International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)
Web Site: www.ijettcs.org Email: editor@ijettcs.org, editorijettcs@gmail.com
Volume 2, Issue 2, March – April 2013 ISSN 2278-6856

Performance Analysis for QoS-Aware Two-Layer Scheduling in LTE Networks

TONY TSANG

Hong Kong Polytechnic University.
Hung Hom, Hong Kong

Abstract: ***Long Term Evolution (LTE) has been proposed as a promising radio access technology to bring higher peak data rates and better spectral efficiency. However, scheduling and resource allocation in LTE still face huge design challenges due to their complexity. In this paper, the optimization problem of scheduling and resource allocation for separate streams is first formulated. By separating streaming scheduling and packet sorting, the scheduler is aware of probabilistic state information, fairness among the streams, and the frame weight. We integrate our algorithm in a parallelized modification of the PRISM simulation framework. Extensive validation with both new and PRISM benchmarks demonstrates that the approach scales very well in scenarios where symbolic algorithms fail to do so. Simulations results with video sequences show that significant gains could be observed by our scheme in terms of spectrum efficiency, QoS of packet delay, and video quality while maintaining the fairness among the streams.***

Keywords: Long Term Evolution, radio access technology, QoS.

1. INTRODUCTION

Recent advance research has developed a large variety of smart mobile devices, which are powerful enough to support a wide range of multimedia traffic (e.g. VoIP, video streaming, multiplayer interactive gaming) and also legacy mobile services (e.g. voice, SMS, MMS). These new multimedia applications require high data rates and power to provide better Quality of Service (QoS). However, due to the low transmission rate and high service costs, the 3G (third generation) technology has been unsuccessful in delivering ubiquitous/high-speed mobile broadband.

To address the mobile broadband requirements, the 3GPP introduced the new radio access technology Long Term Evolution (LTE) which has the capability to move towards fourth generation (4G) wireless systems. LTE is designed to be a high data rate and low latency system that aiming to support different types of services, including web browsing, FTP, HD video streaming, VoIP, online multi-user interactive gaming and real time video. However, the use of enriched 4G services is still limited because the receivers of these services require computationally complex circuitry that drains the user equipment (UE) battery power quickly.

In our study, we first formulate the optimization problem of resource allocation for separate streams. Then, we show that it is reduced to the problem of packet scheduling. Various packet scheduling strategies for video transmission over wireless have been discussed including [1–3]. Regarded as a delay-limited capacity problem, The Earliest Deadline First (EDF) strategy is put forward to satisfy the delay constraints in [1]. Moreover, in content-aware schemes, the importance of the scheduled packet for decoders is considered as well [2, 3], i.e., Minimization Cost (MC) strategy. Nevertheless, these strategies don't refer to e-Multimedia Broadcast/Multicast Services (MBMS) system due to the following considerations. (I). each OFDM-based frame including multi-subcarriers is apt to be scheduled to the data from more than one stream. Obviously, it is inappropriate for multicast in view of power consumption, since each terminal needs to decode more frames including the data for its desired contents. [4]. (II). as for MBMS over a Single Frequency Network (MBSFN), the data entity is separated from the control entity. The control entity which is responsible for allocating resources has no idea of the related factors used by packet scheduling [5].

To resolve the problems above, we propose a suboptimum scheduling scheme, called the QoS-aware two-layer scheduling. The innovations lie in (I). it is up to specification of e-MBMS that a frame is allocated to one stream, thus the terminal is enabled to turn into sleep mode during several frames, when its undesired streams are being transmitted [6]. (II). the process of resource allocation is divided into two layers. In the longterm scheduling, we add the QoS-aware Scheduling Module (QASM) to the control entity, and it is able to acquire the information of queue state from the data entity, such as the packet urgency and fairness, to help decide the transmission order of streams. After that, the data entity ensures the prior transmissions of more important packets based on the frame weight in the short-term scheduling.

PRISM [8] is a state-of-the-art probabilistic model checker. We implemented our algorithm in Java, using a parallelized version of PRISMs simulation framework for trace generation. This allows us to seamlessly use PRISMs specifications for MDPs. We take care to ensure that our multi-threaded modification of the framework remains statistically unbiased. We apply our algorithm to both the PRISM benchmark suite as well as to new benchmarks and perform an extensive comparison. The results show that the algorithm is highly scalable and efficient. It also runs successfully on problems that are too large to be tackled by PRISMs exact engine.

Simulations results show that QoS-aware two-layer scheduling scheme performs well in exhaustive QoS metrics including spectrum efficiency, packet delay, and video quality, while maintaining the adequate fairness among the streams

2. QOS-AWARE TWO-LAYER SCHEDULING SCHEME

The conceived novel scheduling strategy targets real time service provisioning in the LTE downlink. It has been built on two distinct levels (see Fig. 1) that interact together in order to dynamically assign radio resources to user equipment (UE). They take into account the channel state, the data source behaviors, and the maximum tolerable delays.

At the highest level, an innovative resource allocation algorithm, frame level scheduler, namely FLS, defines frame by frame the amount of data that each real time source should transmit to satisfy its delay constraint. To solve the problem using a solution with a low computational complexity, FLS exploits a discrete-time linear control loop [9]. Once FLS has accomplished its task, the lowest layer scheduler, every transmission time interval (TTI), assigns resource blocks (RBs) using the proportional fair (PF) algorithm [10] by considering bandwidth requirements of FLS.

In other words, FLS defines on the long run (i.e., in a single frame) how much data should be transmitted by each data source. The lowest layer scheduler, instead, allocates resource blocks in each TTI to achieve a trade-off between fairness and system throughput. It is important to note that FLS does not take into account the channel status. On the contrary, the lowest layer scheduler assigns RBs first to flows hosted by UEs experiencing the best channel quality and then (i.e., when these flows have transmitted the amount of data imposed by FLS) it considers the remaining ones. In particular, the lowest layer scheduler decides the number of TTIs/RBs (and their position in the time/frequency domains) in which each real time source will actually transmit its packets. It is very important to remark that the proposed approach is very general and it is independent on the model used for describing incoming data. For this reason, we do not need stochastic flow models. In fact, the control theoretic approach describes a flow as a signal modeling the bit-rate produced by the application layer.

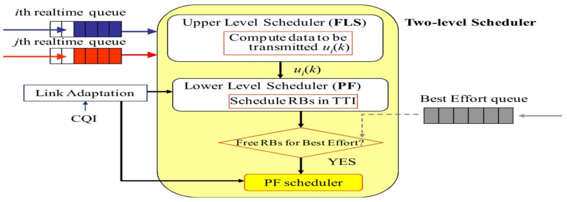

Fig. 1. QoS–aware Two-level Scheduling Algorithm

2.1 Frame Streaming Scheduling

A QoS-ware two-layer scheduling scheme is devised, where *wk* is divided into the streaming weight ws_k and frame weight $I_{k;m}$. In the first layer frame streaming scheduling, streaming weight is determined by Multi-cell/Multicast Coordination Entity (MCE) at Multicast Channel Scheduling Period (MSP) level. And then, evolved Node Bs (eNB) performs the packet sorting based on the results of streaming scheduling at TTI-level.

With the help of QoS-aware Scheduling Module (QASM), the following parameters offered by eNBs at the end of MSP, would help MCE to decide the transmission order for the next MSP. A certain eNB is enough since the action of each eNB is identical. Considering the cost of additional signaling, Delay Tolerance Factor (DT) and Fairness Penalty Factor (FP) are included to guarantee the throughput and fairness among the streams. Such scheduling is called Time-out-Based Scheduling Strategy (TBS) here.

$$DT_k = \frac{\mathrm{T}_{delay_{HoL}}}{\mathrm{T}_{PDB_k}} \qquad (1)$$

$$\mathrm{T}_{delay_{k,HoL}} = t - T_{k,HoL}\ , \qquad (2)$$

$$FP_k = \frac{1}{\dfrac{scheduled\_total_{\mathrm{T}_k}}{received\_total_{\mathrm{T}_k}}}. \qquad (3)$$

where $T_{delayk;HoL}$ is the period from the time spot $T_{k;HoL}$, i.e., when the head of line (HOL) packets arrived at the queue, to the current time spot t for the streaming *k* . T_{PDBk} is the Packet Delay Budget for video streaming *k* indicated by QCI.

The fairness is earliest proposed in unicast systems [11], we adopt it into the e-Multimedia Broadcast/Multicast Service (e-MBMS) system. $Scheduled\_total_{Tk}$ is the throughput of streaming *k* during a period. $Received\_total_{Tk}$ is the amount of received packets in this period for streaming *k*.

After acquiring DT and FP, QASM would determine the streaming weight

$$ws_k = \frac{received\_total_{\mathrm{T}_k}}{scheduled\_total_{\mathrm{T}_k}} \times exp(\frac{\mathrm{T}_{delay_{HoL}}}{\mathrm{T}_{PDB_k}}). \qquad (4)$$

Finally, the transmission order in the bundle is determined along with QCI for non-multiplexed streams. The stream in the bundle with a larger ws_k is prior transmitted.

Finally, the transmission order in the bundle is determined along with QCI for non-multiplexed streams. The stream in the bundle with a larger ws_k is prior transmitted.

2.2 Packet Sorting

To improve the video quality at receivers, the scheduler in eNB performs the packet sorting at TTI-level after streaming scheduling, called Cost-based Sorting Strategy (CSS).

The binary indicator δ_k is used to show whether the streaming *k* is scheduled completely or not, that is, whether there is any packet left in the queue to be scheduled.

$$\delta_k = \begin{cases} 0 & \text{if the streaming k is scheduled completely} \\ 1 & \textit{otherwise} \end{cases} \quad (5)$$

The corresponding streaming k^*, which is to be scheduled could be determined by the following equation

$$k^* = arg\max_{k=1,2,\ldots,K} ws_k \cdot \delta_k \, . \quad (6)$$

Despite being sufficient to achieve maximum probabilities, deterministic schedulers are a poor choice for exploring the state space through simulation: sampling with a deterministic scheduler provides information only for the actions that it chooses. Probabilistic schedulers are more flexible, explore further, and enable reinforcement of different actions. Thus, we always use probabilistic schedulers in the exploration part of our algorithm.

Ideally, σ converges to a near-deterministic scheduler, but due to our commitment to exploration, it will never do so completely. Before using Statistical Model Checking (SMC) to answer the Probabilistic Model Checking (PMC) question, we thus greedily determinise σ. More precisely, we compute a scheduler that always picks the best estimated action at each state. Formally, DETERMINISE (σ^*) is a new scheduler with the help of equation (6), for a determined streaming k^*, CSS could be described as (7)

$$\sigma^* = arg\max_{\sigma} I_{k^*,\sigma} \, , \quad (7)$$

where the more important packet with a higher frame weight
is prior allocated in CSS during the MSP.

We thus hope to redirect the residual probabilities of choosing bad actions to the promising regions of the state space. In practice, this step makes a significant difference.

Generally, QoS-aware Two-Layer Scheduling can be described as follows:

QoS-aware Two-layer Scheduling Scheme

```
1)Initialization
a) Set $\delta_k$ = 1 for all k ∈ {1, 2, · · · , K} .
   b) Set $\omega_{n;k;t}$ = 0 for all n ∈ {1, 2, · · · , N},
   k ∈ {1, 2, · · · , K}, and t ∈ {1, 2, · · · , T}
   c) Set i = 1 and j = 1
2) Determine $ws_k$ in MCE,
   where $ws_k$ is defined as (4)
   for all k ∈ {1, 2, · · · ,K} . Then, MCE informs
   eNBs of the results of resource allocation.
3) eNBs receive the MCH Scheduling Information
   (MSI).
4) While j ≤ T or $\delta_k$ = 0, ∀ k , in eNBs
   a) While i ≤ N
      i. Find k* where it is defined as (6).
      ii. Find the σ* as (7) for a given k*, then the
      selected packet is allocated to the pair i of RBs
      in TTI $T_j$ .
      iii. Update $\delta_k$, ∀ k , according to (5),
      iv. Update i = i + 1 .
   b) Update j = j + 1 .
   c) Set i = 1 .
5) The procedure of resource allocation is complete.
```

Since the interval time T of scheduling is enlarged, our scheme is suboptimum in the case of conventional scheduling strategies at TTI-level. However, from the view of realization, it ensures that one TTI Ti is allocated to one stream. Moreover, differing from the current semi-dynamic scheduling in LTE system, QoS of packet delay, fairness and the frame weight are considered in the long-term and short-term scheduling respectively, to aim to approach the performances achieved by the conventional strategies.

2.3. Number of Runs

Although we will show that the scheduler packet sorting stage converges towards frame streaming schedulers, at any given point we cannot quantify how close to frame streaming the candidate scheduler is. Statistical claims are possible, however. If the current candidate is sufficient to settle the original Probabilistic Model Checking (PMC) query, the algorithm can stop immediately. If it is not, it may be restarted after a reasonable number of improvement iterations. These restarts help our algorithm finding and focusing on more promising parts of the state space it might have missed before. Algorithms like this are called biased Monte Carlo algorithms. Given a confidence parameter (*p*) on how likely each run is to converge, we can make a statistical claim up to arbitrary confidence (η) on the number of times we have to iterate the algorithm, $T_{\eta;p}$:

Bounding Theorem [12] : For a false-biased, p-correct Monte Carlo algorithm (with $0 < p < 1$) to achieve a correctness level of $(1-\eta)$, it is sufficient to run the algorithm at least a number of times:

$$T_{\eta,p} = \frac{\log_2 \eta}{\log_2 (1-p)} \quad (8)$$

This result guarantees that, even in cases where the convergence of the scheduler learning procedure in one iteration is improbable, we will only need to run the procedure a relatively small number of times to achieve much higher confidence. Taking all these considerations into account, the main Statistical Model Checking (SMC) procedure for Markov decision processes (MDPs) is laid out in Algorithm QoS-aware Two-layer Scheduling Scheme. An important requirement of this algorithm and Bounding Theorem is that we have a positive probability of convergence to an frame streaming scheduler during scheduler learning.

3. PERFORMANCE EVALUATION

We evaluate our procedure on several well-known benchmarks for the PMC problem. First, we use one easily parametrisable case study to present evidence that the algorithm gives correct answers and then we present systematic comparisons with PRISM [8]. Our implementation extends the PRISM simulation framework for sampling purposes. Because we use the same input language as PRISM, many off-the-shelf models and case studies can be used with our approach.

Reinforcement Heuristics: Our approach allows us to tune the way in which we compute quality and reinforcement information without destroying guarantees of convergence (under easily enforced conditions) but netting significant speedups in practice. These optimizations range from negatively reinforcing failed paths to reinforcing actions differently based on their estimated quality.

3.1 Parametrisation

Our algorithms parameters generally affect both runtime and the rate of convergence, with dependence on the MDPs structure. In this section we will outline the methods used to decide values for each parameter.

- History h : high h causes slower convergence, whereas small h makes convergence less likely by making sampling variance a big factor. From a range of tests done over several benchmarks, we found 0.5 to be a good overall value by achieving a balanced compromise.
- Greediness ϵ : experimentally, the choice of $0 < \epsilon < 1$ influences the convergence of the algorithm. However, the heuristics we use do not allow us to set ϵ explicitly but still guarantee that $0 < \epsilon < 1$(necessary for convergence). For details, we refer to [13].
- Numbers of samples N and iterations L: the main factor in runtime is the total number of samples $N \times L$. A higher N yields more confidence in the reward information R of each iteration. A higher L makes the scheduler improve more often. Increasing L at the cost of N without compromising runtime ($N \times L$ constant) allows the algorithm to focus on interesting regions of the state space earlier. We ran several benchmarks using combinations of N and L resulting in the similar total number of samples, and found that a ratio of around 65 : 1 $N : L$ was a good overall value. The total number of samples is adapted to the difficulty of the problem. Most benchmarks used $N = 2000$ and $L = 30$, with smaller values possible without sacrificing results. Harder problems sometimes required up to $N = 5000$ and $L = 250$. If the ratio $N : L$ is fixed, N and L are just a bound on runtime. If $\theta > p$, the algorithm will generally run $N \times L$ samples, but if $\theta < p$, it will generally terminate sooner.
- Number of runs T : if a falsifying scheduler is found, the algorithm may stop (up to confidence in SMC). We used between 5 and 10 for our benchmarks.
- System throughput characterizes the average amount of data that is transmitted by the radio network in a certain amount of time. It is estimated to evaluate system resource utilization in this paper, and it is calculated in the following.

$$Throughput = \frac{\sum_{i=1}^{N_{user}} M_i}{\mathrm{T}_{Time_{SIM}}}$$

Where M_i denotes total amount of transmitted data of user i during $\mathrm{T}_{TimeSIM}$. N_{user} is total number of activated users.

- Packet Delay: It is the difference in time when a packet is created and when UE acknowledges that packet.

$$\text{Packet Delay} = \text{Created time - Acknowledged time}$$

- Statistical Model Checking: the Beta distribution parameters used were $\alpha = \beta = 0.5$ and Bayes factor threshold $T = 1000$. For an explanation these parameters, see [14].

3.2 Simulation Results

The performance of the proposed algorithm will be evaluated and compared with three traditional scheduling algorithms Proportional Fair (PF), Round Robin (RR) and Best CQI (BCQI) in normal mode - no Discontinuous Reception DRX). The evaluation and comparison are done in same simulation environment and parameter. Evaluation will be done on key performance evaluation parameters; as described in above subsection.

All the schedulers are used in the same simulation setup as presented in the following Table. The receivers of all the UE are switched-on all the time that means no power is being saved by the UEs. The traditional schedulers which are designed to work in non-DRX environment are being compared with Proposed Scheme.

However, the Proposed Scheme specially considers active and normal modes of UEs. Therefore, other schedulers may overwhelm the Proposed Scheme in one or more performance evaluation parameters.

Parameters	**Values**
eNodeB radius	250 m
Number of sectors per	eNodeB 3
Target area	Single sector
Number of UEs	0-100
eNodeB total TX power	20 W
Number of antennas (MIMO)	4 TX, 3 RX
Fading models	Fast fading
UE Speed	5 km/h
Operating frequency band	2 GHz
System channel bandwidth	5 MHz
Number of RBs	25
GBR	25 kbps
CQI reporting Every	TTI
Traffic model	Video
Video packet generation interval	20 ms
Video delay threshold	100 ms
Power saving mechanism DRX	Sleep
DRX on duration	1 TTI
DRX In-Active duration	4 TTIs

Figure 2, shows systems throughput performance when the simulation is run for 5000 TTI. The results show that Best CQI (B-CQI) scheduler performed the best because it chooses the UEs, which have the best channel conditions in the uplink through CQI feedbacks. The PF scheduler performed the second best in this regard because it tries to balance the system throughput with the fairness. The Proposed Scheme performed not as good as B-CQI and PF scheduler because it is not designed to maximize system throughput rather, it designed to provide good QoS. The three markers point to the time when the Proposed Schemes system throughput performance degraded significantly. The reason is the throughput of some UEs had started to go below the GBR limit due to bad channel condition, and the scheduler tried to compensate it by assigning more resources. The RR scheduler performed not so well, but its throughput is more stable than any other scheduler because it treats all the UEs equally regardless of their channel conditions or requirements.

Figure 3, shows the effect of number of users on packet delays for all four schedulers. The packet delay threshold for video is 100 ms according to LTE QCI otherwise the packet will be discarded. When the number of users increases, the most of the time UE switched off which result in packets start to get delayed. Figure 3, shows that RR performed best and Proposed Schemes performed second best. Both of these curves followed a linear pattern while the PF initially started well, but its performance degraded significantly after 20 ms duration. The B-CQI performed worst in terms of packet delays because it is designed to achieve maximum systems throughput thus it disregards fairness and delay constraints.

4. CONCLUSION

In this paper, we proposed a new QoS-aware Two-layer downlink scheduling algorithm for delay sensitive traffic (Video). QoS-aware Two-layer scheduling algorithm is divided into the streaming scheduling and packet sorting by introducing dynamic QoS-related factors, such as the packet urgency and fairness among the streams. Streaming scheduling determines the transmission order of the multi-streams in MCE. And then, packet sorting ensures the transmissions of more important packets in eNode Bs.

Combining classical SMC and reinforcement learning techniques, we have proposed what is, to the best of our knowledge, the first algorithm to solve the PMC problem in probabilistic nondeterministic systems by sampling methods. We have implemented the algorithm within a highly parallel version of the PRISM simulation framework. This allowed us to use the PRISM input language and its benchmarks. The Proposed Scheme endeavors to provide better QoS by decreasing packet delay, improve fairness among the UE and considering the QoS requirement of multimedia service. It has the capability to assure QoS in non-power saving environment. The Proposed Scheme is compared with the traditional schemes according to different QoS attributes through simulations. In a normal power environment, the Proposed Scheme performs well in terms of throughput among the UEs with acceptable packet delay.

In future work, a longer simulation environment will be used with multiple eNode Bs. The mobility effect on QoS will be evaluated by considering the handover procedure. The performance of Proposed Scheme will be examined with Deep Sleep mode of operation and its comparison with DRX Light Sleep mode.

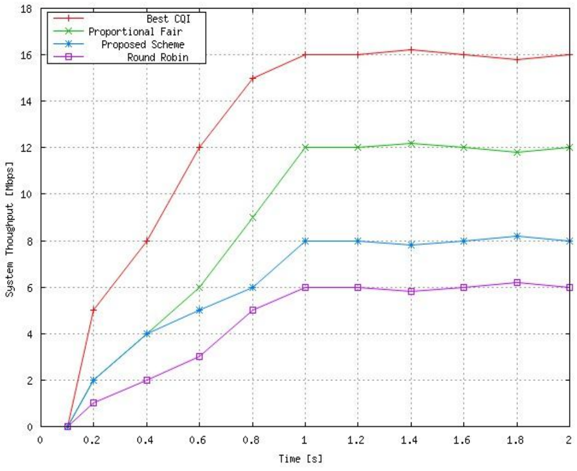

Fig. 2. Downlink System Throughput vs. Time

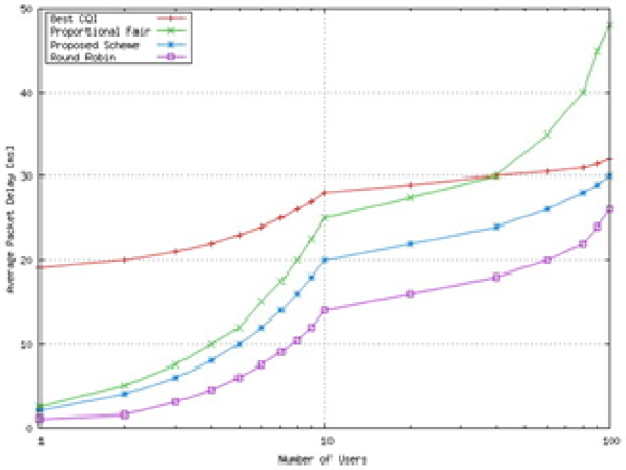

Fig. 3. Average Packet Delay vs. Number of Users

References

[1] M. Andrews, Probabilistic End-to-End Delay Bounds for Earliest Deadline First Scheduling, in Proc. IEEE INFOCOM, March 2000.

[2] G. Liebl, K. Kalman, and B. Girod, Deadline-Aware Scheduling for Wireless Video Streaming, in Proc. IEEE ICME, April 2005.

[3] P. V. Pahalawatta, R. Berry, etc., Content-aware Resource Allocation and Packet Scheduling for Video Transmission over Wireless Networks, IEEE J.Select. Areas Commun., vol. 25, no. 4, pp. 749V759, May 2007.

[4] P. Hosein and T. Gopal, Radio Resource Management for Broadcast Services in OFDMA-Based Networks, in Proc. IEEE ICC, pp.271-275, May 2008.

[5] Y. Chen, Statistical Multiplexing for LTE MBMS in Dynamic Service Deployment, in Proceeding. IEEE VTC, pp.2805-2809, May 2008.

[6] 3GPP TS 36.300, Evolved Universal Terrestrial Radio Access (EUTRA) and Evolved Universal Terrestrial Radio Access Network (EUTRAN); Overall description; Stage 2 (Release 10), v10.0.0, June 2010.

[7] Edmund M. Clarke Jr., Orna Grumberg, and Doron A. Peled. Model Checking. The MIT Press, 1999.

[8] Marta Z. Kwiatkowska, Gethin Norman, and David Parker. Prism 4.0: Verification of probabilistic real-time systems. In Ganesh Gopalakrishnan and Shaz Qadeer, editors, CAV, volume 6806 of Lecture Notes in Computer Science, pages 585-591. Springer, 2011.

[9] K. J. Astrom and B. Wittenmark, Computer Controlled Systems: Theory and Design, 3rd ed. Englewood Cliffs, NJ: Prentice-Hall, 1995.

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

[11] P. Svedam, S. Wilson, etc., A QoS-aware Proportional Fair Scheduling for Opportunistic OFDM, in Proc. IEEE VTC, pp.558-562, September 2004.

[12] Gilles Brassard and Paul Bratley. Algorithmics - theory and practice. Prentice Hall, 1988.

[13] David Henriques, Jo~ao Martins, Paolo Zuliani Andre Platzer, and Edmund Clarke. Statistical model checking for Markov decision processes. Technical Report CMU-CS-12-122, Computer Science Department, Carnegie Mellon University, 2011.

[14] Paolo Zuliani, Andr ‼ e Platzer, and Edmund M. Clarke. Bayesian statistical model checking with application to simulink/stateflow verification. In HSCC, pages 243-252, 2010.

AUTHOR

Tony Tsang (M’2000) received the BEng degree in Electronics & Electrical Engineering with First Class Honours in U.K., in 1992. He received the Ph.D from the La Trobe University (Australia) in 2000. He was awarded the La Trobe University Post-graduation Scholarship in 1998. Prior to joining the Hong Kong Polytechnic University, Dr. Tsang earned several years of teaching and researching experience in the Department of Computer Science and Computer Engineering, La Trobe University. He works in Hong Kong Polytechnic University as Lecturer since 2001. He has numerous publications in international journals and conferences and is a technical reviewer for several international journals and conferences. His research interests include mobile computing, networking, protocol engineering and formal methods. Dr. Tsang is a member of the ACM and the IEEE.