Journal of Computer and Communications, 2020, 8, 28-40
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Steady-State Analysis of the Distributed Queueing Algorithm in a Single-Channel M2M Network

Romeo Nibitanga[1], Elijah Mwangi[2], Edward Ndung'u[3]

[1]Department of Electrical Engineering, Pan African University Institute of Basic Sciences, Technology and Innovation, Nairobi, Kenya
[2]Department of Electrical and Information Engineering, University of Nairobi, Nairobi, Kenya
[3]Department of Telecommunication and Information Engineering, Jomo Kenyatta University of Agriculture and Technology, Juja, Kenya
Email: nibitanga.romeo@students.jkuat.ac.ke, elijah.mwangi@uonbi.ac.ke, ndunguen@jkuat.ac.ke

How to cite this paper: Nibitanga, R., Mwangi, E. and Ndung'u, E. (2020) Steady-State Analysis of the Distributed Queueing Algorithm in a Single-Channel M2M Network. *Journal of Computer and Communications*, **8**, 28-40.
https://doi.org/10.4236/jcc.2020.89003

Received: August 3, 2020
Accepted: September 1, 2020
Published: September 4, 2020

Abstract

The Distributed Queuing (DQ) algorithm is predicted as one of the solutions to the issues currently found in IoT networks over the use of Aloha based algorithms. Since recently, the algorithm has been of interest to many IoT researchers as a replacement of those Aloha variants for channel access. However, previous works analyzed and evaluated the DQ algorithm without any consideration of the stability of its queues, assuming it is stable for any given number of nodes in the network. In this paper, we define the DQ stability condition in a single-channel M2M environment considering a traffic model of periodic and urgent frames from each node in the network. Besides, a steady-state evaluation of the algorithm's performance metrics is also presented. In general, the DQ algorithm, when it is stable, was observed not to efficiently use the contention slots for the collision resolution. In a single-channel environment, the DQ algorithm is found to outperform the Aloha based algorithms only in an idle-to-saturation scenario.

Keywords

Aloha, Collision Resolution, Distributed Queueing, IoT Networks, M2M Communications, Stability Condition

1. Introduction

Massive connectivity is certainly one of the critical challenges in the deployment of the Internet of Things (IoT) networks [1], and specifically for the Low Power

Wide Area Networks (LPWAN) [2]. In order to handle the multitude number of connections in these networks, a Distributed Queuing (DQ) algorithm has been proposed as the key solution to the issues of the currently used Aloha based protocols. The basic principle of the DQ algorithm is to divide an initial group of contending sensors into virtual queues before trying to resolve another contention. Several works have analyzed the DQ algorithm in a massive Machine-to-Machine (M2M) communication environment [3]-[11]. They focused mainly on the evaluation of the algorithm performance metrics such as the throughput, the access delay and energy consumption under a given technology at the physical layer like Long Term Evolution (LTE) [3] [4] [5] [6] [7], Narrow Band Internet of Things (NB-IoT) [8], Long Range (LoRa) [9], crowd sourced networks [10] or Radio Frequency Identification (RFID) networks [11]. However, most studies fail to give any condition of the stability for the DQ system or do not clarify how frames are processed after the collision resolution is finished. Consequently, the algorithm is assumed stable for any given number of contending sensors in the network.

By stability of the DQ algorithm, we aim to define the maximum number of sensors or other nodes allowed contending in an IoT network in the long run, so that the DQ algorithm queues do not grow without bound over time each new contention. The DQ algorithm is stable in the sense that a sensor does not wait for an indefinite time in any of the queues [12]. In this paper, we present a derivation of the stability condition for the DQ algorithm in a single-channel IoT setting. Moreover, a steady-state performance analysis of the DQ protocol, under the stability condition, is also evaluated. We validate the analytical results with numerical simulations based on a discrete event model developed and executed in Matlab.

In general, the heterogeneity of data from M2M networks have led to a diversity of traffic models in the literature [13]-[18]. Some applications are characterized by traffic that is triggered by external events, while others are defined by frames generated at a regular time interval. Most of the time, a sensor generates both types of traffic; however, depending on the application, one type of traffic will be dominant over the other. For example, smart grid applications (e.g., smart meters) are well described by periodic traffic whereas smart home applications (e.g., motion detectors) are modelled by event-driven traffic. Therefore, in an M2M communication environment, the network traffic is a mix of a large number of sources with different periods and rates. In this paper, in the long run, the superposition of traffic processes from the sensors is aggregated into a Poisson arrival process. In [13], such an approximation has been shown to result in a small bias when the number of nodes in the network is sufficiently large and when the traffic is not purely homogeneous.

In [12] [19], the authors analyzed the condition for which the DQ algorithm is stable, considering a Poisson distributed input traffic. The Contention Resolution Queue (CRQ) was modelled using Markov chain theory whereas the Data

Transmission Queue (DTQ) was modelled as G/D/1 queue. They found that the CRQ was stable for any given input rate even greater than unity as long as the average contention time was less than the average length of the enable transmission time. Hence, the stability of the whole system was only defined by the DTQ. In general, the DQ algorithm was stable when the traffic intensity was less than unity. In this paper, in contrast to those studies, we consider a source traffic model where each sensor in the network generates periodic frames at a regular time interval and urgent frames following a Poisson arrival process with a given rate parameter. Moreover, a performance analysis of the DQ algorithm for both types of frames is also proposed.

The rest of this paper is organized as follows: in Section 2, the system model, together with a brief description of the DQ algorithm, are given. In Section 3, we present the analytical derivation of the stability condition for the DQ algorithm for a network with several applications. Section 4 is dedicated to the steady-state performance analysis of the DQ algorithm, and both analytical and numerical results are presented. Lastly, in Section 5, we give a conclusion and present our future work where a steady-state performance analysis of the DQ algorithm in a multichannel environment will be of interest.

2. System Model and Algorithm Description

In this section, we describe the system model used to define the stability condition for the DQ algorithm and the evaluation of its steady-state performance metrics. We also present a brief description of the DQ algorithm.

The system model is a star network topology comprised of a base station and n sensors. The sensors are in the vicinity of the base station and can communicate with the network coordinator. We assume that N applications may exist in the network leading to heterogeneous traffic. However, a sensor can only belong to one application. Sensors from the same application, exhibit similar traffic characteristics. Thus, a sensor may be in three different states: normal, alarm, and off. In the normal state, a sensor generates periodic frames at regular ith application time interval T_i (with $i = 1, 2, \cdots, N$). In contrast, in the alarm state, it generates urgent frames following a Poisson arrival process with an ith application rate of λ_i. In the off state, the sensor is in sleep mode and does not generate any frames.

Sensors contend for access to the wireless channel following the rules of the DQ algorithm. The DQ protocol is a tree-splitting algorithm used for channel access. It virtually divides the contending sensors into two different queues. These are the collision resolution queue (CRQ) and the data transmission queue (DTQ). The first queue contains sensors that have not secured a place in the DTQ, and the second queue is for the sensors that are waiting their turn to transmit their data. A DQ frame is divided into two parts (**Figure 1**): first, the up-link channel divided into several contention slots and one or multiple data slots and second, the downlink channel formed by one or multiple feedback data

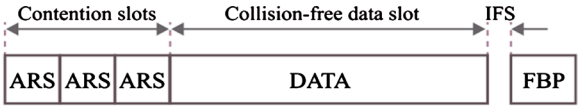

Figure 1. DQ frame structure.

slots. At the end of DQ frame, active sensors know their position in both the CRQ and the DTQ. A detailed description of the DQ algorithm can be found in [19]. In this paper, we chose to use "DQ slot" instead of "DQ frame" to distinguish between a frame from a sensor and the DQ frame.

At the beginning of the contention process, the CRQ and the DTQ are empty. Besides, there are no sensors scheduled for channel access or data transmission. We only consider the up-link channel in our analysis, because the downlink channel is reserved for the base station and is contention-free. Sensors have a perfect slot and contention slot synchronization. Thus, they can contend either in a synchronized or unsynchronized manner per application. In the synchronized scenario, sensors from the same application are scheduled to contend during the same slot at a regular time interval. However, in a massive M2M network, a synchronized scenario is not easy to achieve across a large number of sensors and could lead to a more complex sensor. Therefore, we assume that sensors contend asynchronously. On its incorporation into the network, a sensor from an ith application chooses randomly a slot j for $j = 1, 2, \cdots, T_i$ and regularly generates periodic frames at $j + T_i$. k slots for $k = 0, 1, 2, \cdots$. Moreover, each frame is assumed to have a timeout period after which it is dropped. A frame is dropped if a new frame is generated before it is sent. The type of the frame defines the timeout period: periodic or urgent. We assume that a sensor requires only one frame for its data transmission. It should also be noted that no sensors enter the CRQ before it is emptied.

In the long run, following the fundamental Palm-Khintchine theorem [20], [13], the aggregated traffic generated from n sensors in a network with N applications can be modelled as a Poisson arrival process with an overall parameter λ_{tot}.

Theorem 1 (Palm-Khintchine Theorem). Let $\{N_j(t), t \geq 0\}$ be independent renewal processes for $j = 1, 2, \cdots$ with identically and independent distributed times T_j for each renewal process. The superposition $N(t) = \sum_{j=1}^{n} \{N_j(t), t \geq 0\}$ is asymptotically a Poisson process for $n \to \infty$, if:

1) Overall load is finite, $k = n / \sum_{j=1}^{n} E[T_j]$,

2) No single process dominates the superposition $E[T_j] \ll 1/k$.

Let assume that the application time periods are $T_1 < T_2 < \cdots < T_{N-1} < T_N$. Therefore, in the long run for a time interval T_N, the total traffic load λ_{tot} from n sensors in a network with N applications is defined by the traffic from both the periodic and urgent frames from all the sensors during that interval of time:

$$\lambda_{tot} = \sum_{j=1}^{n} \frac{T_N}{T_j} + \sum_{j=1}^{n} \lambda_j T_N \tag{1}$$

where T_j and λ_j are the application period and the rate parameter for the jth sensor. For all $j = 1, 2, \cdots, n$, we have $T_j = T_i$ and $\lambda_j = \lambda_i$ if a sensor j belongs to an application i.

3. Derivation of the Stability Condition for the DQ Algorithm in a Single-Channel Environment

The DQ algorithm is comprised of two subsystems: the collision resolution subsystem and the data transmission subsystem. The former deals with frames from sensors trying to get access to the channel. The latter contains frames from sensors waiting for their turn for data transmission. These two subsystems are related in series as two queues in tandem with the first being the CRQ and the second the DTQ. Thus, for the DQ algorithm to be stable, both queues need also to be stable. The DQ algorithm is stable if the algorithm never reaches a point where a frame may wait in the system an infinite time before the complete transmission [12].

The total waiting time $t_{DQ/frame}$ for any given frame in the DQ system is composed of three components:

$$t_{DQ/\,frame} = t_{w/\,frame} + t_{crq/\,frame} + t_{dtq/\,frame} \tag{2}$$

where $t_{w/frame}$ is the waiting time before the contention, $t_{crq/frame}$ is the time spent contending in the CRQ, and $t_{dtq/frame}$ is the waiting time in the DTQ. The first two components of the total waiting time in the DQ system are both finite. The upper limit of the waiting time before the contention $t_{w/frame}$ corresponds to the timeout period before a frame is dropped, depending on whether the frame is periodic or urgent. The CRQ waiting time $t_{crq/frame}$ is also stable because once a frame is granted access to the channel, it is assured to secure a place in the DTQ [12] [19]. As for the DTQ waiting time $t_{dtq/frame}$, it is defined by the number of frames entering and leaving the DTQ. Consequently, the waiting time in the DTQ is stable only if the speed of frames leaving the DTQ is not less than the speed of frames leaving the CRQ. Therefore, only the data transmission subsystem defines the stability of the DQ algorithm [19]. The frames exiting the CRQ need to be controlled so that the DQ algorithm is stable. Therefore, the task of establishing the stability condition for the DQ algorithm gets into a problem of defining the maximum number of sensors in the network before the algorithm is unstable.

In **Figure 2**, we present different cases of the evolution in time of the DTQ depending on whether the stability condition is observed or not. It can be noted that:

1) when the DQ algorithm is unstable, the length of the DTQ tends to grow linearly with each new contention. Consequently, in the long run, the time each sensor waits in the DTQ increases without bound.

2) when the algorithm is stable, the length of the DTQ is quasi-periodic and does not grow linearly with each new contention; thus, the waiting time in the DTQ is finite.

Let CRQ_{out} be the service rate from the CRQ, DTQ_{in} the arrival rate in the DTQ and DTQ_{out} the service rate from the DTQ. In a single-channel environment, the DQ algorithm can service only one frame during a slot time:

$$DTQ_{out} = 1 \tag{3}$$

The arrival rate DTQ_{in} in the DTQ corresponds to the CRQ service rate CRQ_{out} because when a frame finishes the channel contention it is sent in the DTQ. Thus, we have:

$$DTQ_{in} = CRQ_{out} \tag{4}$$

Therefore, in the long run, the DQ algorithm, in a single-channel network, is stable if and only if the arrival rate in the DTQ (*i.e.*, the service rate from the CRQ) is not longer than the service rate from the DTQ:

$$CRQ_{in} < 1 \tag{5}$$

Following the Equation (1), it can be noted that, in the long run, for a network with n sensors from N applications, the total traffic load corresponds to λ_{tot} for an interval of time T_N. In Table 1, we present the average service rate CRQ_{out} from the CRQ during the average collision resolution time t_{crq}. The results are obtained through a DQ performance analysis executed in Matlab. The average service rate is evaluated during the collision resolution time for a network with n ($n \geq 2$) frames at the initial collision and for m contention slots.

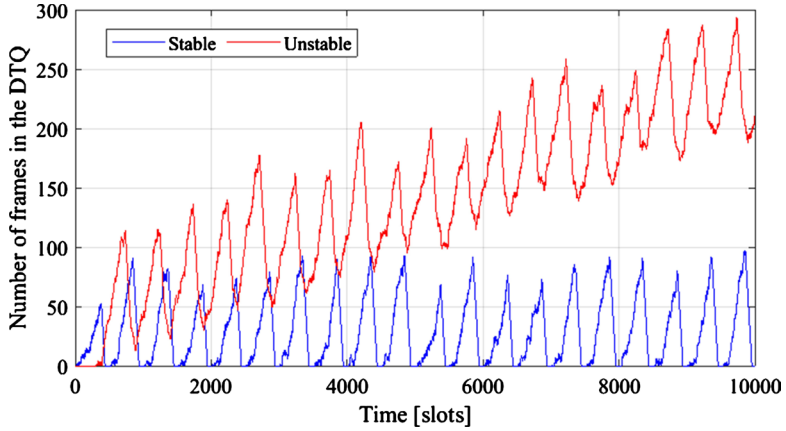

Figure 2. Evolution in time of the DTQ when the DQ algorithm is stable (blue) and unstable (red) for the same application.

Table 1. Average service rate from the CRQ for the DQ algorithm for n number of contending frames.

Number of contention slots	t_{crq}, slots	CRQ_{out}, frame/slot
$m = 3$	$0.91n$	1.10
$m = 4$	$0.72n$	1.39
$m = 8$	$0.48n$	2.08
$m = 12$	$0.41n$	2.44
$m = 16$	$0.35n$	2.86

From Table 1, it can be observed that, on average, the service rate CRQ_{out} from the CRQ is over a frame per slot when the initial number of contending frames is greater than unity. Therefore, in a single-channel setting, the DQ algorithm is stable if the average initial number of frames contending per CRQ session is not greater than unity.

$$n_{crq} < 1 \tag{6}$$

Let us define n_x the average number of frames generated in a slot during the period T_N, *i.e.*, the average traffic load. The probability for a sensor choosing a given slot for its periodic or urgent frame during the period T_N follows a discrete uniform distribution with the probability mass function:

$$p = \frac{1}{T_N} \tag{7}$$

The probability of having x sensors generated during a given slot (X is the corresponding random variable) follows a binomial distribution with parameters p and λ_{tot}:

$$p_x\left(X = x\right) = \binom{\lambda_{tot}}{x} p^x q^{\lambda_{tot} - x} \tag{8}$$

where $q = 1 - p$. The average number of frames n_x generated in a slot during the period T_N is:

$$n_x = \sum_{x=0}^{\lambda_{tot}} x p_x = \frac{\lambda_{tot}}{T_N} \tag{9}$$

Taking into consideration the Equations (6) and (9), in the long run, the stability condition for the DQ algorithm, for a network with n sensors from N different applications and a traffic comprised of periodic and urgent frames from each sensor, is:

$$v_1 n_{s,1} + v_2 n_{s,2} + \cdots + v_{(N-1)} n_{s,(N-1)} + v_N n_{s,N} < T_N \tag{10}$$

where:

1) $v_i = T_N / T_i$ are constants;

2) T_i is the application period for the ith application;

3) $n_{s,i}$ is the total number of frames generated during the period T_i for the ith application:

$$n_{s,i} = n_{p,i} + n_{u,i} \tag{11}$$

where $n_{p,i}$ and $n_{u,i}$ represent respectively the number of periodic and urgent frames for the ith application.

Therefore, $n_{p,1}, n_{p,2}, \cdots, n_{p,(N-1)}$ and $n_{p,N}$ are the maximum number of sensors in the network from the first, the second, ..., the $(N-1)$th and the Nth application before the DQ algorithm is unstable. The number of urgent frames for the ith application is defined as follows:

$$n_{u,i} = \lambda_i n_{p,i} T_i \tag{12}$$

where λ_i is the ith application rate parameter.

4. Steady-State Performance Analysis of the DQ Algorithm with One Single Channel

Let us consider a network with n sensors from N applications, and sensors from the same application have similar traffic characteristics. As stated earlier, a sensor from an ith application generates periodic frames at regular period T_i, and urgent frames following a Poisson arrival process with parameter λ_i. Moreover, sensors are unsynchronized because such scenario guarantees a low complex, low cost, low power and small in size sensor.

The DQ algorithm is evaluated under the condition that it is stable following the criterion presented in Equation (10). Therefore, on average, only one sensor is contending at any moment in a given slot in the long run. The condition presented in Equation (6) implies that on average for any type of frame from any application, we have:

$$\begin{cases} t_{w/\,frame} = 0 \\ t_{crq/\,frame} = 0 \\ t_{dtq/\,frame} = 0 \\ attempts = 1 \end{cases} \tag{13}$$

here $t_{w/frame}$, $t_{crq/frame}$, $t_{dtq/frame}$, and $attempts$ are respectively the average waiting time before the contention, the average waiting time in the CRQ, the average waiting time in the DTQ, and the average number of attempts before accessing the channel for any frame in the network. The first three metrics are measured in slots. Thus, a frame is transmitted in the same slot as it is generated as no concurrent sensor tries to access the wireless channel at the same time.

As for the average channel throughput, it is defined through the average number of successful frames accessing the channel during the period T_N. A frame is successful if it has been assigned a DQ data slot for its transmission. Therefore, the average number of successful frames is:

$$Throughput = \frac{\lambda_{tot}}{T_N} \tag{14}$$

Moreover, as for the contention slots in DQ slot, they are allocated as follows:

$$\begin{cases} sCs = 1 \\ eCs = m - 1 \\ cCs = 0 \end{cases} \tag{15}$$

where sCs, eCs, and cCs are respectively the average number of successful, empty and collided contention slots in a DQ slot. As it can be noticed, an increase in the number of the contention slots m in the DQ slot leads to an inefficient allocation because only the average number of empty contention slots is increased.

In order to validate our analytical results, a steady-state event-driven simulation model has been developed and executed in Matlab for the evaluation of the performance metrics of the DQ algorithm. The sample averages of the consid-

ered metrics are obtained over a single replication. We use the method of *batch means* for the estimation of the throughput because it is less susceptible to the initial effects of the simulation [21]. The batch size corresponds to the largest application period. Furthermore, the number of batches varies between 30 and 40, following the recommendations from [22]. As for the other metrics, we consider each contention as a terminating simulation. Therefore, the estimate of the mean is averaged over the number of contentions occurring during the observation time.

In **Figure 3**, we present the steady-state numerical results of the performance metrics of the DQ algorithm for a network when sensors are asynchronous in their traffic generation. However, for comparison purposes, the synchronized scenario is also considered. A frame requires one data slot to transmit its payload. We choose to use the source traffic model for its accuracy in capturing the behaviour of each sensor in the model [14] [16]. As periodic and urgent frames are identically and independently distributed, we consider only the periodic frames in our example. From **Figure 3**, for the asynchronous scenario, it can be observed that:

1) From **Figure 3(a)**, on average, a frame is sent after it is generated ($t_{DQ/frame} = 0$). The average waiting time in the DQ system for any given frame in the network is obtained as given in Equation (2). It should be noted that for $n_x = 1$, the overall waiting time in the DQ system is limited by the simulation time; otherwise, it would increase without bound over time.

2) From **Figure 3(b)**, on average, a frame requires one attempt to get access to the channel. As n_x tends to unity, the average number of attempts per sensor increases significantly compared to its value when $n_x < 1$.

3) From **Figure 3(c)**, the average number of successful slots increases with n_x to reach its maximum when $n_x = 1$.

4) From the **Figure 3(d)**, on average, $m - 1$ contention slots are empty as long as we have $n_x < 1$.

From **Figure 3**, it can also be observed that the synchronized scenario would outperform the unsynchronized case in terms of efficient use of the contention slots. However, when frames are synchronized, they would not only last a significant time in the DQ system but also more energy would be spent for the channel contention compared to the unsynchronized scenario. In terms of channel throughput, both scenarios have the same performance. It should be noted that both the waiting time per frame in the DQ system and the number of attempts per frame increase with the number of frames in the network for the synchronized case.

In general, the numerical results validate our analytical assumptions. However, as it can be observed, when the average number of frames n_x generated during a slot in the network tends to unity, the DQ algorithm begins to be unstable. The average waiting time per frame in the DQ system and the average number of attempts per frame vary significantly from their average values compared to the

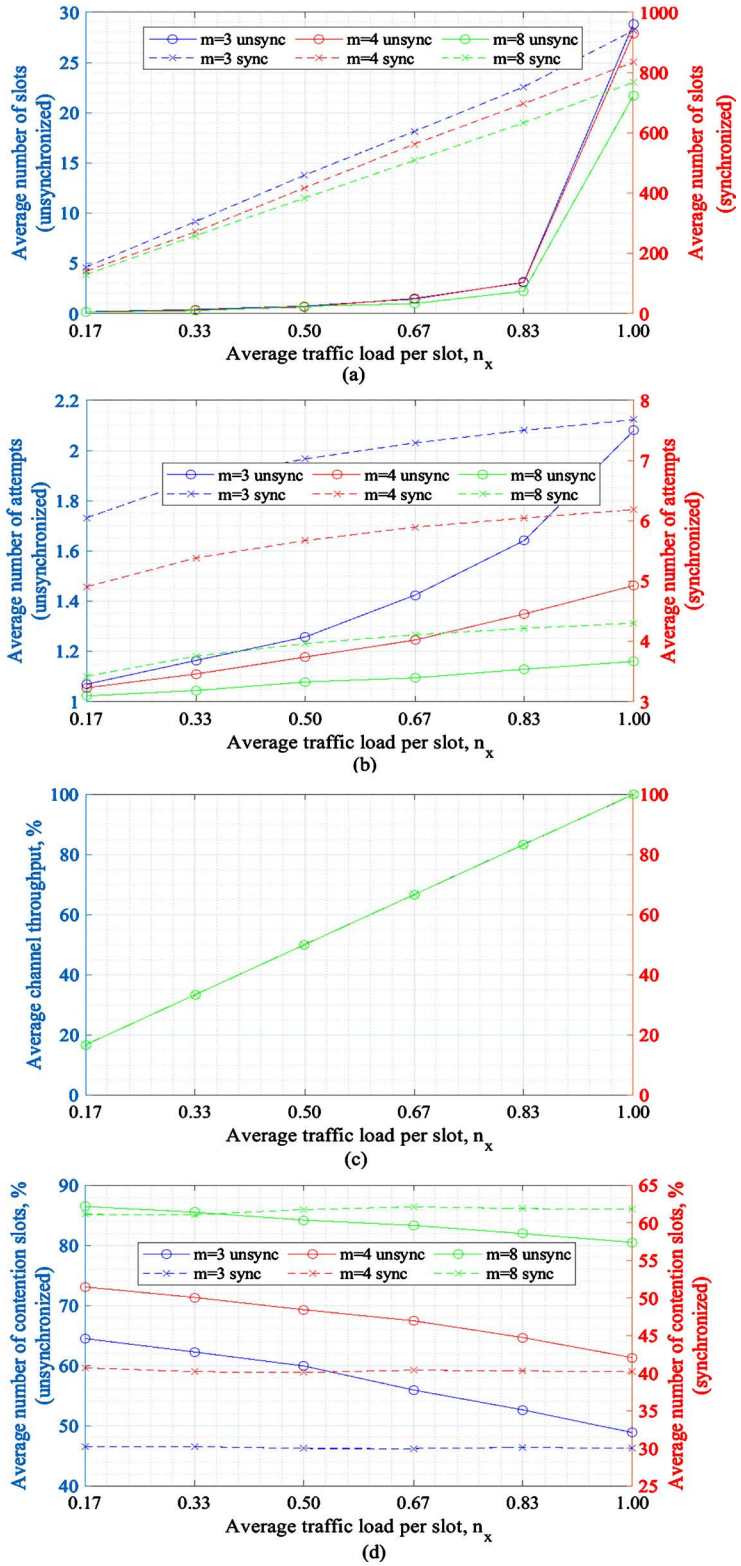

Figure 3. Simulation results of the DQ algorithm for a network with one application and a frame period of 1500 slots both for the unsynchronized and the synchronized traffic scenarios: (a) Average waiting time per frame in the DQ system, (b) Average number of attempts per frame, (c) Average channel throughput for both the synchronization cases and any number of contention slots, and (d) Average number of empty contention slots.

case when $n_x < 1$, whereas the average throughput is at its maximum. Moreover, except for the average distribution of the contention slots, other performance metrics vary slightly with the number of contention slots m.

In a single-channel M2M environment, where sensors are asynchronous in their contention and the traffic model is comprised of periodic and urgent frames, the DQ algorithm behaves like a Time Division Multiple Access (TDMA) algorithm where sensors are allocated frames for transmission randomly. In such a setting, the DQ network performance would be similar to those of Aloha based algorithms in terms of contention resolution. However, in case of an idle-to-saturation scenario in which sensors try to access the channel simultaneously and in a synchronized manner, the DQ algorithm would outperform the Aloha algorithms as the later becomes unstable as the number of contending sensors increases [5] [12] [23].

5. Conclusions and Future Work

In this paper, we have presented a derivation of the stability condition for the DQ algorithm in a single-channel M2M communication environment. Moreover, a steady-state evaluation of the DQ performance metrics has also been conducted when the algorithm is stable. To achieve those goals, we considered a traffic model where each sensor from an ith application generates periodic frames at regular time interval T_i and urgent frames following a Poisson arrival process with the rate parameter λ_i. It was assumed there are N applications in the network with $T_1 < T_2 < \cdots < T_{N-1} < T_N$ and that the sensors are asynchronous in their channel contention. Sensors from the same application were considered to exhibit similar traffic characteristics.

The DQ algorithm is unstable if the DTQ increases without bound over time after each new contention process. Therefore, we found that the maximum number of periodic and urgent frames generated during the period T_N from all the applications needs to be less than the Nth application period in the long run. That number was also observed to be independent of the number of contention slots m when the aggregated network traffic from the sensors is considered to be a Poisson process. In a single-channel communication setting, the DQ algorithm is stable only if on average sensors are allowed to contend individually per slot. Therefore, for any type of frame, a sensor requires on average one attempt to get access to the channel and is guaranteed to transmit its frame instantaneously. However, such an environment does not efficiently use the contention slots for the collision resolution. Therefore, in a low power wide area network, the DQ algorithm and the Aloha based algorithms would perform equally. Nevertheless, the DQ protocol outperforms the Aloha algorithms in case of an idle-to-saturation scenario. Additionally, numerical simulations were used to validate the analytical results.

In the future, we plan to perform a steady-state evaluation of the algorithm metrics in a multichannel M2M communication network.

Acknowledgements

The Pan African University funded this research.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Ding, J., Nemati, M., Ranaweera, C. and Choi, J. (2020) IoT Connectivity Technologies and Applications: A Survey. *IEEE Access*, **8**, 67646-67673. https://doi.org/10.1109/ACCESS.2020.2985932

[2] Raza, U., Kulkarni, P. and Sooriyabandara, M. (2017) Low Power Wide Area Networks: An Overview. *IEEE Communications Surveys & Tutorials*, **19**, 855-873. https://doi.org/10.1109/COMST.2017.2652320

[3] Laya, A., Alonso, L. and Alonso-Zarate, J. (2015) Contention Resolution Queues for Massive Machine Type Communications in LTE. *IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications*, Hong Kong, 30 August-2 September, 2314-2318. https://doi.org/10.1109/PIMRC.2015.7343684

[4] Cheng, R.G., Becvar, Z. and Yang, P.H. (2018) Modeling of Distributed Queueing-Based Random Access for Machine Type Communications in Mobile Networks. *IEEE Communications Letters*, **22**, 129-132. https://doi.org/10.1109/LCOMM.2017.2755020

[5] Bui, A.T.H., Nguyen, C.T., Thang, T.C. and Pham, A.T. (2018) Free Access Distributed Queue Protocol for Massive Cellular-Based M2M Communications with Bursty Traffic. *IEEE 88th Vehicular Technology Conference*, Chicago, 27-30 August, 1-5. https://doi.org/10.1109/VTCFall.2018.8690880

[6] Lee, K. and Jang, J.W. (2018) An Efficient Contention Resolution Scheme for Massive IoT Devices in Random Access to LTE-A Networks. *IEEE Access*, **6**, 67118-67130. https://doi.org/10.1109/ACCESS.2018.2876438

[7] Yoon, C. (2016) Distributed Queuing with Preamble Grouping for Massive IoT Devices in LTE Random Access. *International Conference on Information and Communication Technology Convergence*, Jeju Island, 19-21 October, 103-105. https://doi.org/10.1109/ICTC.2016.7763445

[8] Xing, S., Wen, X., Lu, Z., Pan, Q. and Jing, W. (2019) A Novel Distributed Queuing-Based Random Access Protocol for Narrowband-IoT. *IEEE International Conference on Communications*, Shanghai, 20-24 May, 1-7. https://doi.org/10.1109/ICC.2019.8762057

[9] Wu, W., Li, Y., Zhang, Y., Wang, B. and Wang, W. (2019) Distributed Queueing-Based Random Access Protocol for LoRa Networks. *IEEE Internet of Things Journal*, **7**, 763-772. https://doi.org/10.1109/JIOT.2019.2945327

[10] Marchiori, A. (2017) Maximizing Coverage in Low-Power Wide-Area IoT Networks. *IEEE International Conference on Pervasive Computing and Communications Workshops*, Kona, 13-17 March, 467-472.

https://doi.org/10.1109/PERCOMW.2017.7917608

[11] Vazquez-Gallego, F., Tuset-Peiró, P., Alonso, L. and Alonso-Zarate, J. (2018) Combining Distributed Queuing with Energy Harvesting to Enable Perpetual Distributed Data Collection Applications. *Transactions on Emerging Telecommunications Technologies*, **29**, e3195. https://doi.org/10.1002/ett.3195

[12] Zhang, X. and Campbell, G. (1993) Performance Analysis of Distributed Queueing Random Access Protocol-DQRAP. DQRAP Research Group Report, 93(1).

[13] Metzger, F., Hoßfeld, T., Bauer, A., Kounev, S. and Heegaard, P.E. (2019) Modeling of Aggregated IoT Traffic and Its Application to an IoT Cloud. *Proceedings of the IEEE*, **107**, 679-694. https://doi.org/10.1109/JPROC.2019.2901578

[14] Laner, M., Nikaein, N., Svoboda, P., Popovic, M., Drajic, D. and Krco, S. (2015) Traffic Models for Machine-to-Machine (M2M) Communications: Types and Applications. In: *Machine-to-Machine (M2M) Communications, Architecture, Performance and Applications*, Woodhead Publishing, Cambridge, Chapter 8, 133-154. https://doi.org/10.1016/B978-1-78242-102-3.00008-3

[15] Sansoni, M., Ravagnani, G., Zucchetto, D., Pielli, C., Zanella, A. and Mahmood, K. (2018) Comparison of M2M Traffic Models against Real World Data Sets. *IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks*, Barcelona, 17-19 September, 1-6. https://doi.org/10.1109/CAMAD.2018.8515000

[16] Centenaro, M. and Vangelista, L. (2015) A Study on M2M Traffic and Its Impact on Cellular Networks. *IEEE 2nd World Forum on Internet of Things*, Milan, 14-16 December, 154-159. https://doi.org/10.1109/WF-IoT.2015.7389044

[17] Laner, M., Svoboda, P., Nikaein, N. and Rupp, M. (2013) Traffic Models for Machine Type Communications. *The Tenth International Symposium on Wireless Communication Systems*, Ilmenau, 27-30 August, 1-5.

[18] Smiljkovic, K., Atanasovski, V. and Gavrilovska, L. (2014) Machine-to-Machine Traffic Characterization: Models and Case Study on Integration in LTE. *4th International Conference on Wireless Communications, Vehicular Technology, Information Theory and Aerospace & Electronic Systems*, Aalborg, 11-14 May, 1-5. https://doi.org/10.1109/VITAE.2014.6934482

[19] Xu, W. and Campbell, G. (1992) A near Perfect Stable Random Access Protocol for a Broadcast Channel. *SUPERCOMM/ICC'92 Discovering a New World of Communications*, Chicago, 14-18 June, 370-374. https://doi.org/10.1109/ICC.1992.268230

[20] Heyman, D.P. and Sobel, M.J. (1982) Stochastic Models in Operations Research. 1. Stochastic Processes and Operating Characteristics. McGraw-Hill, New York.

[21] Law, A.M. and Kelton, W.D. (2015) Simulation Modelling and Analysis. McGraw-Hill, New York.

[22] Banks, J. (2014) Discrete Event System Simulation. Pearson Education India, Bengaluru.

[23] Vázquez-Gallego, F., Alonso-Zárate, J., Tuset-Peiró, P. and Alonso, L. (2014) Energy Analysis of a Contention Tree-Based Access Protocol for Machine-to-Machine Networks with Idle-to-Saturation Traffic Transitions. *IEEE International Conference on Communications*, Sydney, 10-14 June, 1094-1099. https://doi.org/10.1109/ICC.2014.6883467