Simulating LTE Cellular Systems: An Open-Source Framework

Giuseppe Piro, *Student Member, IEEE*, Luigi Alfredo Grieco, *Member, IEEE*, Gennaro Boggia, *Senior Member, IEEE*, Francesco Capozzi, *Student Member, IEEE*, and Pietro Camarda

***Abstract*—Long-term evolution (LTE) represents an emerging and promising technology for providing broadband ubiquitous Internet access. For this reason, several research groups are trying to optimize its performance. Unfortunately, at present, to the best of our knowledge, no open-source simulation platforms, which the scientific community can use to evaluate the performance of the entire LTE system, are freely available. The lack of a common reference simulator does not help the work of researchers and poses limitations on the comparison of results claimed by different research groups. To bridge this gap, herein, the open-source framework *LTE-Sim* is presented to provide a complete performance verification of LTE networks. *LTE-Sim* has been conceived to simulate uplink and downlink scheduling strategies in multicell/multiuser environments, taking into account user mobility, radio resource optimization, frequency reuse techniques, the adaptive modulation and coding module, and other aspects that are very relevant to the industrial and scientific communities. The effectiveness of the proposed simulator has been tested and verified considering 1) the software scalability test, which analyzes both memory and simulation time requirements; and 2) the performance evaluation of a realistic LTE network providing a comparison among well-known scheduling strategies.**

***Index Terms*—Long-term evolution (LTE), modeling, performance evaluation, simulation.**

List of Acronyms

AMC	Adaptive modulation and coding.
CQI	Channel quality indicator.
eNB	Evolved node B.
EPS	Evolved packet system.
E-UTRAN	Evolved universal terrestrial radio access.
EXP	Exponential proportional fairness.
LTE	Long-term evolution.
MCS	Modulation and coding scheme.
M-LWDF	Modified largest weighted delay first.
MME/GW	Mobility management entity/gateway.
OFDM	Orthogonal frequency-division multiplexing.
OFDMA	OFDM access.
PDBC	Physical broadcast channel.
PDCCH	Physical downlink control channel.
PDCP	Packet data control protocol.
PF	Proportional fair.
RB	Resource block.
RLC	Radio link control.
RRC	Radio resource control.
SC-FDMA	Single-carrier frequency-division multiple access.
SINR	Signal-to-interference-and-noise ratio
TTI	Transmission time interval.
UE	User equipment.

Manuscript received May 31, 2010; revised September 23, 2010; accepted October 25, 2010. Date of publication November 11, 2010; date of current version February 18, 2011. The review of this paper was coordinated by Dr. C.-C. Chong.

The authors are with the Dipartimento di Elettrotecnica ed Elettronica, Politecnico di Bari, 70125 Bari, Italy (e-mail: g.piro@poliba.it; a.grieco@poliba.it; g.boggia@poliba.it; f.capozzi@poliba.it; camarda@poliba.it).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2010.2091660

I. Introduction

To FACE the ever-growing demand for packet-based mobile broadband systems, the Third-Generation Partnership Project (3GPP) [1] has introduced LTE specifications [2] as the next step of the current 3.5G cellular networks. An enhanced access network (i.e., the E-UTRAN, evolved-UMTS terrestrial radio access network) and an evolved core network have been defined [3]. At present, more than 20 cellular operators worldwide, representing together more than 1.8 billion of the total 3.5 billion mobile subscribers in the world, have already stated a commitment to LTE, and more than 32 million LTE subscribers are foreseen by 2013 [4].

Starting from this premise, it is clear that the optimization of all LTE aspects is a topic worth investigating for both the industrial and academic communities. It is important to remark that, at the present time, a complete system-level simulator is not available for these communities. In fact, the most important vendors of mobile communication equipment have implemented their own simulators. Moreover, other simulators [5]–[8], which were developed in academia–industrial cooperations, can be purchased using a commercial license, and their source codes are not publicly available. In [9], a Matlab-based LTE simulator has been proposed, implementing a standard-compliant LTE downlink physical (PHY) layer with AMC, multiple users, multiple-input multiple-output transmission, and a scheduler. Unfortunately, although it is open source and freely available, it does not consider relevant aspects of LTE simulation, such as realistic applications, a complete LTE protocol stack, and multicell environments with uplink flows. In [10], a system-level simulator for LTE networks has been proposed as a supplement of the previous one to support cell planning, scheduling, and interference. However, it does not support a complete LTE protocol stack, uplink flows, and bearer management.

Since no open-source simulation platforms are freely available for the community, the design of innovative optimization strategies is today seriously impaired. Moreover, the lack of a common reference simulator poses serious problems in the comparison of results presented by different research groups.

To bridge this gap, herein, we present an open-source framework to simulate LTE networks, namely, *LTE-Sim*, which is able to provide a complete performance verification of LTE systems.

LTE-Sim encompasses several aspects of LTE networks, including both the E-UTRAN and the EPS. In particular, it supports single-cell and multicell environments, quality-of-service (QoS) management, multiuser environments, user mobility, and handover procedures. Three kinds of network nodes are modeled: UE, eNB, and MME/GW. Several traffic generators at the application layer have been implemented, and the management of data radio bearer is supported. Finally, well-known scheduling strategies (such as PF, M-LWDF, and EXP [11]), the AMC scheme, CQI feedback, frequency reuse techniques, and models for the PHY layer have been developed.

It is important to note that features covered by *LTE-Sim* will allow both researchers and practitioners to test enhanced techniques for improving fourth-generation (4G) cellular networks, such as new PHY functionalities, innovative network protocols and architectures, and high-performance scheduling strategies. *LTE-Sim* is freely available under the GPLv3 license [12]. We believe that the high modularity of *LTE-Sim* will allow a convergence of efforts toward improved versions of the software enriched with increasingly more features. Such enhancements could cover new versions of the standard, innovative resource management techniques, advanced handover procedures, protocol architectures, and so on.

The rest of this paper is organized as follows. Section II describes the software design, highlighting the most important simulator components. Sections III and IV describe the flows and the radio resource management, respectively. Section V provides a performance evaluation of the proposed simulator. Finally, Section VI draws our conclusions.

II. Software Design

To ensure modularity, polymorphism, flexibility, and high performance, *LTE-Sim* has been written in C++, using the object-oriented paradigm, as an event-driven simulator. At present, the software is composed by 90 classes, 220 files, and approximately 23 000 lines of code. Fig. 1 shows the unified modeling language diagram of the most important classes implemented, highlighting their most important methods and variables.

There are four main components:

1) the *Simulator*;
2) the *NetworkManager*;
3) the *FlowsManager*;
4) the *FrameManager*.

For each of them, a dedicated class has been developed. When a simulation starts, only one object for each of the aforementioned components is created. Furthermore, to ensure that each of these classes will have only one instance during the simulation (with a global point of access), a singleton design pattern has been used [13].

The most important functionalities of main components of the *LTE-Sim* are reported in Table I.

For simplicity, from this moment on, we will use the notation *Class::Function()* to indicate a *Function ()* defined into the *Class*.

A simulation scenario is composed of several objects, modeling the main elements of an LTE system. Each of them can issue, if needed, a new event using a *Simulator::Schedule()* method to enable a realistic interaction among nodes.

The *Calendar* sorts events in a chronological order, according to their timestamps. Event scheduling is handled by the *Simulator* class. In detail, at the beginning of each simulation, the *Calendar* is populated by only three events: 1) the start of the simulation, using the *Simulator::Run()* method; 2) the start of the *FrameManager*, using the *FrameManager::StartFrame()* method; and 3) the end of the simulation, using the *Simulator::Stop()* method. Then, the calendar will be populated by other events generated by LTE system elements that constitute the simulated scenario, e.g., *Application::CreatePacket()*, *NetworkNode::SendPacketBurst()*, and *ENodeB::ResourceAllocation()*.

Three kinds of LTE network nodes have been implemented: UE, eNB, and MME/GW. They are created, destroyed, and handled by the *NetworkManager*. Each LTE network node can be a source or a destination of data flows, which is defined by the classical five-tuple: source and destination IP addresses, sender and receiver ports, and the transport protocol type.

LTE-Sim provides a support for radio resource allocation in a time–frequency domain. According to [14], in the time domain, radio resources are distributed every TTI, each one lasting 1 ms. Furthermore, each TTI is composed by two time slots of 0.5 ms, corresponding to 14 OFDM symbols in the default configuration with short cyclic prefix; ten consecutive TTIs form the LTE frame.

In the frequency domain, instead, the whole bandwidth is divided into 180-kHz subchannels, corresponding to 12 consecutive and equally spaced subcarriers. As the subchannel dimension is fixed, for different system bandwidth configurations, the number of subchannels varies accordingly.

A time/frequency radio resource spanning over one 0.5 ms time slot in the time domain and over one subchannel in the frequency domain is called RB and corresponds to the smallest radio resource that can be assigned to a UE for data transmission.

The present implementation of the LTE frame structure is guaranteed by the *FrameManager* component. It is in charge of the correct scheduling of frames and subframes (i.e., TTIs) and of the synchronization of all the eNBs.

PHY-layer aspects are managed for both UEs and eNBs. In particular, a PHY object storing PHY parameters and the radio channel model (as proposed in [15]) is connected to each device. Here, information such as the channel quality and the perceived interference level is saved. Further PHY information (e.g., frequency carrier, available bandwidth, list of available RBs for both downlink and uplink, and frequency reuse parameters) are stored in a bandwidth manager object.

Finally, four different traffic generators running at the application layer (trace based, ON–OFF, constant bit rate, and infinite buffer) have been developed.

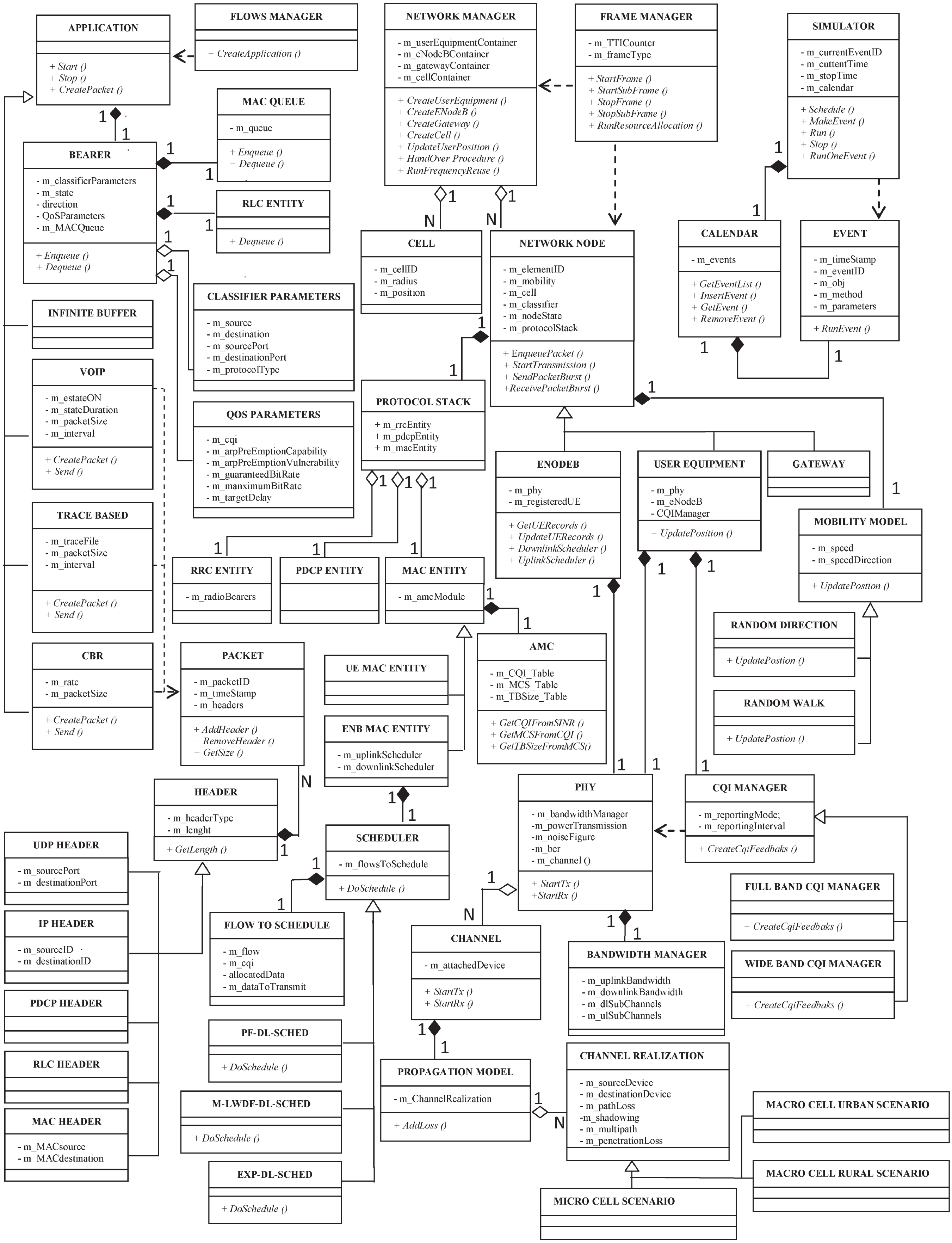


Fig. 1. *LTE-Sim*. Class diagram.

TABLE I
MAIN COMPONENTS OF THE *LTE-Sim*

Component	Functionalities	Important Methods	Method description
Simulator	- Creates/Handles/Ends an event	Schedule()	Creates a new event and insert it into the calendar.
		RunOneEvent()	Executes an event.
		Run() / Stop()	Starts / ends the simulation.
FrameManager	- Defines LTE frame structure - Schedules frames and sub-frames	StartFrame() and StopFrame()	Handles the start and the end of the LTE frame.
		StartSubFrame() and StopSubFrame()	Handles the start and the end of the LTE sub-frame.
FlowsManager	- Handles applications	CreateApplication()	Creates an application
NetworkManager	- Creates devices - Handles UE position - Manages the hand over - Implements frequency reuse techniques	CreateUserEquipment()	Creates an UE device
		CreateCell()	Creates an LTE Cell
		UpdateUserPosition()	Updates the UE position
		HandOverProcedure()	Handles the hand over procedure
		RunFrequecyReuse()	Implements frequency reuse techniques

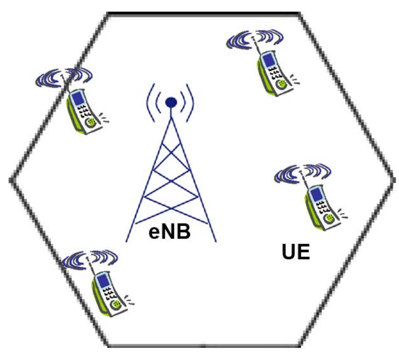


Fig. 2. Single-cell/multiuser simulation environment.

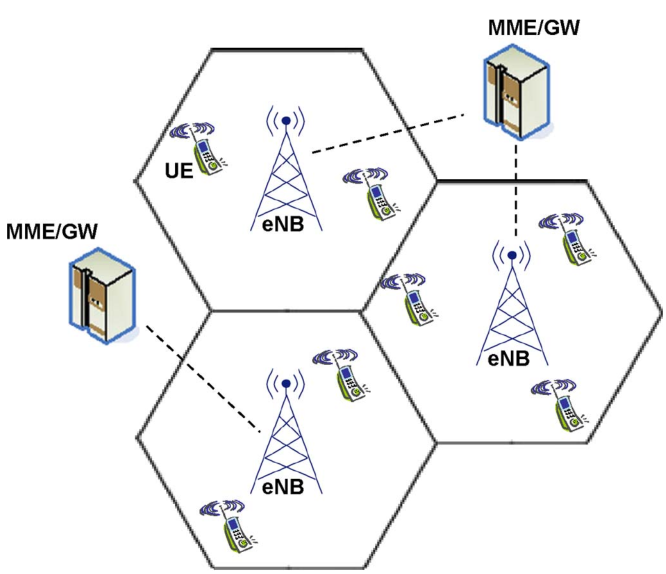


Fig. 3. Multicell/multiuser simulation environment.

With *LTE-Sim*, an LTE scenario can be created as a static function in a C++ header file. A reference to this function should be added into the main program (see Section V-A for details about how to create a scenario).

A. Basic Network Topology

Both single-cell and multicell simulations can be run (see Figs. 2 and 3).

The network topology is composed of a set of cells and network nodes (including eNBs, one or more MME/GW, and UEs), which are distributed among cells. All the methods for the creation and the management of the network topology are provided by the *NetworkManager* component (see Fig. 1 for details).

An LTE cell, which is implemented by the *Cell* class, is identified by a unique identifier (ID). Its attributes are the radius and the position defined in a Cartesian system.

For each kind of LTE network node, a dedicated class has been developed, extending the basic *NetworkNode* class (i.e., *ENodeB*, *UserEquipment*, and *MME-GW* classes). Each network node is identified by a unique ID, and its position in a Cartesian system is also defined.

Support for several functionalities of both user- and control-plane protocol stacks is provided by the *ProtocolStack* class, which is developed as a container of RRC, PDCP, and media access control (MAC) entities.

The eNB performs radio resource management for the evolved radio access. Both downlink and uplink scheduling strategies are defined into its MAC entity. In particular, downlink and uplink schedulers are defined into the *m_downlikScheduler* and *m_uplinkScheduler* variables, respectively. In Section IV, scheduling strategies that *LTE-Sim* provides are described.

As previously mentioned, for eNB and UE devices, an instance of a PHY object, namely, *m_phy*, has been defined. PHY objects, moreover, are attached to an LTE channel, which is modeled by the *Channel* class. This class manages the transmission through the PHY channel of each packet among the attached PHY entities, also implementing a propagation loss model. The PHY object has been developed for providing an interface between the LTE device and the channel, for storing and managing information about the radio channel (such as the bandwidth and list of available subchannels for both downlink and uplink), and for offering an access to the radio channel to simulate the packet transmission and reception. Further details about the implemented channel and PHY models are provided in Section IV.

CQI reporting is another important feature performed by the UE. In particular, *LTE-Sim* also supports the CQI reporting feature [16], i.e., the UE estimates the channel quality and converts it into a set of CQI feedbacks reported to the eNB.

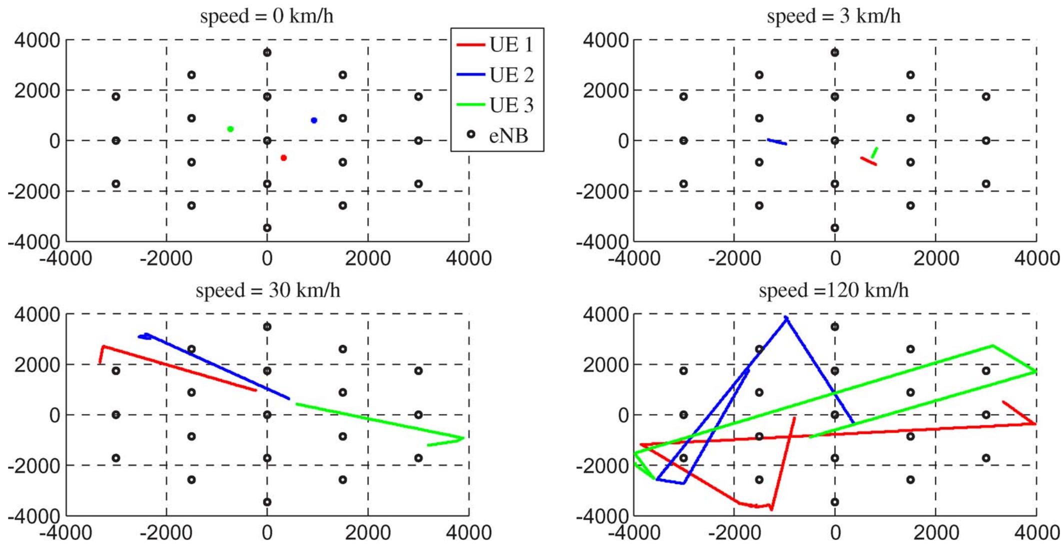


Fig. 4. Traveling pattern of UEs with the random direction mobility model.

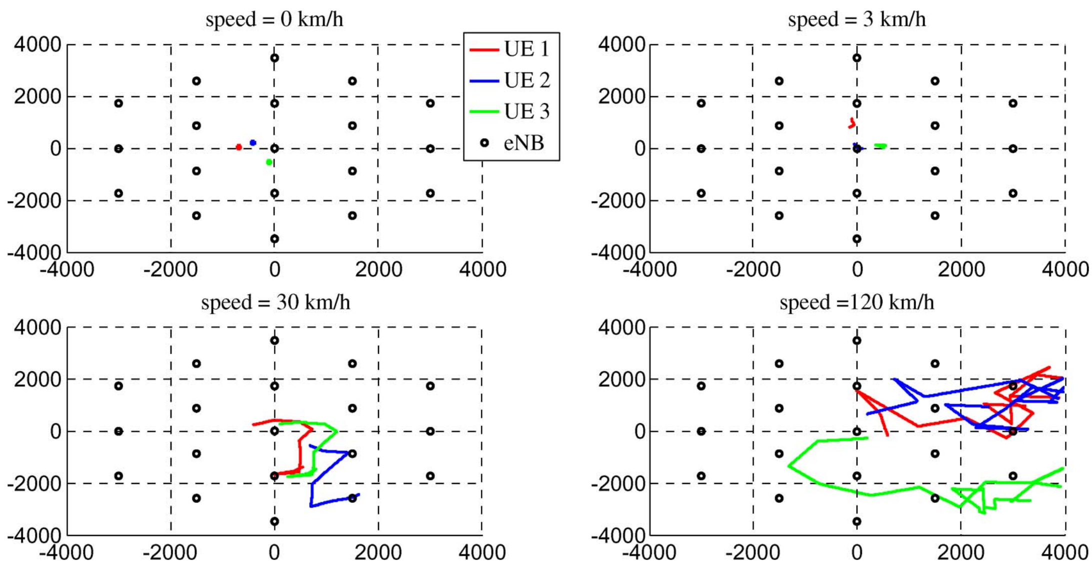


Fig. 5. Traveling pattern of UEs with the random walk mobility model.

Our simulator supports both periodical and aperiodical CQI reporting and both full-band and wide-band reporting modes. During the simulation, each eNB maintains the list of UEs associated with it, storing, for each of them, the ID and the latest CQI feedbacks. Furthermore, eNBs and UEs are aware of the LTE cell they belong to. In fact, each UE keeps up to date the ID of the cell to which it belongs to and the ID of the serving eNB.

B. Users Mobility

To support user mobility, a system-level intercell handover procedure has been implemented. Two types of mobility models are supported: random direction and random walk [17]. For each of them, a dedicated class has been developed, extending the basic *MobilityModel* class, i.e., the *RandomDirection* and *RandomWalk* classes. In the *MobilityModel* class, the *m_speed* and *m_speedDirection* variables are used to define the speed and the travel direction of the user, respectively. For each UE, an *m_mobility* element has been created to manage mobility. According to [15], user speed should be chosen among the values 0, 3, 30, and 120 km/h, which are equivalent to static, pedestrian, and vehicular scenarios, respectively.

When the random direction model is used, the UE randomly chooses the speed direction, which remains constant during the time, and moves toward the simulation boundary area. Once the simulation boundary area is reached, the UE chooses a new speed direction.

When the random walk is used, the UE randomly chooses the speed direction and moves accordingly for a given travel distance that depends on the user speed. The UE changes its speed direction after covering this distance or, as in the previous model, once the simulation boundary area is reached. As default, the travel distance is equal to 200, 400, and 1000 m when the user speed is equal to 3, 30, and 120 km/h, respectively.

Figs. 4 and 5 show an example of a traveling pattern in a network topology composed by 19 cells with a radius equal to 1 km, using both the random direction and random walk mobility models. It is important to note that more sophisticated mobility models can be easily defined by extending the *MobilityModel* class.

The user mobility is managed by the *NetworkManager* that every TTI updates the user position according to the selected mobility model and parameters and verifies, through the *NetworkManager::HandOverProcedure()* function, if the handover procedure is necessary. In *LTE-Sim*, both cell reselection and hard handover procedures are implemented. Moreover, handover decisions are carried out by the *HandOverManager*, which is defined for each UE. Hard handover management consists of the following steps.

1) For the UE that triggered the handover procedure, the function *HandOverManager::SelectTargetENodeB()* is used to select a new target eNB. Among all eNBs, the one closest to the moving UE is chosen. However, more sophisticated policies can be defined for selecting a new eNB by extending the *NetworkManager* and *HandOverManager* classes.
2) All the information about the UE is transferred from the old eNB to the new one.
3) Between the UE and the new target eNB, a new radio bearer is created.
4) The UE updates the list of available subchannels for downlink and uplink, according to those assigned to the new target eNB.

During the handover, the UE switches to a detached state for a given time interval so that no flows directed to and coming from the UE can be scheduled; such a time interval is a simulator parameter and can be modified (the default value is 30 ms).

III. Flow Management and Long-Term Evolution Protocol Stacks

In LTE specifications, the EPS bearer has been introduced to provide QoS differentiation. It maps a flow into a logical channel established between the UE and the GW. Moreover, a radio bearer is associated to each EPS bearer as a logical channel between the UE and the eNB [18].

Both EPS and radio bearers can be classified as default or dedicated. The former is created when the UE is associated to the network and remains established during the whole lifetime of the connection, providing basic connectivity and exchange of control messages. The latter, instead, is established to support a new specific service, e.g., call, video call, and web browsing. Orthogonally, bearers can also be classified as guaranteed bit rate (GBR) or non-GBR, according to the QoS requirements of the flow they map. In particular, the default bearer is a non-GBR; a dedicated bearer can be GBR or non-GBR [19]. In the current version of the software, only dedicated radio bearers have been developed.

A. QoS and Radio Bearer

The *Bearer* class models the dedicated radio bearer. When a downlink (uplink) flow starts, it activates a dedicated radio bearer between the eNB and the UE (the UE and the eNB) and *vice versa*. Moreover, for each UE and eNB, it is possible to activate more than one bearer.

The *QoSParameters* object provides, for each *Bearer*, the definition of flow QoS requirements [20], such as the QoS class identifier, the allocation and retention priority (composed of the preemption flow capability and the preemption flow vulnerability), the GBR, and the maximum bit rate.

An IP-based packet classifier is used to map packets coming from the IP layer to a given radio bearer, according to the parameters stored in the *ClassifierParameters* class. The packet classification is done using a packet filter based on the classical five-tuple: source and destination IPs, sender and receiver ports, and protocol type. This functionality has been implemented in the *Classifier* class, which is defined for all the network elements.

B. Application Layer

Packets transported by a dedicated radio bearer are generated at the application layer by four different traffic generators: trace based, voice over IP (VoIP), constant bit rate (CBR), and infinite buffer. The *trace-based* application sends packets based on realistic video trace files, which are available in [21].

The *VoIP* application generates G.729 voice flows. In particular, the voice flow has been modeled with an ON/OFF Markov chain, where the ON period is exponentially distributed with a mean value of 3 s, and the OFF period has a truncated exponential probability density function with an upper limit of 6.9 s and an average value of 3 s [22]. During the ON period, the source sends 20 bytes sized packets every 20 ms (i.e., the source data rate is 8 kb/s), while during the OFF period, the rate is zero because the presence of a voice activity detector is assumed. It is important to note that these parameters can be modified by defining their values into the constructor of the *VoIP* class or using the methods we provided to set them.

The *CBR* application generates packets with a CBR. In particular, the packet size and the interarrival packet time can be defined for this kind of traffic.

Finally, the *infinite-buffer* application models an ideal greedy source that always has packets to send.

For each of these applications, a dedicated class has been developed, namely, *InfiniteBuffer*, *VoIP*, *CBR*, and *TraceBased*. It is important to note that these classes are extended from the *Application* class that provides methods and parameters common to all of them, such as starting/stopping time instants.

Each application uses the *Application::Send()* method to generate a packet. Then, the application delivers it to the network device, calling the function *NetworkNode::EnqueuePacket()*. When a network device receives a packet from the application, it forwards the packet through the user-plane protocol stack (see the next section for details) to add protocol headers. Then, the packet is enqueued at the MAC layer and associated to a particular bearer, using the *Classifier* functionalities. Now, the packet can be sent on the channel and received from another network device, according to the scheduling decisions. When a network device receives packets from the channel, it forwards them to the upper layer through

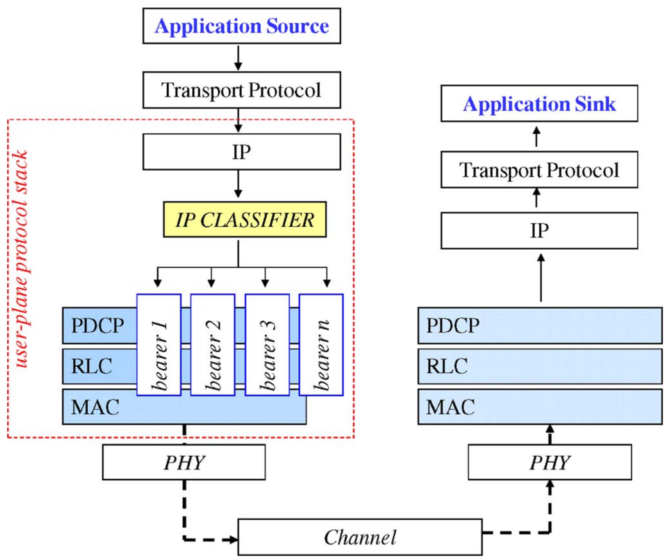


Fig. 6. Implemented protocol stack.

the same user-plane protocol stack; then, it delivers them to the proper application sink.

C. Protocol Stack

LTE-Sim implements several functionalities of both user-plane and control-plane LTE protocol stacks [23]. To this aim, the *ProtocolStack* class has been created as a container of RRC, PDCP, and MAC entities. An instance of this class, namely, *m_protocolStack*, has been defined for each device. Furthermore, as proposed in [24], an RLC entity has been created for each dedicated radio bearer.

The RRC entity manages downlink and uplink dedicated radio bearers for a given device. It interacts with the classifier to classify a packet into a proper radio bearer.

The PDCP entity provides the header compression of packets coming from the upper layer that will be enqueued into a proper MAC queue.

The RLC entity models the unacknowledged data transmission at the RLC layer. We have chosen to implement the unacknowledged mode for the RLC because it represents the most used data transmission mode, particularly by delay-sensitive and error-tolerant applications (such as VoIP and video streaming). The most important functionalities we have defined for the RLC layer are the segmentation and the concatenation of service data units.

The MAC entity provides, for both UE and eNB devices, an interface between the device and the PHY layer designed to delivery packets coming from the upper layer to the PHY one and *vice versa*. Moreover, in this class, the AMC module is also defined. *EnbMACEntity* class adds further functionalities such as uplink and downlink packet schedulers to the eNB. An example of the interaction among the entities included in the implemented protocol stack is illustrated in Fig. 6; it shows the packet path starting from the application layer and following the user-plane protocol stack.

The LTE packet is modeled by the *Packet* class. Three important variables are defined for the packet object: *m_timeStamp*, *m_size*, and *m_packetHeaders*. The *m_timeStamp* variable represents the instant of packet generation at the application layer. This value is used to compute one-way packet delay (it could also be used for statistical purposes and by scheduling strategies). The *m_size* and *m_packetHeaders* variables represent the packet size and the list of protocol headers that have been added to the packet, respectively. When the packet is created, *m_size* is equal to the amount of data generated by the application layer. As soon as a new header protocol is added to the packet, *m_size* is updated according to its size.

In the current release of *LTE-Sim*, only the user datagram protocol (UDP) has been developed at the transport layer. However, it is possible to implement other transport protocols (i.e., the transmission control protocol), extending the *TransportLayer* class.

During the simulation, the application layer creates packets that are passed to the UDP and the IP. As we have explained in Section III-A, an IP-based packet classifier is used to map IP datagrams to radio bearers.

Each bearer maintains its own first-in first-out transmission queue, using the *MACQueue* class. When an IP packet is enqueued, a PDCP header is added. Since the PDCP protocol provides a header compression functionality using the robust header compression protocol [25], the packet size is updated to compress real-time transport protocol/UDP/IP headers to 3 bytes [26].

Finally, when the packet is dequeued, a cyclic redundancy check (CRC) trailer and RLC and MAC headers are added.

IV. Channel Structure and Resource Management

The LTE radio access is based on OFDM and provides a highly flexible bandwidth (from 1.4 to 20 MHz). Both frequency-division duplex (FDD) and time-division duplex (TDD) multiple-access techniques are supported [14].

Radio resources are distributed among users in a time–frequency domain. The eNB schedules radio resources among uplink/downlink flows at the beginning of each subframe.

LTE-Sim supports all six channel bandwidths (i.e., 1.4, 3, 5, 10, 15, and 20 MHz) available for the LTE system [27] and the cellular frequency reuse. Finally, TDD and FDD are handled by the *FrameManager*.

A. Bandwidth Manager

All devices should know the operative bandwidth and available subchannels for both uplink and downlink. Thus, a dedicated class, i.e., the *BandwidthManager*, has been developed to store this information. An instance of the *BandwidthManager* class is defined for each PHY object, and instances of devices belonging to the same cell store the same information.

B. Frequency Reuse

A fundamental implemented feature is the frequency reuse concept [28]. As it is well known, the frequency reuse increases both the coverage and capacity of the cellular network and reduces the intercell interference. The idea is that each cell does

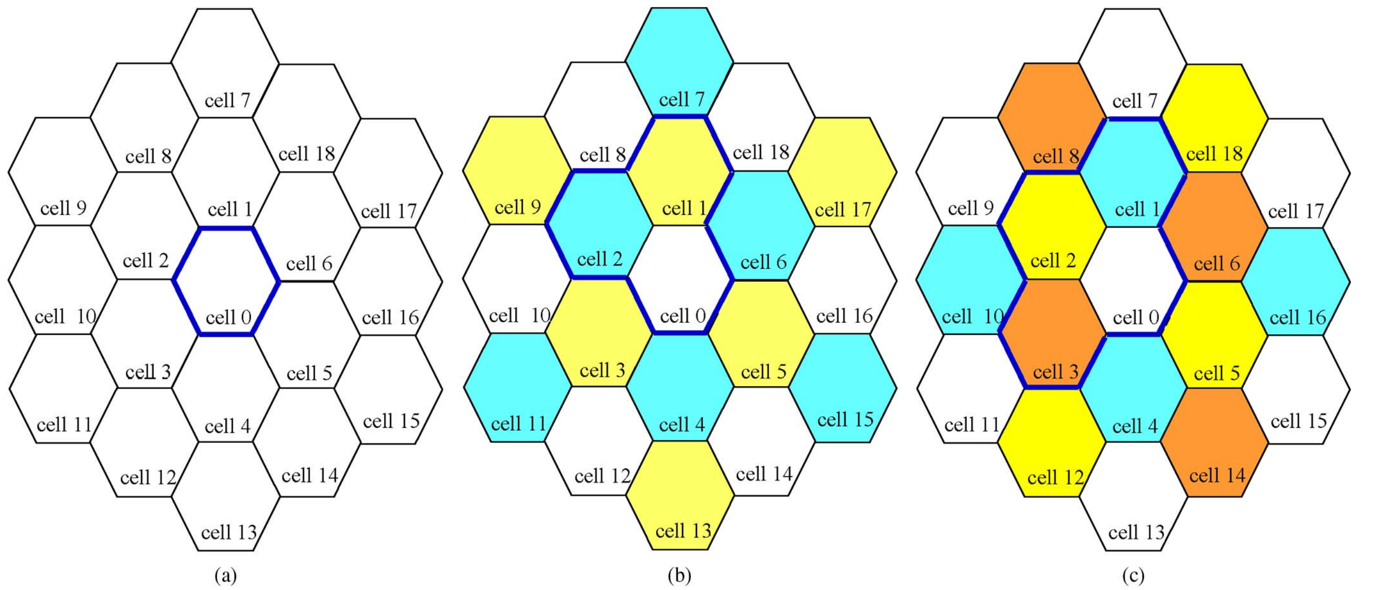


Fig. 7. Implemented frequency reuse techniques with cluster of (a) one, (b) three, and (c) four cells.

not use all the available subchannels but only a subset, in a way that adjacent cells utilize different subsets of subchannels. A group of cells, using together the complete set of available subchannels, forms a *cluster* that is regularly replicated to cover the whole service area.

The *NetworkManager::RunFrequencyReuse()* function has been developed to apply frequency reuse techniques and distribute the available bandwidth among cells. As described in [29], the number of cells that form the cluster depends on the E-UTRAN operative band and the downlink (uplink) bandwidth.

At present, *LTE-Sim* supports all possible cluster configurations available for the first E-UTRAN operative band (i.e., [1929–1980] MHz for the uplink and [2110–2170] MHz for the downlink, in FDD mode) [27].

The *NetworkManager::RunFrequencyReuse()* function receives the following parameters as input: 1) the number of cells; 2) the cluster size; and 3) the downlink (uplink) bandwidth. It distributes the whole available bandwidth among clusters so that all cells belonging to the same cluster have no overlapping channels. After applying the frequency reuse technique, it returns a list of *BandwidthManager* objects that should be assigned to devices belonging to each cell. Fig. 7 shows examples of the result of the implemented frequency reuse technique.

It is important to note that, even if currently the list of subchannels assigned to a cell remains constant during the simulation, it is possible to implement more sophisticated frequency reuse algorithms, extending the *NetworkManager* class.

When the handover procedure occurs, the UE updates the list of available subchannels for downlink and uplink, according to those assigned to the new target eNB.

C. Frame Structure

LTE-Sim supports two frame structure types proposed in [14] for the E-UTRAN. The first one is defined for FDD mode, and it is called *frame structure type 1*. The second one is called *frame structure type 2* and is defined for TDD mode. For the frame structure type 1, the bandwidth is divided into two parts, allowing downlink and uplink data transmissions simultaneously. For the frame structure type 2, the LTE frame is divided into two consecutive half-frames, each lasting 5 ms (see Fig. 8).

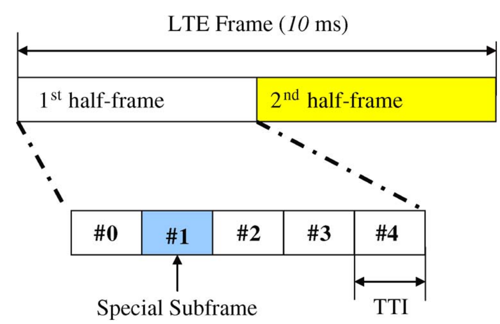


Fig. 8. Frame type 2 for TDD mode.

Moreover, a special subframe in each half-frame is reserved for other purposes and is not used for data transmission. In a real LTE network, this subframe is used to send downlink and uplink pilot symbols, separated by a guard period.

According to [14], Table II reports seven implemented uplink–downlink configurations of type-2 (TDD) frame. We note that subframes 0 and 5 are always reserved for downlink transmission.

The frame structure and the TDD frame configuration have been defined in the *FrameManager*. During the simulation, the *FrameManager* schedules the LTE frame and subframes and decides, according to the frame structure, if a subframe will be used for the uplink, the downlink, or both of them.

D. Radio Resource Scheduling

The most important objective of LTE scheduling is to satisfy QoS requirements of all users by trying to reach, at the same time, an optimal tradeoff between utilization and fairness [23].

TABLE II
FRAME-TYPE-2 CONFIGURATIONS

configuration number	sub-frame number: 1^{st} half frame 0	1	2	3	4	2^{nd} half frame 5	6	7	8	9
0	D	S	U	U	U	D	S	U	U	U
1	D	S	U	U	D	D	S	U	U	D
2	D	S	U	D	D	D	S	U	D	D
3	D	S	U	U	U	D	D	D	D	D
4	D	S	U	U	D	D	D	D	D	D
5	D	S	U	D	D	D	D	D	D	D
6	D	S	U	U	U	D	S	U	U	D

D = downlink sub-frame; U = uplink sub-frame; S = Special Sub-frame.

This goal is very challenging, particularly in the presence of real-time multimedia applications, which are characterized by strict constraints on packet delay and jitter.

In the LTE system, the concept of channel-sensitive scheduling has been introduced. It exploits the independent nature of fast fading across users. When there are many users that measure a different channel quality, it is highly likely to find a user with a good or relatively good channel condition at a given time. Based on this idea, maximum throughput and PF have become the most important well-known scheduling strategies for LTE networks [23].

Scheduling decisions are strictly related to the channel quality experienced by each UE, which periodically measures such a quality using reference symbols. Then, CQI feedback is sent to the eNB, using the uplink control messages [23]. This information is used by the scheduler to properly distribute RBs among users. Moreover, CQI feedbacks are also exploited by the link adaptation module to select, for each UE, the most suitable modulation scheme and coding rate at the PHY level, trying to maximize the spectral efficiency. This approach is known as AMC, and it has been adopted by several wireless technologies such as EDGE [30] and WiMAX [31]. Further details on AMC will be provided below.

To provide an important support to research activities on scheduling, we have implemented virtual uplink and downlink scheduler classes (called *ULScheduler* and *DLScheduler*, respectively) that offer a basic implementation of some methods common to all scheduling strategies. Moreover, to demonstrate the flexibility of the *LTE-Sim*, we have implemented PF, M-LWDF, and EXP downlink schedulers [11] in the *PF-DLScheduler*, *M-LWDF-DLScheduler*, and *EXP-DLScheduler* classes, respectively.

It is important to remark that it is very simple to add a generic scheduling algorithm by 1) creating a new scheduling class or extending an existing one; 2) inheriting methods from the virtual class; and 3) defining methods for the implementation of scheduling functions.

Both downlink and uplink scheduler objects are defined in the *ENodeB* class. Each scheduler runs the uplink/downlink scheduling algorithm using the *DoSchedule()* method, which is defined in the proper scheduler class. In detail, at the beginning of each subframe, the *FrameManager* handles the uplink/downlink resource allocation algorithm for all eNBs.

In the next section, we will give an overview of the implementation of all downlink schedulers provided within the current release of *LTE-Sim*.

E. Downlink Schedulers

At the beginning of each subframe, the scheduler selects all flows that can be scheduled. We remark that a flow can be scheduled if and only if it has data packets to transmit at the MAC layer and the receiver UE is not in the idle state.

Every TTI, the scheduler computes a given *metric* for each flow that can be scheduled. We will refer to $w_{i,j}$ as the *metric* assigned to the i-th flow for the j-th flow subchannel. Scheduling algorithms differ in the way the *metric* is calculated.

The scheduler works by assigning each j-th subchannel to the flow with the highest $w_{i,j}$. Thus, the scheduling procedure can be summarized as follows

1) The eNB creates a list of downlink flows having packets to transmit (i.e., *FlowsToSchedule*), that is, the list of flows that can be scheduled in the current subframe.
2) In such a *FlowsToSchedule* list, MAC queue length and CQI feedbacks are stored for each flow.
3) According to the scheduling strategy, the chosen metric is computed for each flow in the *FlowsToSchedule* list.
4) The eNB assigns each subchannel to the flow that presents the highest metric. It is important to remark that the eNB, during the resource allocation procedure, considers the quota of data that each flow has already sent. Therefore, as soon as a flow sends all enqueued packets, its record is deleted from the *FlowsToSchedule* list.
5) For each scheduled flow, the eNB computes the size of transport block (TB), i.e., the quota of data that will be transmitted at the MAC layer during the current TTI. In detail, the eNB uses the AMC module (defined into the *Phy* object) to map the CQI feedback with the proper MCS. Then, it can obtain the TB size from the selected MCS, which is the quota of binary data at the PHY layer.

At the end of the scheduling procedure, the eNB calls the *Bearer::Dequeue()* function of all scheduled flows that provides the dequeue of packets at the MAC layer.

To obtain the metric, scheduler algorithms usually need to know the average transmission data rate $\overline{R_i}$ of the i-th flow, as well as the instantaneous available data rate of the receiver UE for the j-th subchannel. This knowledge is useful when the metric has to take into account information about the performance guaranteed in the past to each flow to perform fairness balancing. In particular, every TTI, the estimation of $\overline{R_i}$ is given by

$$\overline{R_i}(k) = 0.8\overline{R_i}(k-1) + 0.2R_i(k) \tag{1}$$

where $R_i(k)$ is the data rate achieved by the i-th flow during the kth TTI, and $\overline{R_i}(k-1)$ is the estimation in the previous TTI.

In what follows, we will describe how each implemented downlink scheduler computes the metric.

1) PF Scheduler: It assigns radio resources taking into account both the experienced channel quality and the past user throughput [32]. The goal is to maximize the total network throughput and to guarantee fairness among flows. For this scheduler, the metric $w_{i,j}$ is defined as the ratio between the instantaneous available data rate (i.e., $r_{i,j}$) and the average past data rate. That is, with reference to the i-th flow in the j-th flow subchannel, i.e.,

$$w_{i,j} = \frac{r_{i,j}}{\overline{R_i}} \tag{2}$$

where $r_{i,j}$ is computed by the AMC module considering the CQI feedback that the UE hosting the ith flow have sent for the jth subchannel, and $\overline{R_i}$ is the estimated average data rate.

2) M-LWDF Scheduler: It supports multiple data users with different QoS requirements [11]. For each real-time flow, considering a packet delay threshold τ_i, the probability δ_i is defined as the maximum probability that the delay $D_{\mathrm{HOL},i}$ of the head-of-line packet (i.e., the first packet to be transmitted in the queue) exceeds the delay threshold.

To prioritize real-time flows with the highest delay for their head-of-line packets and the best channel condition, the metric is defined as

$$w_{i,j} = \alpha_i D_{\mathrm{HOL},i} \cdot \frac{r_{i,j}}{\overline{R_i}} \tag{3}$$

where $r_{i,j}$ and $\overline{R_i}$ have the same meaning as the symbols in (2), and α_i is given by

$$\alpha_i = -\frac{\log \delta_i}{\tau_i}. \tag{4}$$

Instead, for non-real-time flows, the considered metric is the one of the simple PF.

Note that, in the current implementation of the M-LWDF allocation scheme, packets belonging to a real-time flow are erased from the MAC queue if they are not transmitted before the expiration of their deadline. This operation is required to avoid wasting bandwidth. This implementation is not available for the PF, because it is not designed for real-time services.

3) EXP Scheduler: It has been designed to increase the priority of real-time flows with respect to non-real-time ones, where their head-of-line packet delay is very close to the delay threshold [11]. For real-time flows, the considered *metric* is computed by using the following equations:

$$w_{i,j} = \exp\left(\frac{\alpha_i D_{\mathrm{HOL},i} - \chi}{1+\sqrt{\chi}}\right) \frac{r_{i,j}}{\overline{R_i}} \tag{5}$$

where the symbols have the same meaning as the ones in (2) and (3), and

$$\chi = \frac{1}{N_{\mathrm{rt}}} \sum_{i=1}^{N_{\mathrm{rt}}} \alpha_i D_{\mathrm{HOL},i} \tag{6}$$

with N_{rt} being the number of active downlink real-time flows.

Instead, for non-real-time flows, the considered metric is the one of the simple PF. Also, with the EXP algorithm, packets belonging to a real-time flow are erased from the MAC queue if they are not transmitted before the expiration of their deadline.

F. Channel and PHY Layer

The simulation of a complete PHY layer is not suited for complex network scenarios (including the entire protocol stack), as it requires a high computational effort [33]. For example, the simulator proposed in [9], which implements a complete LTE PHY layer, requires more than 3 h to simulate a few seconds of a network composed of only two nodes. This effort can greatly be reduced by employing system-level simulators in which the PHY layer is described by an analytic model.

For this reason, we chose an analytical model approach in *LTE-Sim*, which has also successfully been used in other wireless network simulators that have already been developed for analogous technologies and are freely available for the community (such as WiMAX in ns-3 [34], Numbat module in Omnet++ [35], and UMTS model in ns-2 [36]).

The *Channel* and *Phy* classes model the channel and the LTE PHY layer, respectively. For all network devices, an instance of the *Phy* class, named *m_phy*, has been defined, and it is attached to a given channel to accomplish several functionalities: 1) the estimation of SINR; 2) the selection of a proper MCS before packet transmission (in particular, such a function is provided by the AMC module); and 3) the access to the channel to allow transmission and reception of packets.

1) Channel: It has been developed to handle packet transmission, taking into account the propagation loss model. To separately manage downlink and uplink, a couple of channels must be defined for each cell. Moreover, each PHY object stores into variables *m_downlinkChannel* and *m_uplinkChannel* a pointer to channels created for a cell the device belongs to.

The *Channel* class has a private structure named *m_devices* that handles all the PHY objects connected to it.

When a PHY instance has to send packets on a set of subchannels, it calls the *Channel::StartTx* method of a proper channel object (i.e., the downlink channel for the eNB and the uplink channel for the UE), passing a list of packets to send and the transmission power. *Channel::StartTx* handles packet transmission in two consecutive steps. It first calculates, for each attached PHY device, the propagation losses according to the propagation loss model (see below for details) and updates the power of the transmitted signal. Then, it forwards packets to all PHY devices attached to it and calls the reception procedure (see below for details on such a procedure).

2) Propagation Loss Model: Class *PropagationLossModel* models the channel propagation of the E-UTRAN interface. This class has been developed to compute the transmitted signal propagation losses.

To support various cell scenarios (i.e., urban microcell, suburban macrocell, urban macrocell, and rural macrocell), a virtual class, i.e., *ChannelRealization*, has been developed to provide a basic implementation of both propagation and channel models; it realizes the channel condition in terms of loss. As proposed in [15], a realization of the channel condition should consider four different phenomena: 1) the path loss; 2) the penetration loss; 3) the shadowing; and 4) the effect of fast fading due to the signal multipath.

The variable *m_channelRealizations*, which is defined in the *PropagationLossModel* class, stores a *ChannelRealization* object for each couple of devices attached to a given channel. During packet transmissions, knowing the source device and the destination device, a *ChannelRealization* object

TABLE III
IMPLEMENTED PATH LOSS MODELS

Cell scenario	Path loss model
Macro Cell - Urban and Suburban Areas	$L = 128.1 + 37.6 \log_{10} d$ @ 2GHz.
Macro Cell - Rural Area	$L = 100.54 + 34.1 \log_{10} d$ @ 2GHz.
Micro Cell	$L = 24 + 45 \log_{10}(d + 20)$.

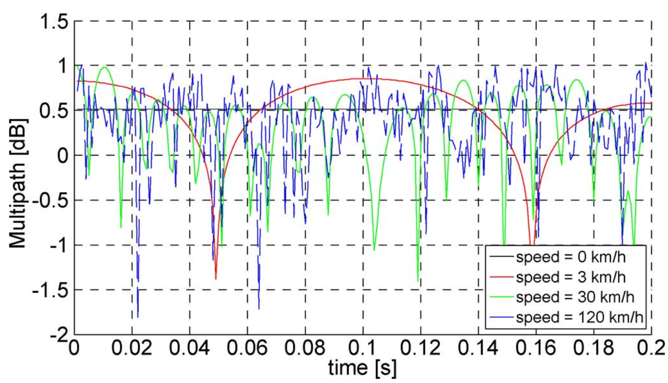


Fig. 9. Fast-fading realization.

is selected and associated with them, forming the *m_ channelRealizations* variable. Then, the *ChannelRealization:: ComputePropagationLoss()* is called to compute the loss due the propagation.

To understand how the propagation loss model works, we can analyze what happens in a downlink transmission.

Let $P_{TX,j}$ and $P_{RX,i,j}$ be the eNB transmission power and the reception power of the i-th UE for the j-th subchannel, respectively. $P_{RX,i,j}$ is given by

$$P_{RX,i,j}|_{dB} = (P_{TX,j} - M_{i,j} - L_i - T_i - S_{i,j})|_{dB} \quad (7)$$

where $M_{i,j}$, L_i, T_i, and $S_{i,j}$ are the losses due to multipath, path loss, penetration, and shadowing, respectively. Note that all of these variables are expressed in decibels.

To simulate cell scenarios with different propagation loss models, we have implemented three possible channel realizations: 1) a macrocell channel realization for urban and suburban areas; 2) a macrocell channel realization for the rural area; and 3) a microcell channel realization. For each of these models, a dedicated class, which inherits from the *ChannelRealization* class, has been developed.

As default, the large-scale shadowing fading has been modeled trough a lognormal distribution with 0 mean and 8 dB of standard deviation. The penetration loss, instead, is set to a default value of 10 dB [15]. It is possible to modify these values according to the simulated cell environment.

Table III reports path loss models used for each of the implemented cell scenarios [37], [38]. Note that d is the distance between the eNB and the UE in kilometers.

At present, the fast fading has been modeled for all the implemented propagation models with the Jakes model [39] for Rayleigh fading, taking into account the user speed, the subcarrier frequency (i.e., the central frequency of the jth subchannel), and a number of multiple paths uniformly chosen in the set (6, 8, 10, 12) [40]. Fig. 9 shows an example of multipath realizations when user speeds are equal to 0, 3, 30, and 120 km/h.

It is worth noting that a new channel model can easily be added to the *LTE-Sim*, extending the *ChannelRealization* class.

3) UE Reception Procedure: When the i-th UE receives packets, it executes the follows steps.

1) The PHY layer computes, for each subchannel, the SINR for the received signal considering the received power, the noise, and the interference as follows:

$$\text{SINR}i, j = \frac{P_{RX,i,j}}{(FN_0B) + I}. \quad (8)$$

where F, N_0, B_j, and I are the noise figure (with a default value of 2.5), the noise spectral density (with a default value of -174 dBm), the bandwidth of a resource block (i.e., 180 kHz), and the interference, respectively.

We remark that the interference is the total power received from the eNBs sharing the same frequency resources. The propagation loss of the interfering power are calculated by the *NetworkManager* through the *ComputePathLossForInterference ()* method, which selects the proper propagation loss model, depending on the cell scenario.

2) According to the CQI reporting mode, the UE creates CQI feedbacks to send to the eNB.
3) The PHY layer determines if packets have been correctly received. To this aim, for each subchannel used to transmit that packet, the block error rate (BLER) is estimated, that is, the ratio between the number of erroneous received blocks (i.e., the TB with wrong CRC) and the total number of sent blocks [41].

The BLER is obtained considering both the MCS used for the transmission and the SINR that the device has estimated for the considered subchannel. In particular, the BLER value is drawn using stored BLER-SINR curves obtained through an LTE link-level simulator [9]. In *LTE-Sim*, different sets of BLER–SINR curves are stored, and the choice of the proper set depends on several PHY parameters. As an example, the curves for a 1.4 MHz bandwidth and a single-input single-output transmission scheme over an additive white Gaussian noise channel are shown in Fig. 10. According to the proper BLER–SINR curve (depending on the used MCS), the simulator estimates if the packet has correctly been received or not. In the latter case, the packet is considered erroneous and discarded.
4) If the packet has been correctly received, it is forwarded to the upper layers.

4) CQI: During the resource allocation process, the eNB can select the most suitable MCS for each scheduled flow by trying to minimize the packet loss due to the channel error.

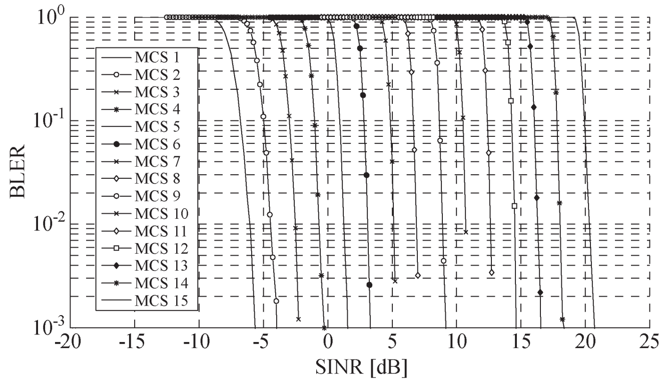


Fig. 10. BLER–SINR curves for 1.4 MHz.

The UE uses the *CQIManager* to create CQI feedbacks. When the UE receives packets in the downlink, it estimates the SINR for each downlink subchannel. Then, according to the CQI reporting rules, it creates CQI feedbacks to send to the eNB at which it is registered.

The CQI is used by the UE to report to the eNB the highest data rate that can be achieved over a given subchannel while guaranteeing a BLER that is at least equal to a certain BLER target (the default value is 10%). In particular, the CQI value is obtained as a quantized version of the estimated SINR. The mapping procedure between the SINR and the CQI is performed again through the BLER–SINR curves (see Fig. 10). Using these mapping tables, it is possible to select the best MCS (in terms of the data rate) that for the given SINR guarantees a BLER value smaller than the target BLER. Finally, the reported CQI corresponds to the index of the selected MCS.

As said before, *LTE-Sim* supports both periodical and aperiodical CQI reporting with both full-band and wide-band reporting modes. When the periodical CQI reporting is selected, the variable *m_reportingInterval*, which is defined in the *CQIManager* class, identifies when CQI feedbacks should be created and sent to the eNB. When the aperiodical CQI reporting is selected, instead, the UE creates and sends CQI feedbacks only when it receives a request from the eNB.

5) AMC Module: The AMC module, which is implemented in the *AMC* class, has been developed to allow eNB to select, during the resource allocation procedure, the proper modulation and coding scheme for the flow that has to be scheduled. To maximize the spectral efficiency, the MCS is chosen considering the latest CQI value sent by the UE and using the *AMC::GetMCSFromCQI ()* method. It is important to note that the selection of the MCS allows us to obtain the efficiency (expressed as the number of informative bits per symbols), as described in [16].

6) Determination of the TB Size: In LTE systems, the TB is the quota of binary data at the PHY layer, coming from the MAC layer during transmission (or passed to the MAC layer during reception) on transport channels [16]. In other words, TB is the number of bytes that a flow can transmit in one or more subchannels at the MAC layer (including the MAC overhead and the CRC trailer) during one TTI. The TB size depends on the MCS chosen by the AMC module, the number of antenna ports, the duration of the prefix code used at PHY layer, and the number of symbols used by the control channel. The *AMCModule* object uses the table *m_TBSize_Table* to evaluate the TB size from the selected MCS value. To this aim, it takes into account the configuration proposed in [42]: normal prefix code, two antenna ports, three OFDM symbols for PDCCH, no sync signals, and the absence of the PHY broadcast channel (PBCH).

V. Simulation and Performance Evaluation

In this section, we describe how an LTE scenario can be defined and how to manage the output results. Then, a performance evaluation of the *LTE-Sim* is reported, considering both the software scalability test and the performance evaluation of realistic LTE networks.

A. Building an LTE Topology

With *LTE-Sim*, an LTE scenario can be created as a static function in a C++ header file, which should be stored into the simulation/scenarios folder. A reference of this function should be added into the main program. This way, the user is able to simulate a proper LTE scenario, directly selecting it from the main program.

A basic scenario can be created using the following guidelines.

1) Create an instance for the *Simulator*, *NetworkManager*, *FlowsManager*, and *FrameManager* components.
2) Create *Cell*, *ENodeB*, and *UE* objects using methods of the *NetworkManager* class. For each of these objects, several parameters can directly be assigned with the constructor of the class.
3) Create applications, defining for each of them the source and destination, the QoS parameters, the IP classifier parameters, the start time, and the stop time.
4) Define the duration of the simulation, and finally, call the *Simulator::Run()* function.

To simplify the use of the simulator, both single-cell and multicell LTE topologies are proposed as examples (both stored into the simulation/scenarios folder), where the number of UEs and the number of flows can be specified by the command line.

B. Tracing Results

The current version of the simulator provides a sophisticated tracing functionality. The trace is directly displayed during the execution of the simulation. An example of the output trace is reported in Fig. 11.

With reference to the figure, the first field describes the event that has triggered the tracing. In particular, rows starting with *TX*, *RX*, and *DROP* are associated with packets that have been sent, received, and dropped, respectively. The second field describes the packet type to which the trace refers. Other fields are described as follows.

1) ID = Identifier. It uniquely identifies the packet.
2) B = Bearer ID. It identifies the bearer used to map the packet.

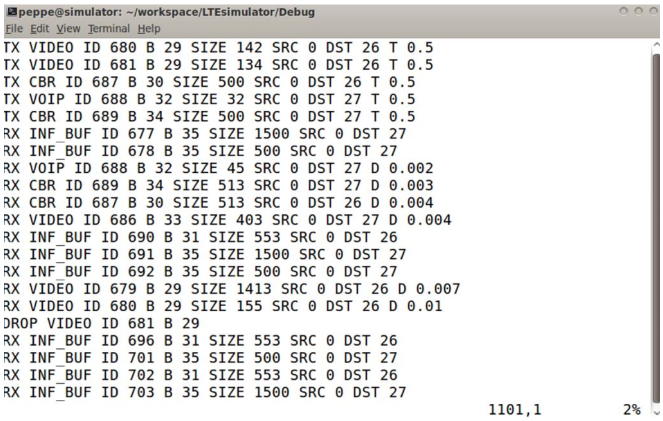


Fig. 11. Example of output trace.

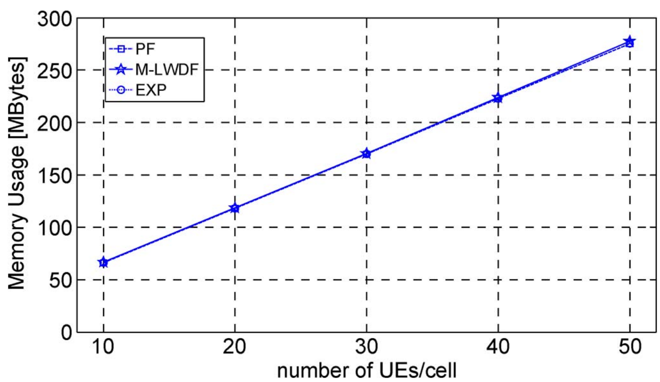


Fig. 12. Memory required versus number of UEs.

3) SRC = Source ID. It identifies the node that sends the packet.
4) DST = Destination ID. It identifies the node that receives the packet.
5) T = Time. It represents the instant in which the packet is created.
6) D = Delay. It represents the delay of the received packet.

C. Scalability Test

To present a scalability test of *LTE-Sim*, several simulations have been executed. An LTE network composed by 19 cells with a radius equal to 1 km has been considered, varying the number of UEs in the range [190–950]. Such users are uniformly distributed among cells. UEs travel with a speed equal to 3 km/h, using the random walk mobility model [17].

For each UE, a downlink VoIP flow has been considered. All flows are active during the whole simulation. Each simulation lasts 100 s. Simulation has been done using a Linux machine with a 2.6-GHz CPU and 4 GB of RAM.

Figs. 12 and 13 show the memory and the time required to execute the simulation, respectively.

The memory used to execute the simulation increases with the number of UEs, and it is independent on the scheduling algorithm used by the eNB. Moreover, the time required to execute the simulation increases with the number of UEs; it depends on the scheduling algorithm used by the eNB. In

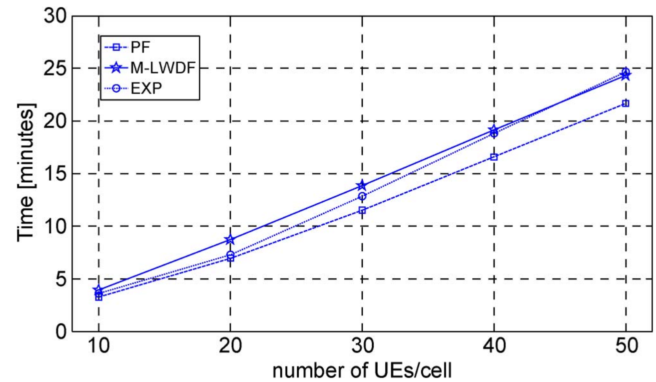


Fig. 13. Simulation time versus number of UEs.

fact, the PF algorithm requires less time than other scheduling strategies due to its very simple implementation.

An important result is that *LTE-Sim* requires a limited amount of memory and simulation time, also in high load scenarios.

D. Performance Evaluation

Herein, a performance evaluation of the implemented downlink scheduling algorithms (i.e., PF, M-LWDF, and EXP) is reported. A realistic scenario is considered: There are 19 cells, with a radius equal to 1 km, where a number of UEs (chosen in the range [10–30]) are uniformly distributed into each cell. UEs travel inside the area following the random walk mobility model in an urban macrocell scenario.

The whole bandwidth is distributed among a cluster of four cells to guarantee 10 MHz of bandwidth in the downlink for each cell.

Each user receives one H.264 video flow (encoded at 128 kb/s), one VoIP flow, and one best-effort flow modeled with the infinite-buffer application.

The performance of the PF, M-LWDF, and EXP schedulers have been analyzed, varying the user speed in the range [3, 120] km/h.

Fig. 14 shows the packet loss ratio (PLR) experienced by video and VoIP flows. As expected, the PLR increases with the user speed because the link adaptation procedure is impaired at high velocity. Furthermore, we note a different behavior between the schedulers designed for real-time services (i.e., M-LWDF and EXP) and the PF scheduler. When M-LWDF and EXP schedulers are used, the PLR increases with the number of UEs because in scenarios with a high number of concurrent real-time flows, the probability of discarding packets for deadline expiration increases. When the PF scheduler is used, instead, the PLR decreases as the number of UEs increases. In fact, this is because in a scenario with a high load factor, the PF scheduler can exploit the multiuser diversity gains, reducing the loss probability at the PHY layer. It is important to note that, in this case, the PLR only counts for the PHY losses and that it cannot grant for bounded packet delays. This effect can easily be seen in Fig. 15, where the average packet delays of real-time flows are reported.

The behavior of best-effort flows has also been studied; Fig. 16 shows the achieved goodput. Its value has been averaged

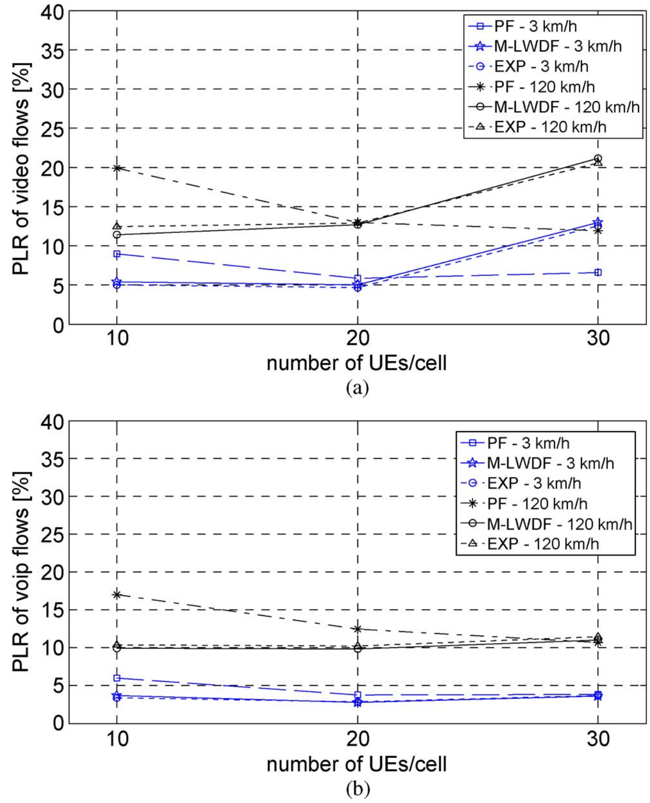


Fig. 14. PLR of (a) video and (b) VoIP flows.

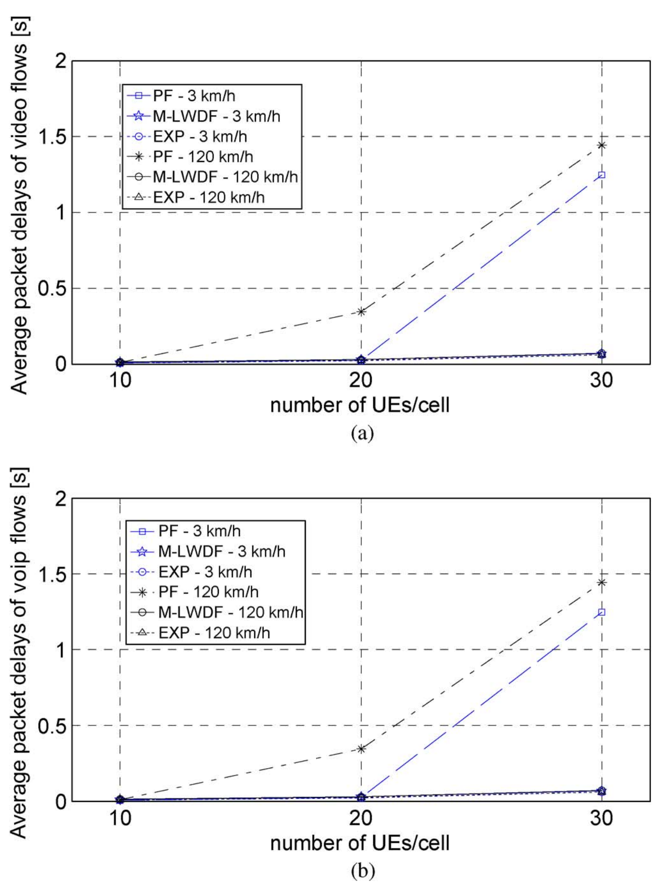


Fig. 15. Average packet delays of (a) video and (b) VoIP flows.

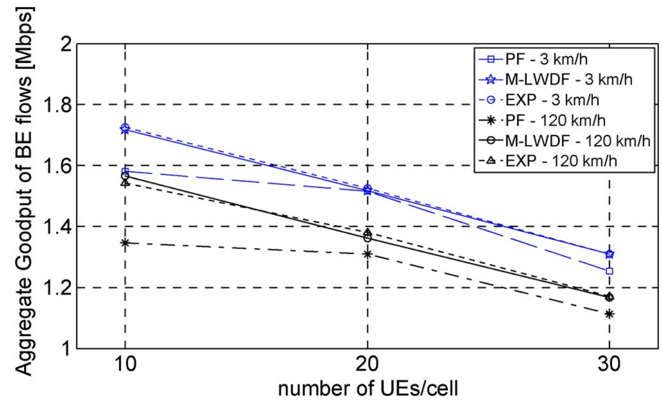


Fig. 16. Goodput achieved by best-effort flows.

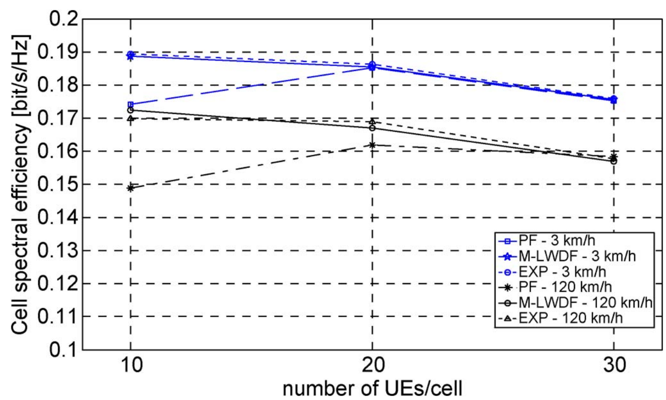


Fig. 17. Cell spectral efficiency.

among all cells. As expected, the goodput decreases as the user speed increases, due to the worse channel quality measured by the receiver UE. Also, the Jain fairness index [43] has been computed, considering the goodput achieved by best-effort flows at the end of each simulation. In all operative conditions, the index is very close to 0.9, meaning that all considered scheduling strategies provide comparable levels of fairness.

Finally, Fig. 17 shows the cell spectral efficiency achieved for the considered LTE scenarios and expressed as the total throughput achieved by all users divided by the available bandwidth [44]. As expected, different schedulers impact differently. When the number of users in the cell increases, QoS-aware schedulers such as M-LWDF and EXP still try to guarantee QoS constraints to a high number of flows, with a consequent negative impact on the system efficiency. On the other hand, when the number of users increases from 10 to 20, the PF scheduler is able to exploit the gain of the multiuser diversity while granting a high fairness index (i.e., 0.9). When the number of users further increases, the same fairness index is reached by the PF scheduler at the cost of a limited drop in the cell spectral efficiency.

VI. Conclusion

In this paper, a new open-source framework to simulate LTE networks, namely, *LTE-Sim*, has been proposed. Features covered by this simulator will allow both researchers and practitioners to test enhanced techniques to improve 4G cellular networks, such as new PHY functionalities, innovative network protocols and architectures, and high-performance scheduling strategies. The open nature of this software can allow people interested in research in this field to contribute to the development of the framework, furnishing a reference platform for testing and comparing new solutions for LTE systems.

The effectiveness of the developed simulator has been verified with several simulations to study the scalability and the performance of the framework. Moreover, *LTE-Sim* has been applied to compare several scheduling strategies and to evaluate their performance. In the near future, we plan to improve the simulator implementing new features, such as HARQ and more sophisticated channel and PHY models, which have not been included in the current version of the software.

REFERENCES

[1] 3GPP. [Online]. Available: http://www.3gpp.org

[2] 3GPP, Tech. Specif. Group Radio Access Network Requirements for Evolved UTRA (E-UTRA) and Evolved UTRAN (E-UTRAN), 3GPP TS 25.913.

[3] 3GPP, Tech. Specif. Group Services and System Aspects Service Requirements for Evolution of the 3GPP System (Rel. 8), 3GPP TS 22.278.

[4] D. McQueen, "The momentum behind LTE adoption," *IEEE Commun. Mag.*, vol. 47, no. 2, pp. 44–45, Feb. 2009.

[5] S. Ascent, 3GPP LTE toolbox and blockset. [Online]. Available: http://www.steepestascent.com/content/default.asp?page=s2_10

[6] mimoOn, mi!Mobile. [Online]. Available: http://www.mimoon.de/pages/Products/miMobile/

[7] Aricent, LTE layer 1, LTE baseband/PHY library. [Online]. Available: http://www.aricent.com/Expertise/LTE.aspx

[8] J. J. Sánchez, G. Gómez, D. Morales-Jiménez, and J. T. Entrambasaguas, "Performance evaluation of OFDMA wireless systems using WM-SIM platform," in *Proc. ACM Int. Workshop MobiWac*, 2006, pp. 131–134.

[9] C. Mehlführer, M. Wrulich, J. C. Ikuno, D. Bosanska, and M. Rupp, "Simulating the long term evolution physical layer," in *Proc. 17th EUSIPCO*, Glasgow, U.K., 2009, pp. 1471–1478.

[10] J. C. Ikuno, M. Wrulich, and M. Rupp, "System level simulation of LTE networks," in *Proc. IEEE VTC—Spring*, Taipei, Taiwan, May 2010, pp. 1–5.

[11] R. Basukala, H. M. Ramli, and K. Sandrasegaran, "Performance analysis of EXP/PF and M-LWDF in downlink 3GPP LTE system," in *Proc. 1st AH-ICI*, Kathmandu, Nepal, Nov. 2009, pp. 1–5.

[12] G. Piro, LTE-Sim—The LTE simulator. [Online]. Available: http://telematics.poliba.it/LTE-Sim

[13] A. Alexandrescu, *Modern C++ Design: Generic Programming and Design Patterns Applied*. Reading, MA: Addison-Wesley, 2001.

[14] 3GPP, Tech. Specif. Group Radio Access Network; Physical Channel and Modulation (Rel. 8), 3GPP TS 36.211.

[15] 3GPP, Tech. Specif. Group Radio Access Network; Physical layer aspect for evolved Universal Terrestrial Radio Access (UTRA) (Rel. 7), 3GPP TS 25.814.

[16] 3GPP, Tech. Specif. Group Radio Access Network; Physical layer procedures (Rel. 9), 3GPP TS 36.213.

[17] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless Commun. Mobile Comput.*, vol. 2, no. 5, pp. 483–502, Aug. 2002.

[18] F. Khang, *LTE for 4G Mobile Broadband, Air Interface Technologies and Performance*. Cambridge, U.K.: Cambridge Univ. Press, 2009.

[19] H. Ekstrom, "QoS control in the 3GPP evolved packet system," *IEEE Commun. Mag.*, vol. 47, no. 2, pp. 76–83, Feb. 2009.

[20] 3GPP, Tech. Specif. Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Overall description; Stage 2, 3GPP TS 36.300.

[21] Video trace library. [Online]. Available: http://trace.eas.asu.edu/

[22] C. Chuah and R. H. Katz, "Characterizing packet audio streams from Internet multimedia applications," in *Proc. ICC*, New York, Apr. 2002, pp. 1199–1203.

[23] E. Dahlman, S. Parkvall, J. Skold, and P. Beming, *3G Evolution HSPA and LTE for Mobile Broadband*. New York: Academic, 2008.

[24] 3GPP, Tech. Specif. Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Link Control (RLC) protocol specification (Rel. 9), 3GPP TS 36.322.

[25] IETF, RObust Header Compression (ROHC): Framework and Four Profiles: RTP, UDP, ESP, and uncompressed, IETF RFC 3095.

[26] M. Rinne, M. Kuusela, E. Tuomaala, P. Kinnunen, I. Kovacs, K. Pajukoski, and J. Ojala, "A performance summary of the evolved 3G (E-UTRA) for voice over Internet and best effort traffic," *IEEE Trans. Veh. Technol.*, vol. 58, no. 7, pp. 3661–3673, Sep. 2009.

[27] 3GPP, Tech. Specif. Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment (UE) radio transmission and reception (Rel. 9), 3GPP TS 36.101.

[28] T. S. Rappaport, *Wireless Communications: Principles and Practice*. Englewood Cliffs, NJ: Prentice-Hall, 2002.

[29] S. Sesia, I. Toufik, and M. Baker, *LTE—The UMTS Long Term Evolution: From Theory to Practice*. New York: Wiley, 2009.

[30] T. Halonen, J. Romero, and J. Melero, *GSM, GPRS and EDGE Performance: Evolution Towards 3G/UMTS*. New York: Wiley, 2003.

[31] L. Nuaymi, *WiMAX: Technology for Broadband Wireless Access*. New York: Wiley, 2008.

[32] J.-G. Choi and S. Bahk, "Cell-throughput analysis of the proportional fair scheduler in the single-cell environment," *IEEE Trans. Veh. Technol.*, vol. 56, no. 2, pp. 766–778, Mar. 2007.

[33] M. Wrulich, W. Weiler, and M. Rupp, "HSDPA performance in a mixed traffic network," in *Proc. IEEE VTC—Spring*, Singapore, May 2008, pp. 2056–2060.

[34] M. A. Ismail, G. Piro, L. A. Grieco, and T. Turletti, "An improved IEEE 802.16 WiMAX module for the ns-3 simulator," in *Proc. SIMUTools*, Mar. 2010, pp. 63:1–63:10.

[35] Omnet++, Numbat—New ubiquitous mobility basic analysis tools: WiMAX, DHCPv6 implementation in Omnet++. [Online]. Available: http://klub.com.pl/numbat/

[36] P. Martin and P. Ballester, The UMTS extension for Network Simulator 2. [Online]. Available: http://www.info.fundp.ac.be/~lsc/Research/ns-2_UMTS/

[37] 3GPP, Tech. Specif. Group Radio Access Networks; Radio Frequency (RF) system scenarios (Rel. 9), 3GPP TS 25.942.

[38] 3GPP, Tech. Specif. Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Radio Frequency (RF) system scenarios (Rel. 10), 3GPP TS 36.942.

[39] W. C. Jackes, *Microwave Mobile Communications*. New York: Wiley, 1975.

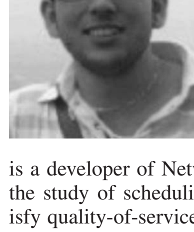
[40] C. Wei, Z. Lili, and H. Zhiyi, "Second-order statistics of improved Jakes' models for rayleigh fading channels wireless communications," in *Proc. Int. Conf. Netw. Mobile Comput., WiCom*, Shanghai, China, Sep. 2007, pp. 1108–1111.

[41] 3GPP, Tech. Specif. Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Terminal conformance specification, Radio transmission and reception (FDD), 3GPP TS 34.121.

[42] 3GPP, Tech. Specif. Group Radio Access Network; Conveying MCS and TB size via PDCCH, 3GPP TSG-RAN WG1 R1-081483.

[43] R. Jain, *The Art of Computer Systems Performance Analysis*. New York: Wiley, 1991.

[44] M. S. Alouini and A. J. Goldsmith, "Area spectral efficiency of cellular mobile radio systems," *IEEE Trans. Veh. Technol.*, vol. 48, no. 4, pp. 1047–1066, Jul. 1999.



Giuseppe Piro (S'10) received the first-level and second-level degrees (both *cum laude*) in telecommunications engineering from the Politecnico di Bari, Bari, Italy, in 2006 and 2008, respectively, where he is currently working toward the Ph.D. degree in electronics engineering.

In June 2009 and November 2009, he collaborated with Planete Project–INRIA, Sophia Antipolis, France, as a Research Assistant. He participated in the Google Summer of Code 2010, working on a project on long-term evolution (LTE) networks. He is a developer of Network Simulator 3. His current research interests include the study of scheduling algorithms and admission control algorithms to satisfy quality-of-service requirements of multimedia flows in wireless networks (IEEE 802.11 WLAN, UMTS LTE, and WiMAX).

Luigi Alfredo Grieco (S'02–M'04) received the Dr. Eng. degree (with honors) in electronic engineering from the Politecnico di Bari, Bari, Italy, in October 1999 and the Ph.D. degree in information engineering from the Università di Lecce, Lecce, Italy, in December 2003.

Since January 2005, he has been an Assistant Professor of telecommunications with the Dipartimento di Elettrotecnica ed Elettronica, Politecnico di Bari. From March to June 2009, he was a Visiting Researcher with INRIA, Planete Project, Sophia Antipolis, France, working on internet measurements and scheduling in WiMax networks. He is the author of more than 90 scientific papers published in international journals and conference proceedings. His main research interests are congestion control in packet-switching networks, quality of service and service discovery in wireless networks, Internet multimedia applications, Internet measurements, and real-time video processing using cellular nonlinear networks.

Gennaro Boggia (S'99–M'01–SM'09) received the Dr. Eng. and Ph.D. degrees (with honors) in electronics engineering from the Politecnico di Bari, Bari, Italy, in July 1997 and March 2001, respectively.

From May 1999 to December 1999, he was a Visiting Researcher with the TelecomItaliaLab (TILab, formerly the Centro Studi e Laboratori Telecomunicazioni), Torino, Italy, where he was involved in the study of the future releases of cellular mobile networks. Since September 2002, he has been with the Dipartimento di Elettrotecnica ed Elettronica, Politecnico di Bari, where he is currently an Assistant Professor. In 2007, he was a Visiting Researcher with Forschungszentrum Telekommunikation Wien, Vienna, Austria, where he was involved in activities on passive and active traffic monitoring in cellular networks. He is the author or a coauthor of more than 70 papers in international journals and conference proceedings. His research interests span the fields of wireless networking, multimedia systems, cellular communication, quality of service in wireless networks, queueing networking, and network performance evaluation.

Francesco Capozzi (S'10) was born in Bari, Italy, on April 14, 1984. He received the first-level and Master degrees in telecommunications engineering, both with honors, from the Politecnico di Bari, in September 2005 and April 2008. He is currently working toward the Ph.D. degree in information engineering jointly with the Politecnico di Bari and the Scuola Interpolitecnica di Dottorato, Bari, Milan, and Turin, Italy.

Since September 2008, he has been collaborating with Nokia Siemens Networks at Aalborg University, Aalborg, Denmark, as a Research Assistant. His current research interests are mainly focused on the study of new radio resource management algorithms for the next generations of cellular networks (long-term evolution networks).

Pietro Camarda received the University degree in computer science from the University of Bari, Bari, Italy, in 1978.

Since 1986, he has been with the Department of Elettrotecnica ed Elettronica, Politecnico di Bari, where he is currently a Professor of telecommunications. He was a Visiting Scholar, from October 1986 to December 1987, with the Computer Science Department, University of California, Los Angeles, under the supervision of Prof. M. Gerla. He has been involved in various research areas, mainly local area network/mobile area network architectures and protocols, optical networks, network reliability, cellular radio networks, telecommunication services, etc. He is the author of more than 100 research papers on these topics. Over the past few years, he has mainly focused on wireless networks (cellular networks, wireless local area networks, and ad hoc networks), developing topics that include resource allocation based on neural networks, header compression schemes for wireless networks, authentication protocols in wireless networks, bandwidth allocation and quality of service in WLANs, medium access control protocols for ad hoc networks, etc.