Efficient, Adaptive and Scalable Device Activation for M2M Communications

Arijit Banerjee[1], Binh Nguyen[1], Vijay Gopalakrishnan[2], Sneha Kasera[1], Seungjoon Lee[3] and Jacobus Van der Merwe[1]

[1]School of Computing, University of Utah
[2]AT&T Labs - Research
[3]Two Sigma Investments

***Abstract*—When traffic arrives from the network for an idled mobile device, the network executes device activation procedures to wake the device up. Current device activation mechanisms are ill suited to support the expected growth of machine-to-machine (M2M) devices and traffic. We propose an adaptive device activation architecture for LTE/EPC cellular networks that adapts to network conditions and M2M application requirements to realize scalable device activation without increasing the resources used for this purpose. Our evaluation shows that our adaptive approach enables the network to handle M2M applications with a large number of devices without negatively impacting existing human-to-human (H2H) and human-to-machine (H2M) traffic.**

I. Introduction

Machine-to-machine (M2M) communication, where smart devices function without direct human mediation, is rapidly becoming commonplace. The scale and unique communication requirements of M2M pose new challenges to wide area wireless communication infrastructure like cellular networks that have been engineered and optimized for human initiated communication.

Our work is concerned with the fundamental cellular network function of *device activation*: To preserve network resources and energy, devices that are not actively communicating enter an idle state after a short period of inactivity. Specifically, when devices enter this idle state, network state associated with the device, specifically across the radio access network (RAN), is released. Device activation then refers to the mechanisms involved in "waking the device up" from this idle state and restoring the network state needed to enable the device to communicate.

Device activation is not a new concept but has been part-and-parcel of cellular networks since their inception, and amounts to almost 30% of the total signaling traffic in current LTE deployments [1]. The expected growth of M2M communication in future cellular networks, however, suggests that existing device activation mechanisms will be inadequate. Current device activation mechanisms were developed to support a single service, namely human-to-human (H2H) voice communication. Growth of data traffic from smart phone use caused this initial use to be replaced by the current predominant human-to-machine (H2M) communication as users use their cellular devices to access Internet services. The device activation workloads associated with these communication patterns are by necessity very different from the workload that might be expected in an M2M environment. For example, a server contacting thousands of smart meters every hour on the hour, will present a radically different step function in terms of offered load to the device activation mechanisms. More importantly, M2M communication is expected to dominate cellular network communication presenting a scalability problem that current mechanisms are simply not able to cope with. A recent M2M study [2] suggests that the percentage of connected M2M devices will grow from 23% in 2012 to 61% in 2022 and that the number of M2M connections is expected to grow at an annual rate of 22% from 2 billion in 2012 to 18 billion in 2022. These numbers suggest that the onslaught of M2M devices and traffic are poised to overwhelm existing cellular network mechanisms, thereby negatively impacting the experience of mobile users at a time when the importance of cellular networks continues its unabated growth.

In this paper, we explore the impact of M2M growth on the specific cellular network function of device activation. We first show that expected M2M use cases will indeed overwhelm current device activation mechanisms, leading to unacceptably large activation times, or even failure to activate devices. We also show that network-agnostic naive strategies to address these concerns simply shift the overload condition from one mechanism to another. Based on these insights we argue that there is a need for a holistic and adaptive approach to device activation. We propose an architecture for cellular device activation that will allow the network to dynamically adapt the manner in which it performs device activation by taking into account current network conditions as well as M2M application specific dynamics.

Our specific aim is for the network to efficiently perform device activation without requiring additional radio resources, with minimal changes to existing mechanisms and without relying on applications to adapt their behavior to protect the network. Towards this aim, the focus of our work in this paper is to develop a detailed understanding of the existing device activation mechanisms in LTE access networks, to identify the primary bottlenecks associated with such mechanisms that impact the network performance and to explore the realm of adaptive techniques that the network could apply to perform efficient device activation even in the presence of large scale M2M arrivals. Specifically, we propose a group-based device activation procedure, using a co-operative paging and random access scheme, that dynamically adapts the group sizes based on the current network conditions.

We note that overload control for M2M arrivals is considered a high priority item by 3GPP standards body [3], and different methods for overload control are proposed including group based access of M2M devices [4]. Our work is aligned with these efforts, and we address many of the issues left open by these earlier proposals concerning group assignment and management, optimal group size selection, adaptive load distribution, and (sub)group activation scheduling. Also, our methods do not require access barring or exclusive resource reservation for M2M to prevent network overload unlike some of the earlier proposed schemes.

One of our key design principle is not to change the legacy H2H devices, and adaptation of the M2M device activation procedures to minimize the impact on the existing H2H communications. Our work, thus, also supports the broader thinking that different device/applications will be treated differentially and our grouping based approach, while following the same adaptive methods within a group, also adapt to different (access) methods used across groups and provides an architecture to support such differential group treatment.

We specifically make the following contributions:

•We demonstrate, using data driven simulation, the fragile nature of current device activation mechanisms and the negative impact that M2M applications have on existing H2H and H2M traffic. Our results show that, large scale M2M arrivals can cause unacceptably high failure rates (as high as 38.58%), and increased device activation time (factor 4 increase) when using the existing mechanisms (Section III).
•We present an adaptive device activation architecture that allows the network to accommodate M2M applications without overloading the network. Specifically, our architecture enables dynamic group formation of M2M devices and execute group based activation mechanism to prevent network overload conditions (Section IV).
•We design group based activation algorithms that allow the network to dynamically adapt to network conditions and to efficiently distribute the M2M load ensuring minimal impact on the ongoing H2H communications. Specifically, we propose a co-operative paging and random access scheme (Section II) that allows efficient distribution of M2M load based on the availability of the limited control plane resources. Our approach consists of an adaptive backoff scheme that enables efficient distribution of M2M load, and an algorithm to select the optimal paging group sizes dynamically, based on current network conditions. Our algorithm also ensures proper separation of consecutive group accesses to achieve very low failure rates (Section IV).
•We implement a simulator that models the LTE device activation procedure to evaluate our algorithms using data

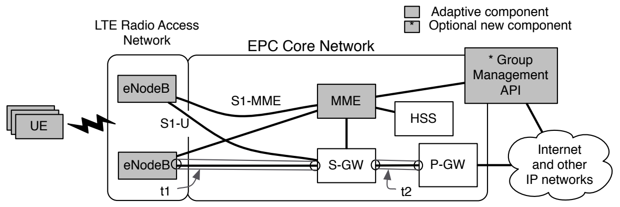


Fig. 1: LTE/EPC architecture with adaptive components.

from a large US cellular provider as background (H2M) traffic (Section V). We evaluate our methods with a demanding M2M use case (30000 arrivals/10s) to show the efficient scalability of our adaptive algorithms under extreme workloads. Our algorithms are equally applicable to other (not so demanding) scenarios as they dynamically adapt to network conditions to ensure efficient and optimal use of limited network resources. Our results show that our adaptive mechanisms can achieve almost zero failure rates in device activation, with negligible impact on H2H communications, even in the presence of very demanding M2M arrivals (Section VI).

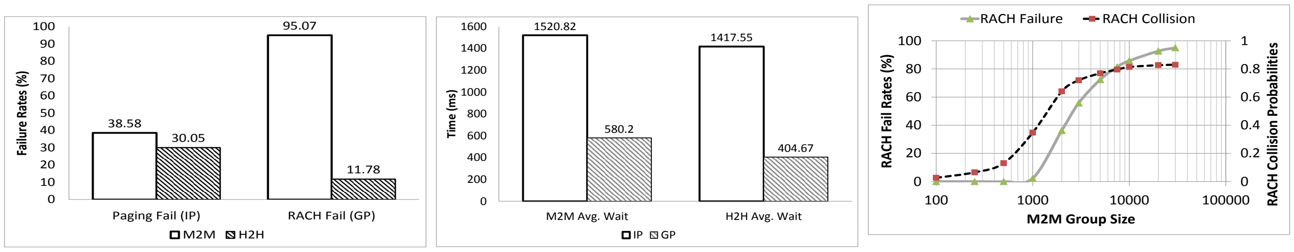
II. Background

In this section we describe the background related to our work in the context of the Long Term Evolution (LTE) and Evolved Packet Core (EPC) technology.

LTE/EPC Architecture. Figure 1 depicts the components of an LTE/EPC mobile system. (The shaded components in Figure 1 form part of our adaptive approach, which we will discuss in Section IV.) LTE RAN consists of eNodeBs, which communicate with mobile devices (i.e., UEs) via the radio link. The eNodeB also performs radio resource control and cooperates with the MME (Mobility Management Entity) for mobility management. The EPC consists of the MME, the S-GW (Serving Gateway), and the P-GW (Packet Data Network Gateway). The MME is a control plane only function responsible for user authentication via the HSS (Home Subscriber Server) and mobility management. It also interacts with the S-GW for data session establishment/release. The S-GW and the P-GW are on the data path, and their main function is packet routing/forwarding.

UE Idle State. When the UE has no data to send, it goes to an idle (low energy state) with no active radio connection and the data tunnel t1 is torn down by the network. In this state, the location of the UE is known to the network in the form of a tracking area (TA) list of eNodeBs, which the MME assigns to the UE. An idle UE wakes up periodically according to a configured discontinuous reception (DRX) cycle to check for potential activation requests (paging) from the eNodeB.

LTE/EPC Device Activation. LTE/EPC device activation consists of two procedures, paging and random access (RACH) [5]. If a data packet arrives from the external network for a UE in the idle state, the MME sends a paging request to all the eNodeBs in the TA list of the UE. The eNodeBs, in turn, calculates the paging slot for the UE based on the UE identifier and sends a paging indication in the calculated slot. The UE wakes up periodically (once per DRX cycle) to monitor the paging channel for incoming paging requests in



(a) Paging and RACH Failure Rates (b) Overall Wait Time Comparison (c) Group Size Impact on RACH

Fig. 2: Individual Paging (IP) and Naive Group Paging (GP): Group Size Impact.

its corresponding slot. If it finds a paging indication in the slot and one of the UEID(s) (multiple UEIDs can map to the same paging slot) sent in the corresponding paging message matches its own identifier, the UE starts connection setup towards the network using the random access channel (RACH) procedure [6]. Both paging and RACH related messages are sent on shared control channel with limited resources. The network is limited in the number of devices it can activate per paging or random access slot. Hence if a large number of devices contend for these limited resources at the same time, it will lead to excessive activation failures and activation delays.

III. Motivation

We illustrate the fragile nature of current LTE/EPC device activation mechanisms by considering a demanding, but realistic, use case involving smart meters in a dense urban environment [3]. Specifically, the use case involves 30,000 M2M devices (smart meters) per cell, being activated by the network in a 10 second interval to report status, for example because of a restored power failure or periodic reporting - frequently used in current M2M deployments [7]. We are assuming a pull-based model in this example which is aligned with the approaches advocated by 3GPP standards body [4].

We use our data driven simulation framework, described in detail in Section V, to show the inadequacy of the current device activation procedure to handle the M2M induced overload scenario described above (30000 request arrivals over 10s). The first column in Figure 2a shows that for the demanding M2M load, the existing individual paging (IP) mechanisms result in a 38.58% paging failure. The impact on background H2H traffic is equally severe with a 30.05% paging failure. Note that, these results are for a network configured with very high paging capacity (detailed experimental setup is described in Section V). In operational networks, the practical capacity is much less [8], so even less demanding M2M use cases will cause similar breakdown of the current procedures.

At the same time, a network agnostic naive grouping strategy (GP), where all the M2M devices are paged as a single group to solve the paging overload problem, will overwhelm the current RACH procedure as shown in the second column of the Figure 2a. Figure 2c shows how the RACH performance deteriorates with increasing M2M group arrival sizes.

The M2M induced overload scenario also increases the device activation time for both M2M and background H2H traffic (Figure 2b). We note that 3GPP target device activation time (excluding paging delay) is below 50 ms [9]. High activation times negatively impact user-perceived performance, and can also lead to timeouts in long-lived TCP connections, typically maintained by application servers to current smartphones [10], [11].

These results motivate us to design an adaptive group-based device activation procedure that dynamically adapts the paging group size based on the available RACH capacity. In essence, we propose a collaborative design that performs a joint optimization of the paging and the RACH procedure, based on the available limited control plane resources, at the same time ensuring that existing H2H communications are minimally impacted by demanding M2M use cases. An alternate design could be to use a single paging-group, and a sufficiently large random backoff window in RACH to reduce contention. We do not consider this alternate design because of its inflexibility to adapt to current channel condition. The available channel capacity continuously changes as devices from different applications with diverse access patterns access the network. A conservative selection of the backoff window to deal with the varying channel capacity will lead to inefficient channel utilization, on the other hand, an aggressive backoff window will result in increased collisions in presence of temporal high load. Spreading M2M paging into batches controlled by the network gives finer control as time evolves, and allows efficient utilization of the scarce resources by opportunistically adapting the batch sizes based on network conditions.

IV. Adaptive Device Activation

In this section, we present the architecture and the adaptive algorithms which enable efficient, scalable device activation in the context of an LTE/EPC network architecture. Fundamental to our approach is the ability for mobile devices to be paged as a group, and dynamically adapt the group assignment based on the network conditions. The adaptive components of our design are described below.

A. Adaptive Activation Architecture:

Figure 3 depicts the key components of our design. Our design enables the network to page a large set of devices using batches of chunks (subgroups), and to dynamically adapt the chunk sizes based on the current network condition.

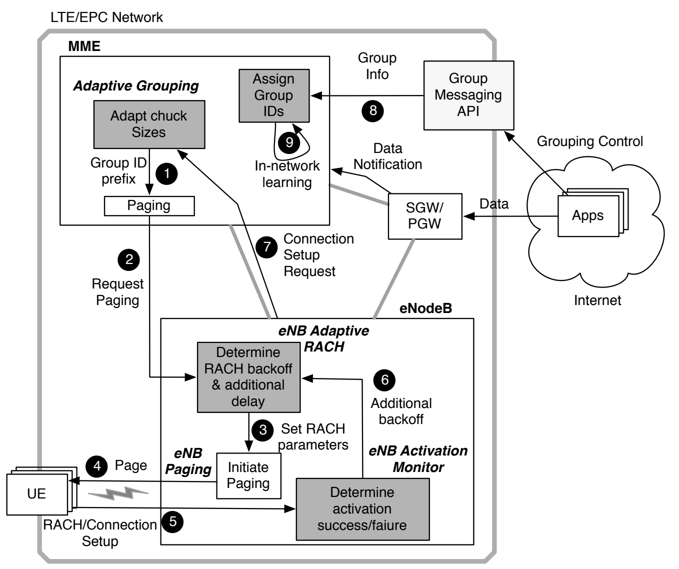


Fig. 3: Adaptive Activation Architecture

The adaptive grouping function in the MME is responsible for selection of a suitable chunk (subgroup) size, and for generation of a proper Group ID (#1) to send a paging request (#2) for the corresponding chunk to the eNB. The eNB, in turn, determines suitable backoff parameters based on the selected chunk size to ensure low collision probabilities in the RACH access (#3), and sends a paging request across the RAN (#4) for the corresponding chunk. The UEs belonging to the chunk initiate their RACH access (#5) based on the eNB indicated backoff parameters. The eNB adjusts the RACH parameters for the subsequent chunks based on the RACH completion rate of the previous chunks (#6). Similarly, the MME adapts the current chunk size based on the connection setup request rates received from the eNB (#7). The Assign Group IDs function is responsible for an initial adaptable group identifier assignment to the UEs based on either explicit group information provided by applications (#8) or by in-network learning in the MME (#9). In section IV-C we present a mechanism that the network can utilize for such adaptive group assignment.

B. Adaptive Methods:

Initial RACH Backoff: Paging a large set of devices as a group improves the paging channel efficiency significantly but as a side effect, leads to dramatic increase in RACH failures and collisions. We present an adaptive backoff scheme, based on the group size, to prevent devices in the same group from competing for the random access channel at the same instant. We propose that when a group of N devices are paged using a single group identifier, all the devices in the group select and wait for a random amount of time B_{init} in $[0, f(N)]$, before starting the RACH procedure. $f(N)$ is calculated as follows:

$$f(N) = (N/N_{RA}^{max}) * T_{RA} \tag{1}$$

where T_{RA} is the periodicity of random access slots as configured by the network, N_{RA}^{max} denotes the maximum number of devices that can complete the RACH procedure in each slot (Section V). Therefore, $f(N)$ is an estimate of the time taken for a group of N devices to complete the RACH procedure in an ideal scenario with no background traffic and no collisions. The intuition behind our choice of the backoff window, $[0, f(N)]$, is that if everything goes well, the devices would backoff just for the right amount of time and initiate the RACH procedures such that there are no collisions while all RACH slots are fully utilized. The eNB adaptive RACH function (Figure 3) computes the value of f(N) and communicates the same to the devices in the paging message itself.

We acknowledge that, the concept of using random backoff to distribute load is not new, and also proposed as solutions for overload control in 3GPP([4]). Our contribution is that we design a simple and yet efficient backoff scheme that can adapt to different M2M arrival patterns and, as we see below, provides the flexibility to distribute a group arrival in smaller chunks according to network conditions, dynamically.

Algorithm 1 Algorithm for finding δ_a for chunk i of size C

```
1: if (At least one of the previous chunks have completed RACH access) then
2:     $R_C$ = RACH access completion time for the most recent completed chunk
3: else
4:     $R_C$ = K * f($C$) (from Equation 1)
5: end if
6: numRemaining = number of devices of chunk $(i-1)$ still accessing the RACH
7: If chunk (i-1) has not started RACH access yet, numRemaining = $C$
8: $\delta_a$ = ($R_C$/$C$) * numRemaining
```

Ensuring Non-Overlap in RACH Access Across Paging Groups: While using smaller chunks to activate a large number of devices, it is possible for the paging request for a chunk to arrive at a time when the RACH procedure for some previous chunk(s) is in progress. In such scenarios, RACH access attempts of the different chunks will overlap with each other leading to increased RACH collisions and failure. To prevent this, we propose that the eNB sends an additional delay parameter δ_a in the paging message and all the devices belonging to the chunk defer their RACH attempt by an additional δ_a time. The eNB activation monitor function (Figure 3) monitors the RACH completion times (R_C) of the previous chunks and passes the information to the eNB Adaptive RACH function, which, in turn, uses the most recent R_C value to dynamically estimate δ_a using Algorithm 1. R_C is estimated based on the measurement of the most recent RACH access completion time of a chunk. In the case where some devices of a chunk fail to complete the access procedure, the eNodeB can detect such failures using existing timeout-based methods, and determine the chunk access completion time based on the information of both successful and failed access counts.

Initially when an estimate of R_C is not available, the eNB conservatively sets $R_C = K * f(C)$ where the factor K accounts for additional delays related to background load, collisions etc. The factor K only impacts the first few chunks until one of the previous chunks completes RACH access, and can be set based on past history (we use $K = 3$ in our evaluation). The eNB sends $f(C)$ and δ_a along with the group identifier in the paging message, where C is the chunk size, and all devices belonging to a chunk first wait for δ_a time and then initiate a backoff for a random time in $[0, f(C)]$ before starting the RACH procedure.

Instead of requiring the devices to wait for an additional δ_a time, the eNB could possibly delay sending the paging message by the same amount of time. However, we do not use this alternative because a paging message can only be sent in the specific slots corresponding to the group identifiers, and waiting for the next paging slot will incur unnecessary delays (at least one paging DRX cycle) in the access completion.

Automatic Chunk Size Selection: We now present a method that allows the network to dynamically estimate, based on the current control channel load, a chunk size that provides low failure rates for both paging and RACH while minimizing the RACH access time. In our method, the network initially selects a default minimum chunk size (e.g., 100) that avoids high paging failures (very small chunk sizes will result in large number of paging messages being sent out, thereby increasing paging collision and failure rates). Then the network dynamically adapts the chunk size based on the observed network conditions.

In our approach, the network selects the chunk size C such that

$$N_C^T = T/d_C \leq P_s \tag{2}$$

and,

$$N_C^T * R_C \leq T \tag{3}$$

where T is the paging DRX cycle, d_C is the arrival duration of a chunk of C devices, N_C^T is the number of chunks that arrive within a paging DRX cycle, P_s is the paging slot capacity, and, R_C is the estimated RACH access delay for chunk size C. Equation 2 ensures that we do not exceed maximum capacity of a paging slot, and Equation 3 ensures that all previous chunks have completed their access procedure before the new set of chunks arrive in the next DRX cycle. If no C that satisfies equations 2 and 3 is found, we repeat the search for a suitable C over two DRX cycles, i.e., we try to find C, such that chunks arriving over $2T$ can complete their RACH access in $2T$ without exceeding the overall paging capacity $2*P_s$. Even after searching over two DRX cycles, if a suitable C is not found, we choose C such that $d_C \;\; = 2T$, i.e., we choose a chunk size whose arrival duration is close to (but less than) $2T$. This choice ensures that the average delay experienced by devices in a chunk does not exceed the paging DRX cycle value, T. Our selected chunk size depends on R_C, which, in turn, depends on the network condition, e.g., R_C is likely to increase when the network is experiencing high load. Our method ensures that the selected chunk size dynamically adapts to changing network conditions.

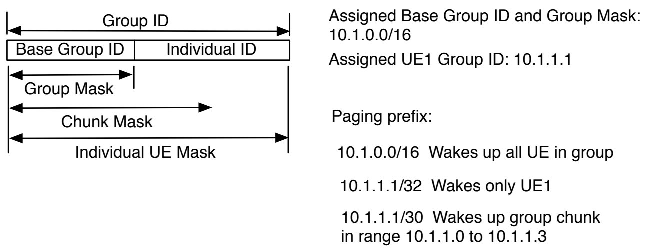

Group ID
Base Group ID
Individual ID
Group Mask
Chunk Mask
Individual UE Mask
Assigned Base Group ID and Group Mask: 10.1.0.0/16
Assigned UE1 Group ID: 10.1.1.1
Paging prefix:
10.1.0.0/16 Wakes up all UE in group
10.1.1.1/32 Wakes only UE1
10.1.1.1/30 Wakes up group chunk in range 10.1.1.0 to 10.1.1.3

Fig. 4: Adaptive Group IDs

C. Adaptive Group Identifiers:

In this section, we present an group assignment approach that the network may utilize to implement the adaptive algorithms described earlier. The network assigns the devices a *group identifier (ID)* which is explicitly communicated by the network using control message exchanges e.g., during initial attachment. As shown in Figure 4, a *Group ID* consists of two parts namely, a *Base Group ID* and an *Individual ID*. The relative size of each of these components is determined by a *Group Mask*. For example, as shown in Figure 4 (and assuming conventional IPv4 like notation to simplify exposition), a base group ID and group mask of respectively $10.1.0.0$ and $/16$ means a base group ID of 16 bits, leaving 16 bits for the individual IDs. Think of the base group ID as an identifier for the application, and the individual IDs as the device identifiers the application sends to and receives data from.

To address a group of devices, the MME uses a *Paging Prefix* consisting of a group ID and a mask. A device compares its full group ID to the paging prefix to determine whether it needs to wake up. For example, as shown in Figure 4, a paging prefix using the group mask, i.e., $10.1.0.0/16$, will activate all UEs in the group. On the other extreme, using an *Individual UE Mask* of $/32$ a specific UE within the group can be woken up. Using a *Chunk Mask* (or sub-group mask) in between these extremes, allows the MME to address subsets of UEs along "prefix-like" boundaries. Once assigned, the group identifiers of the individual UEs within a base group remain the same throughout, and the prefix based approach allows the MME to dynamically address different subgroups (chunks) within the same base group without explicitly informing the UEs each time the subgroups are updated.

V. Evaluation Framework

We evaluate our adaptive device activation architecture through a data driven simulation framework. In this section, we explain our framework and the data used in our evaluation.

A. Simulation Framework

We model the LTE eNodeB paging procedure and the first two steps (preamble transmissions and random access responses) of the RACH procedure as described in Section II. The subsequent messages for connection establishment are sent over dedicated (non-shared) resources, and we therefor do not model them for our evaluation. We implement the dynamic group adaptation (automatic chunk size selection) and

Parameter	Setting
Random Access Slot Period	5ms
Total Number of Preambles	54
Max. Number of Preamble Transmissions	10
Number of UL Grants per RAR (N_{RAR})	3
Ra-ResponseWindowSize (W_{RAR})	5ms
Backoff Indicator (BI)	60
Paging DRX Cycle (T)	640ms
Paging Capacity per Slot (P_s)	16
nB	1
Max. Paging Retry Limit	2

TABLE I: Simulation Parameters

the adaptive RACH algorithms as described in Section IV-B in our Python-based simulator.

The paging slot for an UE is given by a system frame number (SFN) and a paging occasion (PO) which is calculated based on its identifier (UEID), DRX cycle (T), and a network specific parameter nB which specifies how many paging occasions (POs) or slots are available per system frame [12]. Number of UEs that can be paged in a slot is given by the paging slot capacity (P_s) of the eNodeB. After receiving a paging message, the UE starts connection setup using the RACH procedure by selecting a preamble from a set of available (network specific) random access preambles and sends the same to the eNodeB in the next random access slot. In case more than two UEs select the same random access preamble in the same slot, we randomly choose one of the UE as the contention winner. The eNodeB responds with a random access response (RAR) message which should reach the UE within a network configured time window W_{RAR}, otherwise the UE considers the previous step failed and restarts the RACH procedure. The network is also limited in the number of identifiers (preamble reception acknowledgement) it can send in a single RAR (N_{RAR}) due to radio resource constraints. This implies the network can only reply to $W_{RAR} * N_{RAR}$ (N_{RA}^{MAX} in Equation 1) UEs per random access slot. UEs who either collide in preamble transmission or do not receive a RAR, restart the random access procedure after a random backoff indicator (BI) time as indicated by the network. Table I presents our simulator parameters. We choose typical network configuration values ([3], [8]) to evaluate our work. In some cases, our parameter choice (e.g., P_s & nB) is also driven by the fact that we want to investigate whether M2M communication pattern poses a challenge to the fundamental LTE radio access capacity. For example, we choose 640 ms as paging DRX cycle which strikes a good balance between paging slot wait times and frequent device wakeups. Again, we choose the random backoff indicator value (BI=60) to keep the activation delay within a tolerable limit.

B. Data

For M2M arrivals we simulate the arrival of 30000 activation requests uniformly spread over an interval of 10s (the same use case as described Section III). We consider this demanding use case in our evaluation to show that our adaptive algorithms scale efficiently even with extreme workloads.

Loadtype	Data Sample	Scaled Data
	Mean (Max) requests/sec	
Paging	4.2 (15)	8.4 (30)
New Session Requests	2.32 (9)	4.64 (18)
Idle UE Conn. Reestab.	-	4.64 (18)

TABLE II: Data Statistics

For H2H (background traffic) we use data obtained from a large cellular service provider in the US. The purpose of using this data is to show how the projected M2M traffic patterns affect the existing H2H communications. The data sample consists of one hour of LTE control plane data collected during August 2013. This data, anonymized and aggregated over one second, consists of all GTPC-v2 control plane message exchanged between all MME-SGW pairs. We obtained a time series for both paging (Downlink Data Notification GTPC-v2 message) and new session requests (Create Session Request GTPC-v2 message) from this data. The data we obtained was collected during a relatively quiet network period. Therefore, as we explain below, we use a scaled version of the data in our simulations. Table II shows the essential statistics for both the one hour data sample and the scaled data. The table shows statistics associated with a representative eNB we use in our simulations. The "Data Sample" column shows the statistics derived from these time series. To scale the data to more realistic rates, we use the well known diurnal pattern in terms of traffic load, to scale the data by a factor 2, from the low volume hour, to a peak hour.

We note that our data did not contain "connection re-establishment" requests which are sent out when a UE in idle state wakes up because of application initiated activity. To account for these messages in our simulation we assume that connection reestablishment requests are of the same magnitude as new data session requests. The statistics of the resulting scaled data that we use in our simulations are shown in the "Scaled Data" column in Table II. Our scaling of the sample data is conservative along several vectors. First, diurnal patterns often show larger differences (than factor 2) between low and peak times. Second, given predominant H2M use of current mobile devices, connection reestablishment can be expected to happen more frequently than new connection establishment. Finally, given that LTE/EPC deployments are relatively new, the bulk of mobile devices are still being served on 3G networks [13]. For example, recent (collected in March 2014) hourly averaged data from the same provider shows peak paging and peak RRC connection request rates of respectively 36 and 53 requests per second, indicating that our scaling factor is indeed conservative.

The M2M arrivals and the real H2H paging data serves as the input to the paging module in our simulator. The successfully paged UEs, in turn, are used as input to the RACH module. Additionally, the RACH module also takes as input the real H2H new data session requests and connection re-establishment as these messages also contribute to RACH contention. Each simulation run starts with the arrival of the first of the 30000 M2M device activation requests which arrive

Metric	Description
Paging Collision Probability	Ratio of number of paging slots in which paging collision occurs, to the total number of paging slots. (Number of devices paged exceeds slot capacity.)
RACH Collision Probability	Ratio of the number of random access slots in which two or more devices select the same preamble to the total number of random access slots.
M2M Completion Time	Difference between the time when the request for device activation for the first device in a M2M group is received and the time when the entire group (last device in the group) successfully completes the connection setup procedure (excluding the devices that fail).
Average Wait Time	Average time a device needs to wait from the time of device activation request to complete the connection setup. Computed for successful devices only.
Paging Failure Rate	Ratio of total number of devices that could not be paged after the maximum number of paging message retransmission attempts, to the total number of paging requests.
RACH Failure Rate	Ratio of total number of devices that could not complete the RACH procedure after maximum number of preamble retransmission attempts, to the total number of devices participating in the RACH procedure.

TABLE III: Evaluation metrics.

uniformly over 10s, and continues till all the M2M devices has completed (or failed to complete) the access (paging and RACH) procedures. Since the real trace is for 1 hour (much longer than our simulation period), for each simulation run we select a random starting point in the trace to avoid any bias in the evaluation. For each set of experiment, we use 20 simulation runs and present the average result of these 20 runs.

VI. RESULTS

We present results for our evaluation in this section. Our evaluation metrics are defined in Table III.

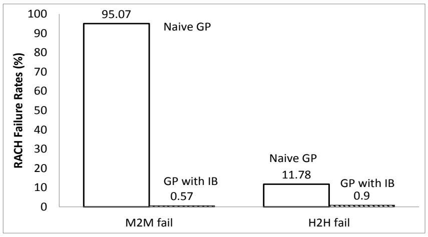
Initial RACH Backoff. Figure 5a shows how our adaptive initial backoff scheme (shown as *GP with IB*) dramatically reduces RACH failure rates as compared to a naive grouping strategy (shown as *naive GP*) with no adaptive backoff. Results are for group arrival of 30000 devices in 10s. In Section III, we have seen that such arrival causes unacceptably high paging failures, whereas a naive grouping strategy simply shifts the bottleneck to the RACH procedure. Our method helps overcome the RACH bottleneck with efficient distribution of load. For example, RACH failure rates for M2M devices reduces to only 0.57% (from 95%) and overall RACH collision rate reduces to 14.83% (from 76.65%) when using our strategy.

Figure 5b shows how RACH failure rate varies with different group arrival sizes (with and without our proposed adaptation). We experiment with group arrival sizes of 500, 1000, 2000, 5000, 10000 and 30000 respectively. We see that, the RACH failure rates with a naive grouping strategy increases significantly as group size is increased. For example, the failure rate is close to 40% even for a moderately large group arrival of 2000. This indicates that, dividing a very large group in moderate sized subgroups and paging the subgroups individually does not work well without any adaptation. Our adaptive backoff scheme ensures that the RACH failure probability is almost zero for moderately large groups and even for a very large group of arrival 30000, the failure rate is only 0.57%.

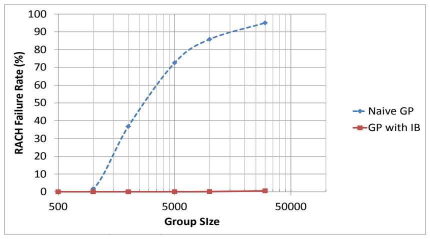
We present the CDF of random access preamble retransmission attempts in Figure 5c. If a device fails to complete the RACH procedure after 10 retransmission attempts (excluding the initial transmission), it aborts the RACH procedure (a RACH failure). With each retransmission the devices increase the transmission power to improve the chances of preamble detection at the eNodeB, and large number of retransmissions negatively impact device battery consumption as well as wastes valuable network resources. From the figure we see that more than 50% of the M2M devices require 9 or more retransmissions to successfully complete the RACH procedure while using a naive grouping strategy, where as our scheme enables 90% of the M2M devices to complete the RACH procedure using only 5-6 retransmissions. Similarly, the 90th percentile for H2H retransmissions is almost 10 for a naive grouping strategy as compared to only 5 for our adaptive backoff scheme.

Using Chunks. Figure 6a and Figure 6b show RACH failure rates & collision probabilities and Paging failure rates & collision probabilities, respectively, when we use different sizes of chunks (or subgroups) to activate 30000 devices who arrive uniformly over a 10 second period. Note that, in these experiments, we do not use the automatic chunk selection mechanism described in Section IV-B. The purpose of these experiments to observe how the selection of different chunk sizes affects the network performance. Chunk size of 1 corresponds to not doing any grouping and paging the devices individually, and chunk size of 30000 corresponds to the case when we wait for all the requests to arrive and page them as a single group. Figure 6a suggests that using smaller chunk sizes reduces RACH failure rates and collision probabilities for both M2M and H2H traffic. However, Figure 6b suggests that if we use chunk sizes less than 100 (or equivalently, more than 300 chunks) the paging failure rate increases significantly (e.g., almost 80% M2M devices fail to complete the paging process for a chunk size of 10). The high failure is because of the fact that smaller chunk sizes means more paging messages need to be sent to activate all the devices, and all these chunks will be paged in slots determined by a common base group identifier, leading to a large number of collisions. The high volume of paging messages also negatively impact the H2H paging success rate (failure rate increases by almost 18 times when using chunk sizes of 10 as opposed to chunk sizes of 100).

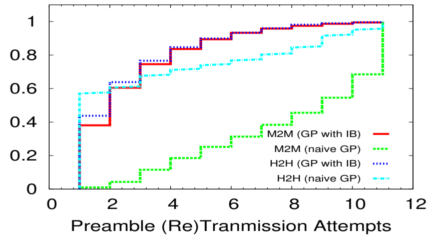
Figure 6c shows another tradeoff the network needs to



(a) RACH Failure Rates

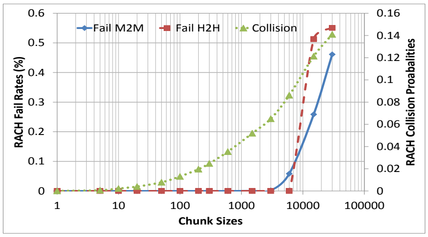


(b) RACH Failure Rates (M2M) for Different Group Sizes

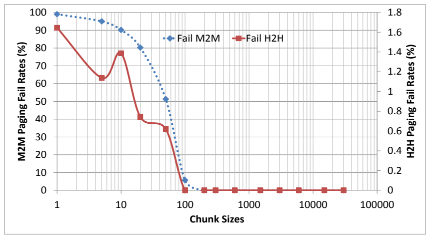


(c) CDF of Retransmission Attempt (Group Size: 30000)

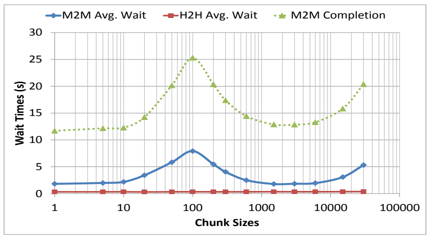
Fig. 5: Results for Initial RACH Backoff



(a) RACH Failure Rates & Collision Prob.



(b) Paging Failure Rates



(c) Wait and Access Completion Time

Fig. 6: Results for Chunking

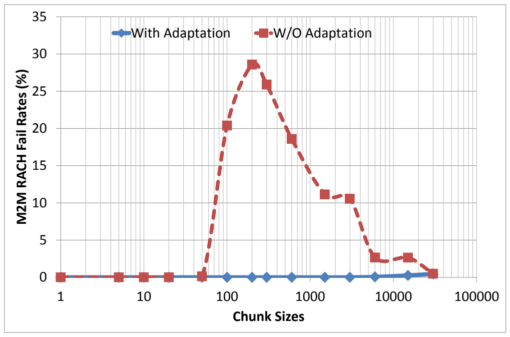


Fig. 7: RACH Failure Rates w/ & w/o Adaptation

consider while selecting the proper chunk sizes. We see that, if we use very large chunk size e.g., 15000 or moderately small chunk size (e.g., 100-1000), M2M average wait time and overall completion time increases. This is due to the fact that large chunk size leads to more RACH collisions (Figure 6a) and hence more retransmissions, whereas, smaller chunk sizes leads to more number of chunks (chunk size of 100 means 300 chunks for a group arrival of 30000), and hence more number of paging messages. The cumulative effect of paging delay (paging message can be sent in only one slot per DRX cycle) and any overestimation of additional RACH delay leads to overall increased wait times and completion times for M2M. As evident from the figure, average wait time for H2H remains mostly unaffected irrespective of the chunk sizes we use because our RACH adaptation methods ensure that the M2M traffic use the RACH in a well distributed manner. Moreover, since all M2M chunks only utilize a single paging slot per DRX cycle, H2H paging traffic do not face much competition from M2M traffics, hence the paging delay also remains unaffected. We omit results for a group arrival of 5000 over 10s which show similar, but less pronounced, results.

Chunks overlapping in RACH access. Figure 7 shows why it is necessary to adapt the RACH process to ensure that consecutive chunks do not overlap with each other. We ignore chunk sizes less than 100 due to high paging failure rates. For the smaller chunk sizes in the range of [100-5000], the RACH failure rates are very high, almost 30% for chunk sizes of 200, 20% for chunk sizes of 500 etc. This is due to the fact that, without any adaptation, it is likely that one chunk is paged at a time when the other chunk is still accessing the RACH, so RACH access attempts from the devices belonging to the newly arrived chunk (albeit initial random backoff) will collide with access attempts from the previous chunks leading to high RACH failure rate. Furthermore, as a consequence of the paging mechanism, all the chunks that arrive within a paging DRX cycle are paged at the same time. Smaller chunk sizes also increases the number of chunks that arrive within a paging DRX cycle and, hence, the number of chunks that start the RACH procedure at the same instant. We see that, with our adaptation method the RACH failure rate remains close to zero irrespective of the chunk sizes used. Thus, we achieve an order of magnitude improvement as far as RACH failure rates are concerned with our adaptation scheme.

Automatic Chunk Size Selection. Figure 8 shows how our adaptive chunk selection method is able to select optimal operating points for different arrival intensities. Recall from Section IV-B, the network initially selects a default minimum chunk size that avoids high paging failures. In our experiment we set the default value to 100. We see that, for group arrivals of 30000/10s, the network converges to the optimal chunk size (3840) within only 100ms of the arrival start time. The access completion time is 12.2 s, whereas the average wait times for M2M and H2H are 1.1s and 0.32s, respectively. Also, RACH and Paging failure rates for both M2M and H2H are zero in

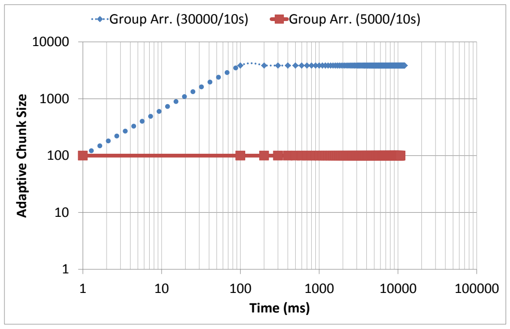


Fig. 8: Automatic Chunk Size Selection

this case. We note that, these results are consistent with the optimal operating points we observe in our experiment with different (fixed) chunk sizes (Figures 6a, 6b, 6c). For a less intense group arrival of 5000/10s, the default minimum chunk size proves to be the optimal for the entire simulation period.

VII. Related Work

Our work is complementary to overload control for M2M efforts in the 3GPP standards body [3]. We specifically address issues left open by these earlier proposals concerning optimal group sizes and group assignment and management. For example, the standards body [4] advocates the use of globally unique identifiers for M2M groups that is statically assigned during device registration. However, this approach is inflexible as it does not allow the network to contact a single device individually or a subset of devices in a group when required. Also, for large group sizes, such static assignment leads to excessive RACH failures. To prevent this, the network may divide a large group into smaller subgroups and assign a unique identifier to activate each subgroup. In this paper, we show that without proper adaptation techniques even such subgroup assignment does not prevent activation failures.

Serror et al. [14] demonstrated that data traffic can cause overload in the paging channel and delay call setup in a CDMA2000 cellular network. Zang and Bolot [15] showed that compared to using fixed location areas across all users, paging/location management based on per-user profile can reduce paging load by up to 95%. Our work focuses on how to reduce control channel load in the presence of synchronized device activation attempts regardless of paging location optimization, as paging channels can be overloaded with perfect optimization when many M2M devices under a single base station are paged at the same time.

The technical vulnerabilities of the LTE paging mechanism is considered in [16]. However, they stop short of applying their work to improve paging. Work perhaps most related to our own suggests group paging to deal with expected performance issues of M2M communication [17]. They developed an analytical model to attempt to capture various aspects of a basic group paging approach. The analytical approach becomes tractable by assuming a known group size, dedicated and isolated radio resources for M2M, and using a Poisson approximation model. They have also proposed a consecutive group paging approach [18], using reserved paging cycles for a group in base stations to page the same group of users consecutively to account for failures in earlier attempts. Neither of these works considers the impact on the existing background H2H traffic. Our work is also different in that we propose adaptive strategies to address problems associated with M2M activation and use a data driven simulation framework to ensure the realism of our evaluation. We do not assume resource isolation for M2M, rather we minimize the impact of large scale M2M arrivals on H2H traffic.

VIII. Conclusion

In this paper, we proposed an adaptive device activation architecture that adapts to network conditions and M2M application requirements to realize scalable device activation without increasing the resources used for this purpose. Our evaluation showed that our adaptive approach enables the network to handle M2M applications with a large number of devices without negatively impacting existing H2H traffic.

References

[1] D. Nowoswiat and G. Milliken, "Managing LTE Core Network Signaling Traffic," http://www2.alcatel-lucent.com/techzine/managing-lte-core-network-signaling-traffic/.

[2] M. Hatton, "The Global M2M Market in 2013," Machina Research White Paper, January 2013, http://www.globalm2massociation.com/machina-research-white-paper-the-global-m2m-market-in-20/.

[3] "3GPP TR 37.868," http://www.in2eps.com/3g37/tk-3gpp-37-868.html.

[4] "3GPP TR 23.888," http://www.3gpp.org/DynaReport/23888.htm.

[5] "3gpp ts 23.401," http://www.3gpp.org/DynaReport/23401.htm.

[6] "3GPP TR 36.321," http://www.3gpp.org/DynaReport/36321.htm.

[7] M. Z. Shafiq, L. Ji, A. X. Liu, J. Pang, and J. Wang, "A First Look at Cellular Machine-to-Machine Traffic: Large Scale Measurement and Characterization," in *Proceedings of ACM SIGMETRICS*, 2012.

[8] "TA Planning Guideline,Ericsson," http://www.telecom-cloud.net/wp-content/uploads/2010/09/LTE-TA-Planning.pdf.

[9] Zeljko Savic, "LTE Design and Deployment Strategies," http://www.cisco.com/web/ME/expo2011/saudiarabia/pdfs/LTE_Design_and_Deployment_Strategies-Zeljko_Savic.pdf.

[10] B. Nguyen, A. Banerjee, V. Gopalakrishnan, S. Kasera, S. Lee, A. Shaikh, and J. Van der Merwe, "Towards Understanding TCP Performance on LTE/EPC Mobile Networks," in *Proceedings of the 4th workshop on All things cellular: operations, applications, & challenges*. ACM, 2014, pp. 41–46.

[11] J. Huang, F. Qian, Y. Guo, Y. Zhou, Q. Xu, Z. M. Mao, S. Sen, and O. Spatscheck, "An In-depth Study of LTE: Effect of Network Protocol and Application Behavior on Performance," in *Proceedings of the ACM SIGCOMM*, 2013.

[12] "3GPP TR 36.304," http://www.3gpp.org/DynaReport/36304.htm.

[13] "Verizon Again Sees Strong Subscriber Growth in Q2, Driven by Smartphones, LTE," http://gigaom.com/2013/07/18/verizon-again-sees-strong-subscriber-growth-in-q2/, 2013.

[14] J. Serror, H. Zang, and J. C. Bolot, "Measurement and Modeling of Paging Channel Overloads on a Cellular Network," *Computer Networks*, 2013.

[15] H. Zang and J. C. Bolot, "Mining Call and Mobility Data to Improve Paging Efficiency in Cellular Networks," in *Proceedings of ACM MOBICOM*, 2007, pp. 123–134.

[16] A. Baraev, U. Ayesta, I. M. Verloop, D. Miorandi, and I. Chlamtac, "Technical Vulnerability of the E-UTRAN Paging Mechanism," in *Proceedings of IEEE WCNC*, 2012.

[17] C.-H. Wei, R.-G. Cheng, and S.-L. Tsao, "Performance Analysis of Group Paging for Machine-Type Communications in LTE Networks," *Vehicular Technology, IEEE Tran. on*, 2013.

[18] R. Harwahyu, R.-G. Cheng, and R. F. Sari, "Consecutive Group Paging for LTE Networks Supporting Machine-type Communications Services," in *Personal Indoor and Mobile Radio Communications (PIMRC)*. IEEE, 2013.