates a comparison between the traditional PF with the proposed MPF. It can be noticed that there is an improvement in the scheduling time with the MPF. This is due to the reduction in the number of steps required in calculating the average data throughput per UE. It is worth stating that although the time complexity of the proposed MPF scheduling algorithm is decreased, it can be noticed that the ST of MPF is also starting to grow up with increasing the size of the network represented by the number of the UEs. Hence, scaling up the network size can diminish the gain that has been achieved with MPF. This aspect can be applied to all former research studies demonstrated in section 2, which are tried to reduce the computational complexity of the algorithm itself without considering the network scalability. This leads to the idea of using the Spark framework as a scalable and parallel processing framework.

Regarding the result of the network with the proposed Spark framework, it is unlike the conventional scheduling strategies, which accommodate the requests of entire UEs and produces high latency from the scheduling process that can finally limit the network scalability, in the Spark framework, the scheduling algorithm is copied into several workers of the Spark to serve a small set of UEs in the form of clusters in a parallel manner as shown earlier in Fig. 9. Hence, the results in Fig. 11 and Table 2 demonstrate the ST with the proposed Spark framework. It can be noticed that the ST is minimized with the proposed Spark framework. This is due to the fact that the ST is dependent now on the size of the cluster instead of the total number of UEs in the network. Hence, it can scale up the size of the network with the proposed design without the barrier of high scheduling latency, where the resources are allocated in this test by the MPF scheduler.

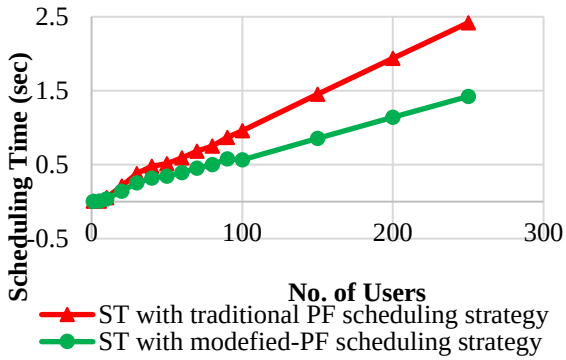

Figure. 10 comparison between the traditional PF and the modified PF in terms of ST

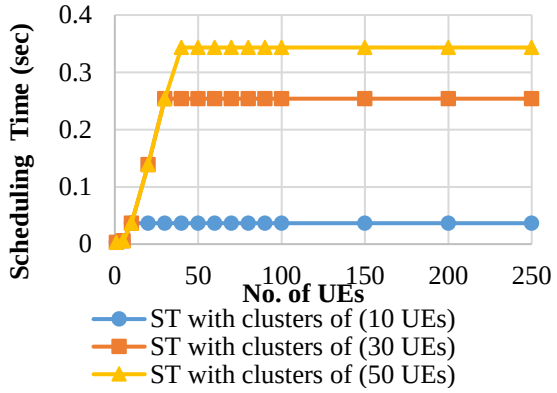

Figure. 11 Scheduling time of the MPF algorithm with the application of Spark framework for clusters of 10, 30, 50 UEs

Table 2. Comparison of ST with two levels of complexity reduction using two sizes of UEs clusters (50, 10), besides the percentage of saving in ST compared to PF

No. of UEs (250)	ST without Spark (sec) using PF 250 UEs	ST without Spark (sec) using MPF 250 UEs	ST with Spark (sec) using MPF 5 clusters of (50 UEs)	ST with Spark (sec) using MPF 25 clusters of (10 UEs)
	2.419	1.423	0.343	0.036
Percentage of saving in ST compared to PF		41%	85%	97%

The power of applying the proposed Spark framework is represented by the scalability and the parallel properties. In other words, it can schedule the requests of for instance 1000 UEs in the form of clusters of a predefined number of UEs such as 10 UEs or 30 UEs or 50 UEs, where this scenario is not applicable or possible to achieve with the

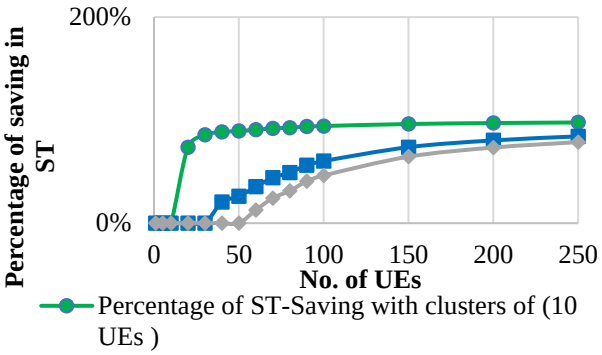

Figure. 12 Percentage of Saving in ST of the PF algorithm with the application of Spark framework for clusters of 10, 30, 50 UEs

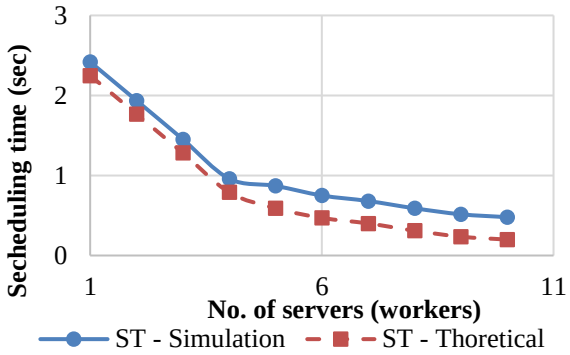

Figure. 13 The correlation between the scheduling time and no. of servers/ workers

conventional scheduling paradigm. The results in Fig.12 show the percentage of saving in the ST for the MPF scheduling algorithm for different UEs cluster sizes. Definitely, in case of increasing the cluster size, the ST also starts to increase too but now the upsurge will be controlled. Therefore, a high gain in the percentage of ST saving is observed at a smaller cluster of (10 UEs), which almost 97% saving in the ST of 250 UEs as shown in Fig.12, whereas this percentage begins to decline when increasing the cluster size. Table 2 illustrates a summary of the amount of saving in the scheduling time with the proposed approaches of this research.

Finally, the obtained result in Fig. 13 illustrates a comparison between the theoretical calculations and simulation results of the overall scheduling time. The results show that as the number of servers or workers increases the scheduling time declines proportionally.

7. Conclusions

Resource allocation via the scheduling algorithm is one of the central aspects of the dense 5G-IoT networks. The scheduling time can be considered as a critical time due to the dynamic nature of the mobile networks that lead to continuous variation in the instantaneous channel conditions. However, scaling

up the network can lead to an increase in the scheduling time accordingly. In this paper, two levels of complexity minimization are achieved in the process of scheduling. The proposed Spark framework is accelerated the scheduling process by shifting the scheduling time complexity from the size of the entire network into the processing of small predefined clusters of UEs, which minimizes the complexity from $O(N^2)$ to $O(N/M)^2$. Additionally, the proposed MPF is reduced the computational complexity of the traditional PF algorithm from $O(N/M)^2$to$O(N/M)$.Table 2 verifies the amount of reduction in the ST with the deployment of both the MPF and the Spark framework. As demonstrated in section 3.3, the PF algorithm is used due to its performance which represents a trade-off between fairness and the data rate. The results reveal that the proposed Spark framework and the MPF can provide a significant improvement in the scheduling process of the cellular network via minimizing the scheduling time that can support network scalability.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Conceptualization, Ali M. Mahmood; methodology, Ali M. Mahmood and Azad R. Kareem; software, Ali M. Mahmood; validation, Ali M. Mahmood and Azad R. Kareem; formal analysis, Ali M. Mahmood and Azad R. Kareem; investigation, Azad R. Kareem and Ali M. Mahmood; resources, Ali M. Mahmood and Azad R. Kareem; data curation, Ali M. Mahmood; writing original draft preparation, Azad R. Kareem; writing review and editing, Ali M. Mahmood and Azad R. Kareem; visualization, Ali M. Mahmood and Azad R. Kareem; supervision, Azad R. Kareem; project administration, Azad R. Kareem; funding acquisition, not funded.

References

[1] G. Liu and D. Jiang, "5G: Vision and requirements for mobile communication system towards year 2020", *Chinese Journal of Engineering*, Vol. 2016, pp. 8, 2016.

[2] K. Abbas, M. Afaq, T. Ahmed Khan, A. Rafiq, and W.-C. Song, "Slicing the Core Network and Radio Access Network Domains through Intent-Based Networking for 5G Networks", *Electronics*, Vol. 9, pp. 1710, 2020.

[3] O. Shmuel, A. Cohen, and O. Gurewitz, "Performance analysis of opportunistic distributed scheduling in multi-user systems", *IEEE Transactions on Communications*, Vol. 66, pp. 4637-4652, 2018.

[4] H. Zheng and L. Li, "Joint backhaul bandwidth and power allocation in heterogeneous cellular networks: A two-level approach", *International Journal of Communication Systems*, pp. e4420, 2020.

[5] H. Santos, P. Pinho, and H. Salgado, "Patch Antenna-in-Package for 5G Communications with Dual Polarization and High Isolation", *Electronics*, Vol. 9, pp. 1223, 2020.

[6] Y. Liu, M. Peng, G. Shou, Y. Chen, and S. Chen, "Toward Edge Intelligence: Multiaccess Edge Computing for 5G and Internet of Things", *IEEE Internet of Things Journal*, Vol. 7, pp. 6722-6747, 2020.

[7] P. Jonsson, S. Carson, S. Davis, G. Blennerud, P. Lindberg, K. Öhman, J. Travers, F. Pedersen, P. Linder, J. Sethi, P. Rinderud, J. Rubio, J. Garcia, H. Hemmer, C. Ashraf, and A. Powell, "Ericsson Mobility Report", *Ericsson: Stockholm*, Sweden, 2020.

[8] A. Mukhopadhyay, G. Das, and V. Reddy, "A fair uplink scheduling algorithm to achieve higher MAC layer throughput in LTE", In: *Proc. of 2015 IEEE International Conference on Communications (ICC)*, pp. 3119-3124, 2015.

[9] N. M. Elshennawy, "Modified Proportional Fair Scheduling Algorithm for Heterogeneous LTE-A Networks", *International Journal of Interactive Mobile Technologies*, Vol. 14, 2020.

[10] F. Ferreira, and P. Guardieiro, "A New Channel-Aware Downlink Scheduling Algorithm for LTE-A and 5G HetNets", In: *Proc. of International Conference on Computing Science, Communication and Security*, Singapore, pp. 173-183, 2020.

[11] Z. Sha, Z. Wang, S. Chen, and L. Hanzo, "Graph Theory Based Beam Scheduling for Inter-Cell Interference Avoidance in MmWave Cellular Networks", *IEEE Transactions on Vehicular Technology*, Vol. 69, pp. 3929-3942, 2020.

[12] M. A. Ibraheem, N. ElShennawy, and A. M. Sarhan, "A Proposed Modified Proportional Fairness Scheduling (MPF-BCQI) Algorithm with Best CQI Consideration for LTE-A Networks", In: *Proc. of 13th International Conference on Computer Engineering and Systems (ICCES)*, Cairo, Egypt, pp. 360-368, 2018.

[13] F. Zabini, A. Bazzi, B. Masini, and R. Verdone, "Optimal performance versus fairness tradeoff for resource allocation in wireless systems", *IEEE Transactions on Wireless Communications*, Vol. 16, pp. 2587-2600, 2017.

[14] S. Schwarz, C. Mehlführer, and M. Rupp, "Low complexity approximate maximum throughput scheduling for LTE", In: *Proc. of Conference Record of the Forty Fourth Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, USA, pp. 1563-1569, 2010.

[15] D. Yuan, H.-Y. Lin, J. Widmer, and M. Hollick, "Optimal joint routing and scheduling in millimeter-wave cellular networks", In: *Proc. of IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, Honolulu, HI, USA, pp. 1205-1213, 2018.

[16] H. Alsheyab, S. Choudhury, E. Bedeer, and S. Ikki, "Near-Optimal Resource Allocation Algorithms for 5G+ Cellular Networks", *IEEE Transactions on Vehicular Technology*, Vol. 68, pp. 6578-6592, 2019.

[17] M. Naeem, S. Bashir, Z. Ullah, and A. Syed, "A Near Optimal Scheduling Algorithm for Efficient Radio Resource Management in Multi-user MIMO Systems", *Wireless Personal Communications*, Vol. 106, pp. 1411-1427, 2019.

[18] Y. Yang, B. Bai, W. Chen, and L. Hanzo, "A low-complexity cross-layer algorithm for coordinated downlink scheduling and robust beamforming under a limited feedback constraint", *IEEE transactions on vehicular technology*, Vol. 63, pp. 107-118, 2013.

[19] A. Karimi, K. Pedersen, N. Mahmood, G. Pocovi, and P. Mogensen, "Efficient low complexity packet scheduling algorithm for mixed URLLC and eMBB traffic in 5G", In: *Proc. of IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*, Kuala Lumpur, Malaysia, pp. 1-6, 2019.

[20] M. Mohseni, S. Banani, A. Eckford, and R. Adve, "Scheduling for VoLTE: Resource allocation optimization and low-complexity algorithms", *IEEE Transactions on Wireless Communications*, Vol. 18, pp. 1534-1547, 2019.

[21] H. Yang, A. Arafa, T. Quek, and H. Poor, "Age-based scheduling policy for federated learning in mobile edge networks", In: *Proc. of ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Barcelona, Spain, pp. 8743-8747, 2020.

[22] D. Wu, Z. Zhang, S. Wu, J. Yang, and R. Wang, "Biologically inspired resource allocation for network slices in 5G-enabled Internet of Things", *IEEE Internet of Things Journal*, Vol. 6, pp. 9266-9279, 2018.

[23] B. Zhang, X. Mao, J.-L. Yu, and Z. Han, "Resource allocation for 5G heterogeneous cloud radio access networks with D2D communication: A matching and coalition approach", *IEEE Transactions on Vehicular Technology*, Vol. 67, pp. 5883-5894, 2018.

[24] H. Khan, M. Ali, M. Naeem, I. Rashid, A. Siddiqui, M. Imran, and S. Mumtaz, "Resource allocation and throughput maximization in decoupled 5G", *In: Proc. of IEEE Wireless Communications and Networking Conference (WCNC)*, Seoul, Korea (South), pp. 1-6, 2020.

[25] A. Mukhopadhyay, and G. Das, "Low Complexity Fair Scheduling in LTE/LTE-A Uplink Involving Multiple Traffic Classes", *IEEE Systems Journal*, pp. 1-12, 2020.

[26] B. Palit and S. Das, "Performance evaluation of mixed traffic schedulers in OFDMA networks", *Wireless Personal Communications*, Vol. 83, pp. 895-924, 2015.

[27] A. Pokhariyal, K. Pedersen, G. Monghal, I. Kovacs, C. Rosa, T. Kolding, and P. Mogensen, "HARQ aware frequency domain packet scheduler with different degrees of fairness for the UTRAN long term evolution", *In: Proc. of 2007 IEEE 65th Vehicular Technology Conference-VTC2007-Spring*, Dublin, Ireland, pp. 2761-2765, 2007.

[28] S. Lee, J. Kim, J. Park, and S. Cho, "Grant-Free Resource Allocation for NOMA V2X Uplink Systems Using a Genetic Algorithm Approach", *Electronics*, Vol. 9, p. 1111, 2020.

[29] D. Mannani, "Modeling and simulation of scheduling algorithms in LTE networks", *Bachelor of Science Thesis*, The Institute of Telecommunications, Faculty of Electronics and Information Technology, University of Technology, Warsaw, 2012.

[30] A. Marinčić and D. Šimunić, "Performance evaluation of different scheduling algorithms in LTE systems", In: *Proc. of 39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, Opatija, Croatia, pp. 595-600, 2016.

[31] E. M. A. M. Osman, "Performance Evaluation of Downlink Scheduling Algorithms in Long Term Evaluation (LTE) Network", Masters Dissertations, Sudan University of Science and Technology, 2016.

[32] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauly, M. Franklin, S. Shenker, and I. Stoica, "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing", In: *Proc. of 9th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 12*), USA, pp. 15-28, 2012.

[33] R. Myung, and H. Yu, "Performance prediction for convolutional neural network on spark cluster", *Electronics*, Vol. 9, pp. 1340, 2020.
[34] N. Vaidya, "Apache Spark Architecture – Spark Cluster Architecture Explained", Spark Training, Accessed 14 September 2020.
[35] M. Laner, P. Svoboda, S. Schwarz, and M. Rupp, "Users in cells: A data traffic analysis", In: *Proc. of IEEE Wireless Communications and Networking Conference (WCNC)*, Paris, France, pp. 3063-3068, 2012.
[36] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things", In: *Proc. of the first edition of the MCC workshop on Mobile cloud computing*, Helsinki Finland, pp. 13-16, 2012.