Industry Use Case Viability Study with Performance Models for Satellite Computing Networks

Fabian Poignée*, Frank Loh*, Florian Zeiger †, Markus Sauer †, Tobias Hoßfeld*
* *University of Würzburg, Chair of Communication Networks*, Würzburg, Germany
{fabian.poignee|frank.loh|tobias.hossfeld}@uni-wuerzburg.de
† *Siemens AG, Technology*, Munich, Germany
{markussauer|florian.zeiger}@siemens.com

***Abstract*—Network access via satellites has been identified as a complementary solution to traditional mobile networks. The roll-out of comprehensive 5G networks is often not appropriate in sparsely populated regions. However, satellite networks provide more than a coverage extension for black spots in the coverage map or as backup in disaster scenarios. For the industry, satellite networks gain interest for overload mitigation and to improve the general service quality to their customers for many applications. Consequently, satellite networks aroused research interest in recent years, providing besides network access also the possibility to perform edge computing with integrated processing entities in satellites. This opportunity moves processing capability closer to every global service and opens many new application areas, but also research questions. Therefore, we thoroughly model such a satellite computing network to identify benefits, challenges, and limitations for edge computing in a satellite constellation. We identify the requirement for a migration mechanism to decrease overall system load and we propose a modeling solution to assess system load limitations with our performance model. We derive different application areas and showcase the feasibility and advantages of satellite computing networks. Finally, we verify our modeling results with a comprehensive simulation study for different use cases.**

***Index Terms*—satellite network, satellite computing, performance model, edge computing, QoS**

I. Introduction

The increasing requirement for access to communication networks and computing services in every part of our world is driven by a multitude of novel applications, services, and technologies. Consequently, a large percentage of the global population expects comprehensive and affordable network connection around the clock for communication with family and friends, social media, and multimedia applications. Additionally, business and especially industrial services require constant connectivity and access to remote computing services. Communication and computing services in this context need to be at industrial grade with a reliable and high quality network to function without any errors under all circumstances.

From the perspective of an infrastructure service provider, it is essential to deal with user requirements and offer networks and computing services in a sufficient Quality of Service (QoS) to avoid customer churn. Consequently, much money is invested to foster the roll-out of mobile networks including 5G or widespread Internet of Things (IoT) deployments to fulfill all Service Level Agreements (SLAs). However, a service provider also needs to operate economically and, given the heterogeneity of end devices and systems, network roll-outs in sparsely populated regions lead to a negative return of invest. Furthermore, a reliable and fast alternative to classical mobile network generations is demanded in case of network failures.

For that reason, using comprehensive satellite networks aroused interest in recent years starting with proprietary and specialised services some decades ago. The launch of satellite constellations like Starlink in 2019 [1] introduced a paradigm shift from specific and expensive deployments tailored to requirements for industrial use cases towards satellite networks being available for everyone as a generic service. Since then, the application of satellite networks for global Internet coverage, disaster recovery and overload mitigation on terrestrial networks is heavily researched. In addition, a large percentage of satellites will be equipped with more computational resources usable for comprehensive edge computing tasks [2].

This availability of generic communication services combined with computing capabilities at the edge for reasonable costs drives an increasing number of novel industrial use cases which require communication with high QoS. Furthermore, computing capacity close to the data sources allows operation of data hungry services with reasonable reaction times. Consequently, it is possible to request specific services from earth, process the request on satellites, and answer it across large geographic distances with a little number of hops and, on the first glance, little overhead. However, besides the numerous benefits of such networks that are already discussed in detail, the challenge of a seamless connection to services running on a satellite still exist. These satellites move with high velocity in complex but deterministic mobility patterns. Consequently, a general assessment of the system load on satellites is challenging and has to the best of the authors knowledge not been investigated yet. This also includes a system analysis of a combined design including communication and computing resources in such a satellite computing network.

For that reason, we present a comprehensive investigation of a satellite computing network in this work. By considering both communication and processing demands, we investigate the feasibility of use cases deployed on satellites and identify key challenges and benefits. Therefore, we develop a generic model for a compute satellite where we consider user applica-

ISBN 978-3-903176-63-8 © 2024 IFIP

978-3-903176-63-8/24/$31.00 ©2024 IEEE

tion and its service traffic, and identify the requirement for our proposed migration mechanism. This mechanism is necessary to cope with mobility and the compute infrastructure in a satellite computing network as without it only few use cases are feasible. By a formal description of the satellite's behavior and a comprehensive simulation study, we assess our model and evaluate the expected system load.

To this end, the contribution of our work is three-fold. First, we present a comprehensive queuing model for a satellite capable of performing edge computing in a satellite network. Our model is applicable to any satellite in a constellation and thus, viable to describe the general behavior, limitations and challenges, but also benefits for the complete network. Second, we incorporate application and migration information and showcase the necessity of user service information migration in a satellite computing network. Thus, we can define general equations to describe processing on a satellite and derive simulative studies about the expected load. Finally, we identify application areas and uses cases that can benefit from a satellite computing network based on our modeling result. Therefore, we identify key parameters inducing system load in a satellite computing network. To this end, we identify three research questions (RQs) that are answered throughout this work.

- RQ1: How can we model a generic satellite computing network including communication and computing capabilities for arbitrary use cases?
- RQ2: How can we formally describe the load for a given system in a satellite computing network to examine service stability and to identify bottlenecks?
- RQ3: Which key parameters influence system load in satellite computing networks most and which use cases can be realized in such an environment?

In the remainder of this work, we provide background and related work in Sec. II, our general queuing model for a satellite computing network in Sec. III. Formal load description, the scenarios, and a use case simulation follow in Sec. IV and are evaluated in Sec. V. We conclude in Sec. VI.

II. Background and Related Work

This section presents background and related work to understand this work. We introduce details on satellite access, a satellite network architecture, and known state-of the art constellations before presenting the concept of satellite computing networks and the required mechanisms to achieve it.

Access to Satellites: Typically, satellite access features user equipments (UEs), terrestrial gateways, and satellites [3]. In general, satellites can communicate with three different entities: (1) UEs on earth communicate via gateways or directly to a satellite via a so-called service link. Then, the satellite operates as an intermediary network component within space via the bent-pipe principle as a relay node [3] with or without altering the received signal. (2) Using a feeder link, the satellite then communicates with a gateway on earth to forward the message received from the UE. (3) Finally, there are inter-satellite links (ISLs) for communication and forwarding between satellites. The inter-satellite communication can be

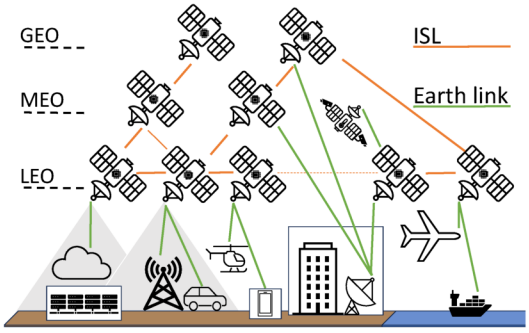


Fig. 1: Exemplary satellite computing network architecture.

required, for example, to use other satellites as relay to gateways on earth across large geographic distances. While such satellite networks are usually closed systems and have only little interoperability to other systems, research in 5G and towards 6G networks (e.g., [4]) considers the direct integration of satellite communication to the network core for better interoperability between systems. This can be achieved by equipping the gateway on earth with base station functionality. Additionally, there are standardization efforts in 3GPP for NTN [5].

To understand capabilities of state-of-the-art satellite access, Yahia et al. [6] provide a comprehensive survey. Prominent LEO satellite examples such as OneWeb and Starlink offer data rates of up to around $20\,\mathrm{Gbit/s}$. In addition, more countries and companies enter this market segment including Amazon [6]. While Starlink plans to deploy tens of thousands of satellites, their system already contains 4,408 satellites in its first phase [7]. Such a high number of satellites is essential in the low orbit since these satellites have a small field of view. This, however, leads to the requirement of more frequent handovers with the ground equipment as the time a satellite is visible from a fixed point on the ground is short. For example LEO satellites have orbital periods of about $90\,\mathrm{min}$, depending on the exact orbit and altitude, and a maximal visibility duration between $5\,\mathrm{min}$ to $20\,\mathrm{min}$. Then, a handover must occur. Considering the Starlink satellites example, a handover is triggered after on average around $114\,\mathrm{s}$ [7], depending on the geolocation of the receiver on earth.

Satellite Network Architecture: Besides simple access to satellites, novel architectural concepts for satellite network architectures assess the interoperability between multiple satellite hierarchies and a multitude of different network entities. Such a hierarchical constellation of satellites including multiple layers is currently subject of research for future deployments [8] and visualized in Figure 1. Such architectures can be enabled through increasing processing capabilities on satellites. Multiple layers are differentiated based on their altitude, in this example, low-earth orbit (LEO), medium earth orbit (MEO), and geostationary orbit (GEO) satellites [9]. In addition, different satellites within the same orbit keep a logical hierarchy [10] and are connected via ISLs. The ISLs in Figure 1 indicate several possibilities as a simplified and

abstract example. Other or different links are also feasible if a direct line of sight between two satellites is available.

On the lowest level or ground level the terrestrial network is deployed with terrestrial devices. These devices can include a terrestrial cloud center, base stations, or different client types including automotive deployments, aircraft, mobile devices, ground stations, or watercraft. The connection between the terrestrial devices and the satellites on different orbits traditionally requires specialized hardware. Consequently, it is established via ground stations or satellite terminals acting as gateways. However, since nowadays even smartphones can directly communicate with LEO satellites [11], a general transition from ground station centric architectures to more individual device and service architectures can be observed.

Satellite Computing Network: To now enable compute capabilities in a satellite network, computational resources are required on satellites to perform edge computing. Therefore, constellations need to be equipped with sufficient resources for processing tasks [12]. Consequently, such satellite constellations form a distributed and mobile edge computing platform named satellite computing network in the following, usable for private or industrial tasks. Note that the proximity for the edge term here is derived from the number of hops between the satellite edge and their clients and not necessarily from the distance to the satellite, similarly to the literature [13], [14].

In this context, in particular LEO satellites are promising as they require only a small sized equipment and have lower power requirements and costs for a specific coverage. Finally, smaller propagation delays due to their low altitude are essential for many use-cases as the Round-Trip-Time (RTT) to a GEO satellite can range between $600\,\mathrm{ms}$ and $700\,\mathrm{ms}$ [7]. However, MEO or GEO satellites can act as a backbone network layer, offering additional ISLs and provide compute capabilities for ground devices with more delay tolerance.

Challenges, Requirements, and Benefits: In general, task offloading and resource allocation are fundamental requirements in terrestrial edge computing. Besides these fundamentals, additional constraints arise to enable edge computing in satellite networks since satellites may be spatially distributed across the globe. Thus, mobility support must be guaranteed and inter-satellite routing opportunities need to be available. Consequently, existing solutions used in terrestrial edge computing may reach or exceed their limits and must be reassessed for satellite computing networks. Consequently, there are several differences between terrestrial edge computing and satellite computing scenarios, summarized in [15]. The network topology changes dynamically in a satellite computing network, but is mostly predictable due to the satellites' orbits. Though, since the satellites are moving with high relative velocity compared to the users and the network is constantly changing, mobility is a larger concern and routing, dependent on the network architecture, is a highly dynamic task. Furthermore, since task offloading is usually considered with short distances and low dynamics in terrestrial networks, long distances and high dynamics are expected when using satellites. In addition, all satellites have specific network limitations and energy consumption plays an important role. However, a comprehensive satellite computing network can then offer edge capabilities to all clients and their devices and various edge use cases anywhere on earth using non-specialized computing hardware. In addition, the satellite computing network architecture can not only offer edge capabilities in regions on earth where terrestrial networks and compute infrastructure are expensive to deploy but also act as a backup infrastructure during a disaster recovery scenario. Additionally, business and especially industrial services require constant connectivity and access to remote computing services. Since such industrial services and systems in, among others, the manufacturing, mobility, energy, and smart infrastructure domain, drive our everyday life and protect our society and the way we live, even higher quality standards are required and can be fulfilled. In addition, the heterogeneity of end devices and systems in an industrial Internet of Things (IoT) domain requires attention. End devices can range from small low-power IoT sensors and respective controllers to complex mobile or stationary machinery. Furthermore, an increasing number of satellites generate or sense data including environmental monitoring, large scale Industrial IoT, and satellite images. Processing on the satellite edge can be beneficial for these use cases by transmitting meaningful and compressed data to the ground [16].

Approaches from the Literature: Since comprehensive measurements on satellites are complex and costly, different authors already conducted simulation studies to evaluate satellite computing networks and, in particular, tackle existing limitations and challenges. For example SatEdgeSim [15] is a simulation toolkit for satellite edge computing environments to study resource configuration and task deployment strategies. In contrast to the SatEdgeSim approach, we investigate service feasibility on a satellite computing network for non-terrestrial edge services based on given service requirements in this work. Therefore, we focus on a data-driven publish/subscribe paradigm implemented for a joint sensing, compute, and networking architecture on the concrete example of MQTT. With regard to routing dynamics in satellite networks, Liu et al. propose a routing algorithm design based on Information Centric Networking or Software Defined Networks [17]. In this work, we present an additional mechanism, including service information migration, required to run arbitrary services on compute satellites with close proximity to the client.

III. Queuing Model for a Compute Satellite

Typically, a satellite computing network consists of several satellites with computational resources named compute satellites hereafter. To this end, we describe a queuing model for a single compute satellite in such a satellite computing network in this work. We start with a general model description and outline details on the input and output to the compute entity within a satellite represented as a CPU in Figure 2.

A. Migration Mechanism

The focus of our model is a LEO constellation because of the lower distance to earth. This enables usability for

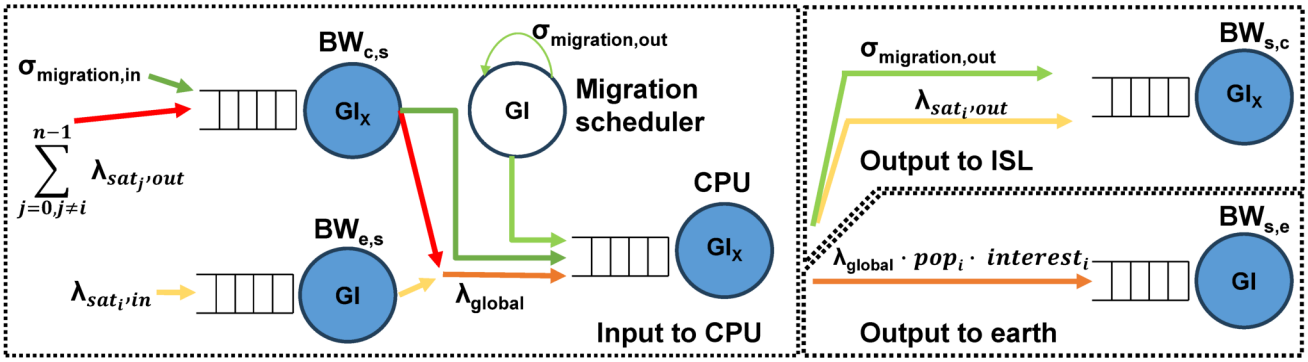


Fig. 2: Overview of queues and resource units at a compute satellite.

various application areas without significant end-to-end delay. But to run arbitrary services on such a satellite computing network, and guarantee a seamless service to all end points on earth, named clients in the following, many satellites for comprehensive coverage are essential. In a LEO constellation, this leads to frequent handovers between satellites. Consequently, from the perspective of a single compute $satellite_i$, its client population is highly dynamic and migration of service information between satellites is constantly required.

The necessity for migration could stem, for example, from technical or business requirements that forbid client information to leave a certain geographical location. It is assumed that each service has specific migration information for each client, e.g., a representation of user interests or client specific service configuration information, that is necessary to offer service functionality to a particular client. Such information could include specific service configuration like access to a certain function subset or in the case of a communication service, information about the topics of interest of a client. This information must be present at the current $satellite_i$ in order to serve the client and before a handover to $satellite_{i+1}$ is required, it must be migrated to the next satellite $satellite_{i+1}$. This migration acts as a client information state transition between $satellite_i$ and $satellite_{i+1}$. This can enable a continuous and seamless service for all end users since the availability of client's migration information is crucial for service continuation. Consequently, traffic prioritization is advisable, but for simplicity reasons omitted in the following.

B. General Queuing Model Definition

As our model describes the behavior of a single satellite, we assume that all inter-satellite traffic is handled by one abstract entity in the network. We name this entity satellite core in the following to represent the interaction of the modeled satellite with all the other satellites in the satellite computing network. However, by applying our model to all satellites in one constellation, a comprehensive description is possible.

An overview of our queuing model for a compute satellite is shown in Figure 2. We split our model in three parts, separated by the dotted boxes. The first part on the left encompasses all input to the compute entity of the satellite abbreviated as CPU. There, migration and client information is processed. The part on the upper right includes the output to other satellites, and thus, to the satellite core in our model. It is composed of service and migration information required by other satellites and is forwarded via ISLs. Finally, the lower right part represents the output to earth including service information for clients on earth. Each satellite has five resources, depicted in blue. The compute resource is named CPU in our system and represents the compute unit of a satellite. Furthermore, the links from and to the satellite core and from and to earth are modeled in a bidirectional way. The links consist of two input link resources, equalling the link capacities, where $BW_{c,s}$ in the upper left handles the incoming traffic from the satellite core to the modeled $satellite_i$ and $BW_{e,s}$ represent the link from earth. Finally, we have two outgoing link resources with $BW_{s,c}$ in the upper right handling the traffic towards the satellite core and $BW_{s,e}$ as link to earth.

C. Input to CPU

For the *Input to CPU* part of our model in the left dotted box of Figure 2, we see two incoming traffic classes for the resource $BW_{c,s}$, symbolizing traffic from the satellite core. The modeled $satellite_i$ has to take care of migration and service event traffic. Thus, we model the rate of the incoming migration information of clients currently connected to another satellite, but which soon need to be serviced by our $satellite_i$, as $\sigma_{migration,in}$. In addition, $\sum_{j=0,j\neq i}^{n-1} \lambda_{satj,out}$ represents the rate of incoming service events from other satellites. Since, in particular $\sigma_{migration,in}$ depends on the satellite orbits, population density, and client mobility, we model the underlying migration information arrival distribution at the satellite as general independent (GI). In contrast, we model the inter-arrival times of service events from the satellite core with an exponential distribution. This holds true as we can assume that service events follow a Poisson arrival process due to the large number of renewal processes, i.e., sending clients, as investigated by Metzger et al. [18]. Both migration information and service events arrive at the same queue on the link resource and consequently, the transmission time at each bandwidth resource is modeled as a GI distribution. For different traffic classes, we allow different transmission time distributions

according to the traffic class X, denoted by GI_X. On the other incoming link, resource $BW_{e,s}$ has to handle service traffic from earth arriving with rate $\lambda_{sat_i,in}$. Together with the incoming service event rate from the core, they sum up to λ_{global} depicted in orange. Finally, the migration scheduler is shown in the center top of the *Input to CPU* part. It schedules outgoing migration information with rate $\sigma_{migration,out}$. This information requires CPU time for processing. In reality, this scheduler might use information on satellite orbits and client movement to estimate user mobility to choose the correct satellite as receiver for migration traffic if multiple satellites will be within reach for a client on earth. Furthermore, the migration information could be sent to multiple satellites to increase reliability. In summary, the CPU resource must handle and process all service events received at and forwarded from other satellites, service events from earth, migration information from other satellites and migration information for other satellites generated by the migration scheduler. Thus, the processing time at the CPU is modeled as GI_X.

D. Output from CPU

In our setup, we assume that any new service event must be processed by each compute satellite's CPU exactly once. Then, it can be decided to which clients on earth the information has to be forwarded to. Additionally, service events that were received directly from earth at each satellite are forwarded to other satellites and thus, to the satellite core in our model. We assume that traffic to other satellites is handled by the satellite core for simplicity reasons and we do not consider relay traffic in our model. Consequently, we can model the output from the CPU as outgoing traffic from the satellite. The output back to the satellite core via ISLs, namely $BW_{s,c}$, is shown in the top right part of Figure 2. It handles the transmission of any outgoing migration information to other satellites as well as the distribution of events $\lambda_{sat_i,out}$ which have their origin within the population of all connected clients to $satellite_i$.

Finally, the output to earth via $BW_{s,e}$, depicted in the bottom right part of our model, takes care of transmitting service events to clients on the ground. Each processed service event from λ_{global}, is sent to each interested client according to the $interest_i$ of the local population pop_i via $BW_{s,e}$. The interest of a client represents the necessity of the client to receive the service event from the satellite for a fully functioning service. For example, in an MQTT service, the interest is represented by the client's subscriptions to topics of interest. The local satellite node population and its interest can be highly dynamic and change with satellite movement, user mobility on earth, or over time for a given population.

To this end, we can answer our first research questions RQ1: *To model a generic compute satellite in a satellite computing network running an arbitrary service, the link and compute resources need to be considered. We identify Poisson arrival processes for the arrival of service events into our system, but the complexity of influencing factors for the migration requires GI arrivals for migration events. All resources also have GI service times and for the CPU and links from and to the core, different traffic classes must be considered.*

TABLE I: Table of Notation.

Parameter	Unit	Description
$BW_{i,j}$	Gbit/s	bandwidth of link from i to j
BW_{minISL}	Gbit/s	minimal inter satellite bandwidth
$CPU_{processing}$	1/s	CPU processing rate
$E[size_{ev}]$	B	expected service event size
$info_{client}$	B	migration information of a client
$interest_i$	–	interest share of pop_i
λ_{global}	1/s	global service event rate
$\lambda_{sat_i,in}$	1/s	service event rate from earth to sat_i
$\lambda_{sat_i,out}$	1/s	service event forwarding rate of sat_i
pop_i	–	client population of sat_i
sat_i	–	satellite$_i$
$\sigma_{migr,in}$	1/s	incoming migration information rate
$\sigma_{migr,out}$	1/s	outgoing migration information rate
U_r	–	utilization of resource r
vp_i	s	visibility period of sat_i

IV. Application Load Estimation and Use Case Simulation

To understand the impact of different services and specific requirements on the compute satellite load, we introduce detail for our generic model that allows load estimation. These considerations are required to quantify the expected resource utilization for a given application. To analyse the load induced by an example application, we present a simulative environment and identify key parameters for system load, and as a consequence, application feasibility using satellite computing with different use-cases afterwards. The general used notation is summarized in Table I.

A. Application Load Estimation

Since our main focus is an in-depth analysis of application generated system load. Additionally, we assume stable connectivity at full bandwidth and the ability to successfully perform seamless and soft handovers between satellites at any time. Consequently, we can derive several equations for system stability and resource utilization from the generic model which are presented in the following.

First, the migration information $info_{client}$ of each client connected to the satellite must be transferable within the visibility period vp_i. After that time frame, the satellite will have lost connection to those clients. If this condition does not hold, clients can not be served after a handover and the system will break. We define the minimum inter-satellite bandwidth BW_{minISL} as $min(BW_{c,s}, BW_{s,c})$, representing the minimum of the link resources connected to the satellite core and present the service stability condition as

$$vp_i \geq \frac{info_{client} \cdot pop_i}{BW_{minISL}}. \tag{1}$$

Besides the service stability condition, we can define the load of each individual resource. In particular we define

$$U_{CPU} = \frac{\lambda_{global} + \sigma_{migr,in} + \sigma_{migr,out}}{CPU_{processing}} \quad (2)$$

as the CPU utilization U_{CPU}. This utilization is achieved as sum of all service event rates λ_{global} and migration information rates ($\sigma_{migr,in}$ and $\sigma_{migr,out}$) divided by the CPU processing rate $CPU_{processing}$. Furthermore, we can describe the link utilization from earth to the satellite $U_{BW_{e,s}}$, with

$$U_{BW_{e,s}} = \frac{\lambda_{sat,in} \cdot E[size_{ev}]}{BW_{e,s}}. \quad (3)$$

The combined arrival rate $\lambda_{sat,in}$ of events from clients on earth connected to the satellite is multiplied with the expected transmitted event size $E[size_{ev}]$ and divided by the link bandwidth from earth to the satellite $BW_{e,s}$. The utilization for the resource with direction towards earth, $U_{BW_{s,e}}$, is accordingly achieved by

$$U_{BW_{s,e}} = \frac{\lambda_{global} \cdot pop_i \cdot interest_i \cdot E[size_{ev}]}{BW_{s,e}}, \quad (4)$$

where all service events, arriving at rate λ_{global}, are replicated and sent to those clients in the local population pop_i of the $satellite_i$ that are interested in specific events according to their interest $interest_i$. The bandwidth of the link is $BW_{s,e}$.

Finally, the first summand in Equation 5 and Equation 6 describes the service load, while the second summand is for the migration load on the link resources from and towards the core, respectively. The utilization $U_{BW_{c,s}}$ of the incoming link from the core is described by

$$U_{BW_{c,s}} = \frac{\sum_{j=0,j\neq i}^{n-1} \lambda_{sat_{j,out}} \cdot E[size_{ev}]}{BW_{c,s}} + \frac{\sigma_{migr,in} \cdot info_{client}}{BW_{c,s}}, \quad (5)$$

where the service load is made up from $\sum_{j=0,j\neq i}^{n-1} \lambda_{sat_{j,out}}$, which includes all service event arrivals from other satellites, multiplied by their expected size $E[size_{ev}]$. The migration load is the rate of incoming migration information $\sigma_{migr,in}$ multiplied by the client migration size $info_{client}$. The utilization $U_{BW_{c,s}}$ of the outgoing link resource towards the satellite core is described by

$$U_{BW_{s,c}} = \frac{\lambda_{sat,in} \cdot E[size_{ev}]}{BW_{s,c}} + \frac{\sigma_{migr,out} \cdot info_{client}}{BW_{s,c}}, \quad (6)$$

where the first summand's numerator is similar to Equation 3, since it describes the forwarding of locally received service events from earth to the core. The migration load in the second summand is similar to the migration load in Equation 5, but changes the direction of the migration events. Consequently, their rate is changed from incoming $\sigma_{migr,in}$ to outgoing $\sigma_{migr,out}$. With this, we answer our second research question RQ2: *For system stability, the presented service stability condition must be met to ensure timely transfer of migration information for all clients. Additionally, the equations for the utilization of each single resource in our system must be considered. The overall system load can then be described through the bottleneck resource, i.e. the resource with the highest utilization, for a given scenario.*

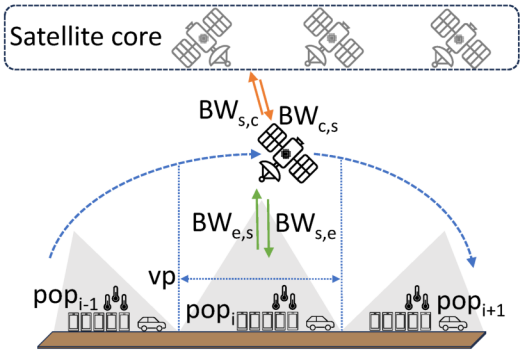


Fig. 3: Symbolic visualization of the simulation setup.

B. Simulative Environment for Load Analysis.

In the following, we present our process-oriented simulation environment for load analysis. Using this, we can simulate a satellite computing network architecture with input traffic grounded in an MQTT environment. We choose MQTT as example application, where each satellite runs an MQTT broker as a service. MQTT is a popular service application well suited for our analysis, as it is a rather light-weight communication protocol with easy deployment and broad usage possibilities that may span global use cases with millions of devices.

The simulation framework is based on SimPy [19] and simulates a satellite with a CPU resource, its current population, a bidirectional link to earth, and a bidirectional link to the satellite core in detail. Figure 3 shows the satellite in the center, and its bidirectional links in orange and green. To comprehensively investigate a complete satellite computing network architecture, we need to deal with multiple satellites. In addition to our compute satellite modeled in Figure 2, we simulate other satellites as part of the satellite core. As a result, our compute satellite only communicates with the satellite core and with the ground on earth. The local population, i.e., the clients with direct connection to the satellite, are located in the grey area which is typically referred to as footprint of the satellite. The clients in the footprint generate load which is sent to the satellite, processed, and forwarded to the satellite core, as well as to interested clients on earth. Consequently, the satellite core accepts migration information from the simulated compute satellite towards successor satellites, and forwards service events from the modeled satellite. Furthermore, it takes care of migration information from the preceding implicit satellite and the accumulated service event traffic $\lambda_{sat_j,out}$ of all implicit satellites. For simplicity, we assume a uniform satellite mobility which means that for a given geographical location, satellite handovers occur periodically. We neglect client mobility and the population is equally distributed over all satellites, as shown in Figure 3. The global service event rate is evenly split across the population and all satellites expect the same load. This allows us to model the migration load as the

TABLE II: Global simulation parameters

Parameter	Value	Parameter	Value
satellites	4,408	publisher ratio	0.1
$BW_{c,s}$, $BW_{s,c}$, $BW_{s,e}$	20 Gbit/s	$CPU_{processing}$	15,000/s
$BW_{e,s}$	8 Gbit/s	visibility period vp	5 min

TABLE III: System specific simulation parameters.

System	IoT	Fed. Learning	Automotive
clients	$200 \cdot 10^6$	5,000	$2.5 \cdot 10^6$
λ_{global}	12,500/s	1/s	18.3/s
$E[size_{ev}]$	19 B	19 MB	19 B
$info_{client}$	32 B	320 MB	3.2 MB

number of clients located in each footprint multiplied by their migration information. After a duration of vp, shown as the footprint width in the figure, the satellite has progressed on its orbit, shown by the blue arc, and the connected population has been fully exchanged. These constraints simplify our model and allow us to fully simulate our single compute satellite within our simulation environment to assess the performance. We do not consider variable population density, e.g., different population density between countries, on land, or on the sea. Nonetheless, this model is valid if it is applied to the bottleneck satellite in a heterogeneous network and can draw insightful conclusions on limitations induced by increasing load.

The overall simulation parameters in Table II are taken from the literature as described in the following. We derive the satellite parameters from existing Starlink constellations in its first phase. As such, we chose 4,408 satellites [7] and a maximum bandwidth of 20 Gbit/s [6] which we assume to reach for the inter-satellite links and the link towards earth using LEO satellites. For the up link from earth to the satellite, 40 % of the down link is set according to [7]. The publisher ratio, i.e., the share of clients that are publishing events into the system, is set to 10 %. Since this is a parameter taken from the HiveMQ benchmark [20] containing more than 200 million concurrent connections, this allows us to model our system rather closely to their measurements. Thus, we assume one MQTT broker on each satellite with a maximum operation event processing rate of 15,000/s on an AWS m5n.8xlarge compute module with 32vCPUs. As we are unable to secure actual measurements for our migration system, and considering that lightweight state migration, i.e., forwarding of the local subscriptions, for MQTT brokers does not yet exist, we neglect the impact of migration handling on the CPU. This approach is valid since it is only one database operation for each lookup of forwarding information and entry of received information, which happens in our simulation only once per visibility period.

C. Exemplary MQTT Use Case Evaluation via Simulation

We evaluate three different MQTT use cases with our simulative environment, namely a large-scale IoT scenario, a federated learning system, and a system for automotive connected vehicles. Those systems differ in the number of clients, the event rate, the event size, as well as the assumed necessary migration information, shown in Table III. Consequently, they are well suited for our analysis. For the large scale IoT scenario, we derive those values from [20] with a large population and small data load. As an example from the device management class, we derive the automotive scenario from aggregated statistics reported by our industry partners, which leads to a scenario with lower population, but higher migration information. Finally, we construct the federated learning scenario with a small number of clients and a low communication rate, but high migration and message load. We acknowledge that the use cases are chosen for their different characteristics and represent classes of services instead for their particular parameter values. In general, the number of possible use cases on a satellite network infrastructure is extensive. For each scenario setup we perform 30 runs to achieve statistically significant statements in our results.

V. Results

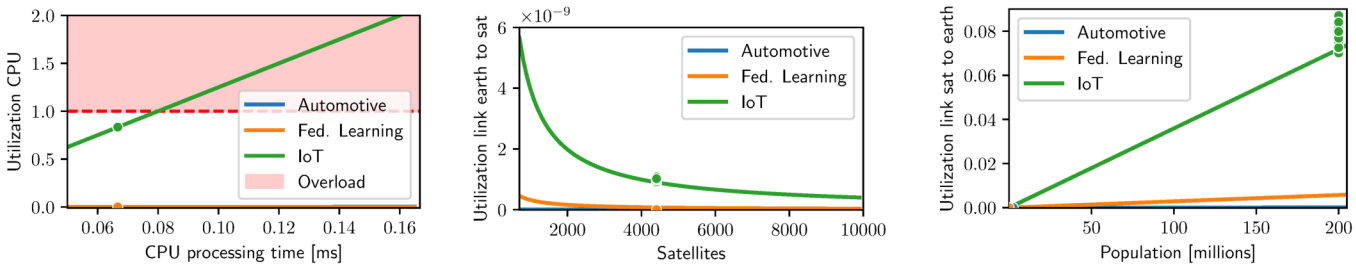
In this section, we investigate the requirement for migration in a satellite computing network and assess our model and, in particular, our utilization equations with our simulation and answer the remaining research question.

A. Requirement for Migration Mechanism

First, we identify that pure compute satellite resources are not sufficient for a seamless service, but a migration mechanism is mandatory. From the HiveMQ benchmark, we obtain a client connection rate of 2,252/s at a CPU load of 30 % to 40 %. This is the utilization generated by connecting clients to the broker and storing their subscription information. Without our proposed migration mechanism between satellites, each client has to establish a new connection with a broker every time a handover to the next satellite is required. With 200 million clients on 4,408 satellites, and a visibility period of 5 min, after which the client population of a satellite is renewed completely, more than 20 s of CPU time within that 5 min period are spent for connecting 45,372 clients to the MQTT broker. That sums up to more than 6 % of the available CPU time in each period. If we reduce the handover interval between satellites to 114 s as reported in [7], this increases to 17.5 %. Similarly, we assume some impact on the CPU resource during client's disconnects. Furthermore, subscription information must be sent from each client to the satellite during the connection process without migration via ISLs. This generates additional load on the link from earth to our compute satellite, which is the slowest link in our network. Thus, our proposed migration mechanism to transfer essential service information for each client between satellites is crucial to reduce the overall system load and allow a seamless handover.

B. Parameter Study and Simulation Results on Utilization

In the following, we discuss for each of our scenarios the resulting theoretical utilization based on the formulas before we present our simulation results for each scenario, and

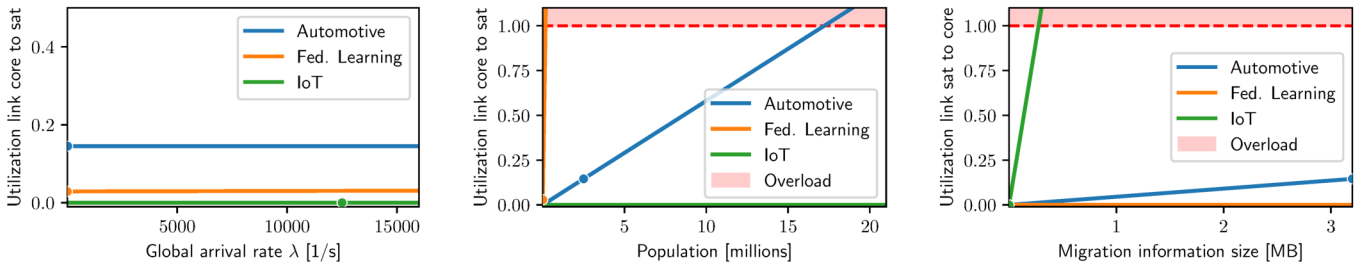


(a) CPU utilization based on required CPU processing time.

(b) Link utilization from earth to satellite based on the number of satellites.

(c) Link utilization from satellite to earth based on population.

Fig. 4: Resource utilization of CPU and links with earth based on different parameters.



(a) Link utilization from core to satellite based on service arrival rate.

(b) Link utilization from core to satellite based on global population.

(c) Link utilization from satellite to core based on migration information size.

Fig. 5: Utilization of inter-satellite links based on different parameters.

TABLE IV: Individual resource utilization for each system.

System	IoT	Fed. Learning	Automotive
U_{CPU}	0.833	$6.666 \cdot 10^{-5}$	0.001
$U_{BW_{s,e}}$	0.065	$1.293 \cdot 10^{-7}$	$1.831 \cdot 10^{-6}$
$U_{BW_{e,s}}$	$8.980 \cdot 10^{-10}$	$7.184 \cdot 10^{-11}$	$1.315 \cdot 10^{-12}$
$U_{BW_{s,c}}$	0.000116	0.029	0.145
$U_{BW_{c,s}}$	0.000118	0.029	0.145

finally, our parameter study to identify key influencing system parameters. We present the theoretical results in Table IV, that we obtain with our additional constraints from the equations shown in Section IV. Depending on the use case, we see different resources as potential bottleneck. For the IoT scenario, CPU utilization is above 80 % and the link utilization towards earth is 6.5 %, while the ISLs experience lower utilization. For the federated learning service, the ISLs are the most utilized link with 2.9 %. In contrast, for the automotive service 14.5 % ISL utilization is derived and a higher CPU utilization of 0.1 % compared to the $6.666 \cdot 10^{-3}$ % with federated learning.

With these values we assess our simulation results and the equations. For each scenario and the utilization of each resource from our simulation, we calculate the 95 % confidence intervals and compare them to the theoretical results. The theoretical value lies within each confidence interval and verifies the simulation results in accordance to the formulas presented in Section IV-A. To get a better understanding of the resource utilization in our architecture, we perform a parameter study to investigate the performance of our system based on our equations. Additionally, we include our simulation results into the following figures of the parameter study. Figure 4a shows the CPU utilization for the different systems when different processing times from 0.05 ms to 0.18 ms are applied. The green line represents the parameter study for the IoT use case, the orange line for the federated learning use case, and the blue line for the automotive use case. Dots in the figure represent our simulation results which we obtain with the parameters from Table II and Table III. On the x-axis the CPU processing time is depicted, while the utilization is shown on the y-axis. Exceeding the red dotted line at a utilization of 100 % means overload at the resource, shown by the area in red. We see that for the IoT use case parameter study in green that at 0.08 ms the red line is crossed and the CPU can not process the number of events in that case. The green dot at 0.83 utilization shows the simulation results. For the other use cases the calculated utilization across the parameter range, shown as lines, and the simulation results, shown as dots, are depicted as well, but due to low utilization and proximity to the x-axis, only the orange line and an orange dot for the federated learning can be seen. The automotive use case in blue is right below the orange line and thus, is not visible.

Figure 4b shows that the utilization on the link from earth to satellite follows the relationship $\frac{1}{x}$, where x is the number of satellites. With more satellites, the local population and their generated events are spread across more satellites. Again we

see a line of green dots at 4,408 on the x-axis, representing the simulation parameters. All the 95 % confidence intervals from our simulation results overlap with the theoretical results again. The link utilization from satellite to earth, shown in Figure 4c, is mainly effected by the replication rate due to the number of subscribers to each event. Since the IoT example has a large population, the link utilization is highest, even though the expected event size is only 19 B compared to 19 MB. For event sizes in the kilobyte range, the IoT service would have a bottleneck on this link. The migration load has a much larger impact on the ISL utilization than the service load shown in Figure 5a. Although, on the link from satellite core to the satellite, the event load from all other satellites is transmitted, a change in the arrival rate from $1/\mathrm{s}$ to $16{,}000/\mathrm{s}$ does not have a visible impact. Changing the population size can have a large impact as shown in Figure 5b. The federated learning use case follows closely to the y-axis and experiences overload on the ISL from core to satellite for a population of less than 17,500 The automotive use case similarly suffers and since deployments of automotive MQTT systems are continuously growing, an increase to a sevenfold population is realistic, but would exhaust this resource. For the IoT use case with little migration information, this resource is not a concern. Finally, in Figure 5c, we see that the ISL utilization from the satellite to the core is highly dependent on the migration information size. For the IoT system, less then 0.5 MB suffice for overload on this resource but the federated learning use case can handle migration information in the gigabyte range.

To summarize, key influencing factors of a service on a satellite compute infrastructure are investigated and, depending on the use case, all of the presented parameters have the potential to become a bottleneck in a system with state-of-the-art satellite resources. This is shown with our simulation framework and the derived utilization formulas. Finally, we can answer our third research question as follows: *Key influencing factors are client population, the satellite computing network's physical resources, and the use case-dependent migration and service load. These change for different systems and with it, the bottleneck resource. Consequently, it is important to investigate the load at each resource. While all of our MQTT use-cases could be realized in our environment, limitations and overload scenarios are observed by our parameter study and without migration, only few services are feasible in a comprehensive satellite computing network.*

VI. Conclusion

The ubiquitous demand for access to communication networks and computational resources across the globe is undeniable for private and industrial applications. However, the global roll-out of next-generation networks is costly and time consuming. Enabled through constant technological advancements, the concept of satellite computing networks emerged as one solution for high quality coverage around the globe but poses challenges for future networks. To offer a seamless service and to guarantee a high QoS for arbitrary existing and future services, it is important to identify benefits and limitations of such deployments. Therefore, thorough network models are essential to determine the expected load. We close this open gap in the existing literature with a generic model for satellite computing networks. We introduced migration as a vital mechanism in our system and provide a formal description of the utilization at each system resource. Without migration we can showcase that only few services are feasible in a satellite computing network. With large scale IoT, federated learning, and automotive as MQTT use cases, we can identify key influencing factors for system load. Our results show that depending on the client population, physical resources, and the use case-dependent migration and service load, each resource can become the bottleneck, necessitating careful consideration during the system planning phase. Moreover, the application architecture must not neglect the network situation in a real deployment during its design phase.

References

[1] Reuters, "First Satellites for Musk's Starlink Internet Venture Launched into Orbit," 2019, accessed: 2024-02-09. [Online]. Available: https://www.reuters.com/article/idUSKCN1SU07Y/

[2] G. Thomas *et al.*, "Eutelsat Quantum a Fully Flexible Software Defined Satellite Successfully Operating on Orbit," 2023.

[3] A. Gaber *et al.*, "5G and Satellite Network Convergence: Survey for Opportunities, Challenges and Enabler Technologies," in *Novel Intelligent and Leading Emerging Sciences Conference*, 2020.

[4] S. Chen *et al.*, "System Integration of Terrestrial Mobile Communication and Satellite Communication — The Trends, Challenges and Key Technologies in B5G and 6G," *China comm.*, 2020.

[5] E. Jaafari *et al.*, "Introduction to the 3GPP-defined NTN standard: A comprehensive view on the 3GPP work on NTN," *International Journal of Satellite Communications and Networking*, vol. 41, 2023.

[6] O. B. Y. others, "Evolution of High Throughput Satellite Systems: Vision, Requirements, and Key Technologies," 2023.

[7] R. Freund, *ITG Tech. Report: Broadband Coverage in Germany*, 2022.

[8] D. He *et al.*, "Security Analysis of a Space-based Wireless Network," *IEEE Network*, 2019.

[9] X. Zhang *et al.*, "A Review of Routing Algorithms Based on Multi-Layer Satellite Networks," in *Journal of Physics*, 2021.

[10] P. Wang *et al.*, "Multi-layer LEO Satellite Constellation Design for Seamless Global Coverage," in *Global Comm. Conf.* IEEE, 2021.

[11] A. Pozdnyakov, "iPhone 14 Will Have Satellite Connectivity. How Exactly It Will Work." 2022, accessed: 2024-02-09. [Online]. Available: https://www.universetoday.com/157474/iphone-14-will-have-satellite-connectivity-how-exactly-it-will-work/

[12] L. Cheng *et al.*, "Dynamic Computation Offloading in Satellite Edge Computing," in *International Conf. on Communications*. IEEE, 2022.

[13] J. Wei *et al.*, "Satellite IoT Edge Intelligent Computing: A Research on Architecture," *Electronics*, 2019.

[14] Q. Tang *et al.*, "Computation Offloading in LEO Satellite Networks with Hybrid Cloud and Edge Computing," *IEEE Internet of Things Journal*, 2021.

[15] J. Wei *et al.*, "SatEdgeSim: A Toolkit for Modeling and Simulation of Performance Evaluation in Satellite Edge Computing Environments," in *International Conference on Communication Software and Networks*. IEEE, 2020.

[16] I. Leyva-Mayorga *et al.*, "Satellite Edge Computing for Real-Time and very-high Resolution Earth Observation," *IEEE Transactions on Communications*.

[17] Z. Liu *et al.*, "Routing Algorithm Design of Satellite Network Architecture based on SDN and ICN," *Int. Journal of Satellite Communications and Networking*, 2020.

[18] F. Metzger *et al.*, "Modeling of Aggregated IoT Traffic and its Application to an IoT Cloud," *Proceedings of the IEEE*.

[19] A. Meurer *et al.*, "SymPy: Symbolic Computing in Python," 2017. [Online]. Available: https://doi.org/10.7717/peerj-cs.103

[20] HiveMQ, "Achieving 200 Million Concurrent Connections with HiveMQ - Whitepaper," 2023, accessed: 2024-02-09. [Online]. Available: https://www.hivemq.com/benchmark-200-million/