Congestion Control for Machine-Type Communications in LTE-A Networks

Chia-Wei Chang, Yi-Hao Lin, Yi Ren, and Jyh-Cheng Chen
Department of Computer Science, National Chiao Tung University, Hsinchu 300, Taiwan

Email: {chwchang, yhlin1377, yiren, jcc}@cs.nctu.edu.tw

***Abstract*—Collecting data from a tremendous amount of Internet-of-Things (IoT) devices for next generation networks is a big challenge. A large number of devices may lead to severe congestion in Radio Access Network (RAN) and Core Network (CN). 3GPP has specified several mechanisms to handle the congestion caused by massive amounts of devices. However, detailed settings and strategies of them are not defined in the standards and are left for operators. In this paper, we propose two congestion control algorithms which efficiently reduce the congestion. Simulation results demonstrate that the proposed algorithms can achieve 20∼40% improvement regarding accept ratio, overload degree and waiting time compared with those in LTE-A.**

***Index Terms*—congestion control, M2M, IoT, LTE-A**

I. Introduction

The emergence of Machine-Type-Communications (MTC) and Internet-of-Things (IoT) introduces new challenges for next generation networks [1]. By 2018, MTC traffic is expected to have a growth rate of 71% from 2014, while MTC connections will be 43% of the total devices and connections [2]. In Long-term Evolution Advanced (LTE-A), a Radio Resource Control (RRC) connection should be established before any data transmission, no matter what size the data is. The procedure to establish an RRC connection has more than 12 interactions in the Radio Access Network (RAN) side and 18 interactions in the Core Network (CN) side [3], [4]. When a tremendous amount of MTC devices ask connections for transmission, they may cause congestions in both RAN and CN. They will deteriorate Quality of Service (QoS) for Human-to-Human (H2H) communications as well.

Specifically, in RAN side, as shown in Fig 1, congestions arise mainly because MTC devices and User Equipment (UEs) contend for the same *random access* channel simultaneously. In CN side, the devices performing Non-Access Stratum (NAS) procedure and data transmission lead to congestions on Mobility Management Entity (MME), Serving Gateway (S-GW), and Packet Data Network Gateway (P-GW). Thus, 3GPP has proposed the following mechanisms to alleviate RAN/CN congestions:

- *Mechanism for RAN congestion*:
  To ease RAN congestion, 3GPP has designed the *Extended Access Baring (EAB)* to prevent delay-tolerant devices from contending the random access channel for a period [3].
- *Mechanism for CN congestion*:

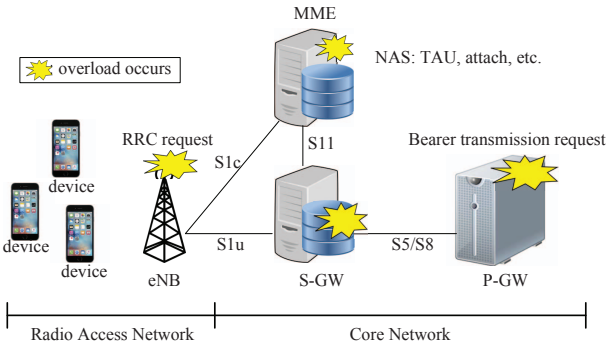

Fig. 1. RAN and CN Congestion.

  Regarding CN congestion, 3GPP has defined a tag called *Low Access Priority Indicator (LAPI)*. Its purpose is to identify which devices are delay-tolerant. M2M operators can configure LAPI to indicate that a device can tolerate delay by using the Over-the-air (OTA) technology. Therefore, CN can postpone the requests of the devices which are labeled as delay-tolerant for a period to ease the congestion.

In summary, in order to solve congestion, 3GPP rejects the requests of delay-tolerant devices and prevents them from accessing the networks during a pre-defined period (i.e., lets devices to backoff). Although the congestion intensity can be distributed by letting device to backoff [4], the major problem is how long this pre-defined period should be. For example, if the pre-defined period is too small, the devices will tend to issue other requests which will impose additional signaling overheads. Contrarily, if the pre-defined period is too large, it may incur idle time for the networks. Thus, the network utilization will be low. How to set a proper time, however, is not standardized by 3GPP. For future networks with massive amounts of IoT devices, it is a challenge to reduce congestion while also maintain high network utilization.

In this paper, we propose algorithms to reject MTC/IoT device's requests and decide an appropriate backoff timer when networks are congested. We use M/M/1/K queueing system to model the impact of MTC traffic on the queue length. Besides, we design a mechanism to detect when an overloaded condition will happen so that network congestion can be mitigated in advance. Simulation results demonstrate that the

proposed algorithms have 20∼40% improvement regarding accept ratio, overload degree and waiting time compared with those in LTE-A.

The rest of the paper is organized as follows: Section II briefly introduces background of congestion control in LTE-A. Section III depicts the state-of-the-art. Proposed algorithms are presented in Section IV, followed by performance evaluation in Section V. Section VI concludes this paper.

II. Overview of M2M Congestion control in LTE-A

In 3GPP Release 10, a device is regarded as low priority when its LAPI is configured as "1", and as normal priority when its LAPI is configured as "0". When networks are overloaded, congestion control function starts to reject the requests of the devices with LAPI = 1. It also decides and sends backoff values to the device. During backoff period, the device cannot initiate an RRC connection until the expiration of the backoff timer. 3GPP has defined congestion control functions for P-GW, MME, and eNB[1], respectively. We describe them as follows:

- *P-GW Congestion Control*:
  P-GW is responsible for IP address assignment for users and devices and acts as the point of entry and exit between the CN and external packet data networks. If a P-GW is overloaded, it will refuse new connections and indicate a random backoff time for a specific Access Point Name (APN) to MME. MME then rejects all incoming requests for the specified APN. Devices will stop trying to access the APN until the backoff timer is expired.
- *MME Congestion Control*:
  The MME is a control entity dealing with the signaling between devices and CN. It is also responsible for bearer activation/deactivation. When an MME is overloaded, it will refuse the NAS requests from low access devices and further specify randomized backoff values to the devices in case of the excessive load by informing RAN with its load level. Thus, RAN can start blocking some incoming traffic heading to the CN. Furthermore, MME will decline the downlink data notification for low-priority devices in idle mode (i.e., MME refuses the paging operations if there are traffic heading to low-priority devices).
- *eNB Congestion Control*:
  The eNB will reject any RRC Connection Requests from low-priority devices when it is overloaded. LTE-A defines six causes for RRC Connection Request: (1) Emergency, (2) High Priority Access, (3) Mobile Originating Data, (4) Mobile Originating Signaling, (5) Mobile Terminating Access, and (6) Delay Tolerant Access. Generally, requests with the lowest priority cause, Delay Tolerant Access, are rejected first. Meanwhile, a random *Extended Wait Time* is configured within the RRC Connection Reject message to a device. Upon receiving an RRC Connection Reject message, a device will start T302 timer[2] with a value specified in the RRC Connection Reject message. During the backoff time, the device will suspend the RRC Connection attempts.

[1]Please refer to [4] for the definitions of P-GW, MME, and eNB.

III. Related Work

According to the report from [2], the number of mobile devices will be over 10 billion by 2018. Upon detecting an event, lots of devices generate lots of small-size data and try to connect to the network for data transmission at the same time. It will lead to congestions at RAN and CN sides. Several solutions were proposed to mitigate the congestions.

To reduce the RAN congestion, the authors in [5] proposed an eNB selection mechanism to distribute the access intensity of highly-loaded eNBs. In [6], [7], the load of RAN is predicted so that the access barring parameter can be determined by the network. The authors preallocated random access resource for different MTC classes to prevent devices from accessing network at the same time [8]. In [9], the authors used Access Class Barring and timing advance information to relieve the random access load. However, RAN congestion control does not always work if considerable network jams occur.

To reduce the CN congestion, the authors of [10] proposed a temporal load balancing mechanism. A backoff timer is used to reject devices from accessing CN. In [11], the authors proposed an admission control algorithm to reject requests from devices. The authors in [12], [13] proposed a congestion-aware admission control mechanism that selectively rejects signaling messages from devices. The probability of rejection follows the congestion level of a relevant CN node. Even though these studies distribute the congestion over the time domain, it may still not work when the devices keep generate lots of data. Therefore, in this paper, we first propose a backoff timer selection mechanism. Furthermore, we provide a mechanism to distribute a congestion over the space domain.

IV. Proposed Algorithms

A. Problem Statement

As mentioned in Section I, LTE-A starts to reject a device with low priority and gives it a backoff value when networks are overloaded. The backoff value can be set from 0 to 1800 ms [4]. However, how to appropriately set the backoff value based on network load is undefined in the standards. Specifically, a very short backoff value may lead to a device issuing another connection request within a short period, imposing extra signaling cost. That is, networks will reject the device again. On the other hand, a very long backoff value may reduce the utilization of the network. Specifying the length of backoff value is a dilemma for operators. Thus, the objectives of this paper are summarized as follows:

- *Backoff value selection (Algorithm 1)*:
  We use M/M/1/K queueing system [14] to determine the backoff time so that the traffic intensity can be scattered over time domain by using a systematic way. The value

[2]Details of T302 timer can be found in [4].

TABLE I
SYSTEM PARAMETERS

Notation	Description
t_i	The i^{th} period.
λ_{t_i}	The estimated mean arrival rate of devices' requests during t_i. That is, $\lambda_{t_i} = \lambda^p_{t_i} + \lambda^b_{t_i}$.
$\lambda^N_{t_i}$	The predicted mean arrival rate from new requests during time t_i.
$\lambda^B_{t_i}$	The mean arrival rate from old requests during t_i due to expiration of backoff timers.
μ	Mean service rate of a network entity.
$P_{t_i}(j)$	The probability that there are j requests in the system during t_i.
$\overline{Q}_{t_i}$	The average number of requests in the system during t_i.
N_c	The number of requests to be served in current system.
N_θ	The number of requests that may pose an overloaded condition.
$\overline{W_{t_i}}$	The average waiting time of requests during t_i.
T_{t_i}	Length of the extend backoff timer.
P_θ	One of the overloaded thresholds for time domain.
ϵ	One of the overloaded thresholds for space domain.
Ω	One of the overloaded thresholds for space domain.

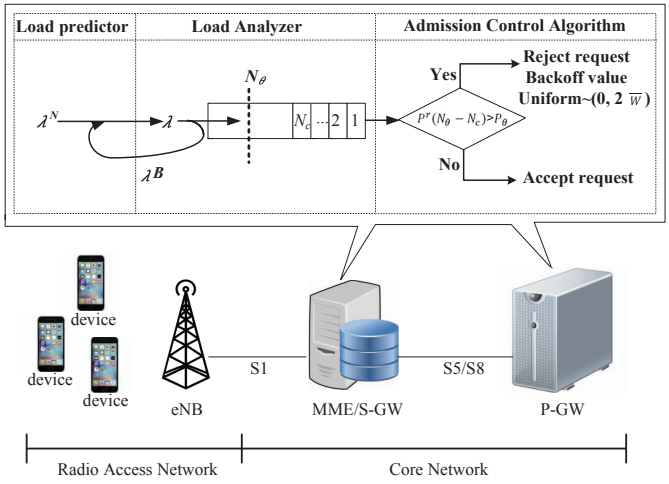

Fig. 2. Algorithm deployed in every NE in CN.

of the backoff depends on the number of requests waiting in the queue. By this way, the probabilities that the device issuing another requests or networks having low utilization will be reduced.

- *Load distribution over space domain (Algorithm 2)* The congestion probability derived in Algorithm 1 will be re-examined in this phase. If the congestion intensity is still high due to a massive number of devices, an algorithm to shift loads to other Network Entities (NE) is proposed (distribute loads over spatial domain). Furthermore, if all NEs' loads in the pool are high, we then use Network Function Virtualization (NFV) [15] technique to create a new instance of a certain NE and distribute the load from existing NEs to the new one.

Algorithm 1 Overload Control in Time Domain

Input: $\lambda^N_{t_0}$=0, $\lambda^B_{t_1} = 0$, α, μ, P_θ, N_θ, $\|t_0\|$;
Output: Length of Extended Backoff Timer: T_{t_i}

while true **do**
  Collect Λ_{t_i};
  /** if exponential moving average is adopted **/
  $\lambda^N_{t_{i+1}} = \alpha\Lambda_{t_i} + (1-\alpha)\lambda^N_{t_i}$;
  $\lambda_{t_{i+1}} = \lambda^N_{t_{i+1}} + \lambda^B_{t_{i+1}}$;
  Whenever receiving a request from devices:
  $\rho_{t_{i+1}} = \frac{\lambda^N_{t_{i+1}}}{\mu}$;
  Compute $P^r_{t_{i+1}}(N_\theta - N_c)$;
  if $P^r_{t_{i+1}}(N_\theta - N_c) \leq P_\theta$ **then**;
    Accept this request;
  else
    Compute $\overline{W_{t_{i+1}}}$;
    $T_{t_{i+1}} = Uniform(0, 2\overline{W_{t_{i+1}}})$;
    Extended Backoff timer=$T_{t_{i+1}}$;
    $x = \lfloor \frac{T_{t_{i+1}}}{\|t_{i+1}\|} \rfloor$;
    $\lambda^B_{t_x}$++;
    Reject this request & send $T_{t_{i+1}}$ to the device;
  end if
end while

B. Backoff value selection

The basic idea of scattering traffic intensity over time domain is depicted in Fig. 2, which consists of three components: Load Predictor, Load Analyzer, and Admission Control Algorithm (ACA). The Load Predictor predicts the arrivals in next period. The Load Analyzer then evaluates the probability of overloaded condition caused by future requests. Next, the ACA makes decision to accept/reject the request according to the probability. If a request is rejected, the ACA forwards a backoff time to the device, where the backoff time is chosen based on the NE's load. The backoff time is larger if the NE's load is high, and vice versa. NE has to locally maintain its own algorithms.

To start overload protection mechanism, it is necessary to detect whether there is an overload condition at any NE. Predicting future load helps us to start overload protection in advance. For this purpose, the time domain is divided into a set of periods. For instance, t_i, is the i^{th} period. Note that $\|t_i\| = \|t_j\|$, $i \neq j \ \forall \ t_i, t_j$ where $i, j \in \mathbb{N}$. The buffer size is K which is finite with First In First Out (FIFO) queueing discipline. The arrival rate of devices' requests is denoted by λ. The service rate in an NE, e.g., MME, S-GW, or P-GW, is denoted by μ. The arrival of devices' requests follows Poisson process, and the service time of a request in one NE is independent and identically exponentially distributed[3]. Other system parameters are listed in Table I. Thus, the estimated arrival requests during t_i, λ_{t_i}, is given as follows:

$$\lambda_{t_i} = \lambda^N_{t_i} + \lambda^B_{t_i}, \tag{1}$$

[3]The distribution of the service time can be replaced by other distributions, here, we simply use exponential distribution as an example.

where $\lambda_{t_i}^N$ is predicted arrival rate of new device connection requests during t_i and $\lambda_{t_i}^B$ is the connection request arrival rate from the devices of which backoff timers are expired during t_i.

Specifically, $\lambda_{t_i}^N$ can be estimated by using any kinds of prediction techniques such as Weighted Moving Average, Exponential Moving Average, etc. [16]. Historical statistics of arrivals can help to estimate the new arrivals for the next period. Here, we adopt Exponential Moving Average into our algorithm. The weight for each older datum will decrease exponentially. Thus, $\lambda_{t_i}^N$ is given by:

$$\lambda_{t_i}^N = \alpha \Lambda_{t_{i-1}} + (1-\alpha)\lambda_{t_{i-1}}^N, \tag{2}$$

where $\Lambda_{t_{i-1}}$ is the actual new arrival during t_{i-1}, and α is a coefficient factor between 0 and 1, representing the degree of weight decreasing. A larger α discounts older observation faster. Especially, most MTC will send data periodically or send data if some events are triggered. Either periodical or event-triggered transmission has a fixed transmission distribution over time. Therefore, it is reasonable to use historical statistics to predict the arrival for the next period. Operators can choose a proper value of α by observing statistics stored in eNB, MME, or P-GW. In this paper, we only focus on such kind of MTC devices with periodical or event-triggered data transmission. To better estimate the arrival in next periods, one has to find this transmission pattern, which is not our focus in this paper. Some related work can be found in [6], [17], [18], [19].

Regarding $\lambda_{t_i}^B$, it is trivial to obtain. Since an overloaded NE will reject MTC device's connection requests and ban them from sending requests until the expiration of backoff timers[4], an NE can easily know how many MTC devices will resend the requests after the expiration of backoff timers.

Once λ_{t_i} and μ are obtained, the probability that there are j requests in the NE during t_i can be obtained:

$$P_{t_i}(j) = \begin{cases} \rho_{t_i}^j \frac{1-\rho_{t_i}}{1-\rho_{t_i}^{K+1}}, & \rho_{t_i} \neq 1,\ j \leq K, \\ \frac{1}{K+1}, & \rho_{t_i} = 1,\ j \leq K, \end{cases} \tag{3}$$

where $\rho_{t_i} = \frac{\lambda_{t_i}}{\mu} < \infty$. For M/M/c queueing system, the system is stationary if the condition $\rho_{t_i} < c$ is satisfied [14]. Whereas, M/M/1/K queueing system is always stationary even if $\rho_{t_i} \geq 1$. Hence, there is no stationary condition [14].

The average number of requests in an NE during t_i is:

$$\overline{Q}_{t_i} = \sum_{j=0}^{K} j \cdot P_{t_i}(j). \tag{4}$$

In order to prevent an NE from overloaded status, we set a warning criterion. The system will be warned if the probability of having at least $N_\theta - N_c$ requests in the system during next period is too high. Therefore, the probability that at least $N_\theta - N_c$ requests in a certain NE during t_i is denoted as $P^r(N_\theta - N_c)$ and is given as:

$$\begin{aligned} P_{t_i}^r(N_\theta - N_c) &= 1 - \sum_{j=0}^{N_\theta - N_c} P_{t_i}(j) \\ &= \begin{cases} \frac{1-\rho_{t_i}}{1-\rho_{t_i}^{K+1}} \sum_{j=N_\theta - N_c+1}^{K} \rho_{t_i}^j, & \rho_{t_i} \neq 1,\ N_\theta \leq K, \\ \frac{K-N_\theta+N_c}{K+1}, & \rho_{t_i} = 1,\ N_\theta \leq K, \end{cases} \end{aligned} \tag{5}$$

where N_θ is the number of requests that may pose an overloaded condition. N_c is the current number of requests to be served in the system. Because an NE can always observe its queue status, it is easy to obtain N_c. Hence, $P_{t_i}^r(N_\theta - N_c)$ can be treated as the load metric for an NE during t_i.

After setting the warning criterion, we have to configure appropriate backoff time for each rejected request. It is necessary to calculate the average waiting time of requests before configuring an appropriate backoff time. We have to ensure that the mean backoff time of all rejected requests approximates to average waiting time of requests in an NE. To do so, MTC devices will not be significantly rejected by the NE again. Hence, the average waiting time in an NE during t_i (denoted as $\overline{W_{t_i}}$) is given by:

$$\overline{W_{t_i}} = \frac{\overline{Q}_{t_i} + N_c}{\lambda_{t_i}(1 - P_{t_i}(K))}. \tag{6}$$

From (6), the backoff time during t_i is denoted as T_{t_i}, and is selected with the following rule:

$$T_{t_i} = Uniform(0, 2\overline{W_{t_i}}). \tag{7}$$

The overload control over time domain is referred in Algorithm 1.

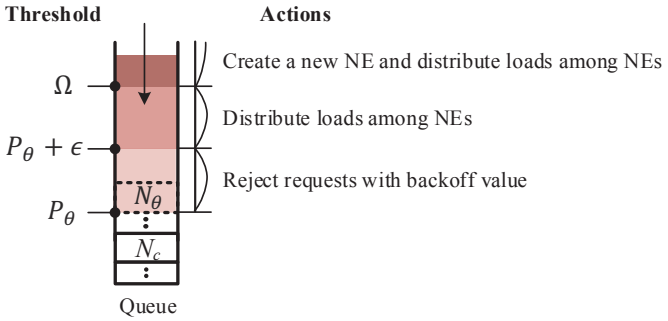

Fig. 3. Threshold configurations and relative actions.

C. Load distribution over space domain

The proposed time domain based congestion control algorithm distributes the loads over time domain. However, if an NE is continuously busy, it will permanently reject devices' request and send back backoff values which increase signaling overhead a lot. Therefore, offloading some devices from one NE to another NEs is a solution to reduce persistent congestion. 3GPP then defines pools of MMEs, S-GWs, and P-GWs for load balancing and redundancy among various NEs [4]. For instance, when an MME is overloaded, we can

[4]An overloaded NE will reply a backoff value to an MTC device.

TABLE II
SIMULATION SETTINGS

$\|\|t_i\|\|$	K	N_θ	P_θ	ϵ	Ω	N_p
100 sec	100	80	0.8	0.1	0.95	2–4

offload traffic by asking devices to perform "Load Balancing Tracking Area Update" to other MMEs. Similarly, loads can also be distributed among pools of eNBs, S-GWs, or P-GWs. Here, we further propose a space domain based algorithm by using the idea of offloading traffic among NEs.

Taking MME as an example, to achieve load balancing among MMEs, a device is assigned to a particular MME based on the load of each MME in the pool. To accomplish this purpose, an MME's *Weight Factor* based on the MME load compared with another MMEs is set [4] and it is forwarded to eNBs through *S1-AP* message. Alternatively, we can change the *Weight Factor* via *MME_Configuration_Update* message [4]. However, the major problem is which value of *Weight Factor* can dynamically reflect the load of a particular MME. In this paper, we use equation (5) as our *Weight Factor*. The detailed overload control over space domain is given in Algorithm 2. In Algorithm 2, two additional thresholds are given: (a) threshold, $P_\theta + \epsilon$, to judge when to distribute loads to other MMEs, and (b) threshold, Ω, to decide when to create a new MME instance. Actions in accordance with their thresholds are demonstrated in Fig. 3. Note that, the selections of ϵ and Ω depend on the preference of the operators. The higher values of ϵ and Ω result in few shifting operations but higher probability of getting an NE overloaded. Due to space limit, we don't discuss this tradeoff in this paper.

V. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the proposed algorithms and compare them with the one in LTE-A. We first investigate the variation of the arrival requests over the time, followed by the discussion between the traffic intensity and the accept ratio of the requests. The overload degree under different traffic intensity then is shown. After that, we examine the average waiting time for a request. The correctness of our model is validated by extensive simulation using ns-2 [20]. The parameters used in the simulation are listed in Table II.

Fig. 4 depicts the relationship between the total arrival requests (i.e., new requests and old requests due to expiring of backoff timer) vs. simulation time when $\rho = 5$. We can see that the congestion-control scheme proposed in LTE-A cannot handle such high ρ and consistently reject the requests which causes another peak of requests from backoff expiration. Therefore, the total arrival requests in LTE-A increase as time goes by. Whereas, our proposed algorithms distribute the load over time and space domains which reduce the total arrival requests effectively.

Fig. 5 shows that the overload degree gradually grows when ρ increases. In LTE-A, the overloaded degree is close to 100% when ρ is approaching 2. The overloaded degree of the proposed algorithms is only around 5% which is

Algorithm 2 Overload Control in Space Domain

Input: Ω, ϵ, P_θ, N_p;
Output: Overload Control among MMEs;

while true **do**
  $\exists\ MME_j,\ \forall j \in N_p$;
  // N_p is the number of NE in the pool.
  Collect $P_{t_i}^{r,j}(N_\theta - N_c^j),\ \forall j \in N_p$;
  if $\sum_j^{N_p} P_{t_i}^{r,j}(N_\theta - N_c^j)/N_p \geq \Omega$ **then**
    if MME_j is the master **then**
      Create a new MME in the MME pool;
      Update $N_p = N_p + 1$ to other MMEs;
    else
      Wait for new N_p & new creation of MME;
    end if
    if $P_{t_i}^{r,j}(N_\theta - N_c^j) \geq (P_\theta + \epsilon)$ **then**
      /** Perform Load Balancing **/
      Force devices to do TAU;
      // Release S1 & attach to other MME according
      // to Weight Factors among all MMEs;
      Calculate $P_{t_i}^{r,j}(N_\theta - N_c^j)$;
      MME_Configuration_Update($P_{t_i}^{r,j}(N_\theta - N_c^j)$);
    end if
  else
    if $P_{t_i}^{r,j}(N_\theta - N_c^j) \geq (P_\theta + \epsilon)$ **then**
      /** Perform Load Balancing **/
      Force devices to do TAU;
      // Release S1 & attach to other MME according
      // to Weight Factors among all MMEs;
      Calculate $P_{t_i}^{r,j}(N_\theta - N_c^j)$
      MME_Configuration_Update($P_{t_i}^{r,j}(N_\theta - N_c^j)$);
    end if
  end if
end while

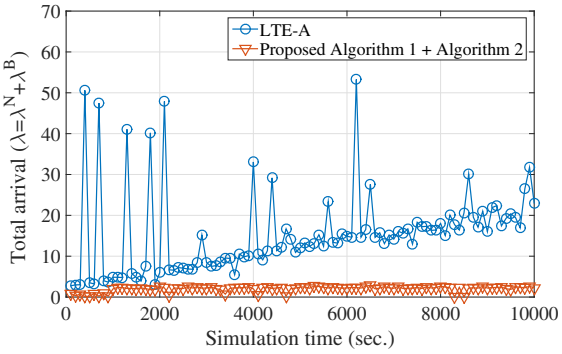

Fig. 4. Traffic intensity ρ vs. total arrival $\lambda = \lambda^N + \lambda^B$.

approximately 20 times as much as that in LTE-A. That is, the proposed algorithms can accommodate a larger number of requests than that in LTE-A under the same ρ. Furthermore, in the proposed algorithms, an NE's load is still not too heavy (around 0.8) even $\rho = 10$. Because of a better backoff selection in the proposed algorithms, each request can be served with

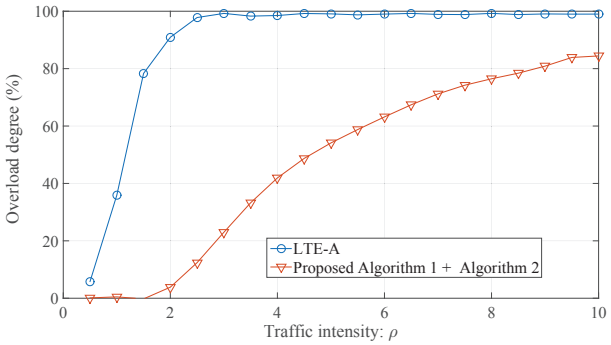

Fig. 5. Traffic intensity ρ vs. overload degree.

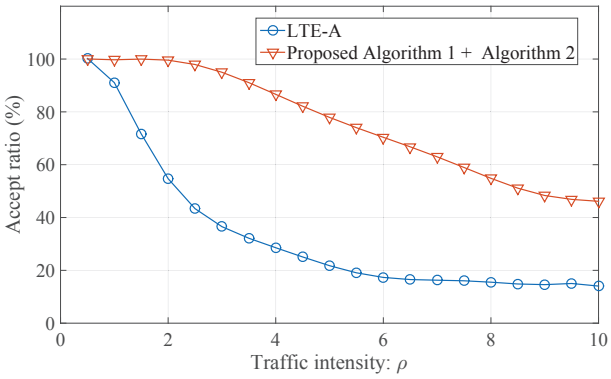

Fig. 6. Traffic intensity ρ vs. accept ratio.

a higher probability (see Fig. 6). However, the congestion control in LTE-A selects backoff value blindly. That is, a device may issue another request even if the NE is busy, resulting in more severe congestion in an NE.

Fig. 6 demonstrates that accept ratio decreases when ρ grows. The proposed algorithms choose an appropriate backoff value according to an NE's load. Thus, the device issuing a request will be rejected with lower probability. In addition, even though the Algorithm 2 uses $2 \sim 4$ NEs to distribute loads, we can see that the performance of the proposed algorithms has at least 3.5 times as much as that in LTE-A.

Fig. 7 depicts the relationship between ρ and average waiting time of an accepted request. From the results in Fig. 6 and Fig. 7, our proposed algorithms have higher accept ratio with less waiting time. However, randomly choosing backoff time from 0 to 1800 in LTE-A without considering an NE's load is not an effective way to reduce the waiting time of the devices. Eventually, congestion control in LTE-A will lead to more and more requests due to higher reject probability. In addition, even though the Algorithm 2 uses $2 \sim 4$ NEs to distribute loads, the waiting time of the proposed algorithms still has 10 times as much as that in LTE-A.

VI. Conclusion

In this paper, we identify a new congestion problem caused by MTC/IoT devices in next-generation networks and propose two algorithms to alleviate the congestion. The results demonstrate that the proposed algorithms have better performance for accept ratio, overload degree and waiting time. They can efficiently decrease and distribute the burden of highly-loaded NEs.

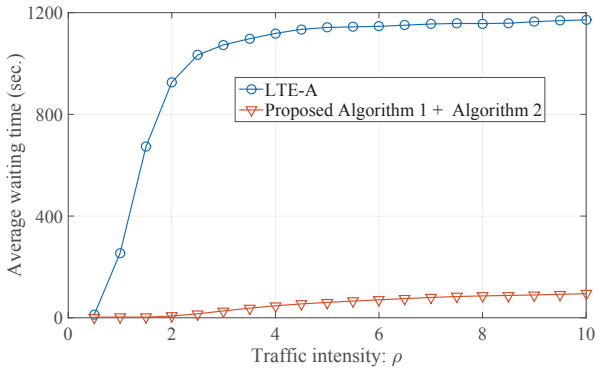

Fig. 7. Traffic intensity ρ vs. average waiting time.

References

[1] 3GPP, *Service Requirements for Machine-type Communications*, TS 22.368, 2014.

[2] "Cisco Visual NetNetwork Index: Global Mobile Data Traffic Forecast Update, 2013-2018," White Paper, 2014.

[3] 3GPP, *E-UTRA; RRC; Protocol Specification*, TS 36.331, 2015.

[4] 3GPP, *GPRS enhancements for E-UTRAN access*, TS 23.401, 2015.

[5] C.-W. Chang, J.-C. Chen, C. Chen, and R.-H. Jan, "Scattering random-access intensity in LTE Machine-to-Machine (M2M) communications," in *Proc. IEEE GLOBECOM*, 2013, pp. 4729–4734.

[6] C. M. Chou, C. Y. Huang, and C.-Y. Chiu, "Loading Prediction and Barring Controls for Machine Type Communication," in *Proc. IEEE ICC*, 2013, pp. 5168–5172.

[7] M. Tavana, V. Shah-Mansouri, and V. WS, "Congestion Control for Bursty M2M Traffic in LTE Networks," in *Proc. IEEE ICC*, 2015.

[8] 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 Trans. Veh. Technol.*, no. 5, pp. 2467–2472, 2014.

[9] Z. Wang and V. WS, "Optimal Access Class Barring for Stationary Machine Type Communication Devices With Timing Advance Information," *IEEE Trans. on Wirel. Communi.*, vol. 14, no. 10, pp. 5374–5387, 2015.

[10] G. Hasegawa, T. Iwai, and Naoki, "Temporal Load Balancing of Time-driven Machine Type Communications in Mobile Core Networks," in *Proc. IEEE IM*, 2015.

[11] A. Ksentini, T. Taleb, X. Ge, and Hu, "Congestion-aware MTC Device Triggering," in *Proc. IEEE ICC*, 2014, pp. 294–298.

[12] A. Ksentini, Y. Hadjadj-Aoul, and T. Taleb, "Cellular-based Machine-to-Machine: Overload Control," *IEEE Network*, pp. 54–60, 2012.

[13] A. Ahmed, A. Ahmed, Y. Hadjadj-Aoul, and T. Taleb, "Congestion Control for Machine Type Communications," in *Proc. IEEE ICC*, 2012.

[14] R. Nelson, *Probability, Stochastic Processes, and Queueing Theory: The Mathematics of Computer Performance Modeling*. Springer Science & Business Media, 2013.

[15] *Network Functions Virtualisation (NFV); Infrastructure Overview*, ETSI Std. NFV-INF 001, 2015.

[16] J. M. Lucas and M. S. Saccucci, "Exponentially Weighted Moving Average Control Schemes: Properties and Enhancements," *Technometrics*, vol. 32, no. 1, pp. 1–12, 1990.

[17] I. Petiz, P. Salvador, and AN, "Characterization and Modeling of M2M Video Surveillance Traffic," in *Proc. IEEE AFIN*, 2012.

[18] J. Park, S. M. Raza, P. Thorat, D. S. Kim, and H. Choo, "Network Traffic Prediction Model Based on Training Data," in *Proc. Spriner ICCSA*, 2015.

[19] D. Doran and Nathan, "Request Type Prediction for Web Robot and Internet of Things Traffic," *PeerJ*, vol. 3, 2015.

[20] "The Network Simulator NS-2," http://www.isi.edu/nsnam/ns/.