A Simple Model of MTC Flows Applied to Smart Factories

A Simple Model of MTC Flows in Smart Factories

Paolo Castagno*, Vincenzo Mancuso†, Matteo Sereno*, Marco Ajmone Marsan‡†
*Università di Torino, Turin, Italy
†IMDEA Networks Institute, Madrid, Spain
‡Politecnico di Torino, Turin, Italy

***Abstract*—In this paper we develop a simple, yet accurate, performance model to understand if and how evolutions of traditional cellular network protocols can be exploited to allow large numbers of devices to gain control of transmission resources in smart factory radio access networks. The model results shed light on the applicability of evolved access procedures and help understand how many devices can be served per base station. In addition, considering the simultaneous presence of different traffic classes, we investigate the effectiveness of prioritised access, exploiting access class barring techniques. Our model shows that, even with the sub-millisecond time slots foreseen in LTE Advanced Pro and 5G, a base station can accommodate at most few thousand devices to guarantee access latencies below 100 ms with high transmission success probabilities. This calls for a rethinking of wireless access strategies to avoid ultra-dense cell deployments within smart factory's infrastructures.**

I. Introduction

Factory automation, under the buzzwords Factories of the Future (FoF), or Smart Factories (SF), is a key pillar of the Industry 4.0 concept, and one of the key vertical sectors for the application of 5G technologies, together with automotive, healthcare, energy, media and entertainment [1]. The 5G PPP identifies SF as one of the main business cases for 5G [2] and classifies the most stringent performance requirements of this application domain in the use case family termed *Tactile Internet / Automation* since they require *Time-critical process optimisation to support zero-defect manufacturing*.

Key Performance Indicators (KPIs) defined by the 5G PPP for SF are exceedingly stringent: end to end (E2E) latency between 100 μs and 10 ms, device densities between 10.000 per square km and 100 per square meter, service reliability higher than 99%. The device density, in particular, is extreme, due to large numbers of sensors and actuators instrumenting robots, and transmitting and receiving data. Such utmost device densities suggested the identification of SF as the paradigmatic environment for massive Machine-Type Communication (MTC) and a very challenging example of the Internet of Things (IoT).

While the present 5G activities are addressing scenarios that are *either* massive (i.e., with extreme user densities) *or* critical (i.e., with stringent latency requirements), it is quite likely that future evolutions of 5G research will also consider massive *and* critical scenarios, which will emerge in several domains, most notably automotive, health, and, in particular, SF. Therefore, investigating how the 5G technology can cope with an extremely demanding environment such as SF is very important, especially to determine the type and the density of base stations (BSs) that can meet the required KPI targets, together with the associated cost.

To accomplish such task, little exists in the literature that can help to understand the impact of those procedures needed to access resources in a cellular network under extreme operational conditions. The most relevant work in this field is the analytic study described in [3]. In there, the authors developed a probabilistic model for MTC using the LTE technology, and compared the model results to simulation predictions, to show a good match between the two approaches. The model in [3] incorporates many features of the LTE procedures, but does not account for blocking at the BS, does not allow for differentiation of traffic classes, does not generate the latency distribution, and does not provide a closed form solution for the main performance indicators.

In our paper we go well beyond existing approaches. Specifically, we describe a stochastic model of the behavior of environments that, like SF, can be massive, or critical, or massive and critical, incorporating features that will be part of the 5G operations, and evaluating the performance of scenarios typical of a SF environment. The model allows us to evaluate operational conditions, and to derive the distribution of latencies experienced by network access requests. Our model proves to be very accurate when results are compared to the predictions of a detailed simulator or to the very detailed analytical model in [3]. In addition, our model allows a closed form solution, except for one fixed point iteration, which however exhibits a very fast convergence.

Our results show that, for example, with standard system parameters (details are given in the section on numerical results), in order to achieve a success probability not less than 0.9, and a latency not higher than 70 ms, one BS should serve no more than ~ 1400 devices. With a device density equal to 10.000 per square km, this means that the BS can cover an area of radius equal to approximately 200 m. Instead, if we consider the most extreme density envisioned for SF, equal to 100 devices per square meter, the BS can cover only 14 square meters, hence a circle of radius just over 2 m, which would be practically unfeasible even in future SF scenarios! This shows how important it is to carefully evaluate the performance of cellular access in massive MTC environments, and how impactful device density is, which should be definitely taken into account in the design of future wireless access techniques for super-dense device layouts.

The rest of this paper is organised as follows. Section II introduces the scenario that we consider in this work, and provides a concise description of the massive MTC environment. Section III presents the analysis, together with the relevant assumptions and the derivation of closed-form results.

TABLE I
NOTATION AND CELL PARAMETERS USED IN THE ANALYTICAL MODEL

description	*notation*	*range*
RACH interval	τ	1 ms
Minimum time needed to reply to a RACH request	$T_{\min}$	0.2 ms
Maximum time allowed to replay to a RACH request	$T_{\max}$	$0.4 \sim 1$ ms
Maximum time needed to establish an RRC connection after a RACH exchange	$W_{\max}$	1 ms
Maximum number of RACH attempts	$k_{\max}$	$10 \sim 40$
RACH collision probability	p_C	$0 \sim 1$
Probability of failure in the RRC connect	$p_{\bar{R}_i}$	$e^{-1} \sim 1$
Maximum number of requests that a base station can serve in a RACH interval	Θ	$12 \sim 24$
Network blocking probability	p_B	$0 \sim 1$
Random RACH backoff at stage i	B_i	10 ms
ACB deferral probability (for flow ℓ)	p_A	$0.05 \sim 0.95$
Random ACB backoff at stage i, after the j-th ACB barring event in that stage	A_{ij}	$4 \sim 512$ s
Primary/secondary flow rate	λ/ℓ	$9 \sim 0.11$
Timeout (primary flow)	T_O	≤ 10 s
Number of Random Access Preambles	N	54

Section IV validates our model by comparing its results to those of the very detailed model in [3]. Section V presents and discusses numerical results. Section VI discusses related work, and Section VII concludes the paper.

II. SYSTEM

We focus our analysis on a single cell, with n Machine-Type Devices (MTDs) that generate new uplink transmissions with a given aggregate rate. In the following, we distinguish two different types of requests: time-critical and non-time-critical, which we identify with two flows, namely the primary and secondary flows. The requests belonging to the primary flow (with intensity λ) can wait at most T_O seconds before being served; otherwise, they are dropped. On the other hand, the requests of the secondary flow (with intensity ℓ) have no timeout, and represent traffic with lower priority, referring to non-real-time applications. Table I shows the notation we use.

In order to access the network, each MTD has to first complete the random access procedure, which initiates as soon as a RACH (Random Access CHannel) opportunity is granted by the BS. The MTD has to go through the RACH each time it has a new message to transmit because downlink traffic is assumed to be sporadic [4].

A request is successful only when resources are actually allocated to the MTD; that is, we take into account also signalling messages that are exchanged *after* the random access procedure successful completion. Indeed, the 3GPP-defined procedure to access resources includes the RACH phase and the RRC (Radio Resource Control) connect phase, resulting in the exchange of four messages. When either of the two phases fails, the MTD retries after a random backoff interval. Multiple timeouts are used in the overall procedure, in the event of a collision, of an early access failure (when no RRC connect message is exchanged before a time $T_{\max}$ from the beginning of the RACH opportunity used by the MTD) or a late access failure (when the RRC connect phase starts, but no final resource allocation is notified to the MTD within a window $W_{\max}$ from the beginning of the RRC connect phase). More details on the timing of a request will be provided in the next section when describing our model. Fig. 1 shows an example of access request that succeeds after 2 retries over the Random Access.

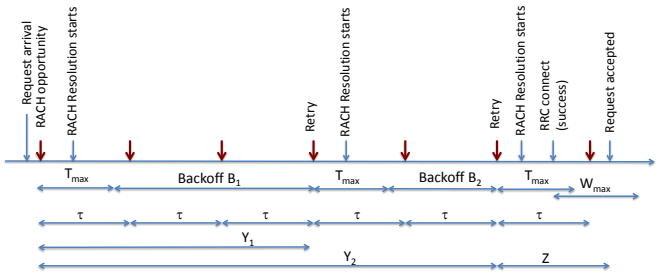


Fig. 1. Timing example for a primary flow request served after 3 attempts.

In the system we just described, a message transfer can take place after the successful completion of two subsequent steps: the RACH and the RRC connection procedures. The RACH can be divided into $k_{\max}$ sequential states, one for each allowed RACH attempt (after $k_{\max}$ attempts, a request is dropped). Access requests move from one state to the next in case of collision (with probability p_C) and in case the request gets lost (i.e., it is not correctly received and acknowledged by the BS). The latter event occurs with probability $p_{\bar{R}_i}$, which is different at each attempt, due to the standard *power ramping* mechanism: nodes progressively increase the power used to transmit RACH requests after each failed attempt [5].

The dynamic of RACH requests in the system is presented in Fig. 2. In each state, a request can leave because of a success (the MTD transmits its data). The request can however also leave the system because of a failure, which can consist in either a network blocking due to a shortage of queueing resources at the network processor after a successful RRC connection procedure or because of a timeout. Moreover, a request can move from state i to $i+1$ because of a collision, or any event that precludes the success of the RRC connection procedure: either the request is not decoded by the BS, or the BS does not have resources to send an acknowledgement and decides to drop the request (we indicate with Θ the maximum number of requests the base station can acknowledge in each RACH interval τ). In addition, a request can retry the RACH procedure at most $k_{\max}$ times, otherwise, it leaves the system with a failure. Notice that passing from a state to the next incurs a random delay due to backoff. In addition, the secondary flow incurs RACH access deferring with fixed "barring" probability p_A at any attempt, and multiple back-to-back deferrals are possible, so that secondary flow requests incur additional delay at each state, due to standard Access Class Barring (ACB) operation [6].

III. ANALYTICAL MODEL

For the sake of compactness and readability, we provide the reader with the definitions of the variables used in the analysis in Table I. Moreover, the random variables representing intervals of time used in the model are pictorially presented in Fig. 1, while flows entering and leaving the RACH system are indicated in Fig. 2 jointly with the system throughput ξ.

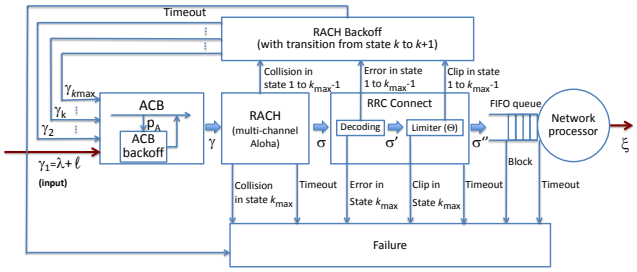


Fig. 2. Block diagram representing the system with primary and secondary flows accessing resources via RACH channels and RRC connect procedure. Flow ℓ is subject to ACB and all accepted requests are served in FIFO order.

A. Structure of latency

A RACH request enters the system in state 1 and leaves in any state $i \in \{1,\dots,k_{\max}\}$ upon a success, a network blocking, an excessive number of retries, or a timeout. If a request leaves the system from state i because of either a success or a network blocking, it has been in the system for a time Y_{i-1} (more precisely, either $Y_{i-1}^{(\lambda)}$ or $Y_{i-1}^{(\ell)}$, depending on the flow considered) which consists of $(i-1)$ times the interval $T_{\max}$ and $i-1$ backoffs, plus a random interval Z (see Fig. 1 for $i=2$)—the latter being independent from Y_i—and a random number of barring backoffs for requests of the secondary flow. Similarly, in the case of an excessive number of retries, the time spent in the system is $Y_{k_{\max}-1}+Z$. In the case of timeout, of course, the time spent is T_O. When passing from state i to $i+1$, the time spent until the state transition is simply Y_i. As we will see later, the above quantities are sufficient to describe the entire sojourn in the system and to evaluate the performance of the system in terms of, among other quantities, network blocking probability, timeout probability, throughput and latency. With the notation described in Table I, the distribution of $Y_i^{(\ell)}$ is

$$F_{Y_i^{(\ell)}}(x)=\Pr\left\{ i\,T_{\max}+\sum_{k=1}^{i}\left(B_k+\sum_{j=0}^{L_k}A_{kj}\right)\le x\right\}, \quad (1)$$

where L_k is the random number of back-to-back deferrals experienced because of ACB in state k. The distribution for the primary flow, namely $F_{Y_i^{(\lambda)}}(x)$ is omitted because it can be derived from $F_{Y_i^{(\ell)}}(x)$ by plugging $L_k=0$. The backoff random variables B_k and A_{kj} are independent among them and from Z, although not necessarily identically distributed. In contrast, variables $Y_i^{(\lambda)}$ depend on $Y_k^{(\lambda)}, \forall k<i$. Similarly, variables $Y_i^{(\ell)}$ depend on $Y_k^{(\ell)}, \forall k<i$.

The distribution of the time spent by a request until the resolution of the i-th RACH attempt, for the secondary flow, is expressed as the distribution of the random variable $Y_{i-1}^{(\ell)}+Z$:

$$F_{Y_{i-1}^{(\ell)}+Z}(x)=\Pr\left\{(i-1)T_{\max}+\sum_{k=1}^{i-1}\left(B_k+\sum_{j=0}^{L_k}A_{kj}\right)+Z\le x\right\}; \quad (2)$$

and for the primary flow it is enough to use (2) with $L_k=0$ to derive $F_{Y_{i-1}^{(\lambda)}+Z}(x)$. Since Z is independent from $Y_i^{(\lambda)}$ and $Y_i^{(\ell)}$, and denoting by f_Z the p.d.f. of Z, the following useful results also hold: $F_{Y_{i-1}^{(\lambda)}+Z}=F_{Y_{i-1}^{(\lambda)}}*f_Z$ and $F_{Y_{i-1}^{(\ell)}+Z}=F_{Y_{i-1}^{(\ell)}}*f_Z$.

B. State probabilities

At state i, a request leaves the system because of either a success, a network blocking, or a timeout (for the primary flow only). In all other cases, the request moves from state i to state $i+1$, with the exception of state $k_{\max}$ for which an attempt to pass to state $k_{\max+1}$ results in a failure due to an excessive number of retries. Here we derive state probabilities for the primary flow only. However, the same equations hold for the secondary flow by replacing λ with ℓ and using $T_O\to\infty$.

State transitions. Denoting by p_C the RACH collision probability, which is the same for all RACH attempts, and by $p_{\bar{R}_i}$ the probability of an error in the RRC connect procedure after the i-th RACH attempt, state transition probabilities $P_N^{(\lambda)}(i)$ are computed as the probability to reach state $i+1$ going through all previous i states. The described quantities only depend on the aggregate load in the RACH and on the resources available at the BS.

For a request in the primary flow, the transition to the next state occurs when there is either a collision or an RRC connect failure, therefore with probability $1-(1-p_C)(1-p_{\bar{R}_i})$, but only if the timeout has not expired before the end of the RACH backoff in that state, i.e., with probability $F_{Y_i^{(\lambda)}}(T_O)$. This results in the following iterative computation, $\forall i\ge 1$:

$$P_N^{(\lambda)}(i)=P_N^{(\lambda)}(i-1)\left[1-(1-p_C)\left(1-p_{\bar{R}_i}\right)\right]F_{Y_i^{(\lambda)}}(T_O); \quad (3)$$

where $P_N^{(\lambda)}(0)=1$ by definition. Note that, since $k_{\max}$ is the maximum retry number, $P_N^{(\lambda)}(k_{\max})$ is a failure probability.

Success. The probability of a request succeeding in state i, $\forall i\ge 1$, is the probability of reaching state i and then have no collision in the RACH, no error in the RRC connect phase, and no network blocking. At the same time, no timeout has to occur while waiting for the resolution of the i-th RACH attempt. Hence, denoting the network blocking probability by p_B, the following recursive relation holds:

$$P_S^{(\lambda)}(i)=P_N^{(\lambda)}(i-1)(1-p_C)\left(1-p_{\bar{R}_i}\right)(1-p_B)F_{Y_{i-1}^{(\lambda)}+Z}(T_O). \quad (4)$$

We denote by $P_S^{(\lambda)}$ the total success probability for the primary flow. Such quantity is computed by summing the success probabilities (4) over the states.

In the case of success, the request receives service, and the time spent in the system before service, for a request on the primary flow, results to be a random variable $Y_{i-1}^{(\lambda)}+Z$.

Blocking. When a request successfully passes both the RACH and RRC connect phases, it can be either admitted to the service or blocked because of lack of resources at the network processor of the BS. The probability that a request is blocked by the network in any state i, can be computed as

$$P_B^{(\lambda)}(i)=P_N^{(\lambda)}(i-1)\,(1-p_C)\,\left(1-p_{\bar{R}_i}\right)p_B\;F_{Y_{i-1}^{(\lambda)}+Z}(T_O). \quad (5)$$

We denote as $P_B^{(\lambda)}$ the total blocking probability of flow λ.

In the case of blocking, the time spent in the system is exactly like in the case of success (now excluding the service time), i.e., for a request on the primary flow, it is $Y_{i-1}^{(\lambda)}+Z$.

Timeout. Requests of flow λ can experience timeout in state i if they reach state i and: 1) either the random access or the

RRC connect fail, and the backoff delay leads to exceeding the timeout; or 2) the RRC connect attempt is not resolved within the timeout. The time spent in the system is of course T_O, but it is also a value obtained from the r.v. $Y_i^{(\lambda)}$ or $Y_{i-1}^{(\lambda)}+Z$. The resulting timeout probability can be expressed via the cumulative functions of those r.v.'s:

$$P_{TO}(i)=P_N^{(\lambda)}(i-1)\left\{(1-p_C)\left(1-p_{\bar{R}_i}\right)\left[1-F_{Y_{i-1}^{(\lambda)}+Z}(T_O)\right]+\left[1-(1-p_C)\left(1-p_{\bar{R}_i}\right)\right]\left[1-F_{Y_i^{(\lambda)}}(T_O)\right]\right\}. \quad (6)$$

We further denote as P_{TO} the total timeout probability.

Closed form for probability expressions. Although we have presented iterative expressions, one can notice that all of the above expressions can be easily re-written in closed form. Indeed, it is enough to notice that state transitions probabilities can be put in the following closed form, $\forall i \geq 1$:

$$P_N^{(\lambda)}(i)=\prod_{k=1}^{i}\left\{\left[1-(1-p_C)\left(1-p_{\bar{R}_k}\right)\right]F_{Y_k^{(\lambda)}}(T_O)\right\}. \quad (7)$$

The above expressions can be used in all other expressions found in this section to derive probabilities in closed form.

Remark on the generality of state probability expressions. All expressions derived in this section are valid independently from the distribution of backoff events and ACB configuration, and can be easily generalised for the case with no limit on the number of RACH attempts (i.e., for $k_{\max}\to\infty$). As it is easy to check, the sum of success, blocking, and timeout probabilities, plus the state transition probability in state $k_{\max}$, i.e., the sum over all events in which a request leaves the system, is identically 1 for all possible values of parameters and distributions used, which has to hold because an MTD request eventually has to leave the system.

C. Analysis of random access operation

To compute the expressions for p_C, p_B and $p_{\bar{R}_i}$ to plug in the state probability expressions derived above, we model the RACH operation as a multi-channel slotted Aloha system with random backoff after a collision and with a finite number $k_{\max}$ of attempts. We consider the typical 3GPP procedure in which access requests are transmitted with increasing power after each failure and the BS can receive corrupted RACH messages even in the case of no collision, with probability e^{-i}, with the power used in state i, as suggested by 3GPP [5]. Moreover, the BS can serve a limited number of requests per RACH opportunity interval, namely Θ access requests each τ seconds, where τ is the spacing between two subsequent Random Access Opportunities (RAOs) and users can choose between N orthogonal RACH preambles to request access.

RACH collision probability. Given that, regardless the actual state, all the requests performing random access share the same resources, the collision rate is the same at all states, and depends on the total RACH load γ, including both primary and secondary flows. Hence, the collision probability in the resulting multi-channel slotted Aloha with N channels and slot duration τ, is simply expressed as $p_C=1-e^{-\frac{\gamma\tau}{N}}$.

With one primary flow of intensity λ arrivals per second, plus a secondary flow of intensity ℓ, the load of the RACH is given by the sum of arrivals at each state of the RACH:

$$\gamma=\gamma^{(\lambda)}+\gamma^{(\ell)}=\sum_{i=1}^{k_{\max}}\gamma_i^{(\lambda)}+\sum_{i=1}^{k_{\max}}\gamma_i^{(\ell)}. \quad (8)$$

where γ_i is the RACH load due to attempts of connections that have already failed the random access $i-1$ times.

In turn, the load entering state i due to the primary flow is simply given by the total intensity of the flow times the probability to reach state i, which is given by (7), i.e.:

$$\gamma_i^{(\lambda)}=\lambda\prod_{k=1}^{i-1}\left\{\left[1-(1-p_C)\left(1-p_{R_k}\right)\right]F_{Y_k^{(\lambda)}}(T_O)\right\}. \quad (9)$$

The expression of $\gamma_i^{(\ell)}$ is similar, but for the fact that $\lim_{T_O\to\infty}F_{Y_k^{(\ell)}}(T_O)=1$, and therefore we omit it.

Failure of RRC connect. After a success in the random access phase, an access request may not receive an answer either because of channel errors or because the BS is saturated, which happens when the output σ of the multi-channel slotted Aloha is greater than a maximum rate Θ.

As concerns channel errors, since the power ramping mechanism is taken into account, at each subsequent state, requests are detected with an increasing probability $1-e^{-i}$, where i is the current state index [5].

As concerns exceeding the base station capacity Θ, let's consider the output of the RACH at each state, namely σ_i, which is simply given by the load at that state, times the probability of having no collision, i.e.: $\sigma_i=\left(\gamma_i^{(\lambda)}+\gamma_i^{(\ell)}\right)(1-p_C)$. However, part of the non-collided RACH requests are received incorrectly by the BS, depending on the state in which they are, so that the actual number of requests to accommodate is $\sigma_i'=\sigma_i\left(1-e^{-i}\right)$, which is $\sigma'=\sum_{i=1}^{k_{\max}}\sigma_i'$ in total.

With the above, the number of correctly received requests in a RAO is, on average, $\sigma'\tau$. Considering that the RACH has N independent output channels with binary output, the number of correctly decoded access requests at the BS can be modeled as a binomial process with success probability $\sigma'\tau/N$. The resulting mass probability function can be written as follows:

$$\pi_j'=\binom{N}{j}\left(\frac{\sigma'\tau}{N}\right)^j\left(1-\frac{\sigma'\tau}{N}\right)^{N-j},\ \forall j\in\{0,\ldots,N\}. \quad (10)$$

At most Θ requests can be answered in a RAO, and we denote by $\sigma''\tau$ the average value of the corresponding random process. The average number of losses due to clipping to Θ is denoted by $E[N_L]$ and it is given by

$$E[N_L]=(\sigma'-\sigma'')\tau=\sum_{j=\Theta+1}^{N}(j-\Theta)\pi_j'; \quad (11)$$

Since clipping is enforced independently of the RACH state, the losses are uniformly spread over the states: $\sigma_i''=\sigma_i'\left[1-\frac{E[N_L]}{\sigma'\tau}\right]$. Hence, combining the probability to incor-

rectly decode a request or that the BS cannot answer the request, we derive the RRC connect failure probability:

$$p_{\bar{R}_i} = 1 - \frac{\sigma_i''}{\sigma_i} = 1 - \left(1 - e^{-i}\right)\left(1 - \frac{E\left[N_L\right]}{\sigma'\tau}\right). \quad (12)$$

Notice that the computation of $\gamma_i^{(\lambda)}$, $\gamma_i^{(\ell)}$, p_C and $p_{\bar{R}_i}$ requires an iterative approach, which can be solved by finding the fixed point for $\gamma = f(\gamma)$, where $f(\gamma)$ results from using the expressions of p_C and $p_{\bar{R}_i}$ in $\gamma_i^{(\lambda)}$ and $\gamma_i^{(\ell)}$ and summing to compute the aggregate RACH load.

Blocking probability. The maximum number of MTDs allowed to access the network for packet transmission per unit of time is constrained by the transmission rate C of the devices (which equals the rate at which the BS operates) and the mean packet length P_L. Denoting with $E[S] = \frac{P_L}{C}$ the network service time, the flow of requests approaching the network exceeds the BS capacity as soon as the offered load $\rho = \sigma'' E[S]$ becomes greater than 1. The latter happens when the number of accepted requests in a RAO, $\sigma''\tau$, is larger than $\frac{\tau}{E[S]}$. The maximum number of MTDs' requests that can fit in a RAO unit is then $m = \lfloor \tau / E[S] \rfloor$. Requests in excess to m are blocked. Since the BS replies to access requests in an interval that can be considered as uniformly distributed and with no memory, to compute the blocking probability, we use σ'' as the arrival rate of a M/D/1/m queue. The resulting blocking probability is [7]:

$$p_B = (1-\rho)E_m/(1-\rho E_m), \quad (13)$$

where $E_m = 1 - (1-\rho)\sum_{j=0}^{m} \frac{(-1)^j \rho^j (m-j)^j e^{\rho(m-j)}}{j!}$.

Network throughput. From the above simple approximate analysis, the resulting flow of requests successfully accessing the network is simply $\xi = \xi^{(\lambda)} + \xi^{(\ell)} = \lambda P_S^{(\lambda)} + \ell P_S^{(\ell)}$.

D. Latency distribution

Primary flow. The distribution of the time spent in the system (not including the service time) for an access attempt in the primary flow is computed by noting that a request exits the system at a generic stage i if one of three disjoint events happens: 1) success, 2) blocking and 3) timeout. In addition to this, at stage $k_{\max}$, any failure in the random access causes a drop as well, even if the timeout has not expired. All the described events are mutually exclusive and cover the entire space of probability for the event of leaving the system. Hence, the CDF of the time $T^{(\lambda)}$ spent in the system by a request can be written by using the total probability formula as follows:

$$\begin{aligned} F_{T^{(\lambda)}}(x) = \Pr\left\{T^{(\lambda)} \le x\right\} &= \sum_{i=1}^{k_{\max}} P_{TO}(i)\,\mathcal{U}(x - T_O) \\ &+ \sum_{i=1}^{k_{\max}} \left(P_S^{(\lambda)}(i) + P_B^{(\lambda)}(i)\right) \frac{F_{Y_{i-1}^{(\lambda)}+Z}(x)}{F_{Y_{i-1}^{(\lambda)}+Z}(T_O)} \\ &+ P_N^{(\lambda)}(k_{\max}) \frac{F_{Y_{k_{\max}-1}^{(\lambda)}}(x - T_{\max})}{F_{Y_{k_{\max}-1}^{(\lambda)}}(T_O - T_{\max})}, \end{aligned} \quad (14)$$

where $\mathcal{U}$ is the step function centred in T_O. However, if we consider that failures for blocking or excess retries are equivalent to timeouts, we consider as T_O the latency in case of any failure and then simplify the above formula as follows:

$$F_{T^{(\lambda)}}(x) = \sum_{i=1}^{k_{\max}} \left(P_S^{(\lambda)}(i) \frac{F_{Y_{i-1}^{(\lambda)}+Z}(x)}{F_{Y_{i-1}^{(\lambda)}+Z}(T_O)} + \left(1 - P_S^{(\lambda)}(i)\right)\mathcal{U}(x - T_O) \right). \quad (15)$$

For designing and dimensioning purposes, a more insightful indicator should only take into account the time spent within the system until a success. Hence, we derive the cumulative probability function of $T^{(\lambda)}$ given a success as

$$F_{T_{|S}^{(\lambda)}}(x) = \sum_{i=1}^{k_{\max}} \frac{P_S^{(\lambda)}(i)}{P_S^{(\lambda)}} \frac{F_{Y_{i-1}^{(\lambda)}+Z}(x)}{F_{Y_{i-1}^{(\lambda)}+Z}(T_O)}. \quad (16)$$

Secondary flow. In case of an access request belonging to the secondary flow, the expressions of the latency $T^{(\ell)}$ are similar to the ones derived for the primary flow, except for the absence of timeout events (i.e., $T_O \to \infty$).

So, all cumulative distributions that appear in (14) to (16) can be replaced by a term 1 and the timeout probability is 0:

$$\begin{aligned} F_{T^{(\ell)}}(x) = \Pr\left\{T^{(\ell)} \le x\right\} &= \sum_{i=1}^{k_{\max}} \left(P_S^{(\ell)}(i) + P_B^{(\ell)}(i)\right) F_{Y_{i-1}^{(\ell)}+Z}(x) \\ &+ P_N^{(\ell)}(k_{\max})\, F_{Y_{k_{\max}-1}^{(\ell)}}(x - T_{\max}); \end{aligned} \quad (17)$$

$$F_{T_{|S}^{(\ell)}}(x) = \frac{1}{P_S^{(\ell)}} \sum_{i=1}^{k_{\max}} P_S^{(\ell)}(i)\, F_{Y_{i-1}^{(\ell)}+Z}(x). \quad (18)$$

E. Expressions with simple distributions

The analysis we just presented holds for any distribution of backoff durations and of Z. Simple expressions can be obtained for special cases of the distributions of such variables.

RACH and ACB backoffs are normally chosen according to (possibly truncated) exponential distributions. Therefore, random variables $Y_i^{(\lambda)}$ and $Y_i^{(\ell)}$ can be approximated by means of Erlang distributions with time offset $i\,T_{\max}$, which exhibits a p.d.f. resulting from the convolution of i independent exponential backoffs, each with average duration$E[B]$:

$$Y_i^{(\lambda)} - i\,T_{\max} \sim \text{ Erl } \left[i, \frac{1}{E[B]}\right]. \quad (19)$$

$Y_i^{(\ell)} - i\,T_{\max}$ is a sum of i exponential RACH backoffs, each with average duration $E[B]$, plus a random number of exponential ACB backoffs, each with average $E[A]$. The number of ACB backoffs per state is geometrically distributed according to the ACB barring probability p_A, so that the distribution of time spent in ACB backoff is the same for each state. Hence, the total time spent in ACB in i states is the i-th convolutional power of the distribution for one state. Summing up all components, we derive the following expression (see the Appendix for details):

$$Y_i^{(\ell)} - i\,T_{\max} \sim \text{Erl}\left[i, \frac{1}{E[B]}\right] * \sum_{j=0}^{i} \binom{i}{j} (1-p_A)^{i-j} p_A^j \,\text{Erl}\left[j, \frac{1-p_A}{E[A]}\right]. \quad (20)$$

The random variable Z results from the sum of a constant term $T_{\min}$ plus two random intervals: $\mathcal{X}$, representing the time needed for receiving a RACH resolution message from the BS after the last RAO slot, and $\mathcal{W}$, which is the time needed to establish the RRC data connection after a successful RACH exchange. Both $\mathcal{X}$ and $\mathcal{W}$ are narrow time windows, and we approximate their distribution as uniform in $[0, T_{\max}-T_{\min}]$ and $[0, W_{\max}]$, respectively. $\mathcal{X}$ and $\mathcal{W}$ are independent, and $T_{\max} < W_{\max}$. Thus, $Z = T_{\min} + \mathcal{X} + \mathcal{W}$, from which we obtain the following p.d.f.:

$$f_Z(x)=\begin{cases}\frac{x-T_{\min}}{W_{\max}(T_{\max}-T_{\min})} & \text{if } x\in[T_{\min},T_{\max}];\\ \frac{1}{W_{\max}} & \text{if } x\in[T_{\max},T_{\min}+W_{\max}];\\ \frac{T_{\max}+W_{\max}-x}{W_{\max}(T_{\max}-T_{\min})} & \text{if } x\in[T_{\min}+W_{\max},T_{\max}+W_{\max}].\end{cases} \quad (21)$$

The Appendix further presentes an expression for the distribution of the latency under the above listed assumptions.

F. Average latency for successful requests

Beside the distribution of latency, we can explicitly write the average latency for transmission requests that are eventually served. We achieve that by considering all events and the corresponding average duration. For the primary flow λ, at each failed stage, a request spends $T_{\max}$ seconds before dropping the current attempt, and $E[B]$ seconds before being allowed to attempt again. Instead, in case of success, the request is guaranteed to access the network, on the average, within $E[Z]$ seconds. Hence, the average time a request waits before access in stage i can be written as follows:

$$E\left[D^{(\lambda)}|i\right] = (i-1)\left(E[B]+T_{\max}\right)+E[Z]. \quad (22)$$

In order to get the unconditional average latency we need to sum all the contributions of observing a success at any stage. Hence, each term can be computed as the average latency of a success at stage i, given by (22), times the probability of a success at the same stage:

$$E\left[D^{(\lambda)}\right] = \sum_{i=1}^{k_{\max}} \frac{P_S^{(\lambda)}(i)}{P_S^{(\lambda)}} E\left[D^{(\lambda)}|i\right]. \quad (23)$$

The computation of the average latency for the secondary flow follows from a similar derivation, taking also into account the time spent due to congestion control:

$$E\left[D^{(\ell)}|i\right] = i\frac{p_A}{1-p_A}E[A] + (i-1)\left(E[B]+T_{\max}\right)+E[Z]; \quad (24)$$

$$E\left[D^{(\ell)}\right] = \sum_{i=1}^{k_{\max}} \frac{P_S^{(\ell)}(i)}{P_S^{(\ell)}} E\left[D^{(\ell)}|i\right]; \quad (25)$$

where the first term in (24) accounts for the average time spent in the ACB stage, by counting the mean number of failed ACB attempts before a success and adding the average time for each one of them.

G. Optimal operational point

The **optimal operational point** of the system, n', originates from two requirements: on the one hand, the need of network operators to maximise the usage of the system and on the other hand users' satisfaction. The former requirement directly maps to maximise network throughput ξ while the latter requires to meet the QoS that users have subscribed, specifically concerning latency. Therefore, the point n' corresponds to the point where, for the first time, an increment of the offered load $\lambda+\ell$ does not correspond to an increment of the throughput. That is, n' corresponds to the minimum exogenous traffic rate capable of saturating network capacity, $\xi = \frac{C}{P_L}$, and for this reason $E[N_L] = 0$. Having no losses due to clipping, at the point n', requests move from one RACH stage to the subsequent due to ether *i)* a collision in the RACH or *ii)* a decoding failure. Assuming the RACH being far from saturation, and therefore $p_C \simeq 0$, (9) can be rewritten as

$$\gamma_i^{(\lambda)} \simeq \lambda e^{-\frac{1}{2}(i-1)i} \prod_{k=1}^{i-1} F_{Y_k^{(\lambda)}}(T_O), \quad (26)$$

and the same applies to the secondary flow component $\gamma_i^{(\ell)}$. From (26) it is easy to derive the probability to get to the i^{th} stage, as

$$P_N^{(\lambda)}(i) \simeq \frac{\gamma_{i+1}^{(\lambda)}}{\lambda} = e^{-\frac{1}{2}(i-1)i} \prod_{k=1}^{i-1} F_{Y_k^{(\lambda)}}(T_O). \quad (27)$$

Notably, for $k_{\max}$ big enough, (27) leads to negligible losses due to excessive RACH retrials, and hence flow rates are linked by the following formula

$$\lambda(1-P_{T_O})+\ell \simeq \xi. \quad (28)$$

The previous approximated expression allows the computation of the exogenous flow (either primary or secondary) required to reach the optimal operational point n' as a function of the other one.

Noting that, when $T_O \gg T_{\max}$, it is possible to use the approximation $F^{(\lambda)}_{Y_{i-1}+Z}(T_O) \simeq F^{(\lambda)}_{Y_i}(T_O)$, so that we can approximate P_{T_O} as follows:

$$P_{T_O} \simeq \sum_{i=1}^{k_{\max}} \left\{ e^{-\frac{1}{2}(i-1)i}\left[1-F^{(\lambda)}_{Y_i}(T_O)\right] \prod_{k=1}^{i-1} F^{(\lambda)}_{Y_k}(T_O) \right\} \quad (29)$$

The existence of $\mathbf{n'}$ supposes that *i)* the RACH is far from saturation, *ii)* clipping does not occur in practice, and *iii)* losses due to excessive retries are negligible. Condition *i)* can be evaluated by considering that, at n', $p_C \simeq \gamma\tau/N$ should be small (consider that $p_C = 1-e^{-1} \simeq 0.63$ when the RACH reaches its peak (i.e., $\frac{\gamma\tau}{N}=1$), and $p_C \simeq 0.4$ with half of the peak load (i.e., $\frac{\gamma\tau}{N}=\frac{1}{2}$), so that with values of p_C below 0.4 we are safely far from RACH saturation. Condition *ii)* should not occur unless clipping occurs before network saturation, which means that the system is not well designed (the capacity of the clipping must be at least as large as the one of the network, otherwise the effective capacity of the system is the one of the clipper). Condition *iii)* means that $P_N^{(\lambda)}(k_{\max})$ and

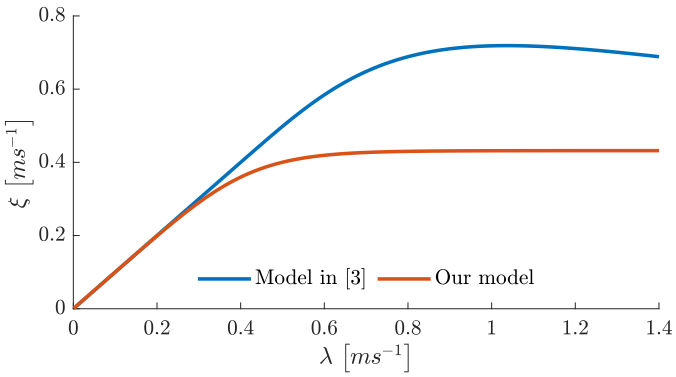


Fig. 3. Comparison between our simple model and the M2M model by Madueño *et al.* [3], which follows the behaviour of LTE-A signalling operations in detail. The latter is not meant to describe the behaviour of the system in saturated conditions, and hence a fair comparison with our model is possible only in the leftmost part of the figure.

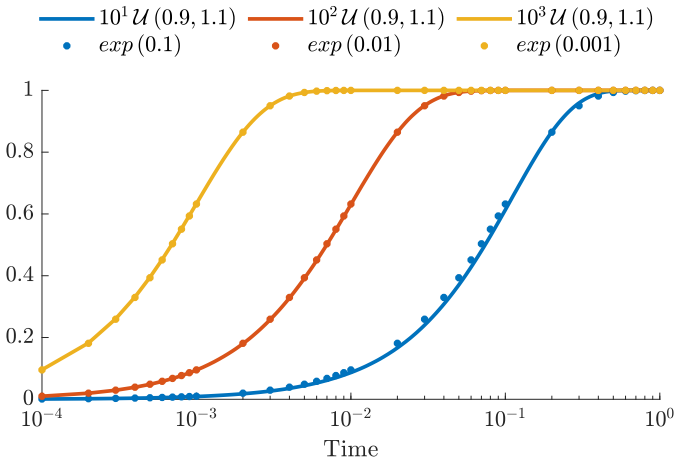


Fig. 4. Comparison of the CDF of a Poisson arrival stream against the one resulting from the superposition of $10, 100, 1000$ arrival streams with interarrival times distributed according to a $\mathcal{U}(0.9, 1.1)$.

$P_N^{(\ell)}(k_{\max})$ are close to zero, which can be checked with the expressions given above. In conclusion, assuming the clipper is not a bottleneck, n' exists and is found at $\lambda(1-P_{To})+\ell = C/S$ if the following conditions are satisfied:

- $p_C < 0.4$ (or equivalently, $\gamma < \frac{N}{2\tau}$),
- $P_N^{(\lambda)}(k_{\max}) \simeq 0$,
- $P_N^{(\ell)}(k_{\max}) \simeq 0$.

IV. Model Positioning

The model described so far is rather simple, and its solutiom requires low computational complexity. The heaviest part consists in computing the CDFs of $Y_i^{(\lambda)}$, $Y_i^{(\ell)}$ and Z, which can be done just once, offline. Moreover, deriving those distributions in closed form is trivial in case of simple distributions of backoffs. We do not show them here for lack of space. After computing the CDFs, one only needs to solve iteratively the equations described above. However, few iterations are enough for accurate results (not reported here for lack of space, we have observed that less than 5 iterations are needed) and each iteration scales linearly with the number of states $k_{\max}$.

Our model is generic, since it can be used for arbitrary population sizes and time constraints, so that it can be useful to design massive as well as mission-critical SF scenarios.

Our model does not consider in full detail the operations of signaling channels and access techniques of real networks, e.g., LTE/LTE-A. This implies that we need to validate our model against realistic simulations. However, before proceeding with a complete validation and performance evaluation, here we show that the results of previous very detailed models do not substantially depart from ours. In particular, we consider a model recently proposed by Madueño *et al.* [3], which can be used for the evaluation of M2M unsaturated scenarios, with sparse traffic and small payloads, RACH retries and dropped requests. The main differences between the model in [3] and ours consist in the fact that [3] models LTE-A signalling channels very accurately, that requests are never dropped because of lack of transmission resources, but only because of user impatience, and that users never return to the network before the RRC timeout.

Fig. 3 compares the predictions obtained with our model and with the model in [3]. In order to perform a fair comparison, we used the same configuration parameters for the two models. Specifically, we used the parameters suggested in [3] for M2M traffic, with a narrowband LTE-A cell (1.4 MHz, resulting in 12 OFDMA resource blocks per ms, $\tau = 10$ ms, $N = 54$) and a slow modulation and coding scheme (3.456 Mb/s) for all data and signalling channels. We use 1 Kbyte as fixed payload size and 40 ms as maximum waiting time for a request queued for service. Accordingly, in our model, we use $m = 4$ and $\Theta = 72$, which correspond to queue and serve RACH request in at most 40 ms. Fig. 3 shows the system throughput vs. the exogenous arrival rate generated by users. The two models behave quite similarly at low loads, i.e., in the range for which the model in [3] was designed. However, when approaching saturation, the two models substantially deviate from each other. Indeed, the model in [3] achieves unrealistically high throughputs, beyond the feasible bound imposed by channel speed (the flat region in the curve of our model) because that model does not consider that messages can be dropped because of lack of transmission resources over the PUSCH channel. Those resources are instead limited, as taken into account by our model. This comparison proves that our simple model can be as accurate as a more complex and detailed one, while at the some time resulting in a much more flexible and suitable tool for the evaluation of SF radio access in a much wider range of scenarios and configurations.

V. Numerical Results

Arrival process Poisson approximation. In this paper, we consider industrial (i.e., SF) scenarios where the network traffic consists of data from large numbers of MTDs. In the case of real-time control, data is normally generated from MTDs at quasi-deterministic intervals. On the contrary, data generation for monitoring and maintenance applications can be assumed more random.

As a consequence, modelling the request arrival processes as Poisson might appear an unacceptable simplification. However, it is well known that (in general) the Poisson process is the limit collective behaviour for increasing number of sources that independently generate arrivals. To support our modelling choice, we performed a set of simple simulation experiments, comparing the interarrival time CDF generated by a Poisson process against the one produced by different numbers of sources. Fig. 4 shows some of the results we have obtained. In particular, in this figure, we compare Poisson arrivals against the process resulting from superpositions of processes with interarrival times distributed according to a uniform distribution in the range $[0.9, 1.1]$ (Fig. 4-b). In the experiments, we vary the number of sources that generate arrivals, as well as the average number of total requests for each case. We can clearly see that the CDFs are very similar already for 10 independent sources, and become identical for 1000 sources. Since in SF scenarios the number of MTD is extremely high, we consider Poisson arrivals a reasonable approximation, even for relatively small numbers of MTDs.

SF experiment parameters. Since the focus of this work is on traffic generated by autonomous and automatic MTDs reporting to a central entity collecting data in the SF, single transmissions are of negligible dimensions and we assume $P_L = 1000$ bits as a realistic value. Furthermore, we will consider two different application respectively generating real-time and non-real-time traffic flows. Based on application-specific constraints, the traffic has a cyclic nature; therefore the duration of the cycle depends on the maximum allowable latency. In the following, the primary flow is characterized by a timeout $T_O = 100$ ms and a message generation interval equal to $\frac{4}{3}T_O$, so that any MTD generates a new message every 133.3 ms, on average. On the other hand, the secondary flow—the one without a tight time constraint—is characterized by a much lower arrival rate, here about one message per second. Moreover, we assume that MTDs can transmit at $C = 10$ Mb/s. With the above, the number of requests that can be served in a RAO is $m = 10$. Latencies strongly depend on the frequency of RACH opportunities. Here we use $\tau = 1$ ms, which corresponds to a RACH opportunity every 10 data slots in upcoming LTE Advanced Pro and 5G systems [8]. RACH and RRC connect timers are set to be of the order of magnitude of τ. Specifically, we use $T_{\min} = 0.2$ ms, $T_{\max} = 0.8$ ms and $W_{\max} = 1$ ms (respectively 2, 8 and 10 time slots). The number of RACH channels is $N = 54$, which is a typical value in 3GPP specifications. Unless otherwise specified, we use $\Theta = 18$ requests/ms which is realistic for 4G/5G base stations in which there can be up to 3 acknowledgements per time slot during $T_{\max} - T_{\min}$, and set the maximum number of retries to $k_{\max} = 10$. As concerns backoff timers, we use $E[B_i] = 10$ ms and $E[A_{ij}] = 4$ s for RACH and ABC retries, respectively, and $p_A = 0.5$, although the importance of $E[A_{ij}]$ and p_A is not shown in the paper for lack of space (they only affect the latency of flow ℓ without impairing any throughput).

A. System behaviour and model validation.

Fig. 5 presents the most significant quantities to characterise the system behaviour in presence of the primary flow only.

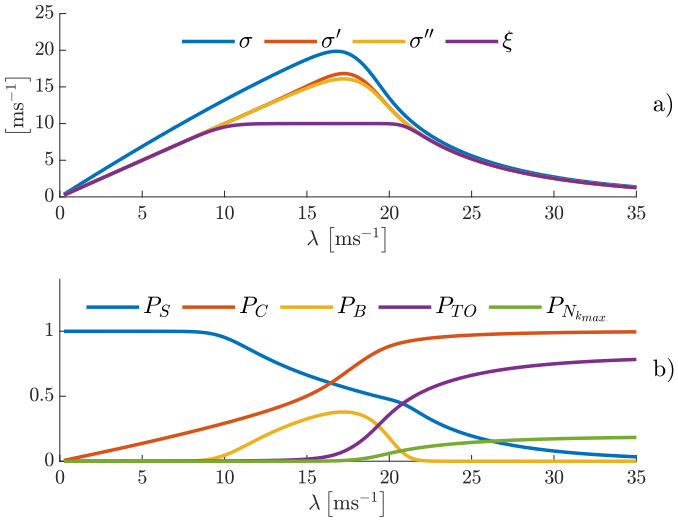


Fig. 5. System behaviour in the reference scenario in presence of flow λ: a) network throughput and flows leaving the RACH and the RRC connect phases with a success; b) state probabilities.

With the parameters described above, the upper part of the figure illustrates the dome-shaped relations between the system input λ and i) the amount or requests per unit time that pass the RACH without collision (σ, which is at most $\frac{N}{e}$, i.e., the max throughput of an N-channel Aloha), ii) the amount or requests per unit time that reach the base station with no decoding error (σ'), iii) that complete the RRC connect phase (σ'', which is limited by Θ), and iv) that eventually receive service (ξ, which is capped by m). Because of the structure of the system, the typical Aloha output flow σ is progressively scaled and flattened to become the system throughput ξ. We can identify 3 regions for ξ. An initial linear region in which the throughput grows almost linearly with the input; a flat region in which the throughput is practically constant or slightly recessing; and a breakdown region in which small increments of the input cause large throughput degradation.

Fig. 5-b gives some insight into the system reactions to progressively higher traffic loads. It is clear that in the linear region, the system works just fine: p_C is quite low, and both $P_{TO} = \sum_{i=1}^{k_{\max}} P_{TO}(i)$ and $P_B = \sum_{i=1}^{k_{\max}} P_B(i)$ are negligible, while the total success probability $P_S = \sum_{i=1}^{k_{\max}} P_S(i) \simeq 1$. However, as soon as the network throughput gets close to its maximum m, P_B begins to grow, and the system enters the flat region. This point corresponds to the first knee of ξ. Then, P_B grows higher, up to its maximum, corresponding to the largest RACH throughput. From this point on, the system behaviour is driven by p_C and P_{TO}. Indeed, the probability to leave the system shifts from low RACH states towards higher ones (not shown because of space limitations) since requests, on average, retry several times before leaving the system. Similarly, it can be observed that the state in which a success occurs shifts to high state numbers, as shown in in Fig. 6, where throughput components are illustrated. This same figure also shows the good accuracy achieved by our model in terms of throughput predictions. Indeed, analytical predictions match well the results of the detailed packet-level simulator we developed in Python. We can observe some limited, yet non-negligible, errors only in the rightmost region

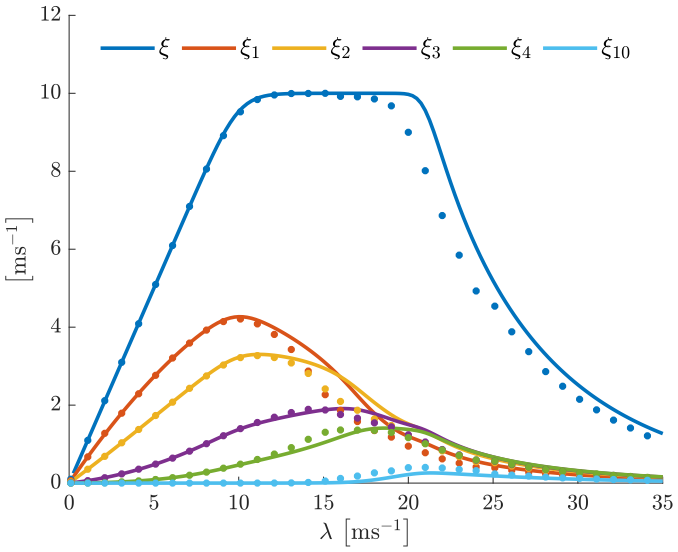


Fig. 6. Validation of the analytical model through simulation with primary flow only: network throughput $\xi^{(\lambda)}$ and its per-state components $\xi_i^{(\lambda)}$ (for readability the figure only shows $\xi^{(\lambda)}$, $\xi_1^{(\lambda)}$, $\xi_2^{(\lambda)}$, $\xi_3^{(\lambda)}$, $\xi_4^{(\lambda)}$, and $\xi_{10}^{(\lambda)}$).

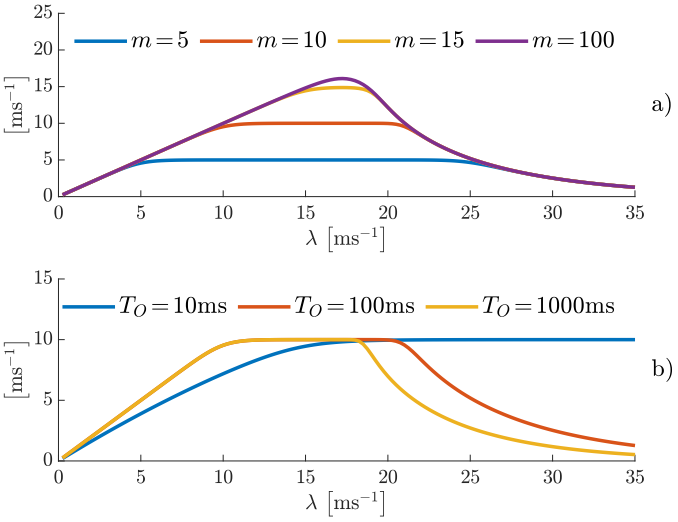


Fig. 7. Effect of varying model parameters on ξ: *a)* Number of clients served per RAO, m; *b)* Timeout

of ξ, which contains, however, no desirable operational points due to low success probability and, as we will show later, very high latency. We conducted many more model validation tests, which cannot be shown here due to lack of space. All tests show extremely good model accuracy, especially for loads below the breakdown region of ξ.

From these initial results, it is already clear that, to obtain a sufficiently good QoS level, it is desirable to keep the system in operational regimes below the point where the RACH saturates, before the beginning of the flat region of ξ.

B. Performance analysis

Impact of transmission rate and packet size. An obvious relation exists among the system throughput (the rate of requests successfully accessing the network, i.e., ξ), the network data rate C, the packet size P_L, and the number of requests that can be processed by the network in a time

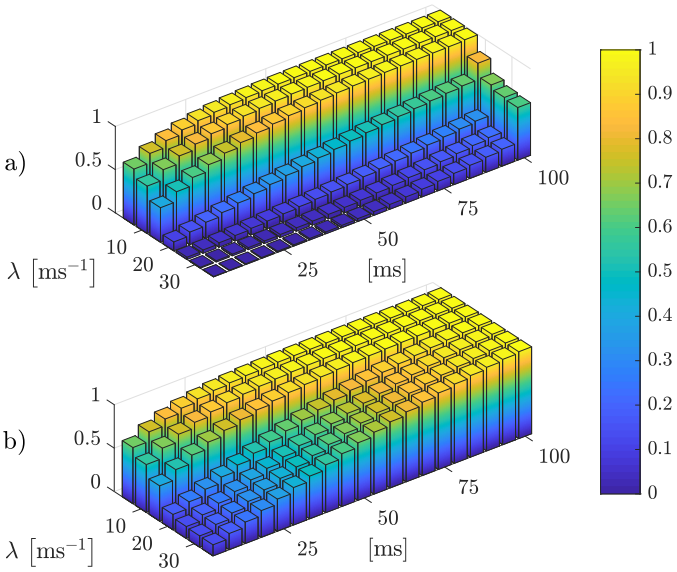


Fig. 8. Latency distributions: *a)* Distribution of latency of successful and unsuccessful requests, based on (15); *b)* Distribution of latency conditioned by a success, based on (16).

interval τ (i.e., m). For fixed τ, m only depends on the ratio $\frac{P_L}{C}$. Therefore, to understand the impact of C or P_L on throughput, it is enough to evaluate the impact of m—results and considerations for the secondary flow are consistent with the ones for the primary flow, for conciseness we omit results for the secondary flow. Fig. 7-a shows the effect of different values of m on the system throughput, while the rest of the parameters is kept as before. It is worth to point out that, independently of m, the throughput is limited by Θ (i.e., the max rate at which the BS can accept requests), so that high values of m perform practically the same. This can be translated into the following very relevant statement for system design and planning: *BS capacity increases can lead to (very) small performance improvements*.

Impact of timeout. Timeout is a very critical aspect of system design, due to the real-time nature of most of the traffic in SF. Fig. 7-b sheds light on the impact of the timeout value on system performance. In particular, we can observe that higher timeout values make MTDs saturate the network sooner. When the network is saturated, increases in λ lead to higher values of p_C, which cause a drastic decrease of ξ. Interestingly, low timeout values impact network throughput also for low input rates, while medium to high values of the timeout only impact the beginning of the breakdown region.

Latency performance. Fig. 8 shows how latency is affected by increasing incoming traffic λ, when $\ell = 0$. The two 3D plots depict the quantized CDFs latency of a request, with a quantum equal to 5 ms, ranging from 0 to the timeout $T_O = 100\,ms$. Specifically, Fig. 8 shows how the latency varies as the incoming traffic λ increases. With respect to the values for the parameter λ we can identify three different behaviours in the latency distribution. The first range of λ values (about $\lambda \in [0, 10]$) is the one for which we saw that the throughput increases linearly, up to the network saturation. The second region ($\lambda \in (10, 20)$) maps to the operational zone where the RACH is still behaving well, $\gamma < N/\tau$, and requests get blocked at the network. As a consequence,

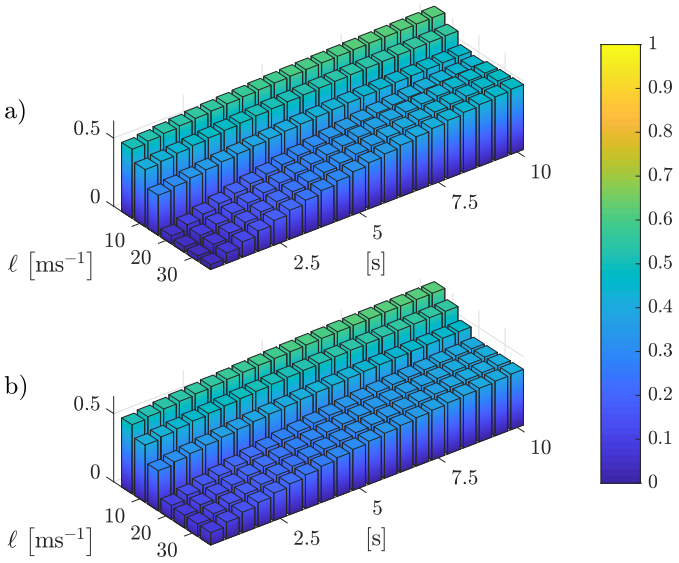


Fig. 9. Latency distributions: *a)* Distribution of latency of successful and unsuccessful requests, based on (17); *b)* Distribution of latency conditioned by a success, based on (18).

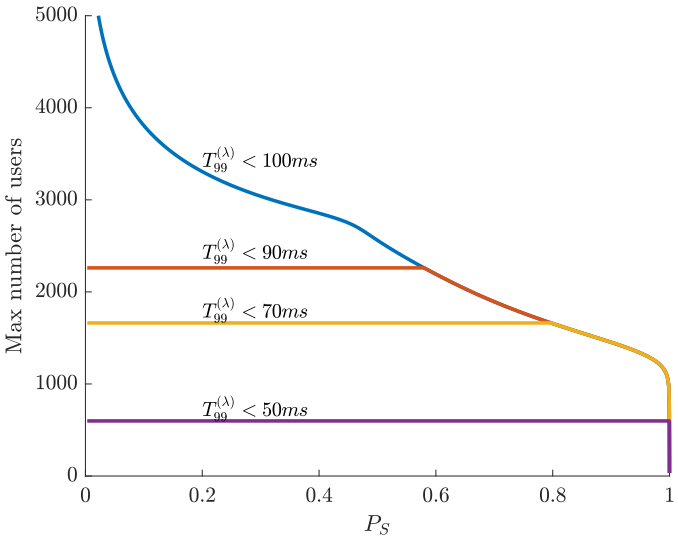


Fig. 10. Max number of MTDs that can be connected to a BS to guarantee success probability above a threshold and latency below a threshold.

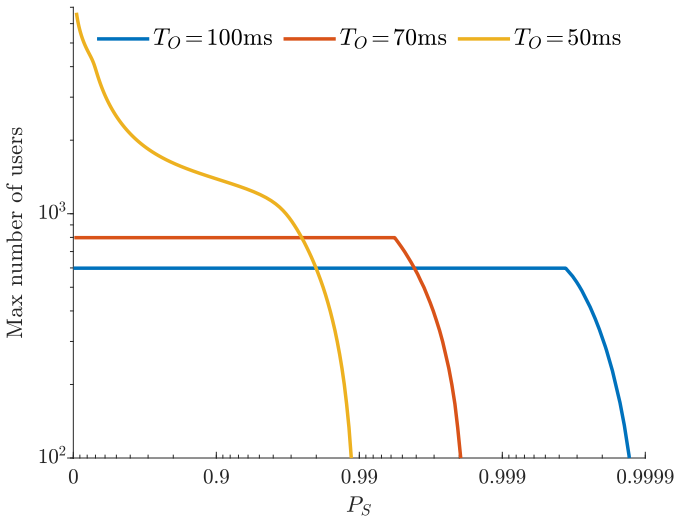


Fig. 11. Max number of MTDs that can be connected to a BS to guarantee success probability above a threshold and latency below a threshold of 50 *ms*

latency increases faster than in the previous region due to the increased collision probability at the RACH. In the range $\lambda \in [20, 35]$ the RACH is already saturated and requests spend their time within the system colliding in the RA procedure. The same kind of results is reported in Fig. 8-b, where latency percentiles are conditioned to a success. Therefore, CDF's shape straighten the definition of n' as the desired operational point. Indeed, having λ in the second region increases the latency but network throughput remains steady. moreover, further increases of λ eventually lead into the breakdown region. Fig. 9 depicts the latency distribution for the secondary flow ℓ, when $\lambda = 0$. Note that the absence of a hard time constraint drops the limits on latency. That is, requests exits the system only either if they exceed the number of available RA attempts or in case of a success. Indeed, as reported in Fig. 9 the latency notably increases, and this occurs because of the congestion control mechanism: at every failure a request has to perform an ACB backoff, which is remarkably longer compared to the other ones.

Sustainable cell population. The key question that an SF network designer has to face is how many cells are necessary to serve a given population of MTD, while providing a predefined QoS level. Our model answers this question by computing the mapping between KPIs and number of MTDs. Let us focus on a single cell operated with the default realistic parameters considered in this section.

Traffic flow with time constraint ($\lambda \in (0, 35)\, ms^{-1}$ and $\ell = 0$): Fig. 10 shows the maximum number of MTDs that can access the network (in the vertical axis) when the 99-th percentile of latency, conditioned to a success, is guaranteed (the value that labels the curves in the figure), as a function of the guaranteed total success probability P_S (in the horizontal axis). That is, the curves provide the greatest value of n that guarantees a latency with a 99-th percentile lower than a threshold (the curve label) and a success probability higher than another threshold (the abscissa). As a possible example, we see that one cell is able to handle (roughly) 2100 MTDs, guaranteeing latencies smaller than 90 ms for 99% of the requests, with $P_S \geq 0.6$ (this can be a condition which is representative of a massive scenario, which is however not critical, due to the low success probability value). However, when it comes to serving MTDs with high success probability (say above 90%), and low latency (say below 50 ms at the 99-th percentile of distribution), only a few hundred devices can be connected to a BS. This can be acceptable in a scenario that is critical, but not massive. On the contrary, in a massive and critical context, with high MTD density layouts in the order of tens or even hundreds of users per square meter, this would require deploying ultra-dense BS sets, each BS covering just a few square meters. This is clearly undoable in SF layouts and calls for further technology enhancements, which are out of the scope of this paper. The same considerations as above follow from Fig. 11, where we show the sustainable population size for several timeout configurations when limiting the admissible

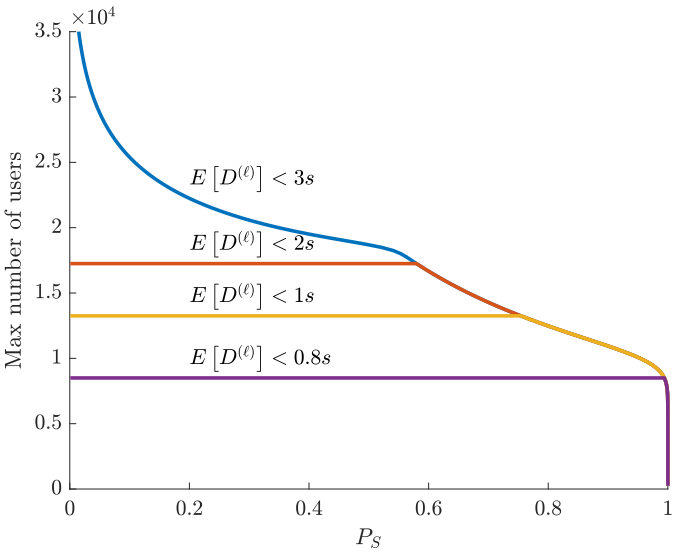


Fig. 12. Max number of MTDs that can be connected to a BS to guarantee success probability above a threshold and latency below a threshold

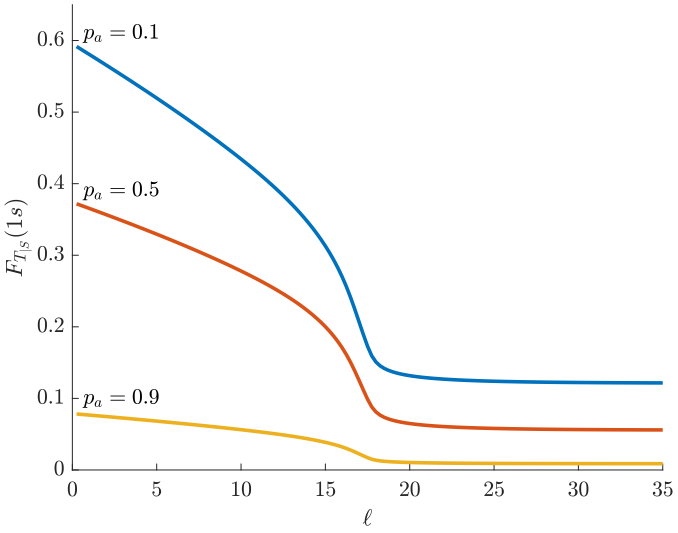


Fig. 13. Max number of MTDs that can be connected to a BS to guarantee success probability above a threshold and latency below a threshold

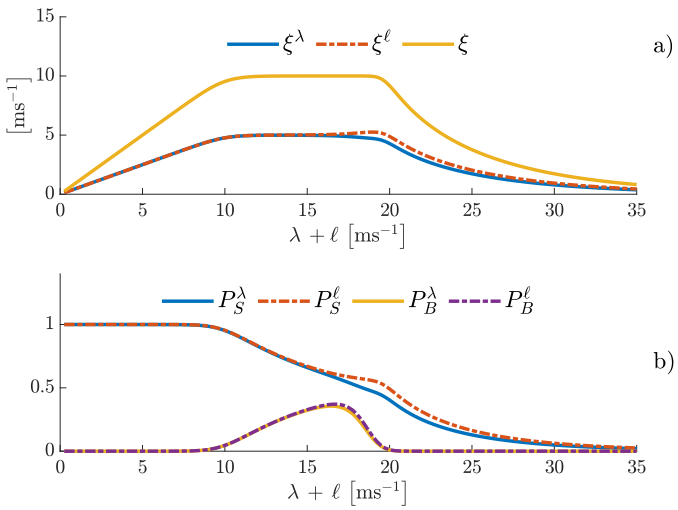


Fig. 14. Flows with different priorities: *a)* system throughput, *b)* P_S and P_B

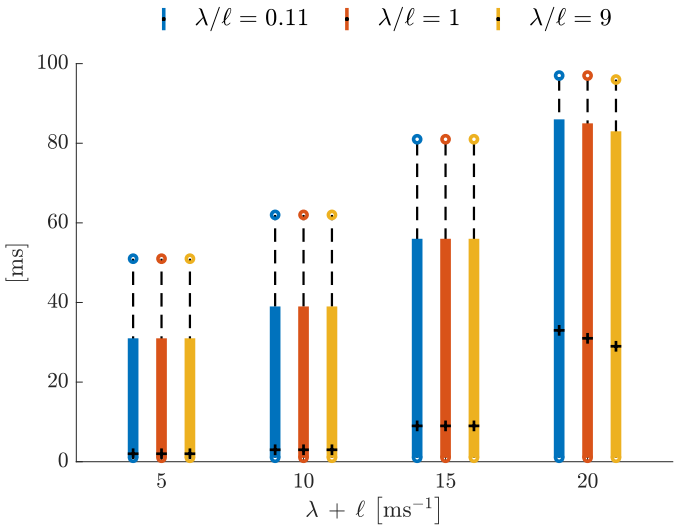


Fig. 15. Primary flow latency distribution in case of a success in the system with multiple flows. Lower and higher values correspond respectively 1st and 99-th percentiles; lower and higher ends of the boxes represent the first and third quartile, whilst the median is represented as a black cross

latency to 50 ms. Obviously, the success probability increases monotonically as the time available to the request increases. Nonetheless, massive scenarios can only be handled with unacceptably low success probability.

Traffic flow without time constraint ($\ell \in (0, 35)\, ms^{-1}$ and $\lambda = 0$): Fig. 12 shows the maximum number of MTDs that can access the network (in the vertical axis) when the average latency, conditioned to a success, is guaranteed (i.e. the labels on the curves), as function of the guaranteed total success probability P_S (in the horizontal axis). That is, the curves provide the greatest value of n that guarantees an average latency lower than a threshold (the curve label) and a success probability higher than another threshold (the abscissa). For instance, we see that a cell is capable of handling around 17500 MTD while assuring mean latency lower than 2 seconds. Clearly, this is not a massive and time critical scenario, but 2 seconds can be a suitable performance for a massive scenario of non-critical IoT applications, such as measurement and reporting applications. The rationale behind such high latency lays in the congestion control mechanism. Indeed, by configuration the flow ℓ gets (probabilistically) delayed of $E[A]$ seconds in favour of a prioritized traffic flow. Specifically, Fig 13 highlights the effect of the deferral on ℓ's latency. It shows how likely a secondary flow request is to face a delay of 1 second as the secondary flow ℓ increases. As arguable, higher barring probability affect greatly the perceived latency showing the two sided effects of the ACB. On the one hand, it is an effective tool to tune the service class access priority, but, on the other, it has to be carefully managed to guarantee minimum QoS requirements to the low priority traffic.

C. Coupling flows' performance

Fig. 14 and Fig. 15 provide results for a scenario where flows of requests with different nature and requirements coexist (i.e., time-critical and non-time-critical request flows). In

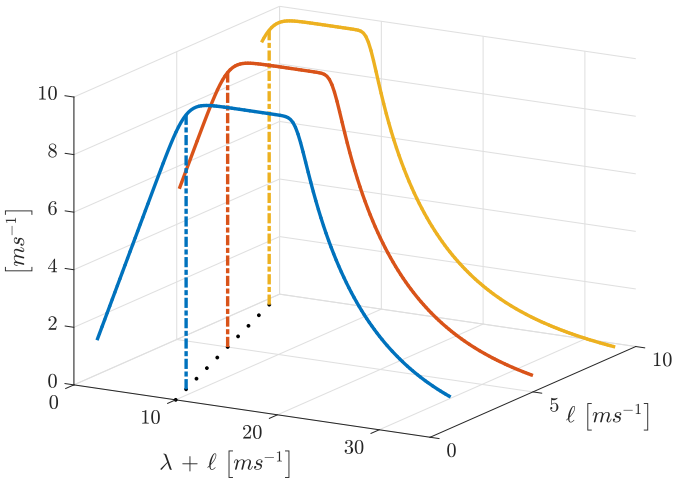


Fig. 16. Primary flow throughput for several values of fixed secondary flow incoming traffic. The vertical dashed lines represent the point n', approximated using Equation (28). The black-dashed line on the $x-y$ plane depicts how the point n' varies as a function of ℓ, again using (28).

particular, the x-axis of Fig. 14 represents the aggregate arrival rate of the two flows ($\lambda+\ell$). The two flows have an equal rate, so that they obtain the same throughput, as long as the timeout probability P_{TO} is negligible. We can observe from a global perspective that the throughput has the same characteristics of the case with requests of only one type. However, by looking separately at the two flows, we can observe that a decrease in $\xi^{(\lambda)}$ (due, for instance, to the effects of P_{TO}) favours the delay-tolerant traffic by increasing $\xi^{(\ell)}$.

For what concerns latency, Fig. 15 compares the latency distributions experienced by successful requests in the primary, time-critical flow, for various ratios λ/ℓ. The result is that the latency performance of the primary flow is barely dependent on the presence of flow ℓ, although it depends on the aggregate arrival rate. We can thereby conclude that regulating the secondary flow with ACB makes the primary flow experience priority when it comes to latency guarantees.

Network operational point approximation. Network orchestration is a very complex task to perform, and therefore, it is of paramount relevance to have efficient tools to foresee the effects of a new configuration, and solving our model might not be a suitable solution in spite of its limited computational complexity. Nonetheless, the presented model provides both indications on the operational point around which network operators ought to be, and an efficient way to compute it. Based on (28), Fig. 16 shows the location of the point n', providing a very accurate approximation. Furthermore, the black dashed line on the $x-y$ plane shows the correlation between the point n' and ℓ, as the secondary flows increases. We can see that the correlation appears to be practically linear, but this occurs because of the little impact of the timeout on the overall performance for the configuration used in Fig. 16.

With our model, we have numerically evaluated the impact of many parameters. Thanks to the set of experiments reported here, we have illustrated the main feature of the system, spotted potentially good features and desirable operational points and, most importantly, we have identified intrinsic limitations in the procedures used to access resources in 4G/5G networks, which will require additional research efforts to accommodate ultra-dense layouts of MTDs in SF scenarios of the future.

VI. Related Work

All forecasts predict that the next generation of cellular networks will support, in addition to traditional services, a wide variety of Machine-to-Machine (M2M) services, in the context of the IoT scenario.

The paper [9] outlines the impact that a massive use of M2M communications, and their coexistence with traditional services, will have on future networks. The paper and its references analyse the issues arising in these networks when a high load of M2M traffic must be served, and identify network access mechanisms as possible bottlenecks that may degrade the system performance.

Other investigations study the access mechanisms in LTE and in 5G networks in the case of M2M communications. Examples of such works are, for instance, [3], [10], [11]. All these papers include the performance modeling and analysis of the network access procedures for LTE and 5G, but, although they include many protocol features, (in general) they only focus on access mechanisms, without accounting for blocking at the BS, and for the reciprocal effects of blocking between access mechanisms and BS. The complex interactions of these two different bottlenecks have been highlighted in the case of massive access by using a measurement-based approach [12] and analysis [13]).

There exist also some recent studies on enhancing the random access procedure, e.g., by using ACB with power control, thus exploiting the so-called *capture effect* to partially solve the RACH collision problem [14], or by resolving collisions in the RACH transmissions instead of avoiding them [15]. Such approaches do not solve the problem of massive MTC scenarios like SF, in which the RRC connect phase can fail with non-negligible probability and cause unacceptable latencies due to multiple access retries.

Authors in [16] introduced a performance model for evaluating M2M communications in heterogeneous settings. This model has been used to study the coexistence between M2M and human-to-human communications in the same networks and for evaluating energy saving strategies.

VII. Conclusions

We have presented and validated a simple, yet accurate, model for the performance analysis and design of cellular networks in smart factory environments characterised by machine-type communications, including the massive and/or mission-critical cases. The model captures many aspects of the dynamics in a cell, such as the different phases of the access procedure, the possible contention preamble collisions and the limited number of uplink grants in the random access response message, the limited number of retrials, the coexistence of different types of traffic (real-time and non-real-time), the use of a timeout for real-time traffic, and the prioritization of different types of traffic flows (e.g., with the ACB technique). Using our model, it is possible to pinpoint the optimal operational point—where the system shows maximum

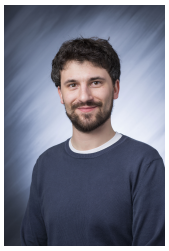
utilization and limited latency—for 3GPP compliant networks. Furthermore, we have also derived an efficient and simple approximation to identify such point. This approximation can therefore be easily plugged in global optimizations tools, giving the relevant information about the system, avoiding the complexity of the whole model.

The model results were validated with detailed simulations, and proved to be in very good agreement with a previously published very detailed model, in spite of the simplifying assumptions introduced for analytical tractability.

The main merit of the proposed model lies in the valuable insight that it brings on cellular system operations and in the possibility to use it to drive the correct dimensioning of the cellular system in smart factory scenarios. The model also unveils some intrinsic limitations of the class of random access procedures adopted in cellular networks, and can be instrumental for the design of more effective algorithms.

REFERENCES

[1] W. Mohr, "5G empowering vertical industries," Tech. Rep., April 2016.
[2] M. Maternia *et al.*, "5G PPP use cases and performance evaluation models," 5G-PPP, Tech. Rep., Apr. 2016, White Paper.
[3] G. C. Madueño *et al.*, "Assessment of LTE wireless access for monitoring of energy distribution in the smart grid," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 3, pp. 675–688, 2016.
[4] "Evolved Universal Terrestrial Radio Access (E-UTRA); Medium Access Control (MAC) protocol specification," 3GPP, TS 36.321 Release 13 V13.1.0, April 2016.
[5] "Technical Specification Group Radio Access Network; Study on RAN Improvements for Machine-type Communications," 3GPP, TR 37.868 Release 11, 2011.
[6] I. Leyva-Mayorga, *et al.*, "Performance Analysis of Access Class Barring for Handling Massive M2M Traffic in LTE-A Networks," in *IEEE International Conference on Communications (ICC)*, 2016.
[7] D.-W. Seo, "Explicit Formulae for Characteristics of Finite-Capacity M/D/1 Queues," *ETRI Journal*, vol. 36, no. 4, pp. 609–616, 2014.
[8] E. Dahlman, S. Parkvall, and J. Skold, *4G, LTE-Advanced Pro and The Road to 5G, Third Edition*, 3rd ed. Academic Press, 2016.
[9] A. Biral *et al.*, "The challenges of M2M massive access in wireless cellular networks," *Digital Communications and Networks*, vol. 1, no. 1, pp. 1 – 19, 2015.
[10] O. Arouk *et al.*, "Performance Analysis of RACH Procedure with Beta Traffic-Activated Machine-Type-Communication," in *Proc. of GLOBECOM*, 2015.
[11] S. Cherkaoui *et al.*, "LTE-A random access channel capacity evaluation for M2M communications," in *Proc. of Wireless Days*, 2016.
[12] M. Zubair Shafiq *et al.*, "A First Look at Cellular Network Performance During Crowded Events," in *Proc. of ACM SIGMETRICS*, 2013.
[13] P. Castagno *et al.*, "Why Your Smartphone doesn't Work in Very Crowded Environments," in *Proc. of WoWMoM*, 2017.
[14] Z. Alavikia and A. Ghasemi, "A multiple power level random access method for M2M communications in LTE-A network," *Transactions on Emerging Telecommunications Technologies*, 2016.
[15] M. S. Ali, E. Hossain, and D. I. Kim, "LTE/LTE-A Random Access for Massive Machine-Type Communications in Smart Cities," *IEEE Communications Magazine*, vol. 55, no. 1, pp. 76–83, January 2017.
[16] D. Niyato, P. Wang, and D. I. Kim, "Performance modeling and analysis of heterogeneous machine type communications," *IEEE Transactions on Wireless Communications*, vol. 13, no. 5, pp. 2836–2849, 2014.



Paolo Castagno is Post-Doc at the Computer Science Department, University of Torino, Torino, Italy. He received its maser and Ph.D. degrees at the Universisty of Torino, in 2014 and 2018 respectively. His research focus is on performance evaluation of computer systems and communication networks, with a specific interest on wireless networks.

Vincenzo Mancuso is Research Associate Professor at IMDEA Networks, Madrid, Spain, and recipient of a Ramon y Cajal research grant of the Spanish Ministry of Science and Innovation. Previously, he was with INRIA (France), Rice University (USA) and University of Palermo (Italy), from where he obtained his Ph.D. in 2005. His research focus is on analysis, design, and experimental evaluation of opportunistic wireless architectures and mobile broadband services.

Matteo Sereno was born in Nocera Inferiore, Italy. He received the Laurea degree in Computer Science from the University of Salerno, in 1987 and the Ph.D. degree in Computer Science from the University of Torino, in 1992. He is currently Full Professor at the Computer Science Department, University of Torino. His current research interests are in the area of performance evaluation of computer systems, communication networks, peer-to-peer systems, compressive sensing and coding techniques in distributed applications, game theory, queueing networks, and stochastic Petri net models.

Marco Ajmone Marsan is a full professor at the Electronics and Telecommunications Department of the Politecnico di Torino in Italy, and a part-time research professor at IMDEA Networks Institute in Leganes, Spain. Marco Ajmone Marsan obtained degrees in EE from the Politecnico di Torino in 1974 and the University of California, Los Angeles (UCLA) in 1978. He received a honorary doctoral degree in Telecommunication Networks from the Budapest University of Technology and Economics in 2002. Since 1974 he has been at Politecnico di Torino, in the different roles of an academic career, with an interruption from 1987 to 1990, when he was a full professor at the Computer Science Department of the University of Milan. Marco Ajmone Marsan has been doing research in the fields of digital transmission, distributed systems and networking. He has published over 350 papers in the leading conferences and journals of his research area. He is also coauthor of two books: Performance Models of Multiprocessor Systems (MIT Press, 1987) and Modelling with Generalized Stochastic Petri Nets (John Wiley, 1995). Marco Ajmone Marsan has been a member of the editorial board and of the steering committee of the "ACM/IEEE Transactions on Networking". He is a member of the editorial boards of the journals "Computer Networks"and "Performance Evaluation"of Elsevier, and of the "ACM Transactions on Modeling and Performance Evaluation of Computer Systems". He served in the organizing committee of several leading networking conferences, and he was general chair of INFOCOM 2013. Marco Ajmone Marsan is a Fellow of the IEEE, a member of the Academy of Sciences of Torino, and a member of Academia Europaea. Marco Ajmone Marsan was the Vice-Rector for Research, Innovation and Technology Transfer at the Politecnico di Torino from 2005 to 2009. From 2002 to 2009 he was the Director of the Istituto di Elettronica e Ingegneria dell'Informazione e delle Telecomunicazioni of the Italian National Research Council. He was the Italian delegate in the ICT and IDEAS committees of FP7.

APPENDIX

Here we derive the general expression for the distribution of $Y_i^{(\ell)}$, which is the time needed to enter stage $i+1$ and is key to compute the distribution of the latency of the secondary flow. We also derive the distribution of the latency before a success for the secondary flow.

$Y_i^{(\ell)}$ is a composed by i exponential i.i.d. RACH backoffs with average $E[B]$, and a geometrically distributed number of exponential i.i.d. ACB backoffs with average $E[A]$ and parameter p_A for each passage through the RACH (i.e., i times). RACH and ACB backoffs are independent, so that the above translates into the following expression:

$$Y_i^{(\ell)} - i\,T_{\max} \sim \text{Erl}\left[i, \frac{1}{E[B]}\right] * \left(\sum_{j=0}^{\infty} (1-p_A) p_A^j \,\text{Erl}\left[j, \frac{1}{E[A]}\right]\right)^{*i}, \quad (30)$$

where $\text{Erl}[n,\mu](x) = \frac{(x\mu)^{n-1}}{(n-1)!}\mu e^{-x\mu}u(x)$ is the Erlang distribution built by summing n i.i.d. negative exponentials, each with average $1/\mu$.

Consider now the LST of the summation in between parentheses in (30), which is the distribution of a r.v. modeling the time spent in the ABC subsystem at each passage:

$$\begin{aligned}
&\mathcal{L}\left[\sum_{j=0}^{\infty} (1-p_A) p_A^j \,\text{Erl}\left[j, \frac{1}{E[A]}\right]\right](s) \\
&= \sum_{j=0}^{\infty} (1-p_A) p_A^j \left(\frac{\frac{1}{E[A]}}{s+\frac{1}{E[A]}}\right)^j = (1-p_A)\,\frac{1}{1-\frac{\frac{p_A}{E[A]}}{s+\frac{1}{E[A]}}} \\
&= (1-p_A) + p_A \frac{\frac{1-p_A}{E[A]}}{s+\frac{1-p_A}{E[A]}}. \qquad (31)
\end{aligned}$$

The above is the LST of a weighted sum between a constant (a value 0 w.p. $1-p_A$, i.e., the probability of skipping the barring) and a negative exponential with average $E[A]/(1-p_A)$ (weighted with a probability p_A, i.e., the probability to be barred). Therefore, we can re-write (30) as follows:

$$Y_i^{(\ell)} - i\,T_{\max} \sim \text{Erl}\left[i, \frac{1}{E[B]}\right] * \left((1-p_A)\,\delta(x) + p_A \text{Exp}\left[\frac{1-p_A}{E[A]}\right]\right)^{*i}, \quad (32)$$

Consider now the LST of the convolutional power in the above expression:

$$\begin{aligned}
&\left[(1-p_A) + p_A \frac{\frac{1-p_A}{E[A]}}{s+\frac{1-p_A}{E[A]}}\right]^i \\
&\qquad = \sum_{j=0}^{i} \binom{i}{j} (1-p_A)^{i-j} p_A^j \left(\frac{\frac{1-p_A}{E[A]}}{s+\frac{1-p_A}{E[A]}}\right)^j, \qquad (33)
\end{aligned}$$

which is a weighted sum of Erlang distributions expressed in the LST domain, so that (30) is finally re-written as (20).

With the above, the latency before a success in stage i, i.e., the r.v., $Y_i + Z$ is distributed as follows:

$$\begin{aligned}
f_{Y_i^{(\ell)}}(x) &= f_Z(x - iT_{\max}) * \text{Erl}\left[i, \frac{1}{E[B]}\right](x) \\
&\quad * \sum_{k=0}^{i} \binom{i}{k} (1-p_A)^{i-k} p_A^k \,\text{Erl}\left[k, \frac{1-p_A}{E[A]}\right](x). \qquad (34)
\end{aligned}$$

Assuming now that $Z \sim U(0, T_{\max})$ (which has LST $\frac{1-e^{-sT_{\max}}}{s}$) and recalling that the LST of a time shift $iT_{\max}$ is $e^{-siT_{\max}}$, if we remove the conditioning on i in (34), i.e., on having a success in stage i, we obtain the following LST for the latency of a successfully served request:

$$\begin{aligned}
\hat{f}_{T_{|S}^{(\ell)}}(s) &= \frac{1-e^{-sT_{\max}}}{s} \cdot \frac{1}{P_S^{(\ell)}} \\
&\cdot \sum_{i=1}^{k_{\max}} P_S^{(\ell)}(i) \left(\frac{e^{-sT_{\max}}}{1+sE[B]} \cdot \frac{(1-p_A)(1+sE[A])}{(1-p_A)+sE[A]}\right)^i. \qquad (35)
\end{aligned}$$