Randomization of Data Generation Times Improves Performance of Predictive IoT Networks

Mert Nakip
Institute of Theoretical and Applied Informatics
Polish Academy of Sciences, IITIS-PAN
44-100 Gliwice, Poland
ORCID: 0000-0002-6723-6494

Erol Gelenbe
Institute of Theoretical and Applied Informatics
Polish Academy of Sciences, IITIS-PAN
44-100 Gliwice, Poland, &
Lab. I3S, Université Côte d'Azur,
Grand Château, 06103 Nice Cecex 2, France
ORCID:0000-0001-9688-2201

***Abstract*—A challenge of IoT networks is to offer Quality of Service (QoS) and meet deadline requirements when packets from numerous IoT devices must be forwarded. Thus this paper introduces the Randomization of flow Generation Times (RGT) that smooths incoming IoT traffic so that QoS improves and packet loss is avoided. When the "Earliest Deadline First" (EDF) or "Priority based on Average Load" (PAL) scheduling algorithms are used, simulation results show that RGT provides significantly better performance, for a small additional computational cost at each device, providing a useful performance improvement for IoT networks.**

***Index Terms*—Internet of Things (IoT), Scheduling, Massive Access Problem, Predictive networks**

I. Introduction

The number of devices on the Internet is constantly growing, and expected to attain 30 billion by the year 2023 [1]. In addition, while the IoT is the key enabler for "smart cities" and Machine-to-Machine (M2M) communications [2], it is expected that the majority of devices on the Internet will be machine-type devices [3], while the usage of the IoT in smart cities has als expanded considerably [4] [5]. Moreover, it is estimated that 52% of all the IoT will be comprised of devices that fall in the Massive IoT segment by the end of 2025 [6] where a large number of low-cost devices will exist within the coverage area of a single base station or gateway, causing congestion due to Physical Random Access Channel (PRACH) overload [3] [7], also known as the "Massive Access Problem" (MAP).

Early work has addressed the MAP in various ways. In [8] adaptive routing was proposed when multiple network gateways and multiple paths are available to reduce congestion created by the MAP, while in [9] information theoretic techniques were suggested to selectively reduce IoT packet traffic. Other work [10] has suggested priority routing of IoT traffic near network access points, while adaptive random routing to reduce congestion was proposed in [11].

More recently, MAP was again addressed with reactive techniques [3], [7], [12]–[27] for IoT traffic, and it has been shown [28] that IoT traffic characteristics at the MAC-layer can be predicted with acceptable accuracy via machine-learning techniques for distinct IoT traffic classes.

Another trend of research [29]–[34] has suggested proactive network solutions, and other work has used the predictability of traffic generation patterns by IoT devices to address the MAP in a "predictive network" framework. To this end, a recent paper [35] proposed the Joint Forecasting-Scheduling (JFS) system and the Priority based on Average Load (PAL) scheduling algorithm which allocates available resources to IoT devices by forecasting the traffic generation pattern of each device for the case of single frequency channels.

In addition, in order to improve the performance of JFS and make it more applicable to longer time windows, a Multi-Scale Algorithm was proposed [36], and in [37] optimal scheduling and a Multi-Channel version of PAL was suggested to extend JFS for multi-frequency channels. The results in [35]–[37] showed that predictive networks with JFS are a promising for the solution to the MAP, and that scheduling heuristics are crucial when optimal scheduling policies cannot be implemented due to their high computational requirements in practical circumstances.

In this paper, an additional heuristic named "Randomization of Generation Times" (RGT) is proposed in order to improve performance and decrease the gap with respect to optimality scheduling for predictive networks. RGT can be implemented at each device and it is a novel pre-processing method that can be applied to any scheduling heuristic under the JFS system. In particular, in this paper RGT is applied to the PAL and to the EDF algorithm.

The simulation results presented here show that RGT improves the performance of each scheduling algorithm significantly, even when forecasting is conducted with high error, as well as in the case of zero-error perfect forecasting. Furthermore, an estimate of the computation time of RGT is shown to be under 2.5 μs per IoT device, and it increases linearly with the number of devices. Thus one can conclude that RGT is an important addition that can facilitate the practical usage of JFS.

The rest of this paper is organized as follows. Section II provides background on the scheduling of IoT traffic as well

This research has been supported by the European Commission H2020 Program under the IoTAC Research and Innovation Action, under Grant Agreement No. 952684.

as regarding the PAL and EDF algorithms. Section III proposes our RGT process. Section IV presents our results. Section V summarizes our main conclusions and sugget further research avenues.

II. Scheduling of IoT Traffic

The scheduling problem of the IoT device traffic is now introduced in predictive networks, startig with the JFS system proposed in [35] which is comprised of the Forecasting and Scheduling modules. The performance of the system for perfect forecasting and for the increasing forecasting error are discussed in Section IV. Between the Forecasting and the Scheduling modules, a module that operates the RGT process is now described.

First, define each burst j of any device i by the triple (r_j, d_j, a_j) where r_j is the instant when it is generated, its deadline is d_j and the number of bits is forwards in a particular burtst is a_j. In addition, if c_j denotes the maximum number of bits that can be transmitted by burst j in a single MAC-layer slot, and p_j is the total number of slots p_j needed to transmit burst j, one has:

$$c_j = R_i.\tau_{\text{MAC}},\ p_j = \lceil \frac{a_j}{c_j} \rceil, \tag{1}$$

where R_i is the data rate of device i which generated burst j, and τ_{MAC} denotes the duration of a MAC-layer slot, while each burst j has a strict delay constraint $\Delta_j = d_j - r_j$. Furthermore, let N denote the total number of devices in the coverage area of IoT gateway and $\mathcal{J}$ denote the set of bursts that are generated by those devices.

Next, in the JFS system, based on the output of the Forecasting module, the Scheduling module allocates the available resources to IoT devices for a given upcoming time interval, namely the scheduling window. That is, the inputs of the Scheduling module is the future traffic generation pattern and its output is the binary schedule matrix $\mathbf{S}$, where $\mathbf{S}(j,m) = 1$ if MAC-layer slot m is allocated for burst j and $\mathbf{S}(j,m) = 0$ otherwise. Also, let u_j be a binary variable which equals 1 if burst j is successfully transmitted, and 0 otherwise.

Although a variety of scheduling algorithms can be used in the Scheduling module of JFS, in this paper, only PAL and the EDF are considered. Note that EDF is optimal [38], [39] in the sense that it will minimize the number of bursts whose deadline cannot be met. PAL aims to maximize the total number of bits in transmitted bursts, while EDF aims to maximize the total number of successfully transmitted bursts without considering the number of bits in each burst.

A. Priority based on Average Load (PAL)

The greedy scheduling algorithm PAL was proposed in [35] for resource allocation in predictive networks, and especially for the JSF system, and it schedules the bursts by prioritizing each burst j based on its average load over Δ_j. For each time slot m, define the set of "active bursts" as follows:

$$\mathcal{J}_{\text{active}}[m] = \{j :\ r_j \le m \le d_j \text{ and } p_j \le d_j - m\}, \tag{2}$$

and the average load per burst:

$$\alpha_j = \frac{a_j}{\Delta_j}. \tag{3}$$

PAL schedules the bursts by prioritizing each burst j based on its average load over the duration Δ_j, and works as follows starting for $m = 1$:

- It computes the burst j^* with the largest average load:

$$j^* = \arg \max_{j \in \mathcal{J}_{\text{active}}[m]} \alpha_j\ . \tag{4}$$

- Then for j^*, PAL allocates the upcoming p_{j^*} slots starting with current slot m.
- Finally, it updates $m \leftarrow m + p_{j^*}$ and updates $\mathcal{J}_{\text{active}}[m]$ accordingly.

As the definition of PAL suggests, it is a non-preemptive algorithm, which means that it completes the transmission of each burst j which has been started, i.e., when the transmission of a burst has started, PAL waits for the completion of the transmission before it schedules another burst.

B. Earliest Deadline First (EDF)

In this paper, a non-preemptive EDF algorithm is used, which is optimal to maximize the total number of successfully delivered bursts for our scheduling problem [40]. In short, the EDF algorithm works as follows: first sort the bursts in $\mathcal{J}$ with respect to d_j's into vector J_{sorted}. Then, for each j in J_{sorted} starting with the first burst, if there are enough available slots to transmit j, EDF reserves the first p_j available slots between r_j and d_j.

C. Performance Metrics

Throughout this paper, in order to measure the network performance, the following metrics are defined. First, η denotes the cross-layer network throughput and is defined as

$$\eta \equiv \frac{\sum_{j \in \mathcal{J}} u_j a_j}{\sum_{j \in \mathcal{J}} a_j} \tag{5}$$

Second, ζ denotes the fraction of the bursts that are successfully delivered, which is defined as

$$\zeta \equiv \frac{\sum_{j \in \mathcal{J}} u_j}{|\mathcal{J}|} \tag{6}$$

Thirdly, $\mathcal{E}$ denotes the transmit energy consumption per successfully delivered bit, and it is assumed that one unit of energy is consumed by an IoT device for each time slot at which the IoT device continues its transmission. Thus, $\mathcal{E}$ is defined as

$$\mathcal{E} \equiv \frac{\sum_{j \in \mathcal{J}} u_j p_j}{\sum_{j \in \mathcal{J}} u_j a_j} \tag{7}$$

III. Randomization of the Generation Times

Since the heuristic algorithms are fast, inexpensive with respect to the computational hardware requirements and able to achieve relatively high QoS, they are promising for the scheduling of the IoT traffic in the predictive networks. Thus, to improve the performance of the heuristic scheduling algorithms. To this end, in this section, Randomization of the Generation Times (RGT) is proposed as a way to relieve the system by distributing the traffic generation instants over the scheduling window with duration of T_{sch}. Thus the generation time, r_j of each burst j is updated by adding to it an offset which is a uniformly distributed random variable $U[\Delta_j - S_j] \in [0, \Delta_j - S_j]$:

$$r_j^{new} \leftarrow r_j + U[\Delta_j - S_j] \tag{8}$$

Here S_j is a *safety delay* that limits the upper bound of r_j^{new} such that $0 \leq S_j \leq \Delta_j$ which indicates the maximum randomization and $S_j = \Delta_j$ indicates that there is no randomization. In order to determine the value of S_j, One can use one of the following approaches:

- (M1) One can search for the best value of S_j; however, this will be expensive in computation time and not suitable for real-time implementation. On the other, one may set a loose upper bound for the performances of EDF and PAL algorithms under RGT.
- (M2) One may estimate the value of S_j while aiming to achieve the close to the performance of the upper bound. Even though the performance of the scheduling under the estimation of S_j might be inferior to the upper bound, it is fast and cheap in computation, and is therefore suitable for real-time applications.

A. Search for S_j to Achieve the Performance Upper Bound

To determine the best value of S_j so as to approach the upper bound for scheduling the performance using RGT, one can search for the best value for the fraction γ of the range $[0, \Delta_j]$. Thus for each $j \in \mathcal{J}$, one first defines $S_j = \gamma\Delta_j$, and then one searches for γ in range $[0, 1]$ as follows:

1) Set $\gamma = 0$.
2) Set $S_j = \gamma\Delta_j \qquad \forall j \in \mathcal{J}$
3) Update r_j for each j according to (8)
4) Schedule IoT traffic for the updated r_j's
5) Compute the performance metric (e.g. η, ζ or $\mathcal{E}$) and save that in a vector
6) Update $\gamma \leftarrow \gamma + 0.05$
7) If $\gamma \leq 1$, return to step 2); else, continue.
8) Find the best value of γ which leads to the best values of S_j's to maximize performance metric

Note that in this search, η, ζ and $\mathcal{E}$ are used as the performance metrics.

B. Estimation of S_j using Queueing Theory

S_j can also be estimated based on queueing theory [41], with the simple assumption that the successive instants of the bursts from the devices constitute a Poisson process, whose arrival rate λ is simply the total number of bursts $|\mathcal{J}|$ divided by the scheduling window expressed in number of slots:

$$\lambda = \frac{|\mathcal{J}|}{\frac{T_{sch}}{\tau_{mac}}} = \frac{|\mathcal{J}|.\tau_{mac}}{T_{sch}}, \tag{9}$$

and assuming exponentially distributed service times, where the average service time $E[S]$ is the average of the burst lengths of all the bursts, in number of slots:

$$E[S] = \frac{\sum_{j\in\mathcal{J}} p_j}{|\mathcal{J}|} \ . \tag{10}$$

The traffic intensity ρ is then:

$$\rho = \lambda E[S] = \frac{\tau_{mac}\sum_{j\in\mathcal{J}} p_j}{T_{sch}} \ . \tag{11}$$

If the system is not saturated, i.e. $\rho = \lambda E[S] < 1$, the average waiting time W to access the channel for any burst, excluding the channel service time for the burst, will be:

$$E[W] = \frac{\rho}{\frac{1}{E[S]} - \lambda} = \frac{E[S]}{\frac{1}{\rho} - 1} \ . \tag{12}$$

Thus for any burst j one has:

$$S_j \approx p_j + \frac{E[S]}{\frac{1}{\rho} - 1} \ , \tag{13}$$

where the term $p_j \geq 0$ is added to insure that the burst may forward all its data when it accesses the channel. Of course, S_j should in no case exceed Δ_j, so that one must take:

$$S_j \approx min[\ p_j + \frac{E[S]}{\frac{1}{\rho} - 1},\ \Delta_j\ ] \tag{14}$$

IV. Results

In this section, the performance of each of the R-PAL and R-EDF schedulers is evaluated using the IoT dataset [42] which described in [37]. This data is comprised of the bootstrapped traffic generation patterns of real IoT devices whose traffic output belongs to one of the following traffic classes: Fixed Bit Periodic, Fixed Bit Aperiodic, Variable Bit Periodic, and Variable Bit Aperiodic.

In addition, for each burst j of each bootstrapped device i, the deadline takes one of the following six values $\Delta_j \in \{0.5,\ 1,\ 2,\ 180,\ 600,\ 3600\ (in\ seconds)\}$.

On this dataset, the performance of the scheduling techniques is measured for $N = 12$ devices as well as for integer values in the range$[400 \leq N \leq 6400]$ with increments of 400 devices. For each N, one can randomly select $N/4$ devices from each device traffic class. Thus in our simulations, the set of devices from each device class is composed of 25% of all devices that are simulated.

Also, in order to increase the load of the system and evaluate the algorithms on a highly loaded network, for each device i in this dataset, one may decrease the data rate R_i by 30% (i.e. $R_i \leftarrow 0.7R_i$). Furthermore, $\tau_{\text{MAC}} = 100$ ms was selected and the simulations were performed for scheduling windows with a duration of 900 seconds.

A. Performance Comparison of Scheduling Algorithms under Perfect Forecasting

The performance of each of the R-PAL and R-EDF algorithms and their comparison with PAL and EDF, and their upper bounds with respect to each of η, ζ and $\mathcal{E}$ are now presented. In Figure 1, one sees that the PAL-based scheduling algorithms significantly outperform the EDF-based algorithms for $N > 4800$ devices. The reason is that PAL aims to maximize the total number of bits in successfully delivered bursts while EDF aims to maximize the total number of successfully delivered bursts as described in Section II. one also sees that both R-PAL and R-EDF are able to achieve their upper bounds.

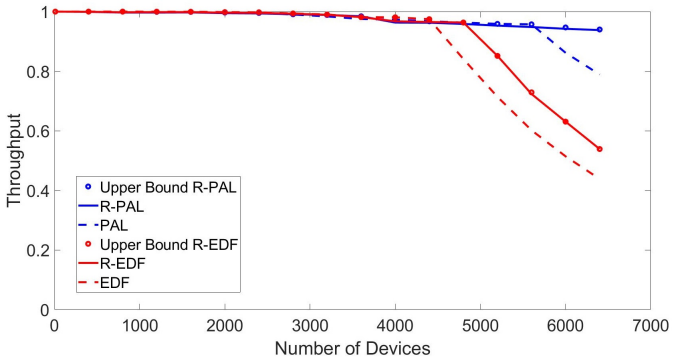

Fig. 1. Comparison of the Upper Bound R-PAL, R-PAL, PAL and the Upper Bound R-EDF, R-EDF, EDF algorithms with respect to throughput η for 12 to 6400 devices

Furthermore, within the PAL-base algorithms in Figure 1, the R-PAL outperforms the PAL for $N > 5400$ devices and the throughput difference between R-PAL and PAL increases with N. As explained in Section II-A, the PAL algorithm is a greedy algorithm that schedules the job with respect to the generation times. This is why the RGT process improves the performance of the PAL significantly (about 0.15 for $N = 6400$ devices). Within the EDF-base algorithms (Upper Bound R-EDF, R-EDF and EDF) in Figure 1, one sees that the R-EDF outperforms the EDF for $N > 4400$ devices, so that the RGT process significantly improves the performance of EDF for $N > 4400$.

In Figure 2, one sees that the EDF-base algorithms are able to schedule more bursts for successful transmission than PAL-based algorithms. The reason is that the EDF aims to schedule maximum number of bursts without considering the number of bits that is carried by that burst. On the other hand, PAL aims to maximize the total delivered bits. This shows that the EDF is a more fair algorithm than PAL throughout the devices in the network since its values all burst equally. In this figure, one may also notice that the ζ performance of the R-EDF and R-PAL are comparable with EDF and PAL respectively.

In Figure 3, it is seen that each device consumes more energy to transmit a single bit under EDF-based algorithms than that under PAL-based algorithms. The reason is that PAL schedules bursts with a larger number of bits; in other words,

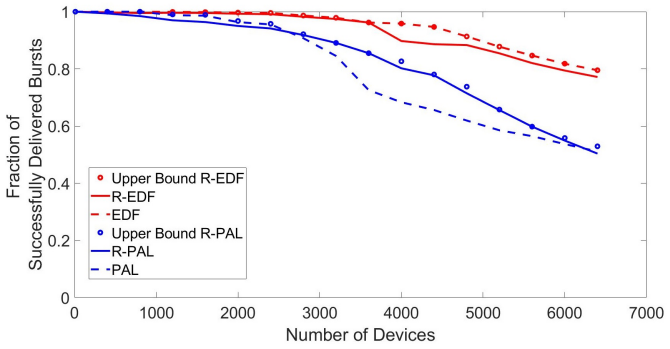

Fig. 2. Comparison of the Upper Bound R-PAL, R-PAL, PAL and the Upper Bound R-EDF, R-EDF, EDF algorithms with respect to the fraction of the successfully delivered bursts (ζ) for 12 to 6400 devices

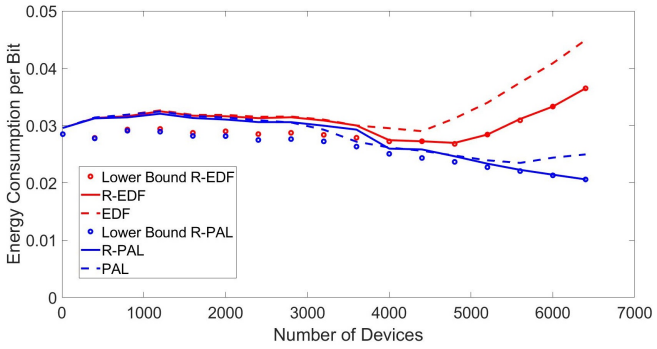

Fig. 3. Comparison of the Lower Bound R-PAL, R-PAL, PAL and the Lower Bound R-EDF, R-EDF, EDF algorithms with respect to the energy consumption per bit in delivered bursts ($\mathcal{E}$) for 12 to 6400 devices

it maximizes the denominator of (7). Moreover, our results in Figure 3 shows that the RGT process significantly decreases the energy consumption of both the PAL and EDF algorithms.

B. Sensitivity of Scheduling Algorithms to Forecasting Error

One may evaluate the performance of the scheduling algorithms for the increasing forecasting errors in the predictive network. To this end, the forecasting error can be modelled as the realization of a normally distributed random variable[1]. For each burst j, the forecasted number of bits is given by $\hat{a}_j = a_j + n$, where n is the realization of a Normally Distributed random variable with zero mean and variance σ. Thus in this subsection,the network performance is analyzed against the increasing value of forcarsting error via the parameter σ.

In Figure 4, the network throughput η is shown for increasing values of σ, i.e. as the forecasting error increases, with $N = 6400$ devices. One sees that the RGT process significantly improves the network throughput performance of each of both the PAL and EDF algorithms. That is, even while the forecasting error is very high compared to the maximum number of bits ($\sigma = 60$ and $\max_{j \in \mathcal{J}} a_j = 128$),

[1]During the sensitivity analysis, in order to control the forecasting error, one may assume that a linear forecasting model minimizes the mean squared error, assuming that the traffic generation pattern is Normally Distributed. Thus, one may also assume that the forecasting error is also Normally Distributed.

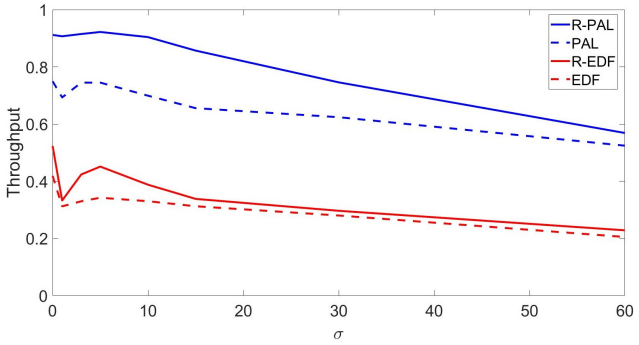

Fig. 4. Comparison of the R-PAL, PAL and R-EDF, EDF algorithms with respect to throughput (η) for $\sigma \in \{0, \ldots, 60\}$ for $N = 6400$ devices

RGT impacts the scheduling performance. On the other hand, the performance difference between R-PAL and PAL as well as the difference between R-EDF and EDF decreases as the forecasting error increases.

C. Computation Time

The computation time of the RGT process as the number of devices is increased is measured as the corresponding computation time using MATLAB on a Laptop with Intel Core i7-10750H cpu and 16 GB ram.

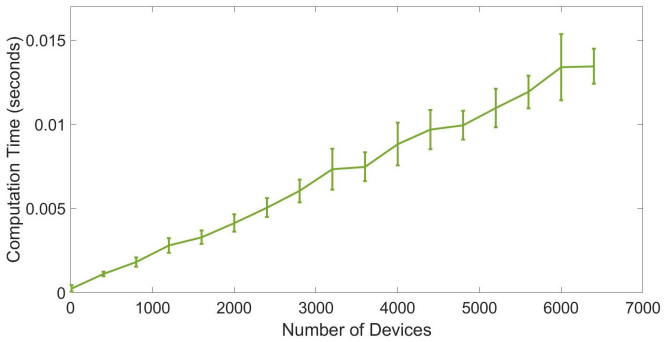

Fig. 5. The mean and the standard deviation of the computation time of the RGT process over 100 runs for 12 to 6400 devices

In Figure 5, the mean of the computation time is shown with a standard deviation bar for each N over 100 simulation runs. In this figure, one notices the computation time of RGT is under 0.015 seconds for all values of N and increases linearly with N. Thus, the computation time cost of RGT for the scheduling algorithm is very small but the performance improvement is relatively high.

V. Conclusion

The MAP arises when a large number of IoT devices can access a single mobile gateway or WI-FI device, leading to network congestion at the entry points of IoT networks, leading to numerous deadlines being missed at the receivers of the IoT data. Thus to address this issue, forecasting algrithms have been used to understand the actual arrival instants and patterns of the data bursts generated by IoT devices. In addition, the resulting sequences of data bursts identified by forecasting techniques can then be exploited to create schedulers that insure that the deadlines are met as much as possible.

In this paper, an additional tecnique to improve the overall system performance is proposed based on randomization. Indeed, it is shown that if the arrival stream of data bursts from IoT devices can be smoothed via randomization, then a higher level of performance can be attained, with fewer deadlines being missed. Thus the RGT technique is proposed for scheduling heuristics that allocate resources in a predictive IoT network. RGT can be implemented together with any scheduling heuristic to smoothly distribute the load of the network over a scheduling window. One may also use queueing analysis to fine tune RGT so that the randomization does not encroach on the inevitable queueing delay to the gateway.

The performance of the RGT process has been evaluated extensively for two known scheduling heuristics: PAL and EDF algorithms for an IoT network with up to 6400 devices accessing a single gateway. The results show that the RGT process significantly improves the performance of both PAL and EDF heuristics in terms of the QoS metrics (throughput and energy consumption) while the fairness of each heuristic remains almost the same. In addition, it was shown that the computation time of RGT is under 15 ms for 6400 devices and increases linearly with the number of devices. That is, the RGT is a fast and effective preprocessing algorithm that significantly improves the performance of the scheduling algorithms for IoT. Thus, since RGT enables to achieve much higher performances via fast heuristics, it will pave the way to include the devices with much lower delay constraint into predictive IoT networks.

Future work will extend the approach to other scheduling heuristics, and it will be useful to examine access points which use a random access channel. The advantages of multi-path access networks to reduce the MAP with RGT will also be studied.

References

[1] Cisco, *Cisco Annual Internet Report (2018–2023)*, Mar. 2020. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html

[2] M. Hasan, E. Hossain, and D. Niyato, "Random access for machine-to-machine communication in LTE-advanced networks: issues and approaches," *IEEE communications Magazine*, vol. 51, no. 6, pp. 86–93, 2013.

[3] F. Ghavimi and H.-H. Chen, "M2M communications in 3GPP LTE/LTE-A networks: Architectures, service requirements, challenges, and applications," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 2, pp. 525–549, 2015.

[4] O. Bello and S. Zeadally, "Toward efficient smartification of the Internet of Things (IoT) services," *Future Generation Computer Systems*, vol. 92, pp. 663–673, 2019.

[5] G. Fortino, W. Russo, C. Savaglio, M. Viroli, and M. Zhou, "Modeling opportunistic IoT services in open IoT ecosystems." in *WOA*, 2017, pp. 90–95.

[6] Ericsson, *Ericsson Mobility Report*, Nov. 2019. [Online]. Available: https://www.ericsson.com/en/mobility-report

[7] A. Zanella, M. Zorzi, A. F. dos Santos, P. Popovski, N. Pratas, C. Stefanovic, A. Dekorsy, C. Bockelmann, B. Busropan, and T. A. Norp, "M2M massive wireless access: Challenges, research issues, and ways forward," in *2013 IEEE Globecom Workshops*. IEEE, 2013, pp. 151–156.

[8] E. Gelenbe and E. C.-H. Ngai, "Adaptive qos routing for significant events in wireless sensor networks," in *2008 5th IEEE International Conference on Mobile Ad Hoc and Sensor Systems*. IEEE, 2008, pp. 410–415.

[9] E. C.-H. Ngai, E. Gelenbe, and G. Humber, "Information-aware traffic reduction for wireless sensor networks," in *2009 IEEE 34th Conference on Local Computer Networks*. IEEE, 2009, pp. 451–458.

[10] E. Gelenbe, E. Ngai, and P. Yadav, "Routing of high-priority packets in wireless sensor networks," in *IEEE Second International Conference on Computer and Network Technology, IEEE*, 2010.

[11] E. Gelenbe and E. Ngai, "Adaptive random re-routing for differentiated qos in sensor networks," *The Computer Journal*, vol. 53, no. 7, pp. 1052–1061, 2010.

[12] L. Liang, L. Xu, B. Cao, and Y. Jia, "A cluster-based congestion-mitigating access scheme for massive M2M communications in Internet of Things," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 2200–2211, 2018.

[13] Z. Alavikia and A. Ghasemi, "Collision-aware resource access scheme for LTE-based machine-to-machine communications," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 5, pp. 4683–4688, 2018.

[14] L. Tello-Oquendo, D. Pacheco-Paramo, V. Pla, and J. Martinez-Bauset, "Reinforcement learning-based ACB in LTE-A networks for handling massive M2M and H2H communications," in *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–7.

[15] N. Shahin, R. Ali, and Y.-T. Kim, "Hybrid slotted-CSMA/CA-TDMA for efficient massive registration of IoT devices," *IEEE Access*, vol. 6, pp. 18 366–18 382, 2018.

[16] L. Tello-Oquendo, I. Leyva-Mayorga, V. Pla, J. Martinez-Bauset, J.-R. Vidal, V. Casares-Giner, and L. Guijarro, "Performance analysis and optimal access class barring parameter configuration in LTE-A networks with massive M2M traffic," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 4, pp. 3505–3520, 2018.

[17] H. Jin, W. T. Toor, B. C. Jung, and J.-B. Seo, "Recursive pseudo-Bayesian access class barring for M2M communications in LTE systems," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 9, pp. 8595–8599, 2017.

[18] J. Liu, L. Song *et al.*, "A novel congestion reduction scheme for massive machine-to-machine communication," *IEEE Access*, vol. 5, pp. 18 765–18 777, 2017.

[19] M. Shirvanimoghaddam, M. Dohler, and S. J. Johnson, "Massive non-orthogonal multiple access for cellular IoT: Potentials and limitations," *IEEE Communications Magazine*, vol. 55, no. 9, pp. 55–61, 2017.

[20] I. Park, D. Kim, and D. Har, "MAC achieving low latency and energy efficiency in hierarchical M2M networks with clustered nodes," *IEEE Sensors Journal*, vol. 15, no. 3, pp. 1657–1661, 2015.

[21] P. Si, J. Yang, S. Chen, and H. Xi, "Adaptive massive access management for QoS guarantees in M2M communications," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 7, pp. 3152–3166, 2015.

[22] A. Aijaz and A. H. Aghvami, "Cognitive machine-to-machine communications for Internet-of-Things: A protocol stack perspective," *IEEE Internet of Things Journal*, vol. 2, no. 2, pp. 103–112, 2015.

[23] Y. Liu, C. Yuen, X. Cao, N. U. Hassan, and J. Chen, "Design of a scalable hybrid MAC protocol for heterogeneous M2M networks," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 99–111, 2014.

[24] Y.-C. Pang, S.-L. Chao, G.-Y. Lin, and H.-Y. Wei, "Network access for M2M/H2H hybrid systems: A game theoretic approach," *IEEE Communications Letters*, vol. 18, no. 5, pp. 845–848, 2014.

[25] T.-M. Lin, C.-H. Lee, J.-P. Cheng, and W.-T. Chen, "PRADA: Prioritized random access with dynamic access barring for MTC in 3GPP LTE-A networks," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 5, pp. 2467–2472, 2014.

[26] A. Aijaz, S. Ping, M. R. Akhavan, and A.-H. Aghvami, "CRB-MAC: A receiver-based MAC protocol for cognitive radio equipped smart grid sensor networks," *IEEE Sensors Journal*, vol. 14, no. 12, pp. 4325–4333, 2014.

[27] S.-Y. Lien, T.-H. Liau, C.-Y. Kao, and K.-C. Chen, "Cooperative access class barring for machine-to-machine communications," *IEEE Transactions on Wireless Communications*, vol. 11, no. 1, pp. 27–32, 2012.

[28] M. Nakip, B. C. Gül, V. Rodoplu, and C. Güzeliş, "Comparative study of forecasting schemes for IoT device traffic in machine-to-machine communication," in *Proceedings of the 2019 4th International Conference on Cloud Computing and Internet of Things*, 2019, pp. 102–109.

[29] Y. Edalat, J.-S. Ahn, and K. Obraczka, "Smart experts for network state estimation," *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 622–635, 2016.

[30] V. Petkov and K. Obraczka, "Collision-free medium access based on traffic forecasting," in *2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*. IEEE, 2012, pp. 1–9.

[31] ——, "The case for using traffic forecasting in schedule-based channel access," in *2011 IEEE Consumer Communications and Networking Conference (CCNC)*. IEEE, 2011, pp. 208–212.

[32] D. Raca, A. H. Zahran, C. J. Sreenan, R. K. Sinha, E. Halepovic, R. Jana, and V. Gopalakrishnan, "On leveraging machine and deep learning for throughput prediction in cellular networks: Design, performance, and challenges," *IEEE Communications Magazine*, vol. 58, no. 3, pp. 11–17, 2020.

[33] L. Ruan, M. P. I. Dias, and E. Wong, "Machine learning-based bandwidth prediction for low-latency H2M applications," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 3743–3752, 2019.

[34] H.-Y. Kim and J.-M. Kim, "A load balancing scheme based on deep-learning in IoT," *Cluster Computing*, vol. 20, no. 1, pp. 873–878, 2017.

[35] M. Nakip, V. Rodoplu, C. Güzeliş, and D. T. Eliiyi, "Joint forecasting-scheduling for the Internet of Things," in *2019 IEEE Global Conference on Internet of Things (GCIoT)*. IEEE, 2019, pp. 1–7.

[36] V. Rodoplu, M. Nakıp, D. T. Eliiyi, and C. Güzelis, "A multi-scale algorithm for joint forecasting-scheduling to solve the massive access problem of IoT," *IEEE Internet of Things Journal*, 2020.

[37] V. Rodoplu, M. Nakip, R. Qorbanian, and D. T. Eliiyi, "Multi-channel joint forecasting-scheduling for the Internet of Things," *IEEE Access*, vol. 8, pp. 217 324–217 354, 2020.

[38] C. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment," *Journal of the ACM*, vol. 20, no. 1, pp. 46–61, 1973. [Online]. Available: https://doi.org/10.1145/321738.321743

[39] E. L. Lawler and J. Labetoulle, "On preemptive scheduling of unrelated parallel processors by linear programming," *Journal of the ACM*, vol. 25, no. 4, pp. 612–619, October.

[40] L. George, N. Rivierre, and M. Spuri, "Preemptive and non-preemptive real-time uniprocessor scheduling," 1996.

[41] E. Gelenbe and G. Pujolle, "Introduction to Networks of Queues," *J. Wiley & Sons, Chichester*, 1998.

[42] "IoT Traffic Generation Pattern Dataset," Jan 2021. [Online]. Available: https://www.kaggle.com/tubitak1001118e277/iot-traffic-generation-patterns