Modeling of Aggregated IoT Traffic and its Application to an IoT Cloud

Florian Metzger*, Tobias Hoßfeld*, André Bauer†, Samuel Kounev†, Poul E. Heegaard‡
*University of Würzburg, Chair of Communication Networks, Würzburg, Germany
†University of Würzburg, Chair of Software Engineering, Würzburg, Germany
‡NTNU - Norwegian University of Science and Technology, Trondheim, Norway
Email: florian.metzger@uni-wuerzburg.de

***Abstract*—As the Internet of Things (IoT) continues to gain traction in telecommunication networks, a very large number of devices are expected to be connected and used in the near future. In order to appropriately plan and dimension the network, as well as the back-end cloud systems and the resulting signaling load, traffic models are employed. These models are designed to accurately capture and predict the properties of IoT traffic in a concise manner. To achieve this, Poisson process approximations, based on the Palm-Khintchine theorem, have often been used in the past. Due to the scale (and the difference in scales in various IoT networks) of the modeled systems, the fidelity of this approximation is crucial, as in practice, it is very challenging to accurately measure or simulate large-scale IoT deployments.**

The main goal of this paper is to understand the level of accuracy of the Poisson approximation model. To this end, we first survey both common IoT network properties and network scales as well as traffic types. Second, we explain and discuss the Palm-Khintiche theorem, how it is applied to the problem, and which inaccuracies can occur when using it. Based on this, we derive guidelines as to when a Poisson process can be assumed for aggregated periodic IoT traffic. Finally, we evaluate our approach in the context of an IoT cloud scaler use case.

I. Introduction

The Internet of Things (IoT) is a networking challenge where billions of new devices fulfilling numerous purposes will be interconnected across the digital landscape. According to the news website Business Insider, IoT devices will account for 24 billion of the 34 billion devices connected to the Internet by 2020.[1] Gartner also gives an estimate of more than 20 billion IoT devices by 2020.[2] IoT refers to the inter-networking of entities such as physical devices and objects. Such objects are equipped with circuitry, software, sensors, actuators, and network connectivity, enabling them to collect data from multiple modalities (e.g., sight, sound, tactile) and react on these inputs. Generally speaking, the Internet of Things consists of generic multipurpose devices usually connected to the Internet. Data to and from the devices is either: (i) collected from the devices, aggregated by an aggregator, and processed or stored (a typical client-server approach), or (ii) pushed to the devices, e.g., in a multicast approach, or (iii) exchanged between the devices in a peer-to-peer manner. In this paper, we consider the client-server scenario in the "IoT cloud" use case, where data is collected from a large number of devices and centrally (sometimes hierarchically) aggregated.

The exponential growth in the number of devices naturally raises the question of scalability of the underlying infrastructure. Scalability can be achieved on different levels. Choosing the right combination of protocols and access technologies provides the flexibility needed to support a specific choice of architecture, where brokers and gateways can be dynamically placed. However, at the same time, it introduces new potential performance bottlenecks, such as gateways or load balancers. Achieving scalability may require the development of new, adaptive load balancing mechanisms that are properly dimensioned. As we consider a scenario where data is collected for further processing in the cloud, the back-end cloud systems have to be scaled in a similar fashion.

In order to evaluate the scalability as the number of devices increases, first and foremost the behavior of IoT devices must be modeled, in particular the traffic patterns. IoT sensors are often sending data in a deterministic periodic manner. Therefore, the aggregated traffic from large numbers of such devices can be considered as a superposition of deterministic point processes. Assuming the point processes to be independent (see for example [1]–[3]), the aggregated traffic can be modeled as a Poisson process, which significantly simplifies the modeling of the aggregated arrival process. However, the deterministic periodicity of the individual devices introduces an error term to the Poisson approximation. This was already addressed by [4], and is, for example, known from works on aggregated periodic cell patterns in ATM networks [5]. Further work published in [6] discusses how the superposition of processes can be applied to modeling packets, flows, and sessions in access and core networks. The aim of our paper is to quantify this approximation by comparing statistical characteristics of the traffic processes. A cloud server case study is applied to compare the Aggregated Periodic traffic Process (APP) with a Poisson Process (PP) approximation. Based on analytical and numerical results, we formulate guidelines for when the Poisson process approximation can be used to model aggregated IoT traffic.

The remainder of this paper is structured as denoted in Fig. 1 and as follows. Section II gives background on IoT data characteristics and reviews related work. Section III provides traffic characteristics of selected IoT applications by surveying related work. Section IV compares the characteristics of the

[1] http://www.businessinsider.com/top-internet-of-things-trends-2016-1
[2] http://www.gartner.com/newsroom/id/3165317

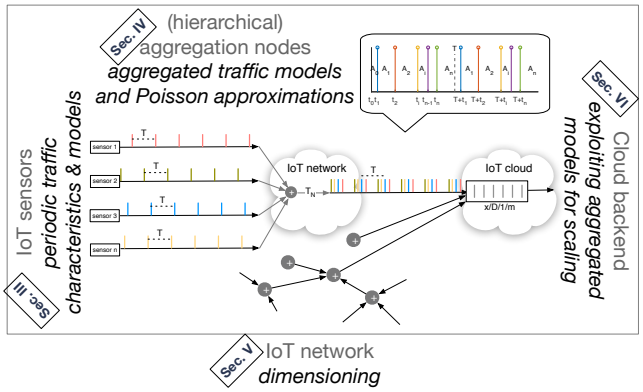

Figure 1. Common architectural elements of IoT networks and related topics in the indicated sections.

APP and the Poisson process traffic approximation. Section V introduces the IoT cloud use case and demonstrates the accuracy and applicability of the Poisson process approximation. The gained knowledge is applied to a cloud autoscaler alongside an experimental evaluation of this scaler in Section–VI. Finally, Section VII concludes this work.

II. The IoT Environment

In order to understand the behavior of IoT traffic, we first take a look at contemporary IoT communication systems and network structures.

A. Common Architectural Elements

Despite having roots deep in Wireless Sensor Networks (WSNs) and mesh networks, IoT networks fundamentally differ from their predecessors' flat meshed structure. Instead, most systems exhibit a simple centralized structure. A number of simple IoT devices — mostly sensors or actuators — that share a commonality (e.g., having the same owner or being situated in the same building or region) connect to one responsible hub, or *aggregator*. Often there are multiple hierarchical levels of aggregation (from the edge to the cloud) to combine different regional hubs together — see also again Fig. 1 for this basic architecture. This results is a network structure not unlike that of a mobile operator with its multiple cells, backhaul, and a common core network, or in our case a central cloud processing platform.

B. Scalability and Flexibility of the IoT Stack

The IoT devices themselves demand a certain rethinking of the entire protocol stack as well. The typical Internet approach of using RESTful HTTP atop TLS and TCP is regarded as too heavyweight (especially due to its statefulness) for many such devices, calling for lighter or better scalable approaches [7]. More suitable examples include, e.g., the protocols Message Queue Telemetry Transport (MQTT) and Constrained Application Protocol (CoAP). With its publish-subscribe approach and a tunable message reliability system MQTT [8], using TCP/IP with or without TLS, offers a variety of features that can be beneficial for many IoT use cases. The protocol is often employed deeper in the IoT hierarchy or in devices with more resources. A lightweight variant of MQTT is MQTT-SN, but it is not directly compatible with MQTT and needs an interconnecting gateway. More in line with HTTP's RESTful approach is IETF's CoAP [9], albeit being trimmed towards IoT through less overhead, UDP/DTLS usage, statelessness, and customizable payload data formats. These communication protocols are designed for centralized or hierarchical architectures with multiple layers of aggregating brokers.

C. Appropriate Radio Access Technologies for IoT

While in many settings, such as home automation, localized Personal Area Network (PAN) communication is desirable (e.g., using an interface from the IEEE 802.15.4 family), other scenarios require RF interfaces with a larger coverage area to reach the aggregation node. This can be provided by new forms of Ultra Narrow Band (UNB) radio connectivity, and opens up an entirely new category of communication modes, aptly dubbed Low-Power Wide-Area Network (LPWAN). This umbrella term subsumes a wide range of different protocols, including the chirp spread spectrum based LoRa [10] and its standardized link layer stack LoRaWAN [11]. Since it operates in unlicensed radio spectrum, LoRaWAN has received public interest through provider-backed installations (e.g., a deployment providing nation-wide coverage in South Korea[3]), but also through community-operated gateway networks where anyone can participate.[4] With a projected range of up to $20\,\mathrm{km}$ in rural areas, such gateways can aggregate the traffic from thousands of devices. LPWAN standards that operate on unlicensed bands may, however, face certain congestion challenges that are not unlike that of $2.4\,\mathrm{GHz}$ WiFi, where the band occupation and the resulting collisions and interference have become worryingly high, especially now that variants of LTE exist that offload into this band [12]. Due to the much higher range of LoRa and other approaches, as well as the number of legacy devices that operate on these bands, this situation may become reality sooner rather than later [13]. Adhering to a strict communication discipline and minimizing the amount of transmitted data and transceiver air time might alleviate the situation.

Other LPWAN options include new IoT-friendly variants of the 3GPP cellular networks, particularly NarrowBand IoT (NB-IoT) [14], which operators are now starting to adopt in their networks. While LoRa specifies only a very shallow protocol stack and leaves all other details to the specific implementation (with the option to use LoRaWAN), NB-IoT brings along the usual deep 3GPP stack.

With the broad selection of protocol stacks and RF interfaces, a few different communication patterns emerge in IoT. Using a RESTful or a publish/subscribe approach, the ability to constantly send large amounts of data or being restricted to minimal data and large periods will directly influence the traffic characteristics. This is discussed in the next section.

[3]https://www.semtech.com/company/press/LoRaWAN-IoT-Network-Deployed-Nationwide-in-South-Korea-by-SK-Telecom-Covers-99-Percent-of-Population

[4]See, e.g., https://www.thethingsnetwork.org/

III. A Survey of IoT Traffic Characteristics of Select IoT Applications

In general, IoT-traffic can be roughly partitioned into periodic and event-based modes of communication (see also, e.g., [15], [16]). Some applications will always be event-driven. Consider for example a smart home equipped with motion detection sensors. They are triggered by events outside of the domain of influence of these devices. But even here, emergent periodicity can ensue. For example, leaving for work and returning home each day at roughly the same time might activate motion sensors installed in the home hallway in a predictable, periodic manner, cf. [17]. Additionally, many IoT devices from other fields of application often intrinsically communicate in a periodic fashion. A prominent example are Smart Grids. This includes not only the measurement and collection of current power usage values from residential and industrial Smart Meters, but also the supervision, management, and maintenance of the power generation and distribution network [18]. Once again, these usually operate periodically with different intervals depending on the type of data, but may switch to pushing events in case of critical readings.

Summing up, for the purpose of this paper one can describe IoT traffic by describing either the communication periods, i.e. the period lengths (and, if applicable, the variability of the period) and the amount of data sent, or using a probabilistic model to describe the triggering event (and again the data amount). Depending on the scenario other factors might play a role as well, e.g. the directionality of the transmission. This is then combined with QoS criteria. In the case of IoT this is usually the expected maximum end-to-end delay and the loss rate.

A. IoT Traffic Models and Characteristics in Literature

Due to their shared heritage and similarities, the literature covered here includes works from WSN and Machine-Type Communications (MTC), where traffic models have been investigated more closely in the past.

For example, [19] provides numerical simulation results and investigates aggregate packet counts in which both periodic and event-driven communication appears. [20], [21] attempt to show that in MTC the classical Markovian arrival process assumption does not hold due to the burstiness of the traffic. Instead, a Beta distribution should be employed. On the other hand, the work conducted in [22] strives to verify that a Poisson distribution can indeed be applicable at least to the general (LTE-A) connection establishment process (without limitation to IoT devices). [23] explores a large-scale mobile network measurement dataset for well-known Machine-to-Machine (M2M) device types and evaluates the traffic characteristics of these devices.

When speaking of IoT traffic characteristics, of special note are as mentioned the periodic patterns (or the session Inter Arrival Time (IAT)) and message sizes that stand apart from typical mobile phone session arrival processes. The minimum period length can even depend on the underlying communication technology. GPRS for example can not support arbitrarily short messaging periods for a large number of devices without modification due to the imposed signaling interactions and limited available radio resources [24]. In a typical scenario the shortest period is estimated to be $5\,\mathrm{min}$ [25]. Additional work proposes to better utilize the Random Access Channel (RACH) in current and future mobile technologies to allow for more devices and be more resource efficient [26], [27].

With such limitations in mind, Table IV compiles measured, assumed, and modeled traffic characteristics from various publications and standards with a focus on their communication periods. Further publications overview existing and proposed applications of IoT, e.g. in industry automation and supervision [28], cloud-backed at home or in enterprise-settings [29], or smart environment scenarios [30] as well. In the table it is immediately evident that most scenarios assume at least some kind of periodic component, usually with a period length in the order of minutes, and a very high density of distributed devices, albeit with a rather low amount of data per device per period.

B. Traffic Projections Using a Toy Model

But one does not need to solely rely on traffic data from past publications and can instead set up some rough toy models for IoT traffic as well. The predictability of the household smart meter distribution and their communication behavior can be exploited for such a forecast [21], [25], [31], since there should always be only one per household and their installation is mandated by law in many countries.

Assuming that every household will have a smart meter that connects to the same LPWAN network, we can for example map population data from the German Federal Statistical Office [32] to the expected radio coverage from LPWAN gateways. The statistical offices give data on city population and population density for all major German cities, as well as a forecast on the average household size in Germany (1.97 for the year 2020, 1.9 in 2035).

This data can now be combined in a simple model, using a naive radial range model, e.g., for LoRa gateways, to calculate an estimate of the expected number of households — and thus the number of household smart meters along with it — per gateway for a given city. A toy mapping for a few exemplary cities is integrated into the aforementioned Table IV. The model considers a conservative urban LoRa range of one mile, and thus results in an estimated mean number of smart meter numbers per gateway ranging from roughly 2,000 (for the city of Salzgitter) up to 19,000 (for Munich).

One hierarchy level above, in the case of a citywide aggregation of the smart meter data, the number of households can reach $1.7\,\mathrm{M}$ (e.g. in Berlin). Aggregators would have to deal with these numbers of devices, with the sending intervals probably even getting shorter in the future (depending on upcoming legal regulations, since, e.g., time-precise smart meter readings can have direct repercussions for the household's privacy). Naturally, this raises immediate dimensioning and scalability questions for those aggregation nodes.

IV. Analysis of Aggregated Traffic Patterns

As outlined in the previous section, IoT traffic emerges from a large amount of sensor nodes which is aggregated

in the IoT architectures at different points, like the IoT gateways or IoT load balancer at an IoT cloud. For the performance analysis of such an IoT system, queuing theory provides fundamental results that are applied in this section. Such analytical approaches are required e.g. to investigate the dimensioning of gateways and scalability of the entire system, since simulations or testbeds are limited in size due to the necessary computational time and incurred costs. In particular, we will take a closer look at the superposition of periodic traffic processes from a large number of unsynchronized IoT nodes. Thereby, the Palm-Khintchine theorem tells that the aggregated traffic can be approximated with a Poisson process under certain conditions and for a large number n of nodes. To this end, we will investigate in this section whether the Poisson approximation is valid in the IoT case or whether certain traffic characteristics are not properly reflected. For this investigation, we define several metrics to compare the Poisson approximation with the aggregate of periodic IoT traffic. Then, we analyze how large n must be in order to have a sufficiently high accuracy between the Poisson approximation and the aggregated IoT traffic in terms of those metrics.

A more comprehensive treatment can be found in our previous work in [33].

A. Superposition of Traffic Processes

The fundamental theorem for the superposition of traffic processes is the Palm-Khintchine theorem which shows that the superposition of a large number of independent renewal processes will be described by a Poisson process. A point process is a renewal process if and only if the interarrival times are independent and identically distributed (iid). For a Poisson process, the interarrival times X follow an exponential distribution with rate λ with cumulative distribution function $F_X(t)$ and probability density function $f_X(t)$.

$$X \sim \text{Exp}(\lambda) : \; F_X(t) = 1 - e^{-\lambda t} \; , \; f_X(t) = \lambda e^{-\lambda t} \qquad (1)$$

Theorem 1 (Palm-Khintchine Theorem)**.** *Let us consider n independent renewal processes with iid interarrival times X_i. The expected interarrival time for each process is $E[X_i] = 1/\lambda_i$ where λ_i is the arrival intensity. Then the superposition is asymptotically a Poisson process for $n \to \infty$, if the following assumptions hold.*

1) *The intensity λ of the superposition process is finite, $\sum_{j=1}^{n} \lambda_j = \lambda < \infty$ when $n \to \infty$*
2) *No single process dominates the superposition process, $\lambda_i \not\gg \lambda/n = \sum_{j=1}^{n} \lambda_j/n \; ; \forall i$.*

The implication of the theorem for an IoT system is that a superposition of the periodic traffic processes that models the collection of data from a large number n of sources can potentially be approximated by a Poisson process. This is because it is reasonable to assume that the sources generate messages independent of each other and with a sampling frequency in the same order of magnitude, as discussed in Section III. This means that the interarrival time X_i reflects the sampling period of sensor i. The aggregated traffic can be described by the interarrival times, which are the times between the sensor messages as seen by the aggregator e.g. gateway. Please note that the Palm-Khintchine theorem makes no further assumptions about the individual renewal process which may also include periodic processes.

B. Asynchronous Periodic Traffic with same Periodicity

As described in the previous sections in an IoT system the nodes are often periodically generating messages. We formally define such periodic traffic as follows.

Definition 1 (Periodic Traffic)**.** *In a periodic traffic process messages from a single node i are generated at time $\tau_{i,k}$ in period k such that:*

$$\tau_{i,k} = t_i + kT_i \;\; (k \in \mathbb{N}). \qquad (2)$$

The time between messages is constant and equal to a constant T_i, and the arrival rate is also constant $1/T_i$. The first message is sent at time $t_i = \tau_{i,0}$.

The superposition of periodic traffic from IoT nodes is constituted by

1) Synchronous periodic traffic $\tau_{i,k} = \tau_{j,k}$ for any node i and j, and thus identical sampling period $T_i = T, \forall i$,
2) Homogeneous asynchronous periodic traffic with the same sampling period, $T_i = T \; ; \forall i$, but independent start times $t_i \neq t_j$, $\forall i \neq j$,
3) Heterogeneous asynchronous periodic traffic with different periodicity, $\exists T_i \neq T_j$, $\forall i \neq j$

where t_i is the time of the first sample from process i, and T_i the period (inter-message time) of process i.

Synchronous sources can simply be modeled through periodic batch arrivals of size n. The superposition process is then of the same type as the individual processes. However, in a realistic scenario it is reasonable to assume that the IoT traffic sources are asynchronous. In this section we mainly consider homogeneous asynchronous periodic traffic with the same periodicity[5], which is defined as

Definition 2 (Asynchronous Homogeneous Periodic Traffic (APP))**.** *The system consists of n nodes with the same message sampling period, T. In asynchronous mode, the nodes start randomly at time $t_i \sim U(0, T)$. Each node i periodically generates messages at time $\tau_{i,k} = t_i + k \cdot T$ for $k = 0, 1, 2, \ldots$. The interarrival time between the messages of node i and node $i+1$ in $(0; T)$ is denoted $A_i = t_{i+1} - t_i$ $(i = 1, \cdots, n-1)$, with $t_0 = 0$, and $A_n = (T + t_1) - t_n = (T - t_n) + (t_1 - t_0)$ (which is the interarrival time between the first message in a window and the last message in the previous window).*

Note that the message sequence of the APP in $(0; T)$ will be periodically replicated every T.

Figure 2 illustrates a message sequence at sample times t_i, with constant period T (interarrival times). In this paper we consider asynchronous sources $t_i \neq t_j$, $\forall i \neq j$, with the same sampling period, $T_i = T$, $\forall i$, where each source i is sending once (and only once) in the interval $[t_0; t_0 + T]$.

[5]In Section IV-D heterogeneous asynchronous periodic traffic is briefly revisited.

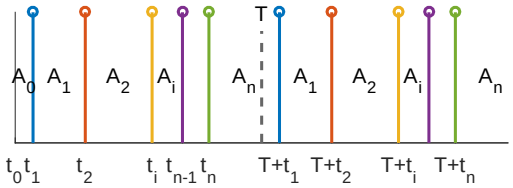

Figure 2. Asynchronous periodic traffic processes with n sources with identical period T, with uniformly distributed start time $t_i \sim U(0,T)$. The interarrival times A_i between the i-th and the $(i+1)$-th $(i = 1,\ldots,n)$ message are identical in each period and $t_i + kT$ for $k \in \mathbb{N}_0$.

C. Performance Metrics

In this section we will compare the performance of the APP and the Poisson Process (PP) approximation. The purpose is to investigate when a Poisson Process can approximate an APP, and when not. In the following we define and look at different metrics that express the difference (in relative error) of the arrival times, interarrival distribution, correlations. An IoT aggregator sees the aggregate traffic and may be implemented in such a way that certain thresholds of interarrival times or timeout values are utilized to dynamically adapt the aggregator, e.g. switching to stand-by mode to save energy after a certain idle time. Hence, the interarrival time patterns are crucial (Sec. IV-C1), as they may trigger thresholds and timeouts. This could lead to differences for APP and PP. We quantify this by the relative error of the interarrival times (Sec. IV-C2) as well as the shift of expected arrival times (Sec. IV-C5).

The variance and distribution of the interarrival times are relevant when for example considering the processing of the sensor node messages (see the IoT load balancer in Section V). In such a case, our aggregator reflects a queuing system, which are well-known to be sensitive to variances of the arrival process, e.g. regarding the waiting and response times of the messages. To this end, we compare the variances of APP and PP in terms of the coefficient of variation (Sec. IV-C3). We compare the interarrival distributions by utilizing the Kolmogorov-Smirnov statistic, which quantifies the maximum difference of the interarrival time CDF values of the APP and the PP (Sec. IV-C4). Of course, when scaling the resources of the aggregator (e.g. scaling cloud resources for processing IoT data as in Section VI), a relevant measure is the current load in the system. To this end, we consider the number of arrivals within a certain period and quantify the differences between APP and PP based on the variances of message arrivals (Sec. IV-C6).

Finally, we take a closer look at the autocorrelations of the interarrival times (Sec. IV-C7) which may influence queuing systems as well. In particular, we are interested in the autocorrelation of the interarrival times of the n-th lag. For the APP, the n-th lag indicates the messages from the same sensor node, i.e. the n-th lag corresponds to the fixed sending period T and the autocorrelation of lag n is 1. However, for the PP, we will observe a random value of the interarrival times due to the Poisson process approximation, hence, the autocorrelation at lag n is < 1. Such differences may be crucial when the IoT system triggers actions for individual sensors based on the interarrival time of messages, e.g. to dynamically adjust the sampling period for reducing the overall load at an aggregator, to reduce the energy consumption of the sending node, or to improve the accuracy of sensor information. As we will see in this section, depending on the concrete metric (and hence the concrete use case which justifies the consideration of the related metric), there may be strong differences between measures for which size n the Poisson process approximation gives acceptable results for an APP.

The Palm-Khintchine theorem only holds when n is sufficiently large and with independent and identically distributed (iid) interarrival times for each node. This raises two questions. *When is* n sufficiently large *such that the superposed process can assumed to be a Poisson process? How large of an error does this assumption introduce, and which traffic characteristics are affected by it?* As discussed in Sec.III the expected scale of IoT applications spans a wide range, the theorem must be carefully investigated before it can be applied to a given scenario. As mentioned above, the queuing performance depends on the autocorrelation and variance of the arrival process so if the PP approximation has very different characteristic w.r.t. these properties then the assessment of the queuing performance will be wrong.

The following sections explore this notion on the basis of several measures and using the following definition of the Poisson process approximation.

Definition 3 (Approximating Poisson Process (PP))**.** *The Poisson process approximating of an APP has arrival rate* $\lambda^* = \frac{n}{T}$*, where* T *is the sampling period and* n *is the number of asynchronous nodes of the corresponding APP. The interarrival time distribution in the Poisson approximated traffic process is exponentially distributed with intensity* λ^**, that is* $A^* \sim \mathrm{Exp}(\lambda^*)$.

1) Interarrival Time (IAT) — Expected value: By definition, the arrival intensities of the APP and the PP are identical. The expected interarrival time of the PP is $\mathrm{E}[A^*] = 1/\lambda^* = T/n$. However, for the APP we observe different values of the expected interarrival times $\mathrm{E}[A_i]$ for $i = 1,\ldots,n$.

Clearly the average of these expected interarrival times over all n must be equal to the expected interarrival time of the PP, since the rates of the two processes are set to be equal. Nevertheless, in order to gain a better understanding of the relationship of these processes we prove this equality in the following by explicitly considering the probabilistic behaviors of the (offset) random variables A_i.

We consider the *first order statistic* X of the uniform distribution. The t_i are iid and uniformly distributed in $(0;T)$, i.e. $t_i \sim U(0,T)$ with Cumulative Distribution Function (CDF) $F_{t_i}(t) = P(t_i \leq t) = t/T$ for $0 \leq t \leq T$. Let $X = \min\{t_i\}$ be a random variable (RV) that describes the minimum of the t_i. Then, the CDF of X is $F_X(t) = 1 - P(t_1 > t,\ldots,t_n > t) = 1 - \prod_{i=1}^{n} 1 - F(t) = 1 - (1 - t/T)^n$. Thus, the first order statistic X follows a four parameter Beta distribution with $a = 1, b = n, c = 0, d = T$, i.e. $X \sim \mathrm{Beta}(a,b,c,d)$ in the interval $[c;d]$. The Beta distribution converges to an exponential distribution as the number of nodes n increases [34].

From Definition 2, we know that the interarrival time between the messages of source i and source $i+1$ is $A_i = t_{i+1} - t_i$, $i = 1, \cdots, n-1$, with $t_0 = 0$, and that $A_n = (T+t_n) - (t_1 - t_0)$. A proof in [35, pp. 122–123] shows that all $A_i, i = 0, \ldots, n-1$ follow the Beta distribution X. It is $\mathrm{E}[A_i] = T/(n+1)$. For $A_n = (T+t_n) - (t_1 - t_0)$, the interarrival time is the sum of two interarrivals, $A_n = X + X$, i.e. a sum of two Beta distributions, and we observe $\mathrm{E}[A_n] = \mathrm{E}[T_n] + \mathrm{E}[A_0] = 2T/(n+1)$.

An intuitive explanation for the expected interarrival times is as follows. The interval $(0;T)$ is divided equally by the n arrivals, and we thus observe $n+1$ interval segments of length $\frac{T}{n+1}$. Hence, $E[A_i] = \frac{T}{n+1}$ for $(i = 0, \cdots, n-1)$ and $\mathrm{E}[A_n] = \mathrm{E}[T_n] + \mathrm{E}[A_0] = 2T/(n+1)$.

2) Interarrival Time (IAT) — Error between average expected IATs: First, we consider the average $\bar{A}$ of the expected interarrival times $\mathrm{E}[A_i]$ of APP and $\mathrm{E}[A^*]$ of the Poisson process. For this we calculate the average of the expected interarrival times of APP over the n nodes

$$\bar{A} = \frac{1}{n}\sum_{i=1}^{n} \mathrm{E}[A_i] = \frac{1}{n}\left(\frac{(n-1)T}{n+1} + \frac{2T}{n+1}\right) = \frac{T}{n} \tag{3}$$

The expected interarrival time of the Poisson process is also $\mathrm{E}[A^*] = T/n$. Hence, there is no difference between the average expected interarrival times in the APP and Poisson process. The error is zero.

3) Interarrival Time (IAT) — Coefficient of variation: The coefficient of variation (CoV) of the IAT of the aggregate arrivals in the APP is $C_i = \mathrm{Std}[A_i]\,/\mathrm{E}[A_i]$, which gives an average CoV of

$$\bar{C} = \frac{1}{n}\sum_{i=1}^{n} C_i \tag{4}$$

The average coefficient of variation over all n nodes of APP can be numerically derived and fitted. The numerical derivation was conducted in our previous work [33] and led to

$$\bar{C} = \frac{n-1}{n} = 1 - \frac{1}{n} \tag{5}$$

For the Poisson process, the interarrival times follow an exponential distribution with a CoV of $C_{A^*} = 1$. The relative error between the average coefficient of variation $\bar{C}$ of interarrival times A_i of APP and PP is

$$r_C = 1 - \bar{C}/C_{A^*} = \frac{1}{n} \tag{6}$$

Thus, for $r_C < \epsilon$, then follows $n > 1/\epsilon$, which implies that the relative error of the coefficient of variation is smaller than ϵ, e.g. an $\epsilon = 1\%$ requires $n > 100$.

4) Interarrival Time (IAT) — Kolmogorov-Smirnov statistic: The average Kolmogorov-Smirnov (KS) statistic $\bar{\kappa}_n$ between distributions of interarrival times in the APP and Poisson process should approach zero as n increases. We use κ_i to quantify the distance between interarrival distributions $F_{A_i}(t) = P(A_i \le t)$ and $F_{A^*}(t) = P(A^* \le t)$, such that

$$\bar{\kappa}_n = \frac{1}{n}\sum_{i=1}^{n} \kappa_i < \epsilon \text{ with} \tag{7}$$
$$\kappa_i = \max_t |F_{A_i}(t) - F_{A^*}(t)|$$

The mathematical analysis has been omitted here for brevity, but instead we derive the values numerically. For $n \to \infty$ the maximum value (emerging from the n-th interarrival) converges towards

$$\lim_{n\to\infty} \kappa_n = \frac{1}{\mathrm{e}} \approx 0.3679\,, \tag{8}$$

while the average and median converge towards zero.

$$\lim_{n\to\infty} \bar{\kappa}_n = 0 \tag{9}$$

Figure 3 plots the mean, median, and maximum KS statistic as a function of n. Thereby, the mean, median, and maximum KS statistic are computed over the n KS values κ_i. As an example, this gives $n \ge 136$ and $\bar{\kappa} < \epsilon$ for $\epsilon = 0.01$.

5) Arrival process — Shift in expected arrival time: When considering the interarrival times, using a Poisson process as an approximation of the APP will introduce a shift in the difference between the expected arrivals of the two processes. As illustrated in Figure 4 we consider the *expected arrivals* in a Poisson process which implies that we have equidistant

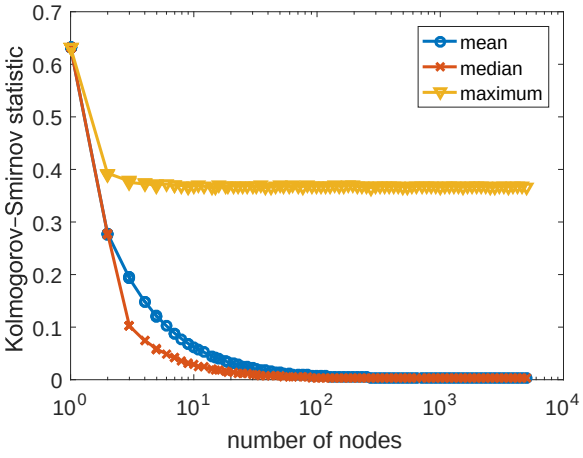

Figure 3. For all n nodes, the mean, median, and maximum Kolmogorov-Smirnov statistics are derived (over all κ_i).

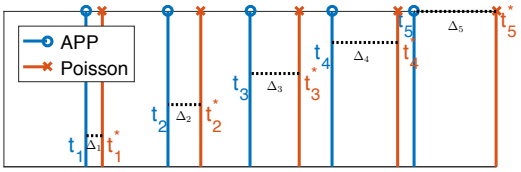

Figure 4. Expected arrivals $t_i, i = 1, \ldots, n$ of aggregated periodic process (blue o) and expected arrivals t_i^* of a Poisson process (red x) with $\lambda = n/T$ in $[0;T]$. The shift in expected arrivals between APP and Poisson process is $\Delta_i = t_i - t_i^*$

arrivals with expected interarrival of $1/\lambda^*$. The expected shift Δ_i of the i-th arrival is then defined as

$$\Delta_i = i \cdot (\frac{T}{n} - \frac{T}{n+1}) \tag{10}$$

The expected shift over the measurement period $(0;T)$ is then

$$\mathrm{E}[S] = \sum_{i=1}^{n} \Delta_i = \frac{T}{2} \tag{11}$$

and the expected shift per node in $(0;T)$ is $E[S]/n = \frac{T}{2n}$. For large n the average shift becomes small, e.g., with $n > 50$ nodes, the error is smaller than $\epsilon = 1\%$ for $T = 1$.

6) Arrival process — Deviation from Expected Number: The APP generates a *fixed* number of n arrivals in $(0;T)$, while for the Poisson process the *expected* number of arrivals in $(0;T)$ is $E[N^*] = \lambda^* T = n$. The Poisson distribution yields the probability that exactly n arrivals will occur in $(0;T)$.

$$P_{N^*}(n) = \frac{(\lambda^* T)^n}{n!} e^{-\lambda^* T} = \frac{n^n}{n!} e^{-n} \tag{12}$$

This probability is *decreasing* with increasing n, from the maximum at $n = 1$, $P_T(1) = 0.3679$.

When considering the coefficient of variation of the Poisson distribution, it is

$$C_{N^*} = \frac{1}{\sqrt{n}} < \epsilon \tag{13}$$

which should be close to zero, as for the APP the CoV of the number of arrivals is 0. For $n > \epsilon^{-2}$, the error is smaller than ϵ.

7) Arrival process — Autocorrelation: The autocorrelation $\rho^*(\tau)$ in a Poisson process is 0 for any lag $\tau > 0$, since the interarrival times are iid. However, in an APP the autocorrelation is $\rho^*(\tau) = 1$ for $\tau = kn$, due to the deterministic periodic pattern of arrivals. Thus, a Poisson approximation is not able to properly capture any autocorrelation characteristics of the APP see also Fig. 8. For many scenarios, this may not be relevant. However, when, for example, considering the waiting times of a queue where the offered traffic is an APP, then the nodes will always observe the same waiting times in every period. Section V-C looks at the waiting times for a single server queuing system in such a scenario.

D. Heterogeneous Traffic Mixes

Besides homogeneous, single-period traffic, there will also be heterogeneous traffic from sources operating on different sending periods. For this, we may consider k APP classes with sending frequencies T_j. In addition, as seen in Table IV, many scenarios also exhibit a mix between periodically sending sources as well as event-based transmissions, which could be represented as a mix of APPs with additional Poisson traffic. To model this kind of rare events, we use a factor $0 \leq \alpha \leq 1$ for the share of Poisson traffic. Since the total load in the system is λ the rate of Poisson traffic is $\alpha\lambda$. That leaves a ratio of $(1-\alpha)\lambda$ for the periodic traffic portion.

Taking a look at the interarrival time in a numerical simulation of this setup a rather close fit can already be observed for $\alpha = 10\,\%$ and $n = 20$ when compared to pure Poisson traffic

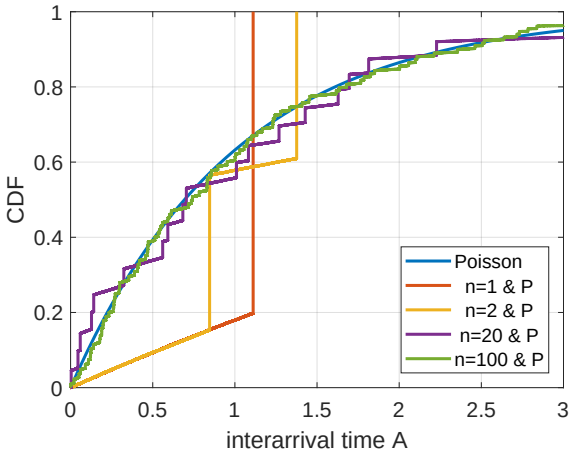

Figure 5. CDF of the interarrival time in a single simulation run of a mixed traffic scenario with 10 % Possionian traffic and a varying number of periodic nodes n.

Table I
RELATIVE ERROR OF THE COEFFICIENT OF VARIATION OF IAT WITH INCREASING NUMBER OF NODES n AND AN INCREASING POISSON TRAFFIC PORTION α.

n	α 0	0.1	0.5
10^0	1	0.7332	0.3127
10^1	0.2139	0.1803	0.0781
10^2	0.0763	0.0624	0.023
10^3	0.0246	0.0199	0.0076
10^4	0.0081	0.0068	0.0035

(cf. also Fig. 5). For small n, the aggregated mixed traffic has a bounded maximum interarrival time (which is at most T). Better approximations for the expected maximum interarrival time exist, but are out of scope for this paper. Investigations of the relative error of the coefficient of variation between mixed and Poisson traffic shed further light on the heterogeneous scenario. The results in Tab. I indicate that an increasing portion of Poisson-modeled event-based traffic quickens the convergence towards Poissonian behavior in the mixed APP case. Since this is not unexpected, and only improves the fidelity of the approximation, we continue the examination with the homogeneous assumption that can serve as a lower limit. Since in praxis, traffic mixes can change or are not entirely predictable, it can be advisable to work under this worst case assumption in any case.

E. Clock Drifts

Up until now, this section has assumed that each source has a constant sending period over the whole duration of the experiment. But in reality this may not always be the case, especially with low-cost IoT devices that do not have a high quality crystal oscillator or even a PLL on their PCB. This causes deviations in the frequency generation and thus also in the clock source. E.g. a typical ceramic resonator as used as the primary clock source on an Arduino Uno board has a frequency tolerance of 0.5 %. Aggregated drifting periodic

Table II
GUIDELINES FOR THE MINIMUM NUMBER n OF NODES SUCH THAT THE RELATIVE ERROR DUE TO POISSON APPROXIMATION IS BELOW A THRESHOLD ϵ. WE CONSIDER $T = 1$, SUCH THAT n DEPICTS THE NUMBER OF MESSAGES PER SECONDS $\lambda = n/T$. 'COV' ABBREVIATES THE COEFFICIENT OF VARIATION OF A RANDOM VARIABLE.

Measure	Description	Formula	$\epsilon = 0.1$
Bias of Poisson process to approximate APP arrival pattern			
r_A	mean interarrival time (IAT)	$r_A = 0$	any $n > 0$
$\bar{S}$	avg. shift of IAT	$n > T/2\epsilon$	$n/T > 50$
r_c	CoV of IAT Eq.(5)	$n > 1/\epsilon$	$n > 100$
$\bar{\kappa}$	KS statistic of IAT	numerically	$n > 136$
c_{N^*}	CoV of arrivals in T	$n > 1/\epsilon^2$	$n > 10000$
Example: Waiting times at IoT load balancer			
$r_W(\rho)$	rel. error waiting time	numerically	depends on ρ
		for $\rho = 0.95$	$n > 38{,}899$
		for $\rho = 0.55$	$n > 486$
		for $\rho = 0.15$	$n > 110$
–	autocorrelation waiting times	not possible	not possible

systems will therefore not have identical interarrival times in each period, as assumed in Fig. 2 but drift relative to each arrival over the course of a number of periods. In the simplest case, the drift can be assumed constant for each source, meaning that each source has a different but constant period. This effectively results in the case of periodic heterogeneous traffic again. But this model does not cover varying drifts, e.g., due to temperature variations of the frequency source.

F. Guidelines for the Lower Limit of n

With the help of the metrics introduced here one can set up guidelines for a lower limit of nodes n in order to keep the relative error below certain thresholds. This has been conducted in Table II. Depending on the concrete use case or characteristic under consideration, the minimal value of n varies significantly. Let us consider a concrete example of an IoT aggregator which goes to stand-by after a certain idle time L. For that use case, the maximum difference between the interarrival time distributions is considered to be on the safe side. Hence, we need to consider the KS statistic from the guideline table. For $n > 136$, the PP approximates the APP (to be more precise: the entire distribution) with a small bias. For example with $L = T/2$, the probability that the IAT is larger than the threshold L is $P(X > L) = e^{-n/2}$ for the PP. For the APP, the beta distribution (see Section IV-C1) is considered. The difference is already below 0.1% for $n \geq 14$ and the guideline table gives a safe recommendation, since the KS statistic considers n wrt. the maximum difference between the two distributions, not only the difference of the tail probabilities.

While many realistic traffic scenarios (looking back at Table IV) might fulfill this requirement due to their scale, others may not be large enough for the Poisson approximation to apply. And for some metrics, like the autocorrelation of waiting times, the Poisson process is simply not able to capture the characteristics of the APP. This has to be kept in mind when one wants to employ this Poisson approximation as basis of their traffic model for scalability and dimensioning decisions.

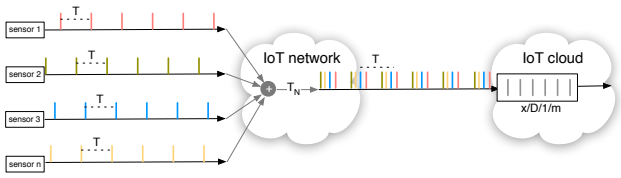

Figure 6. Illustration of the load balancer at an IoT cloud.

V. USE CASE: PERFORMANCE OF IOT CLOUDS

In this section, we take a look a more complex use case where the performance of an IoT cloud load balancer is considered. To this end, we need to examine the load, as we are interested in the waiting times of messages of the IoT load balancer. From the guidelines in Table II, we need to pick the CoV of the number of arrivals during an arbitrary sample period T, as the number of arrivals per time interval determines the load of the IoT load balancer and hence the waiting times. The guideline table tells us that only for large systems with $n > 10{,}000$ nodes the Poisson approximation is appropriate and the CoV C_{N^*} of the number of arrivals is close to zero. However, when studying waiting times of the aggregated traffic in high load scenarios, an even larger number of nodes is required for a small bias, since we need to consider the processing of the message. The waiting system is sensitive to the overall system load and hence the relative error $\epsilon_{C_{N^*}}$ for the CoV should be adjusted to the system load in the queue. We postulate that the higher the load ρ is, the smaller the acceptable bias $\epsilon_{C_{N^*}}$ is for C_{N^*} (i.e. the higher n), such that the relative error in waiting times $r_W(\rho)$ is lower than a threshold, $r_W(\rho) < \epsilon_W$.

$$\epsilon_{C_{N^*}}(\rho) = \epsilon_W(1-\rho) \tag{14}$$

Based on that assumption in Eq. (14), we derive the required number of nodes depending on the system load of the IoT load balancer and postulate that Eq.(15) will lead to a sufficiently small bias between the Poisson process approximation queuing model and the aggregated periodic IoT traffic.

$$n(\rho) > \frac{1}{\epsilon_W^2(1-\rho)^2} \tag{15}$$

We will analyze the performance of the load balancer in the next section and check the postulated number of required nodes for a small bias of the waiting times.

A. Poisson Process Approximations Queuing Models

We now take a look at the concrete case of an IoT cloud, where n nodes are periodically sending messages to a cloud instance to be processed. The nodes are asynchronous, but have the same sending period T. The processing time S to handle the messages at the load balancer is considered to be constant. This system is modeled as an $nD/D/1$ queuing system [5], [36]. We consider $S = 1$ time units and express time-related measures relative to S. The crucial performance

measure for dimensioning this load balancer is the waiting time. We simulate the autocorrelation of the waiting times as well as the impact of additional network transmission delays, which is modeled as an $nG/D/1$ system. The use case demonstrates the limits of the Poisson process approximation, but also shows that for the analysis of scalability it is a very good approximation.

Roberts and Virtamo analyze the state probability for the $nD/D/1$ queue in [5], [37] based on [38]. They compare the system to $M/D/1$ and find that the Poisson approximation can lead to a significant overestimation of buffer requirements, particularly in case of heavy load. A summary of waiting time approximations for high load is for example available in [36]. For the $M/D/1$-approximation, in which the arrivals are generated by a Poisson process with rate $\lambda = \frac{n}{T}$, Iversen and Staalhagen provide an efficient calculation of the $M/D/1$ system state probabilities [39], i.e. the number of customers i in the system. The state probabilities $P(i)$ are recursively computed based on Fry's equation [40]. The load in the system is $\rho = \lambda S$ with constant service time S. We can define $P(i,\rho)$ as $P(i,\rho) = \frac{\rho^i}{i!}\mathrm{e}^{-\rho}$. When comparing the system state of $nD/D/1$ and $M/D/1$ under high load ($\rho = 0.95$), the Poisson approximation overestimates the buffer requirements when dimensioning the load balancer according to a certain threshold $P(X > x)$. However, for fairly low load ($\rho = 0.55$), the difference between $nD/D/1$ and $M/D/1$ is negligible.

B. Mean Waiting Time and Relative Error

For $M/D/1$, the expected waiting time is

$$\mathrm{E}\big[W_{M/D/1}\big] = \frac{\lambda \bar{S}^2}{2(1-\lambda\bar{S})} = \frac{\bar{S}\cdot\rho}{2(1-\rho)}\,. \tag{16}$$

For $nD/D/1$, the expected waiting time is derived based on a result from Eckberg [41] which depends on the Erlang-B formula $B(M,a)$ quantifying the blocking probability in an $M/GI/n/n$ loss system. For the computation of the Erlang-B formula, the iterative method is used, $B(0,a) = 1, B(n,a)^{-1} = 1 + \frac{n}{aB(n-1,a)}$.

$$\mathrm{E}\big[W_{nD/D/1}\big] = \frac{(n-1)\bar{S}\rho}{2nB\left(n-2,\frac{n}{\rho}\right)} \tag{17}$$

$$\lim_{n\to\infty} B\left(n-2,\frac{n}{\lambda\bar{S}}\right) = 1-\lambda\bar{S} \tag{18}$$

Figure 7 shows the mean waiting times in relation to the service time S on the y-axis, while the system load is depicted on the x-axis. For heavy load, there are significant differences. The higher the number of nodes the closer the $nD/D/1$ system approaches $M/D/1$. Under high load ($\rho = 0.95$) the relative error

$$r_W(\rho) = \left|1 - \frac{\mathrm{E}\big[W_{nD/D/1}\big]}{\mathrm{E}\big[W_{M/D/1}\big]}\right| \tag{19}$$

of the expected waiting time is smaller than epsilon for $n > 38{,}900$. An intuitive explanation of the differences between the mean waiting times of $M/D/1$ and $nD/D/1$ is the boundedness of individual waiting times. In $nD/D/1$, the individual waiting time is bounded by $S(n-1)$, which happens if all n arrivals occur simultaneously. Ramamurthy and Sengupta explain it with the busy period [42]: "*In the $M/D/1$ queue both the waiting time and the busy period arej unbounded as the utilization gets close to one* [i.e. heavy load]. *On the other hand, in our* [$nD/D/1$] *model the busy period (and consequently the waiting time) is always upper bounded by one. For this reason, it is not surprising that the $M/D/1$ results overestimate those of our* [$nD/D/1$] *model.*"

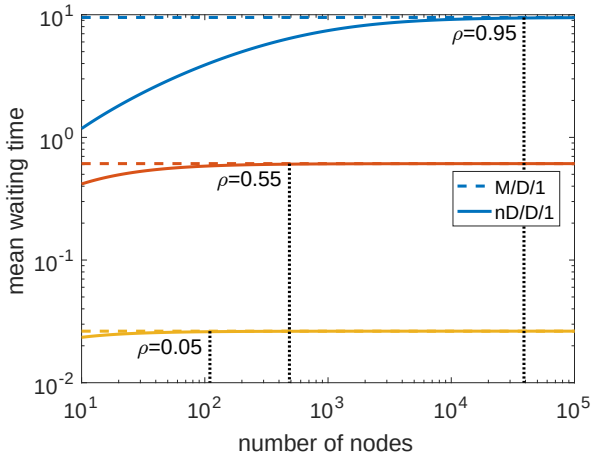

Figure 7. Mean waiting times (normalized by service time S) for $nD/D/1$ and $M/D/1$ reveal significant differences for high load, e.g. $\rho = 0.95$.

However, when investigating the scalability of the IoT load balancer, we observe that the Poisson process approximation fits very well to the $nD/D/1$ system. Especially, in situations where the load is not very high, traffic from just a few hundred sensor nodes already allows using the simple Poisson approximation, and the differences are negligible. In general, basic queuing theory gives us a powerful tool for scalability investigations to easily derive exact performance measures. We will also utilize the Poisson approximation in Section VI for IoT cloud scaling.

Please note that the numerically derived numbers in the guideline Table II fit very well to our basic thoughts in Section IV-F, where we came up with a simple relationship to estimate the number of nodes, see Eq. (15). For the considered system loads ($\rho = 0.15, 0.55, 0.95$), we postulated $n = 138; 494; 40{,}000$ for $\epsilon = 0.1$, which is very close to the numerically derived exact numbers $(110; 486; 38{,}899)$.

C. Autocorrelation of Waiting Times

Periodic systems naturally exhibit deterministic arrival time and the system state pattern. The autocorrelation of the waiting times at lag n is 1, i.e. $w_i = w_{i+kn}$ for $k = 0, 1, 2, \ldots$. Figure 8 compares the autocorrelation of the waiting times for $nD/D/1$ and $M/D/1$ by means of a numerical simulation. In the $M/D/1$ system, the autocorrelation converges towards zero, while the convergence rate depends on the system load ρ. Thus, the characteristics of the autocorrelation of $nD/D/1$ cannot be approximated by a Poisson process.

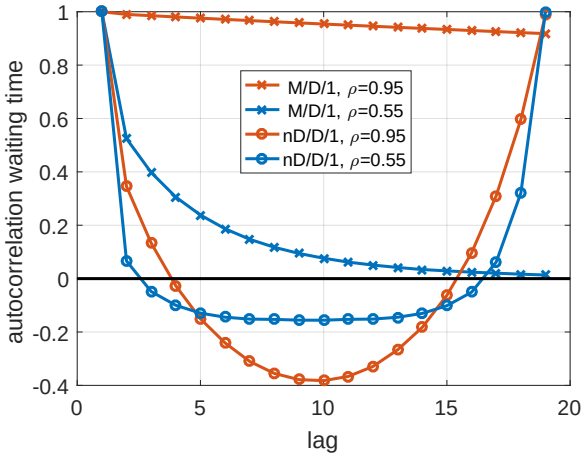

Figure 8. Autocorrelation of waiting times in $nD/D/1$ and $M/D/1$ for medium and high load.

Although the autocorrelation of the waiting time is not crucial for the scalability analysis of an IoT load balancer, as the aggregated load is of importance, the autocorrelation plot and the deterministic pattern show that messages from the same sensor nodes will experience the same waiting times. Hence, such a Poisson approximation may lead to different results when e.g. considering the timeliness of sensor data.

D. Impact of Network Transmission Times (nGI/D/1)

We now refine the $nD/D/1$ IoT cloud model by additionally taking into account a shift in the arrival pattern caused by different and varying transmission (and propagation) times from the sources to the cloud instances. We assume the delay to be an exponentially distributed random variable $\delta \sim \text{Exp}(\mu)$ with mean delay $\delta_m = \frac{1}{\mu}$. Thus, any packet sent at time t arrives at the cloud at time $t+\Delta t$ with $\Delta t \sim \delta$. Consequently, the interarrival times I per node do not follow a deterministic distribution, instead they are the convolution of the network transmission delay, $I = (i+1)T + \delta - (iT + \delta) = T + \delta - \delta$, with

$$F(t) = \begin{cases} \frac{1}{2}e^{\mu(t-T)} & ,t \leq T \\ 1 - \frac{1}{2}e^{-\mu(t-T)} & ,t > T \end{cases} \quad (20)$$

However, additional network delay does not have an impact when n is sufficiently large and the mean waiting time primarily depends on the number of nodes. Therefore, the minimal number of nodes n to keep the error below a threshold is similar. Only in the (unrealistic) case that the additional network delay δ is much larger than the period T then the aggregated process leads to a Poisson process, as the sending period T has no significant influence anymore. Detailed results can be found in [33].

VI. Use Case: Perfomance of IoT Cloud Scalers

The aggregated IoT arrival patterns can also cause load fluctuations in the cloud backend and thus also introduce variations in the required number of cloud instances. Therefore, we investigate how this interacts with cloud auto-scaling. Based on the survey of Lorido-Botran et al. [43], auto-scalers reconfigure the cloud depending on the application and the deployment in the order of minutes. Islam et al. [44], for instance, uses a scaling interval of 12 minutes. Due to the high frequency of the IoT devices, we scale every minute in this use case.

We compare how both the effects of an APP, a Poisson process approximation, and mixed scenarios (similar to the ones in Sec. IV-D) affect the auto-scaling performance. In the following experiments, we first take a look at a representative auto-scaler in the form of *React* [45], since it is a simple and straight-forward approach. Second, we specifically design a threshold-based auto-scaler that targets the mean waiting time.

A. Introducing the Auto-Scalers

In 2009, Chieu et al. [45] presented a reactive scaling algorithm for horizontal scaling, called React. React provisions resources based on a threshold or a certain scaling indicator of a web application. The considered indicators include: the number of concurrent users, the number of active connections, the number of requests per second, and the average response time per request. React gathers these indicators for each resource and calculates the moving average. Afterward, the current web application resources with active sessions which are above or below the given threshold are determined. Then, if all resources have active sessions above the threshold, new web application instances are provisioned. If there are resources with active sessions below the threshold and with at least one resource that has no active session, idle instances are removed. In this work, we used a version modified by Papadopoulos et al. [46] that is available online.[6]

Besides the investigation of React, we design a custom auto-scaler that models each service unit as a queue and takes the mean waiting time into account. The decisions how many instances should be provided are made based on the arrival rate λ, the service rate μ, and the number of instances n. The scaling depends on predefined thresholds of the maximum utilization ρ_max, minimum utilization ρ_min, and the maximum waiting time w_max. Algorithm 1 depicts the pseudo code of the scaling logic of this auto-scaler. In lines 2–4, the current system state is retrieved consisting of arrival rates, service rates, and the number of running instances. Based on this information, the average utilization for each service unit (line 6) and the mean waiting time w (line 7) is calculated. If the utilization exceeds the maximum threshold or the waiting time is higher than the associated threshold, the up-scaling procedure is started: As long as ρ and w exceed their thresholds, the number of supplied instances is theoretically increased (lines 8–10). Otherwise, if the utilization falls below the minimal threshold, the number of provided instances is decreased analogously (lines 13–15). Finally, the new number of supplied instances n is returned in line 16.

B. Quantifying Scaling Behavior

To evaluate the scaling decisions made by the auto-scalers, we consider both user- and system-oriented metrics. For the

[6]Competing auto-scalers: https://github.com/ahmedaley/Autoscalers

Algorithm 1: Pseudo code of scaling logic.

```
Scaling Logic
    λ = getArrivalRate();
    μ = getServiceRate();
    n = getNumInstances();
    ρ = λ/(μ·n); // calculates the average utilization
    w = calcWaitingTime(ρ, μ,...); // calculates E[W]
    if ρ ≥ ρ_max or w > w_max then
        while ρ ≥ ρ_max or w > w_max do
            ρ = λ/(μ·(++n)); // calculates the new average utilization
            w = calcWaitingTime(ρ, μ,...);
        n = min(n, maxInstances());
    else if ρ < ρ_min then
        while ρ < ρ_min do
            ρ = λ/(μ·(--n)); // calculates the new average utilization
        n = max(n, minInstances());
    return n;
```

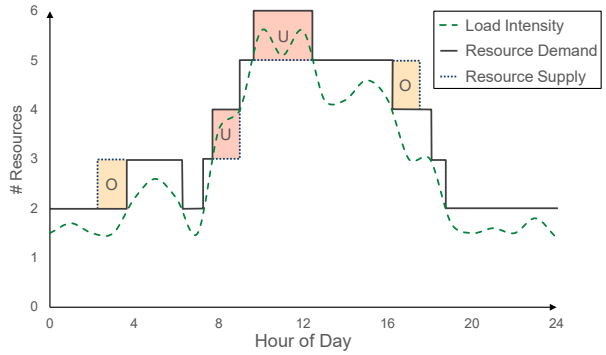

Figure 9. The core idea of elasticity is depicted with **o**ver-provisioning (yellow boxes) and **u**nder-provisioning (red boxes) .

user-oriented metrics, we use the mean waiting time. In literature, there are many approaches on how to measure the auto-scaling quality at the system level. Some of the approaches are intuitive while others seem to be arbitrary. In this work, we want an intuitive comparison that can be precisely described using mathematical formulas. Thus, we consider for the system-oriented metrics the elasticity, which is commonly considered as a central characteristic of the cloud paradigm [47]. Herbst et al. [48] introduce metrics endorsed by the Research Group of SPEC[7] and define the elasticity as follows:

> "Elasticity *is the degree to which a system is able to adapt to workload changes by provisioning and de-provisioning resources in an autonomic manner, such that at each point in time the available resources* match *the current demand as closely as possible.*" [48]

Figure 9 shows the core idea behind the elasticity. The green dashed curve is the load intensity, the solid black curve is the minimal resource demand to handle the load, and the dotted blue curve shows the supplied resources. The red areas labeled with an U represent the under-provisioning, i.e., the demand is higher than the supply. Analogously, the yellow areas labeled with O represent over-provisioning. The width of each area is used for the time share and the surface of each area is used for the accuracy. In this work, we focus on metrics describing the system during under-provisioning: The *under-provisioning time share* τ_U captures the time relative to the measurement duration in which the system has less instances than required. The *under-provisioning accuracy* θ_U represents the relative amount of resources that are under-provisioned during the measurement interval. The best value of $0\,\%$ is achieved when the system is not under-provisioned. We define both metrics as

$$\tau_U[\%] := \frac{100}{T} \cdot \sum_{t=1}^{T} \max(sgn(d_t - s_t), 0)\Delta t$$

$$\theta_U[\%] := \frac{100}{T} \cdot \sum_{t=1}^{T} \frac{\max(d_t - s_t, 0)}{d_t}\Delta t \ ,$$

where d_t is the minimal amount of resources (see black solid curve in Figure 9) required under the load intensity at time t, s_t is the resource supply (see dotted blue curve in Figure 9) at time t, and T is the experiment duration. Δt denotes the time between two scaling intervals, i.e., in this case $1\,\mathrm{min}$.

C. Experiment Discussion

In the first experiment, we compare the scaling with React in three different scenarios. Each scenario represents the arrivals of smart meter readings per gateway with the same sampling period (see the highlighted section of Table IV) during a whole day. We investigate the maximum distance of the waiting time cumulative distributions (in the form of the KS-statistic κ) between the two arrival types (deterministic and Markovian), the average waiting time (WT), the under-provisioning time share τ_U, the under-provisioning accuracy θ_U. The results are listed in Table III. In accordance to our guidelines in Table II, we set the threshold for κ, i.e. the relative error, to 0.1.

In the context of the approximation of Markovian with deterministic arrivals under consideration of a tolerating relative error of 0.1, we see that 10,496 ($\kappa = 0.02 < 0.1$) sending devices are enough to sufficiently approximate Markovian arrivals. In contrast, with 2,422 ($\kappa = 0.11 > 0.1$) sending devices the relative error exceeds the threshold. In the context of auto-scaling, React achieves better values for τ_U and θ_U in all scenarios with the deterministic arrival pattern than for the Markovian ones. This result is also reflected by the average waiting time as in the deterministic case the time is lower than in the Markovian case. That is, the auto-scaler can better handle deterministic load than Markovian load.

Next, we also compare the scaling of React under heterogeneous arrival periods and additional exponentially distributed arrivals (at a portion of $10\,\%$ and $50\,\%$). We focus in this experiment on τ_U and θ_U that are listed in Table III. In each scenario, the auto-scaler begins to struggle with the increasing number of random events and achieves worse performance.

In a final experiment, we compare React to the performance of a custom auto-scaler that scales based on the mean waiting

[7]Standard Performance Evaluation Corporation (SPEC)

Table III
SCALING WITH DETERMINISTIC VS. MARKOVIAN ARRIVAL PATTERN. THE FIDELTY OF THE SCALING IS MEASURED BY THE KS-STATISTIC κ, THE AVERAGE WAITING TIME WT, THE UNDER-PROVISIONING TIME SHARE τ_U AND THE UNDER-PROVISIONING ACCURACY θ_U.

#Devices	Arrival Type	κ	WT	τ_U	θ_U
Homogeneous Scenarios					
2,422	Deterministic	0.11	0.56 s	5.00 %	2.22 %
	Markovian		0.81 s	11.66 %	5.75 %
10,469	Deterministic	0.02	0.14 s	26.67 %	2.58 %
	Markovian		0.15 s	33.81 %	2.65 %
16,098	Deterministic	0.05	0.11 s	20.00 %	1.16 %
	Markovian		0.13 s	35.83 %	2.28 %
#Devices	**Arrival Type**	—	—	τ_U	θ_U
Heterogeneous Scenarios mixed with Poisson Traffic					
2,422	Deterministic			20.00 %	0.99 %
	Deterministic + 10% Markovian			25.00 %	1.42 %
	Deterministic + 50% Markovian			31.67 %	1.59 %
10,469	Deterministic			36.67 %	0.73 %
	Deterministic + 10% Markovian			38.33 %	0.79 %
	Deterministic + 50% Markovian			40.00 %	1.06 %
16,098	Deterministic			26.66 %	0.55 %
	Deterministic + 10% Markovian			36.66 %	0.74 %
	Deterministic + 50% Markovian			43.33 %	0.79 %

time, which is calculated in two variants with the first using Eq. 16 and the second Eq.17. In each scenario, the custom auto-scaler shows for both variants the same values quantifying the under-provisioning. This result is aligned with Table II as each experiment has a sufficient amount of sending devices ($n > 486$). Further, the custom auto-scaler outperforms React in each scenario. While React provides in $5\,\%$ of the experiment time in the first scenario too few instances, the custom auto-scaler reduces this time to $3.33\,\%$. Also θ_U is improved by the custom auto-scaler from $2.22\,\%$ to $1.67\,\%$. For the second scenario, the custom auto-scaler achieves $\tau_U = 6.67\,\%$ and $\theta_U = 0.36\,\%$. In the final scenario, the custom auto-scaler results in $\tau_U = 0.26\,\%$ and $\theta_U = 6.67\,\%$.

D. Threats to Validity

In order to conduct the measurements as realistically as possible, we analyze experiments covering smart meter grids from three German cities with simulated data derived from the toy model in Sec. III-B. Note that the results may not be generalizable to every kind of IoT scenario. As the defined thresholds influence the scaling behavior of React and our custom auto-scaler, we cannot prove that we have chosen the optimal ones. However, React shows a comparable performance as in the related work on auto-scaler evaluation [46]. We address the threat of possible bias by using established sets of metrics that have been officially endorsed by SPEC [48]. In general, some of the result statements rely on the guidelines presented in Table II. That is, if other guidelines or values are used, some of them may be different.

VII. CONCLUSION

Traffic models for IoT applications often reveal periodic traffic patterns from asynchronous sources. This superposition of traffic streams from n nodes can be — and often will be — approximated by a Poisson process, allowing for a simple computation of even large-scale IoT systems. However, the error introduced by the Poisson approximation is often neglected in reality, raising issues of the fidelity of the approximated model, the magnitude of which depends on the statistic under investigation (recall Table II). Depending on the concrete use case or characteristic under consideration, the minimal value of n varies significantly.

Especially, in many practical IoT scenarios — both already existing scenarios today as well as forecasted ones — (see Table IV) the number of nodes is sufficiently large to result in only a small bias. However, the number of nodes also depends on to which hierarchical level of the IoT aggregation system it is applied to. While the cloud backend may usually proof to be large enough, especially when combining data from a whole region or city, the first level of aggregation near the IoT could prove to be too small to accurately apply a Poisson approximation. However, if the arrival process here is not just purely homogeneously periodic and includes a mix of other, e.g. Markovian, components, possibly due to an event-based nature, the approximations might just become valid again.

In summary, this means that before one can apply the Poisson process approximation to any scenario, one has to first both characterize the IoT environment and traffic properties under scrutiny and needs to know what one wants to achieve with this model (i.e. determine the observed metrics). Only if all these conditions have been satisfied can the validity of the approximation be determined — under the additional constraint that the Poisson process is not able to capture the characteristics of the APP for characteristics like the autocorrelation of waiting times at all. However, if those characteristics are not relevant, then the Poisson process might be a good approximation.

Following the data further down the trail towards the cloud infrastructure and looking at the scaling of these systems for IoT scenarios, the auto-scaling experiments confirm that in many realistic IoT scenarios the Poisson approximation can be safely assumed. However, the scaler might also be able to exploit the autocorrelation of an APP to its advantage, and tune itself to the periodicity of the traffic. If the traffic were just approximated with a Poisson process this advantage would have been overlooked.

Examining load balancing and scaling for highly autocorrelated IoT traffic might be quite an interesting venue for the future to explore, since with an increasing number of random events the scaling performance usually decreases.

REFERENCES

[1] D. R. Cox and W. L. Smith, "On the superposition of renewal processes," *Biometrika*, vol. 41, no. 1-2, p. 91, 1954.

[2] A. Y. Khinchin *et al.*, *Mathematical methods in the theory of queuing*. London: Griffin, 1960.

[3] C. Palm, "Intensitätsschwankungen im Fernsprechverkehr," *Ericsson Technics no.*, vol. 44, p. 189, 1943.

Table IV
IoT DIMENSIONS AND MESSAGE INTERVALS. ACTUAL AND ASSUMED VALUES COLLECTED FROM LITERATURE AND FROM THE TOY MODEL. RATES ARE PER DEVICE IF NOT STATED OTHERWISE.

Type	**Density**	**Rate/Period**	**Source**
IMT-2020 Requirements	10^6 devices per km^2	$10\,(\mathrm{Mbit/s})/\mathrm{m}^2$ (indoor hotspot)	[49]
LTE / smart meter	(per PRB) 7.5×10^4 (urban), 5.6×10^4 (suburban)	2017 B every 9000 s	[50]
LTE / health sensor	(per PRB) 5.3×10^3 (urban), 4.0×10^3 (suburban)	$\frac{128\,\mathrm{B}}{60\,\mathrm{s}}$	[50]
LTE / home security	(per PRB) 1.2×10^5 (urban), 9.2×10^4 (suburban)	$\frac{20\,\mathrm{B}}{600\,\mathrm{s}}$	[50]
Sensor-based alarm/event detection example scenario	3×10^4	$\frac{1000\,\mathrm{B}}{1800\,\mathrm{s}}$	[15]
Mobitex model	—	Poisson, average 300 calls/mobile/hour, uniform pkt distribution 30 ± 15B up, 115 ± 57B down	[51]
ITU use case (first year, aggregate over 10 gateways, 1.5M packet capacity per day)			
Sensor	200	$1/\mathrm{h}$	[51, p.153]
Metering	100	$0.04/\mathrm{h}$	[51, p.153]
Alarm	100	$0.04/\mathrm{h}$	[51, p.153]
Tracking logistics	100	$2/\mathrm{h}$	[51, p.153]
Vehicle tracking	70	$6/\mathrm{h}$	[51, p.153]
Traffic control	150	$10/\mathrm{h}$	[51, p.153]
Agriculture	200	$1/\mathrm{h}$	[51, p.153]
Wearables	1,000	$0.5/\mathrm{h}$	[51, p.153]
Home automation	300	$0.5/\mathrm{h}$	[51, p.153]
Parking spot sensors	500	$\hat{T}_{MSG} = 10/\mathrm{d}$ (over all 500 sensors)	[52]
Automatic Meter Reading (AMR)			
Rural	100 per km^2	100 B to 200 B payload per message	[18], [53]
Suburban	800 per km^2	100 B to 200 B payload per message	[18], [53]
Urban	2,000 per km^2	100 B to 200 B payload per message	[18], [53]
Power transmission line monitoring (per pole)			
Accelerometers	3	$\frac{11\,520\,\mathrm{B}}{300\,\mathrm{s}}$ (total)	[18]
Magnetic field sensors	3	$\frac{2640\,\mathrm{B}}{1\,\mathrm{s}}$ (total)	[18]
Strain sensor	1	$\frac{1920\,\mathrm{B}}{300\,\mathrm{s}}$	[18]
Temperature	1	$\frac{20\,\mathrm{B}}{5\,\mathrm{s}}$	[18]
Scenario: IoT and smart grids in the city			
Water meters	10,000 per km^2	100 B every 43 200 s	[31]
Electricity meters	10,000 per km^2	100 B every 86 400 s	[31]
Gas meters	10,000 per km^2	100 B every 1800 s	[31]
Vending machines	150 per km^2	150 B every 86 200 s	[31]
Bike fleet management	200 per km^2	150 B every 1800 s	[31]
Pay-as-you-drive	2,250 per km^2	150 B every 600 s	[31]
Industrial smart meter	4,500 per cell	2400 B every 3600 s	[25]
Residential smart meter	4,500 per cell	1200 B every 14 400 s	[25]
Enhanced smart meter (distribution grid)	4,500 per cell	3848 B every 1 s	[25]
3GPP TSG RAN WG2 R2-102340 scenarios			
3GPP MTC traffic model 1	1,000, 3,000, 5,000, 10,000, 30,000 per cell	$\frac{1}{60\,\mathrm{s}}$ uniform	[21]
Central London households / smart meters	4,968 per cell	periods of 300 s, 900 s, 3600 s, 43 200 s, or 86 400 s	[21]
Urban London households / smart meters	35,670 per cell	periods of 300 s, 900 s, 3600 s, 43 200 s, or 86 400 s	[21]
German City Home Smart Meter Forecast (1 Mile Radius LoRa Model, excerpt)			
Berlin	16,098 per cell, 1,786,818 total households	15 min periods, depending on regulations	
Offenbach am Main	10,469 households per cell, 62,809 total households	15 min periods, depending on regulations	
Wolfsburg	2,422 per cell, 62,967 total households	15 min periods, depending on regulations	

[4] E. Cinlar and R. A. Agnew, "On the superposition of point processes," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 30, no. 3, pp. 576–581, 1968.

[5] J. W. Roberts and J. T. Virtamo, "The superposition of periodic cell arrival streams in an ATM multiplexer," *IEEE Transactions on Communications*, vol. 39, no. 2, pp. 298–303, 1991.

[6] M. A. Arfeen *et al.*, "Internet traffic modelling: from superposition to scaling," *IET networks*, vol. 3, no. 1, pp. 30–40, 2014.

[7] A. Al-Fuqaha *et al.*, "Toward better horizontal integration among IoT services," *IEEE Communications Magazine*, vol. 53, no. 9, pp. 72–79, Sep. 2015.

[8] A. Banks and R. Gupta, *MQTT Version 3.1.1*, OASIS Standard, Oct. 2014.

[9] Z. Shelby *et al.*, *The Constrained Application Protocol (CoAP)*, RFC 7252 (Proposed Standard), RFC, Updated by RFC 7959, Fremont, CA, USA: RFC Editor, Jun. 2014.

[10] J. Petajajarvi *et al.*, "On the coverage of LPWANs: range evaluation and channel attenuation model for LoRa technology," in *2015 14th International Conference on ITS Telecommunications (ITST)*, Dec. 2015, pp. 55–59.

[11] J. Navarro-Ortiz *et al.*, "Integration of LoRaWAN and 4G/5G for the Industrial Internet of Things," *IEEE Communications Magazine*, vol. 56, no. 2, pp. 60–67, Feb. 2018.

[12] N. Rupasinghe and İ. Güvenç, "Licensed-assisted access for WiFi-LTE coexistence in the unlicensed spectrum," in *2014 IEEE Globecom Workshops (GC Wkshps)*, Dec. 2014, pp. 894–899.

[13] M. Lauridsen *et al.*, "Interference Measurements in the European 868 MHz ISM Band with Focus on LoRa and SigFox," in *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, Mar. 2017, pp. 1–6.

[14] Y. P. E. Wang *et al.*, "A Primer on 3GPP Narrowband Internet of Things," *IEEE Communications Magazine*, vol. 55, no. 3, pp. 117–123, Mar. 2017.

[15] N. Nikaein *et al.*, "Simple Traffic Modeling Framework for Machine Type Communication," in *ISWCS 2013; The Tenth International Symposium on Wireless Communication Systems*, Aug. 2013, pp. 1–5.

[16] M. Laner *et al.*, "M2M Traffic and Models," in *Machine-to-Machine Communications:Architectures, Technology, Standards, and Applications*, V. B. Misic and J. Misic, Eds. Boca Raton, FL: CRC Press, 2014.

[17] I. Petiz *et al.*, "Characterization and modeling of M2M video surveillance traffic," in *IARIA Int. Conf. on Advances in Future Internet-AFIN*, Citeseer, 2012.

[18] R. H. Khan and J. Y. Khan, "A comprehensive review of the application characteristics and traffic requirements of a smart grid communications network," *Computer Networks*, vol. 57, no. 3, pp. 825–845, 2013.

[19] M. A. Mehaseb *et al.*, "WSN Application Traffic Characterization for Integration within the Internet of Things," in *2013 IEEE 9th International Conference on Mobile Ad-hoc and Sensor Networks*, Dec. 2013, pp. 318–323.

[20] X. Jian *et al.*, "Statistical Description and Analysis of the Concurrent Data Transmission from Massive MTC Devices," *International Journal of Smart Home*, vol. 8, no. 4, pp. 139–150, 2014.

[21] 3GPP, "RAN Improvements for Machine-type Communications," 3rd Generation Partnership Project (3GPP), TR 37.868, Oct. 2011.

[22] R. R. Tyagi *et al.*, "Connection Establishment in LTE-A Networks: Justification of Poisson Process Modeling," *IEEE Systems Journal*, vol. PP, no. 99, pp. 1–12, 2015.

[23] M. Z. Shafiq *et al.*, "A first look at cellular machine-to-machine traffic: Large scale measurement and characterization," in *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems*, ser. SIGMETRICS '12, London, England, UK: ACM, 2012, pp. 65–76.

[24] F. Metzger *et al.*, "Exploratory Analysis of a GGSN's PDP Context Signaling Load," *Journal of Computer Networks and Communications*, Feb. 2014.

[25] J. J. Nielsen *et al.*, "What can wireless cellular technologies do about the upcoming smart metering traffic?" *IEEE Communications Magazine*, vol. 53, no. 9, pp. 41–47, Sep. 2015.

[26] M. Vilgelm and W. Kellerer, "Impact of request aggregation on machine type connection establishment in LTE-Advanced," in *Proc. IEEE WCNC*, 2017.

[27] A. Laya *et al.*, "Massive access in the Random Access Channel of LTE for M2M communications: An energy perspective," in *2015 IEEE International Conference on Communication Workshop (ICCW)*, Jun. 2015, pp. 1452–1457.

[28] L. D. Xu *et al.*, "Internet of Things in Industries: A Survey," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2233–2243, Nov. 2014.

[29] J. Gubbi *et al.*, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[30] L. Atzori *et al.*, "The internet of things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[31] *Massive IoT in the City*, Ericsson White paper, Nov. 2016.

[32] S. Bundesamt, "Statistisches jahrbuch 2017," in. Wiesbaden, 2017, ch. 2 Bevölkerung, Familien, Lebensformen, pp. 23–80.

[33] T. Hoßfeld *et al.*, "Traffic Modeling for Aggregated Periodic IoT Data," in *21st International Conference on Innovation in Clouds, Internet and Networks (ICIN 2018)*, Mar. 2018.

[34] M. Kivelä *et al.*, "Multiscale analysis of spreading in a large communication network," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2012, no. 03, P03005, 2012.

[35] A. M. Mathai, *An introduction to geometrical probability: distributional aspects with applications*. CRC Press, 1999, vol. 1.

[36] M. Menth and S. Muehleck, "Packet waiting time for multiplexed periodic on/off streams in the presence of overbooking," *International Journal of Communication Networks and Distributed Systems*, vol. 4, no. 2, pp. 207–229, 2010.

[37] J. Roberts *et al.*, "Broadband Network Teletraffic: Final Report of Action COST 242," 1996.

[38] A. Eckberg, "The single server queue with periodic arrival process and deterministic service times," *IEEE Transactions on communications*, vol. 27, no. 3, pp. 556–562, 1979.

[39] V. B. Iversen and L. Staalhagen, "Waiting time distribution in M/D/1 queueing systems," *Electronics Letters*, vol. 35, no. 25, pp. 2184–2185, 1999.

[40] T. C. Fry *et al.*, *Probability and its engineering uses*. Van Nostrand New York, 1928.

[41] A. Eckberg Jr and L. Green, "Response time analysis for pipelining jobs in a tree network of processors," in *Applied Probability-Computer Science: The Interface Volume 1*, Springer, 1982, pp. 387–416.

[42] G. Ramamurthy and B. Sengupta, "Delay analysis of a packet voice multiplexer," in *Telecommunications Symposium, 1990. ITS'90 Symposium Record., SBT/IEEE International*, IEEE, 1990, pp. 155–160.

[43] T. Lorido-Botran *et al.*, "A Review of Auto-scaling Techniques for Elastic Applications in Cloud Environments," *Journal of Grid Computing*, vol. 12, no. 4, pp. 559–592, 2014.

[44] S. Islam *et al.*, "Empirical prediction models for adaptive resource provisioning in the cloud," *Future Generation Computer Systems*, vol. 28, no. 1, pp. 155–162, 2012.

[45] T. C. Chieu *et al.*, "Dynamic scaling of web applications in a virtualized cloud computing environment," in *E-Business Engineering, 2009. ICEBE'09. IEEE International Conference on*, IEEE, 2009, pp. 281–286.

[46] A. Papadopoulos and more, "PEAS: A Performance Evaluation Framework for Auto-Scaling Strategies in Cloud Applications," *ACM ToMPECS*, vol. 1, no. 4, pp. 1–31, Aug. 2016.

[47] D. C. Plummer and more, "Study: Five Refining Attributes of Public and Private Cloud Computing," Gartner, Tech. Rep., 2009.

[48] N. Herbst *et al.*, "Quantifying Cloud Performance and Dependability: Taxonomy, Metric Design, and Emerging Challenges," *ACM Transactions on Modeling and Performance Evaluation of Computing Systems (ToMPECS)*, vol. 3, no. 4, 19:1–19:36, Aug. 2018.

[49] *Draft new Report ITU-R M.[IMT-2020.TECH PERF REQ] - Minimum requirements related to technical performance for IMT-2020 radio interface(s)*, International Telecommunication Union Radiocommunication Sector, Feb. 2017.

[50] R. Ratasuk *et al.*, "Recent advancements in M2M communications in 4G networks and evolution towards 5G," in *2015 18th International Conference on Intelligence in Next Generation Networks*, Feb. 2015, pp. 52–57.

[51] S. Tabbane, "IoT Network Planning," in *ITU Asia-Pacific CoE Program on "Developing the ICT ecosystem to harness Internet of Things"*, International Telecommunication Union, ITU, Dec. 2016.

[52] B. Martinez *et al.*, "The Power of Models: Modeling Power Consumption for IoT Devices," *IEEE Sensors Journal*, vol. 15, no. 10, pp. 5777–5789, Oct. 2015.

[53] D. Cypher and N. T. Golmie, "Nist priority action plan 2, guidelines for assessing wireless standards for smart grid applications," Tech. Rep., 2011.