IEEE ANTS 2013 1569805705

Adaptive RACH Congestion Management to Support M2M Communication in 4G LTE Networks

Mukesh Kumar Giluka, Aiswarya Prasannakumar, Nitish Rajoria and Bheemarjuna Reddy Tamma
Department of Computer Science and Engineering
Indian Institute of Technology Hyderabad, India
Email: [cs11p1002, cs11m08, cs11m01 and tbr]@iith.ac.in

***Abstract*—Machine to machine communication (M2M) or machine type communication (MTC) facilitates communication of two network enabled devices, without any human intervention, to take some intelligent decision based on the interaction of devices. Because of ubiquitous coverage and global connectivity, cellular networks are playing a major role in the deployment of M2M communications. Due to some unique characteristics of M2M communication, supporting M2M applications in cellular networks is very challenging. One of such challenge is congestion in radio access network (RAN) during RACH procedure. This is because of the fact that there are large numbers of M2M devices which access the radio network at the same time. As a solution, we propose an adaptive RACH congestion management function (ARC) which specifies congestion handling method to be used by all M2M devices based on the current congestion condition of the network.**

I. Introduction

Machine to machine (M2M) communication [1] is an emerging communication concept where the goal of networking can be realized, fully or partially, with limited or no human intervention. The main motivation behind M2M communication is based on the observation [1] that after enabling communication between multiple machines through an underlying network, several applications were found which does not need any human participation. Due to this characteristic, M2M communication is becoming a market changing force for a wide variety of applications. According to the researchers [1] by the end of 2014, 1.5 billion devices and by the end of 2020, 20 billion devices will be part of M2M communication. According to 3GPP specifications [2] [3] M2M communication can also be termed as Machine Type Communication (MTC). In this paper, we have used the terms MTC and M2M interchangeably. Some of the M2M applications are smart grid, e-healthcare, smart homes, environmental monitoring, and industrial automation. These applications can be broadly classified into six categories [4]: fleet management, asset tracking, building security, modem, metering, telehealth.

Presently, cellular networks are optimized for Human-to-Human (H2H) and Human-to-Machine (H2M) communications and in future also, these seem to be uninterrupted because of the percentage of revenue contributed by these applications to the operators. At the same time, characteristics of H2H/H2M are different from that of M2M. In comparison to H2H/H2M, M2M typically has low traffic volume, high uplink to downlink traffic ratio, larger density of devices in a particular geographical area, and limited mobility of devices. With these differences, supporting M2M in current cellular networks is a big challenge to the operators. Among several challenges to be dealt with due to the incorporation of M2M applications in cellular networks, RACH congestion handling in radio access network (RAN) during RACH procedure is one of the biggest challenges. RACH congestion occur when large number of devices attempt for RACH procedure at a time. In this paper, we propose an adaptive RACH congestion management function (ARC) which based on the level of RACH congestion in the network, adaptively chooses the most efficient congestion handling method to overcome from congestion.

Rest of this paper is organized as follows. In section II, random access procedure (RACH procedure) in LTE systems is discussed. In section III, various existing congestion handling methods in RAN have been described. Section IV explains proposed congestion handling method and ARC function. Simulation results were analyzed in Section V and finally Section VI concludes the paper.

II. RACH Procedure

In 3GPP LTE system [5] when user equipment (UE) [6] or MTC device is switched on, they will try to connect to base station (also termed as eNodeB or eNB). Getting connected to the eNB is a four step procedure called RACH (Random Access Channel) procedure [7] [8]. RACH procedure is also required for timing synchronization, handover, etc.

Step 1: This step is called random access request (RA Req). Random access request happens in a special channel called PRACH (physical random access channel). In this step, the device sends a random access preamble to eNB in a PRACH slot. SIB2 (system information block) is broadcasted by the eNB which contains some essential information for devices in a cell which can be used for downlink synchronization. This also contains information about the location of PRACH slot in a frame. Six radio resource blocks are reserved for preamble transmission. Since these resource blocks are not scheduled for a particular device, so, any number of devices can send RA Req at the same time on these resource blocks. This can lead to congestion.

Step 2: This step is called random access response (RAR). RAR is sent by the eNB after receiving RA Req message. It consists of the detected random access preamble, the timing

advance for the device, TC-RNTI which is the temporary identifier for that device and uplink grant for the device to send the subsequent messages.

Step 3: This step is called terminal identification step. In this step, the device sends a message, as a reply of RAR message, to eNB which consists of C-RNTI of the device if it was previously connected to the eNB. If not, it will contain the core network terminal identifier.

Step 4: This step is called contention resolution step. In this step, the eNB will send contention resolution message to devices in DL-SCH (downlink shared channel). Each device receiving the response will check if it contains its identifier. If a device gets a reply then it sends an acknowledgement, otherwise it will backoff.

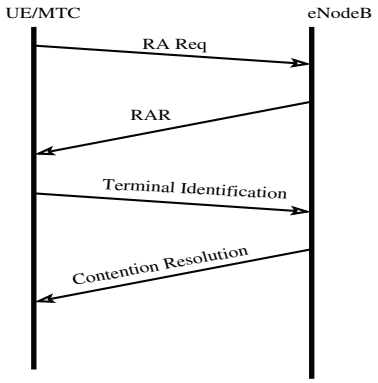

Fig. 1: RACH Procedure

III. Related Work

When a device wants to connect to RAN then it has to perform RACH procedure. If a large number of devices try to access the network at the same time then RACH congestion can occur. This is because there are only 64 preambles in a PRACH slot and if many devices try to access the network they may end up choosing the same preamble as another device and this will lead to collision which can result in failure of that preamble attempt. Such devices will then backoff and try to access the network at a later time. This leads to delays and more congestion at a later stage. Several methods have been proposed to solve the problem of RACH congestion in LTE RAN. These methods can be classified as follows:

Push based method: In push based methods [9] [10], the UE or MTC device will initiate the RACH procedure. One of the approaches under this method is p-persistent approach. In this approach, the UE will not send the random access preamble request message immediately but it will send with probability p. By doing this, congestion can be avoided up to some extent. The problem with this method is that even when congestion is not there, M2M device will send preamble with probability p. This will cause unnecessary delay in case of low congestion. Another approach is called Backoff Indicator Adjustment approach in which the backoff time for MTC devices is kept quite long so that more UE/MTC devices can be served before a particular device start resending the preamble request. This can also cause delay for sending M2M device's data.

Access barring method: In these methods [9], some devices are barred from accessing the network if there is possibility of congestion in both RAN and core network. In access class barring (ACB) approach, MTC devices are classified into different access classes. Single or multiple access classes are barred in case of a possibility of congestion. In extended access barring (EAB) approach, a device is given a status called EAB if it can tolerate some delay in accessing the network. In case of congestion, network can restrict devices having EAB status from accessing the network.

Pull based method: In pull based methods [9] [10], eNB will initiate the RACH procedure. Contention free approach is one of the pull based methods which is generally used at the time of handover. In this approach, the eNB will assign a contention free preamble to UE. UE will send the RA Req with this congestion free preamble. These methods can only be used in special cases like handover and not during normal RACH procedure.

Dynamic PRACH slot allocation method: In this method, UE and MTC devices are not allowed to access the PRACH slot at the same time. In separated PRACH slot allocation approach, PRACH resources are allocated either to UEs or MTC devices. In self-adaptive congestion approach [11], the concept of MTC PRACH is defined. MTC devices can send preamble request only in MTC PRACH. This approach proposes that if a device does successful RACH procedure then probability of success in next RACH procedure will be increased if the previous preamble is used again. If many of UE and MTC devices coming is bursty, this method can cause delay even when load on the network is low.

IV. Proposed Work

Various methods were discussed to handle the RACH congestion in previous section. Some methods avoid congestion up to some extent but as soon as number of devices accessing the network increases, their performances degrade. In this paper, we have divided the RACH congestion level into three categories: (i) No congestion scenario or scenario 1, when there is very less congestion or no congestion. (ii) Moderate congestion scenario or scenario 2, when there is a good amount of congestion (iii) Access barring or extreme congestion scenario or scenario 3, when network is unable to handle the congestion and we need to bar devices from accessing the network. In this paper, we have used the term no congestion scenario and scenario 1 interchangeably.

We propose a new randomized access dispersion based congestion handling method which is more suitable than existing congestion handling methods for scenarios 1 and 2. In the proposed method, Numbering Scheme (NS), when a M2M device does successful RACH procedure, the eNB assigns a number between 0 to n. If a device is assigned a number k then on its next network access, it will send the RA Req on k^{th} PRACH slot from the time of activation of the device.

In general in M2M communications, devices do not access the network continuously like in the case of periodic weather monitoring and reporting to the remote server. So each time when M2M devices access the network, RACH procedure is needed for uplink synchronization of them with eNB. If the device fails in k^{th} PRACH slot, then it will backoff and will send the RA Req on $2k^{\text{th}}$ PRACH slot. If the device fails even after, then it will no longer wait for the next k^{th} PRACH slot. It will follow the normal backoff scheme. Here, the value of n is chosen by eNB in such a way that simultaneous access of a number of devices is spreaded across some PRACH slots to reduce contention. At the same time average access delay should also be controlled.

To find out the suitability of different RACH congestion handling methods for no congestion scenario or moderate congestion scenario over other methods, we propose best congestion handling method selection algorithm (BCHMS). In this algorithm, performance of different RACH congestion handling methods is estimated for a particular condition of congestion in the network and best suitable method is chosen. The method which sustains the network for longer amount of time even if congestion increases, is most suitable for that level of congestion in the network.

After successful preamble transmission, each device informs eNB through terminal identification step that how many times it backed off before getting success. Based on this information, in BCHMS algorithm, the eNB calculates the average backoff (say b) per device at a particular time instant. If value of b is greater than a threshold value (λ), then there is a possibility of moderate or low congestion depending upon the value of λ. We denote λ as λ_1 for the threshold of low congestion or no congestion scenario and as λ_2 for threshold of moderate congestion scenario. So when b is greater than λ, one of the congestion handling methods is applied. After applying the method, average success rate will increase and b will decrease. But, as the number of devices accessing the network increases, average success rate decreases and b increases. Here, we measure the time t_1 elapsed to reach the success rate again same as before applying the method. Similarly, we measure the time (t_2) taken to reach the average backoff equal to λ. For example, let 1000 users or devices per second accessing the network and let presently success rate is 500 devices per second and average backoff per device is 50. Now, if one of the congestion handling methods is implemented, then surely success rate will increase and average backoff per device will decrease. Let the new success rate is 700 devices per second and average backoff is 30. But as number of devices accessing the network increases, success rate will start decreasing and average backoff will start increasing. The time elapsed to come down the success rate from 700 to 500 devices per second is termed as t_1 and the time elapsed to reach the average backoff from 30 to 50 is termed as t_2. After getting t_1 and t_2, we calculate k from following equation:

$$k = t_1 + \alpha t_2$$

Here α is a constant whose value lies between 0 and 1. We have taken α as 0.7. Here, we run the algorithm for different congestion handling methods. k value gives the suitability of that congestion handling method. The method with maximum k value is the most suitable congestion handling method for that particular network condition.

Algorithm 1 BCHMS Algorithm

Input: Congestion handling methods, Start time, λ
Output: t_1, t_2

1: $N = 1000$ {Number of devices present in the cell}
2: **if** $current\_time < start\_time$ **then**
3: wait
4: **else**
5: find b {Average backoff calculated by eNB after receiving terminal identification message (MSG3) by all the devices}
6: **if** $b > \lambda$ **then**
7: find S_1 {Average success rate before applying the algorithm}
8: Apply the algorithm
9: find S_2 and b_2 after next PRACH slot. {S_2 is average success rate after applying the algorithm and b_2 is average backoff calculated by eNB after receiving MSG3 by all the devices}
10: start timer t_1 and t_2
11: **if** $S_1 \geq S_2$ *or* $b_2 > \lambda$ **then**
12: Return $t_1 = -1$ and $t_2 = -1$
13: **else**
14: $N = N + 1000$
15: find S_2 and b_2 after next PRACH slot.
16: **if** $S_1 \geq S_2$ *and timer* t_1 *is not stopped* **then**
17: stop the timer t_1
18: **end if**
19: **if** $b_2 \geq \lambda$ *and timer* t_2 *is not stopped* **then**
20: stop the timer t_2
21: **end if**
22: **if** both timer stopped **then**
23: return t_1 and t_2
24: **else**
25: go to step 14
26: **end if**
27: **end if**
28: **else**
29: $N = N + 1000$
30: go to step 5
31: **end if**
32: **end if**

The ultimate goal of this paper is to propose a function, adaptive RACH congestion management function (ARC), which adaptively chooses most efficient congestion handling method, given by the BCHMS algorithm, for a particular level of congestion in the RAN. The function uses congestion estimation subroutine to decide that RAN is under which

congestion scenario. Before calling congestion estimation subroutine, the function calculates number of devices falling under scenario 1, scenario 2 and scenario 3. Number of devices falling under scenario 1, scenario 2 and scenario 3 are denoted by $devScen1$, $devScen2$ and $devScen3$, respectively. λ_1 and λ_2 are thresholds of scenario 1 or no congestion scenario and scenario 2 or moderate congestion scenario, respectively. So, if the number of times backoff done by a device before successful preamble transmission is less than λ_1 then the device falls under scenario 1 but if it is greater than λ_1 and less than λ_2 then the device falls under scenario 2 otherwise it falls under scenario 3. Now the function calls congestion estimation subroutine and passes these values as parameters. It also passes $prevScen$ and $tolerance\_factor$ as parameters. The parameter $prevScen$ denotes the scenario under which RAN is falling at the time of calling of the subroutine by the function. The subroutine finds that which variable among $devScen1$, $devScen2$ and $devScen3$ is $tolerance\_factor\%$ greater than other two. If that variable is $devScen1$ then the subroutine returns scenario 1 or no congestion scenario and if it is $devScen2$ then subroutine returns scenario 2 or moderate congestion scenario otherwise scenario 3 or extreme congestion scenario.

Algorithm 2 ARC Function

Input: $tolerance\_factor$
1: start timer
2: $devScen1 = 0$, $devScen2 = 0$,$devScen3 = 0$; {number of devices existing in No Congestion Scenario,2 and 3}
3: $prevScen = NONE$;
4: **if** new device successfully connects to eNB **then**
5: **if** $newdevice.backoff \leq \lambda_1$ **then**
6: $devScen1 = devScen1 + 1$;
7: **end if**
8: **if** $\lambda_1 < newdevice.backoff \leq \lambda_2$ **then**
9: $devScen2 = devScen2 + 1$;
10: **end if**
11: **if** $newdevice.backoff > \lambda_2$ **then**
12: $devScen3 = devScen3 + 1$;
13: **end if**
14: **end if**
15: **if** timer expired **then**
16: $congScen$ =CongestionEstimationSubroutine ($devScen1$,$devScen2$,$devScen3$,$prevScen$, $tolerance\_factor$)
17: $prevScen = congScen$
18: broadcast the information to M2M devices to start congestion handling method for $congScen$
19: go to step 1
20: **else**
21: go to step 3
22: **end if**

If no variable among $devScen1$, $devScen2$ and $devScen3$ has sufficient value then subroutine returns previous scenario if the previous scenario is scenario 1 or scenario 2 but if previous scenario is scenario 3 then it returns scenario 2 because in scenario 3, eNB uses EAB as a congestion handing method which bars some of the devices from accessing the network. So, if the difference between the values of $devScen3$ and λ_2 is not much, then it is better to return scenario 2 or moderate congestion scenario in place of returning scenario 3 or extreme congestion scenario. Extreme congestion scenario happens when almost all devices are unable to access the RAN due to congestion. So if we have a good amount of devices in scenario 2 then it is better to keep the network in scenario 2. Now based on the returned value, function chooses a congestion handling method suggested by BCHMS algorithm. After choosing the appropriate method, the function broadcasts this message to all devices in the coverage area of eNB so that devices will perform RACH procedure with the method suggested by eNB.

Algorithm 3 Congestion Estimation Subroutine

Input: $devScen1$,$devScen2$,$devScen3$,$prevScen$, $tolerance\_factor$
Output: Congestion Scenario
1: $x = tolerance\_factor$
2: **if** $devScen1 > (100+x)devScen2/100$ and $devScen1 > (100+x)devScen3/100$ **then**
3: $scenario = scenario1$
4: return $scenario$
5: **else if** $devScen2 > (100+x)devScen1/100$ and $devScen2 > (100+x)devScen3/100$ **then**
6: $scenario = scenario2$
7: return $scenario$
8: **else if** $devScen3 > (100+x)devScen1/100$ and $devScen3 > (100+x)devScen2/100$ **then**
9: $scenario = scenario3$
10: return $scenario$
11: **else**
12: **if** $prevScen == scenario3$ **then**
13: $scenario = scenario2$
14: return $scenario$
15: **else**
16: $scenario = prevScen$
17: return $scenario$
18: **end if**
19: **end if**

V. Simulation Results and Analysis

MatLab has been used for simulation of congestion in RACH procedure. All simulation parameters are taken from 3GPP specifications 37.868 [9]. We have taken user or device arrival distribution as beta distribution. Simulation parameters are given in Table I.

In Table II, for different values of n in numbering scheme, number of successful users and average access delay are shown for different number of users. From $n = 4$ to $n = 8$, number of successful users are increasing but from $n = 8$ to $n = 12$

TABLE I: Simulation Parameters

Parameters	Values
Number of preambles	54
Number of MTC devices	1000 to 30000
Number of preamble retransmissions	10
HARQ retransmission probability	10%
Preamble detection probability	1- $1/e^i$ where i is the i^{th} preamble transmission
PRACH slots per frame	1
$\lambda_1, \lambda_2, \alpha$	3, 10, 0.7
Simulation Time	10s
BackOff Time	20ms
User Arrival Distribution	Beta Distribution

and from $n = 12$ to $n = 16$, rate of increasing of successful users is low. From $n = 4$ to $n = 8$, average access delay is either decreasing or slightly increasing because in case of $n = 4$ success rate is low while for $n = 8$, success rate is more but due to more spread accesses of users, average access delay is affected. From $n = 8$ to $n = 12$ and from $n = 12$ to $n = 16$, spread in access is more, so rate of increasing of average access delay is more. So, from above analysis we can conclude that most suitable value for n is 8.

In Table III, results of BCHMS algorithm is shown. The required parameter λ is calculated by simulating the RACH procedure without any congestion handling method (base case) and we found that value of λ for scenario 1 i.e. λ_1 is 3 and value of λ for scenario 2 i.e. λ_2 is 10. We run the algorithm for various congestion handling methods specified in 3GPP specifications and note the value of t_1, t_2 and k as shown in Table III. From the table we can observe that in case of no congestion scenario, value of k is highest for Numbering Scheme (NS) while in case of moderate congestion scenario value of k is highest for p-persistent. So from Table III, we can conclude that NS is best for no congestion scenario and p-persistent is best for moderate congestion scenario. Further, these results are used by ARC Function. As discussed in previous section, when eNB runs ARC Function at a particular time instant, it first estimates the condition of congestion in the network. If condition of congestion is no congestion scenario then eNB chooses NS as congestion handling scheme as concluded from Table III. Similarly, if it is moderate congestion scenario then eNB chooses p-persistent and if congestion level is extreme congestion scenario, then eNB chooses EAB. So by using this function, eNB alleviates the effect the congestion significantly.

TABLE II: Different values of n in Numbering Scheme

No. of users	No. of successful users				Average access delay			
	n =4	n =8	n =12	n =16	n =4	n =8	n =12	n =16
10000	7031	7126	7193	7291	235	229	222	215
20000	4571	6471	6556	6621	595	564	653	688
25000	3822	5924	6054	6126	530	542	549	684
30000	3697	4919	5032	5037	569	566	692	725

TABLE III: Results of BCHMS algorithm

Schemes	No Congestion			Moderate congestion		
	t1	t2	k	t1	t2	k
Numbering	195	195	331.5	60	65	105.5
Slotted Access	45	55	83.5	25	45	56.5
BackOff Indicator Adjustment	40	60	82	30	45	61.5
p-persistent	160	210	307	80	70	129

To evaluate the performance of ARC function, we use following parameters: (i) number of successful users (ii) average access delay per device (iii) average backoff per device. Average access delay per device is defined as average of difference between the time when a device is ready to start RACH procedure and the time when its RACH procedure finishes successfully. Average backoff per device is defined as average number of backoff done by a device before being successful.

Comparison of performance of ARC function with other congestion handling methods is shown in Figure 2, Figure 3, Figure 4 in terms of number of successful users, average access delay per device and average backoff per device respectively. Since the simulation time is 10 seconds, if number of users are 10000 it means that 10000 users or devices attempted to do RACH procedure in 10 seconds. In the figures base case is the case when there is no congestion handling method is used to evaluate the performance parameters.

Figure 2 shows the number of users versus number of successful users graph. In the figure, we can see that some of the methods like backoff indicator adjustment, base case, slotted access and NS perform good when number of users are less. Among these methods NS performs better. But when number of users increase, their performance degrade. In case of p-persistent, its performance improves upto certain extent when number of users are more. Because, when number of users are less, due to the factor p so many devices are unable to access the network. But when number of users are more, the same p helps to spread the access and consequently success rate increases. But, the ARC function perform best in both conditions because it chooses best congestion handling methods of both congestion scenarios.

Figure 3 shows the number of users versus average access delay graph. Performance of backoff indicator adjustment is not good because of longer backoff. In figure 3 as number of users increase, average access delay increases fastly for ARC function in comparison to NS and p-persistent because in beta distribution occurrence of EAB condition will be more. As we know that during EAB, some of the active devices will be barred from accessing the networks and they will be allowed to access the network only after removal of EAB condition. So, average access delay will be more for ARC function.

Figure 4 shows the number of users versus average backoff per device graph. As number of users increase, success rate decreases rapidly in case of backoff indicator adjustment, base case, slotted access and NS. So, average backoff will increase

rapidly. ARC function performs best because when congestion is higher, EAB will be implemented. So, some of the devices will be barred. So, average backoff per device will decrease. In case of other congestion handling methods, they will not be barred even in high congestion so devices will keep on accessing the network. As a result, average backoff per device will increase.

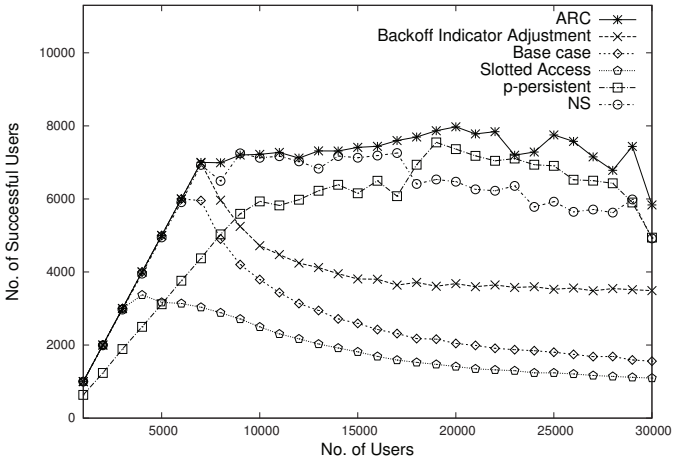

Fig. 2: Number of Successful Users

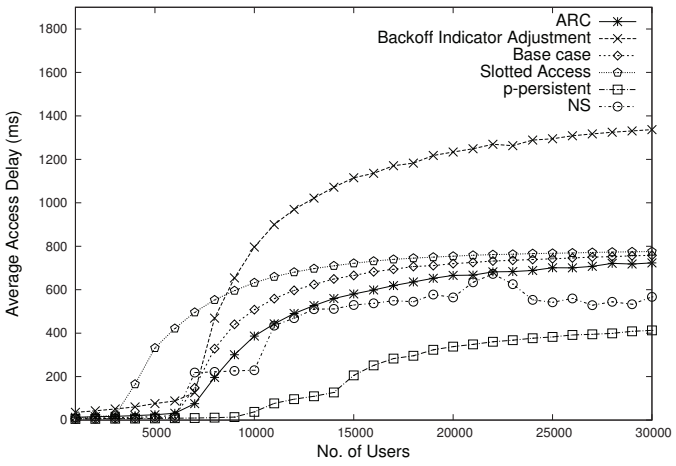

Fig. 3: Average Access Delay

VI. Conclusions

In this paper we studied the performance of various RACH congestion handling methods. We saw that each method performs best with some particular congestion state of the network and does not perform very well when the load in the network is different. We proposed a novel congestion handling method, numbering scheme (NS), which performs very well when the load on the network is low to medium. We also proposed an algorithm, BCHMS Algorithm, by which we can find out which method performs better in a given congestion state of the system. A novel RACH congestion

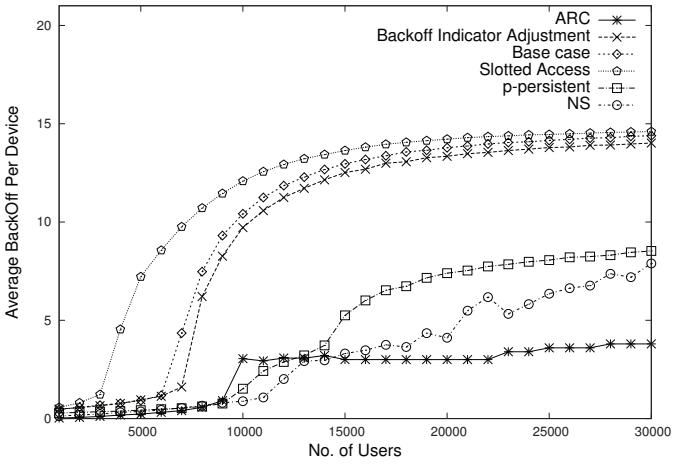

Fig. 4: Average BackOff Per Device

management function (ARC), was proposed which estimates current load in the cell and then decides which congestion handling method is to be used by MTC devices. Simulation results show that ARC function performs better than any single congestion handling method and is able to perform well in both low and high RACH congestion cases. When the load of the system becomes too high for eNB to handle, then it will perform EAB mechanism to do admission control.

Acknowledgment

This work was supported by the Deity, Govt of India (Grant No. 13(6)/2010CC&BT).

References

[1] Min Chen, Jiafu Wan and Fang Li, *"Machine-to-Machine Communications:Architectures, Standards and Applications"*, *Transactions on Internet and Information Systems, vol. 6, no. 2*, February 2012.

[2] 3GPP TS 22.368 V11.3.0, *"Service requirements for Machine-Type Communications (MTC);Stage 1"*, September 2011.

[3] 3GPP TR 22.868 V8.0.0, *"Study on Facilitating Machine to Machine Communication in 3GPP Systems"*, March 2007.

[4] M. Zubair Shafiq, Lusheng Ji, Alex X. Liu, Jeffrey Pang and Jia Wang, *"A First Look at Cellular Machine-to-Machine Traffic Large Scale Measurement and Characterization"*, SIGMETRICS, June 2012.

[5] 3GPP TS 36.321 V10.0.0, *"Evolved universal terrestrial radio access (E-UTRA) medium access control (MAC) protocol specification"*, December 2010.

[6] 3GPP TS 36.101 V10.4.0, *"User Equipment (UE) radio transmission and reception"*, September 2011.

[7] 3GPP RAN2 69bis, R2-102296, *"RACH Intensity of Time Controlled Devices"*, Vodafone, April 2010.

[8] 3GPP RAN2 69bis, R2-102297, *"RAN Mechanisms to Distribute RACH Intensity"*, Vodafone, April 2010.

[9] 3GPP TR 37.868 V0.5.1, *"Study on RAN Improvements for MachineType Communications"*, August 2010.

[10] Ming-Yuan Cheng, Guan-Yu Lin, and Hung-Yu Wei, *"Overload Control for Machine-Type-Communications in LTE-Advanced System"*, *IEEE Communications Magazine*, vol. 50, no. 6, pp. 38-45, June 2012.

[11] Shiann-Tsong Sheu, Chun-Hsiang Chiu, Yen-Chieh Cheng, and Kai-Hua Kuo, *"Self-Adaptive Persistent Contention Scheme for Scheduling Based Machine Type Communications in LTE System"*, iCOST, July 2012.