A Novel Space Based Hosting Approach for Ultra Low Latency Web Services

SUBMISSION DATE / POSTED DATE

19-01-2022 / 01-02-2022

CITATION

Dalai, Debabrata; B. S., Manoj; Bala Sukumaran, Vineeth; Babu, Sarath (2022): A Novel Space Based Hosting Approach for Ultra Low Latency Web Services. TechRxiv. Preprint. https://doi.org/10.36227/techrxiv.18666455.v1

DOI

10.36227/techrxiv.18666455.v1

A Novel Space Based Hosting Approach for Ultra Low Latency Web Services

Debabrata Dalai, Sarath Babu, Vineeth B. S., and B. S. Manoj
Indian Institute of Space Science and Technology, Thiruvananthapuram, India 695547
Email: ddalai@ieee.org, sarath.babu.2014@ieee.org, vineethbs@gmail.com, bsmanoj@ieee.org

Abstract—The lack of 4G and 5G communication infrastructure in rural and remote areas aggravates the digital divide. Satellite communication forms an important technology to achieve the universal goal of *connecting the digitally unconnected population* through global coverage. The key advantages of satellite communication include wide network coverage and availability. We propose a Space-Based Hosting Service (SBHS) approach to deploy the content-servers or copies of terrestrial content-servers in space modules such as satellites to achieve the low-latency service requirements. For SBHS, we develop mathematical models for communication, queuing, and computation. We use B+ tree to estimate computational delay and computational energy consumption of the content-server. Further, we design an optimization algorithm using Markov Decision Process (MDP) for minimizing the queuing delay and computational energy consumption. The efficacy of our SBHS is tested with the Wikipedia case-study using the Iridium-NEXT satellite constellation model. Extensive simulations are carried out to analyze end-to-end delays for various countries using the SBHS approach. Simulation studies show minimum end-to-end delays of 6.56 ms and 23.66 ms achieved for the United Kingdom (UK) using text-based and multimedia-based traffic, respectively. The simulation results revealed the benefits of hosting a content-server in space for achieving ultra-low latency compared to traditional satellite-based web services.

Index Terms—Satellite networks, satellite edge computing, content-server, B+ tree, Markov Decision Process (MDP), Wikipedia.

1 Introduction

The Information Age focuses on new technologies such as Internet of Things (IoT), automation, big data, Artificial Intelligence (AI), cloud computing, and 5G [1]. Internet connectivity is considered one of the basic requirements to connect people worldwide. However, according to the reports of United Nations International Children's Emergency Fund (UNICEF) [2], International Telecommunication Union (ITU) World Telecommunication [3], and ITU analytical publication [4], 47% of the global population is out of the purview of Internet connectivity. Moreover, a report of the United Nations Educational, Scientific and Cultural Organization (UNESCO) [5] revealed that two-thirds of the world's school-age children have no Internet access at their home. Fig. 1 shows the Internet usage pattern across the globe during 2015–2019. In all four years, the penetration rate of Internet connectivity is observed to be very low for Least Developing Countries (LDCs) while the penetration rate of the world is found to be 53%. That is, a significant portion of the world is still remaining unconnected. For instance, a survey by ITU in [6] found that 79.4% population of the African continent, 58.4% of the Arab states, and 58.1% of the Asia and Pacific region are unconnected in terms of Internet connectivity.

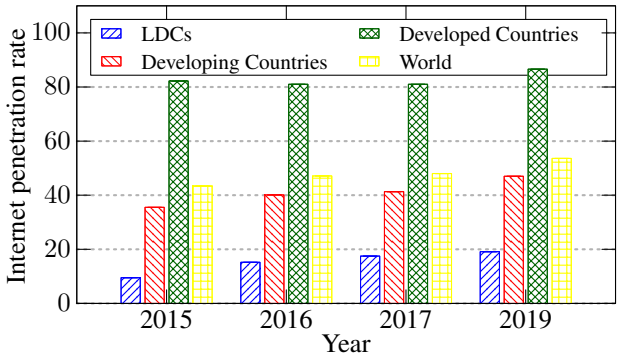


Fig. 1: Percentage of people using the Internet in terms of development status and world [4] [7] [8] [9].

In terms of the access to Internet facilities, rural and remote areas of the world constitute the most affected segments within any nation. The new communication technologies including 4G LTE, and 5G are not accessible to them due to the challenges involved in the remote and rural areas such as lack of infrastructure deployment and network coverage, lack of investment, and high infrastructure cost. Therefore, satellite-based communication technologies such as satellite broadband and integrated 5G-satellite networks can play a vital role in achieving the goal *connecting the digitally unconnected population*, thereby reducing the digital divide. The key advantages of satellite communication include wide network coverage and availability resulting in improved network connectivity in the rural and remote areas. Low infrastructure requirement in the new satellite technologies is an added advantage. For example, a device can connect to the satellites either directly or via a single small antenna such as a satellite dish.

One of the challenges involved in satellite communication is the placement of satellites in different orbits such as Low Earth Orbit (LEO), Medium Earth Orbit (MEO), and Geostationary Earth Orbit (GEO). The distance of these

orbits from the Earth results in high round-trip delays in the order of 240 milliseconds for GEO, 13.3 – 240 milliseconds for MEO, and 1.2 – 13.3 milliseconds for LEO satellite. Moreover, satellite links are of lower bandwidth than optical networks resulting in low data-rates and high end-to-end latency. In addition, the disturbances in the atmosphere negatively affect satellite channels and result in packet loss and a subsequent increase in the number of retransmissions.

One of the important ways to leverage the satellite technology to achieve remote/rural communication is to deploy content-servers in satellites. Such satellite content-servers can provide basic Internet needs required for services such as education especially in the developing and least-developed countries. The remote and rural users can directly access the contents irrespective of their location on the Earth. The contents can be text-based information such as articles and web-pages, or multimedia information embedded within images and videos.

In terrestrial web-based services, organizations maintain multiple copies of the data-center at different locations on Earth in order to reduce access latency. For example, Wikipedia maintains several instances of data-centers, i.e., three in the USA, one in Europe, and another in Asia [10] for their Wikipedia service. Hosting a content-server in satellite can be considered as a type of Internet hosting service that can be directly accessible to the users. In this context, a satellite content-server can be considered as a software running on one or more satellite systems to serve the HTTP client requests on the World Wide Web (WWW). The data-center placed in the satellite network stores the contents such as multimedia files, articles, and web-pages. In a dedicated satellite hosting service, the whole server is dedicated to a specific or multiple services that belong to the same organization. Depending on the user demand, the content-server shares the data or performs the computation.

In this paper, we propose a novel Space-Based Hosting Service (SBHS) approach to deploy content-server in space in LEO and GEO satellites. Our major contributions are enumerated as follows:

1) A novel Space Based Hosting Service (SBHS) approach for hosting an entire content-server in the LEO and GEO satellites.
2) A computation model for SBHS based on B+ tree to calculate the computational delay and computational energy consumption.
3) Theoretical formulation of the average computational delay and average computational energy consumption for SBHS using B+ tree data structure.
4) An optimization algorithm for the LEO satellite queue using Markov Decision Process (MDP) to optimize queuing delay and computational energy consumption.
5) A case-study of hosting Wikipedia content using Iridium-NEXT satellite constellation model using the proposed SBHS.
6) Performance analysis includes Wikipedia case-study results such as average end-to-end delay, total computational energy consumption, and total number of successfully served requests.

The remainder of this paper is organized as follows: In Section 2, we discuss the existing work related to satellite broadband Internet and satellite networks that are used as access networks. Section 3 presents the architecture of our SBHS while the proposed theoretical model and our methodology for SBHS are described in Section 4. In Section 5, we design the optimization algorithm using the MDP policy iteration model to optimize the computational energy consumption and average end-to-end latency in LEO satellite. Section 6 describes our system model of hosting Wikipedia servers in the LEO and GEO satellites. In Section 7, we describe the experimental setup while the performance of our approach is analyzed in Section 8. Finally, Section 9 concludes the paper.

2 Related Work

In this section, we discuss the existing work related to the satellites employed for broadcasting and access network, emphasizing the impact of satellite broadband Internet on rural and remote areas around the world. Further, the next-generation mega satellite constellations and the recent developments in satellite Internet using mega satellite constellations are discussed.

Traditionally GEO satellites are employed to provide Internet to the ground user in the satellite-based Internet access. Authors of [11] discussed different satellite technologies and several aspects related to challenges, system engineering, and protocols to realize satellite broadband communication. The applications of the GEO satellites in mobile and personal communication are emphasized in [12]. The users use their personalized handheld satellite terminals in personal satellite communication. Multimedia content delivery to the end-user or the edge of the network using satellite broadband was discussed in [13]. Authors of [13] reviewed different satellite topologies and underlying protocols for broadband applications by exploiting the broadcasting and ubiquitous coverage characteristics of satellites. The multimedia delivery services using TCP were described in [14] where the performance of TCP and enhanced proxies over the satellite were investigated. A system design using GEO satellites termed as IMAGINE Africa was proposed in [15]. IMAGINE focused on providing Internet to the remote and rural areas of the African continent in a cost-effective way. The system design consists of GEO satellites and easily deployable ground stations.

New satellite technology has transitioned from backup technology to the terrestrial Internet Service Provider (ISP), and further to the stand-alone Internet service provider in recent years. The performance monitoring and analysis of two satellite broadband Internet access services were carried out in [16], i.e., the First-Generation Satellite network (FGS) and the Second-Generation satellite network (SGS). In fact, new mega satellite constellations are planned with Internet access services to realize a fully connected world. The new mega satellite constellations such as OneWeb, Starlink, Telesat, and Kuiper were proposed to provide global Internet access [17]. OneWeb is planned to launch 650 LEO satellites in the first phase that orbit around 1200 km in low Earth polar orbits with 86.4° inclination [18]. Initially, the constellation is planned to use 19 orbital planes with 49 satellites per orbital plane. On the other hand, the

Starlink satellite constellation by SpaceX is designed with 12000 LEO satellites in the Sun-synchronous LEO [19]. The satellites are placed in altitudes varying between 340 km and 1110 km while the orbital inclination varies between 53.0° and 97.6°. The Telesat constellation consists of 120 LEO satellites with 1000 km altitude in an inclined polar orbit [20]. The Kuiper satellite constellation is proposed to have 3,236 satellites in 98 orbital planes in three orbital shells [21]. The LEO satellites are planned to place in three different orbital altitudes 590 km, 610 km, and 630 km.

Apart from the satellite Internet access, the LEO satellites are used as a space-based controller for resource management in terrestrial-space-based networking frameworks [22]. A deep Q-learning-based optimization problem in the space-based Satellite Terrestrial Networks (STN) was discussed in [22] where authors concentrate on the optimization and management of resources for networking, caching, and computing. The data layer includes LEO satellites for networking, content caching, and edge computing. The controller is placed in MEO and GEO satellites as well as on the Earth's surface. The application layer consists of remote sensing, navigation, communication, and monitoring applications. State-space, action-space, and reward function are formulated for both Q-learning and deep Q-learning optimization problems to maximize the expected reward and minimize the loss function, respectively.

A resource allocation method for Edge Computing in Satellite (ECS) was proposed in [23] using Advanced K-means Algorithm (AKA). The nodes in ECS reside in LEO satellites. Requests from the user terminal are classified into three categories based on delay, bandwidth, and connection time. Authors used the concept of virtualization, where the resources of the edge node are divided into logical slices to serve the specific applications with guaranteed QoS.

A multi-tier caching mechanism using LEO satellite was proposed in [24] where a Satellite Edge Server (SES) framework with an inter-satellite cache transfer mechanism between LEO satellites was employed. The GEO satellite is assumed to be the global cache for all files and LEO satellites as the edge server for the end-user. The framework enables caching of the most popular files in LEO satellites to achieve low latency depending on the geographical region. Two case studies were analyzed: (i) single cell with one LEO satellite and (ii) single cell with two LEO satellites. Further, a theoretical model is proposed to calculate the cache hit ratio.

Computation offloading in multi-tier satellite architectures was discussed in [25] where two algorithms were proposed to identify the suitable Mobile Edge Computing (MEC) servers depending on the resource availability to satisfy the client's service requirements. Authors of [25] considered the Iridium constellation with 66 LEO satellites for the simulation. Each satellite cluster has five satellites with inter-satellite links between them. The MEC servers are present in both terrestrial eNodeB and LEO satellites. Spot beam with Time Division Multiple Access (TDMA) is used for communication between eNodeB and LEO satellites. The offloading of computational tasks to the LEO satellite is initiated when the load on the MEC server in the terrestrial node exceeds the specified threshold. Similar to [25], authors of [26] proposed an offloading strategy for mobile devices to offload the tasks to the satellites, using a game-theoretic approach. Each satellite in the LEO constellation acts as an MEC server. The system model is classified into three parts: (i) the orbit model of the satellite, (ii) the communication model of the task of offloading, and (iii) the computation model of task execution.

In [27], authors proposed an architecture to integrate terrestrial and space-based computation to realize edge computing. The architecture consists of four types of nodes: user node, fog satellite node, space-based edge cloud, and remote cloud. User nodes can be satellites or UAVs. A fog satellite node has low computing power and low storage capacity which accepts computing requests from user nodes. Further, the fog satellite node requests assistance to edge cloud or remote cloud nodes. The fog satellite node uses virtualization to deploy different services depending on the user needs. A dual optimization problem in terms of latency and energy in the context of MEC enhanced SaT-IoT architecture was discussed in [28]. The optimization problem is formulated with the help of a dynamic mixed-integer programming problem with end-to-end latency and energy consumption as the optimization parameters. Further, the optimization problem is classified into two subproblems: (i) computing and communication resource allocation and (ii) offloading decisions with a fixed user and joint user association. The subproblems are solved with the Lagrange multiplier method and Deep Reinforcement Learning (DRL).

In [29], authors discussed satellite-based data-centers and concentrated on minimizing energy consumption while minimizing the overflow of queues residing in LEO satellites. The minimization of energy is accomplished through power allocation to the downlink channels. Several duplicate chunks of each file are stored in multiple LEO data-centers at random. Each satellite contains a queue to complete the downloading jobs. The user request is transmitted to all ground stations and further transferred to the LEO satellites for performing the download in parallel. Coflow-like Join the first K-shortest Queues (JKQ) strategy is adapted for the job dispatches. A drift-plus-penalty optimization technique is applied to construct an online scheduling framework for achieving a near-optimal solution to the problem. In [30], satellite-terrestrial networks (STN) based QoE-aware content distribution scheme was discussed. Authors proposed a density-based network division algorithm where the contents are distributed according to the subscriber densities. Using the link usability of STN topology, a cache node selection algorithm was proposed for LEO satellites.

In space-based communication systems, GEO satellites are exploited for providing broadband services while the LEO satellites primarily serve the access network. Even though the concepts of caching and data centers were discussed in LEO satellites, authors concentrated on minimizing the energy consumption in downlink transmission channels. In this paper, we introduce the idea of hosting the entire content-server in LEO-based satellite constellations and propose models and algorithms to minimize end-to-end latency as well as for reducing the computational energy consumption in resource-constrained LEO satellites.

3 Proposed System Architecture for SBHS

For our Space Based Hosting Service (SBHS), we propose a network model consisting of terrestrial content-server, GEO satellite, LEO satellite, and User Equipment (UE). Fig. 2 presents our architecture for hosting a complete content-server in LEO and GEO satellites to serve the UEs.

TABLE 1: **List of notations.**

Notation	Description
U	Maximum number of UEs
L	Maximum number of LEO satellites
G	Maximum number of GEO satellites
CS	Maximum number of terrestrial content-servers
u_i	i^{th} UE
l_i	i^{th} LEO satellite
g_i	i^{th} GEO satellite
cs_i	i^{th} terrestrial content-server
(x_u, y_u, z_u)	Cartesian coordinate of UE u
(x_l^t, y_l^t, z_l^t)	Cartesian coordinate of LEO satellite l at time t
(x_g, y_g, z_g)	Cartesian coordinate of GEO satellite g
c_u	Size of content title requested by UE u
C	Maximum size of content title
P_u	Size of a content
F_1	Single-core computation capacity in GHz
k	Maximum number of cores in a LEO satellite
$T_{endtoend}$	End-to-end delay
T_{prop}	Propagation delay
T_{trans}	Total transmission delay
T_{trans}^{up}	Uplink transmission delay
T_{trans}^{down}	Downlink transmission delay
T_{queue}	Queuing delay
T_{comp}	Total computational delay
T_{comp}^{search}	Delay to search the content
$T_{comp}^{retrieve}$	Delay to retrieve the content
d_{ul}	Distance between UE and LEO
d_{lu}	Distance between LEO and UE
c	Speed of light
R_{ul}^{up}	Uplink physical data-rate from UE to LEO
R_{lu}^{down}	Downlink physical data-rate from LEO to UE
m	Number of key pairs in B+ tree
P_t	Total number of contents in the LEO database
$R_{t_{country}}$	Total number of content requests per country
$P_{E_{country}}$	Total number of UEs present per country
β	CPU cycle required per one bit
e	Energy consumption per one CPU cycle
ϵ_s	Energy required to serve a request at LEO satellite
q	Maximum receive-buffer size in LEO satellite
λ	Arrival rate of requests at LEO
λ_u	Request generation rate at the UE
ρ	Average utilization of LEO queue in %
L_{sys}	Average number of requests in the LEO system
L_{queue}	Average number of requests in the LEO queue
W_{sys}	Average time requests waiting in the LEO system
W_{queue}	Average time requests waiting in the LEO queue
λ_s	Arrival rate per μs at satellite
μ_{sk}	Service rate per μs in the satellite with k cores
S	Set of all states in the MDP model
A	Set of all actions in the MDP model
a_i	Notation for an action i
t_s	Average service time
α	The weight factor in the reward function
TP	Transition probability matrix
$\pi^*(s)$	Near-optimal policy
$V^*(s)$	Near-optimal value function

3.1 Terrestrial content-server

The terrestrial content-servers belonging to the data-centers are present in a very few locations on the Earth. The data-center contains the text and multimedia-based contents. Generally, the terrestrial content-server is always connected to the GEO satellite and performs the required computations to update the contents periodically. Assume, the i^{th} terrestrial content-server is represented as cs_i, where $1 \leq i \leq CS$.

3.2 GEO Satellite

GEO satellite acts as a replica of the terrestrial content-server as a data-center in the SBHS, i.e., a global content-server which is always connected to the terrestrial content-server residing on the ground. GEO satellite gets frequent updates on contents such as addition, deletion, and updating from the terrestrial content-server. Further, GEO satellites are connected to the LEO satellites and share the updated contents.

In Fig. 2, the GEO satellite is placed approximately at 35,786 km circular orbit around the Earth and has an orbital period equal to the Earth's rotation period. Therefore, the position of the GEO satellite remains stationary with respect to the locations on Earth. Assume, the i^{th} GEO satellite is represented as g_i, where $1 \leq i \leq G$ and G is the maximum number of GEO satellites present in the SBHS system. The tuple (x_g, y_g, z_g) represents X, Y, and Z coordinate positions of the GEO satellite g.

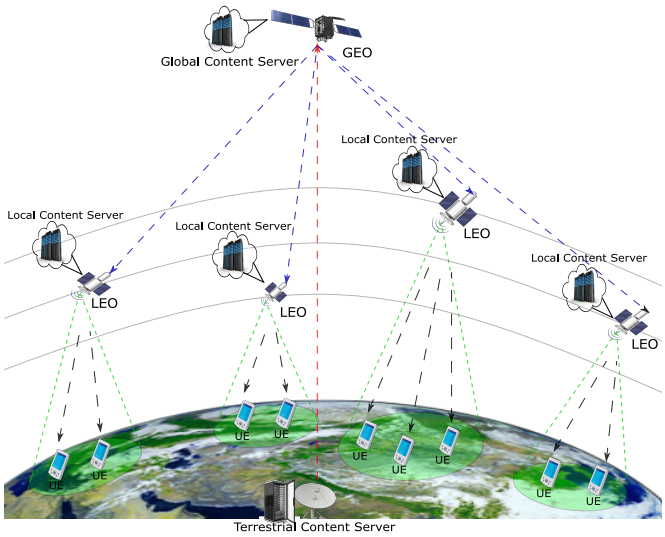


Fig. 2: Space Based Hosting Service (SBHS) architecture.

3.3 LEO Satellite

The i^{th} LEO satellite is represented as l_i, where $1 \leq i \leq L$ and L is the maximum number of LEO satellites present in the SBHS system. The coordinates of the LEO satellites depends on the orbit location and the orbital velocity of the respective LEO satellites. Assume, (x_l^t, y_l^t, z_l^t) are the X, Y, and Z coordinates of LEO satellite l at time t. The LEO satellite positions change over time with respect to the position on Earth.

The local content-server along with the database is assumed to be present in all LEO satellites for the SBHS system (Fig. 2). The contents in the LEO satellites are downloaded and get periodically updated from the GEO satellite. We classify the LEO satellites into (i) single-core and (ii) multi-core satellites. The classifications are in terms of the computational capabilities of LEO satellites. The computational capability of LEO satellite l_i with single-core is represented

as F_i GHz, where $1 \leq i \leq L$. A multi-core LEO satellite is defined with k number of single-cores. Therefore, the computational capability of multi-core LEO l_i is $k \times F_i$ GHz. In our SBHS, we assume the computational capabilities of all single-cores in the LEO satellites as equal.

3.4 User Equipment (UE)

Assume, the i^{th} UE is represented as u_i, where $1 \leq i \leq U$ and U is the total UE population around the globe. UEs are deployed all over the globe (Fig. 2) with stationary coordinate (x_{u_i}, y_{u_i}, z_{u_i}). UEs request the title of the required content to the nearest LEO satellite and served by the same LEO satellite. Assume the size of the requested content title is represented as c_u for a UE u. The value c_u is selected from a discrete uniform distribution between 0 and C, where C is the maximum size of search space for the requested content title.

4 Our System Model for SBHS

To realize the proposed SBHS, we mathematically model each delay that constitutes in a communication session between a UE and the LEO satellite-based server. The mathematical models involved in the estimation of average end-to-end delay and computational energy consumption for SBHS. We define a propagation model, transmission model, computation model, and queuing model. For modelling, we consider the total computational energy consumption in retrieving the requested content from the LEO satellite-based server. We estimate end-to-end delay as the difference between the time at which content is requested from a UE and the same content is received at the UE. The end-to-end delay consists of four major components: (i) propagation delay (T_{prop}), (ii) transmission delay (T_{trans}), (iii) queuing delay (T_{queue}), and (iv) computational delay (T_{comp}). We model each component mathematically in the following subsections. The end-to-end delay ($T_{endtoend}$) is calculated as

$$T_{endtoend} = T_{prop} + T_{trans} + T_{queue} + T_{comp}. \quad (1)$$

4.1 Propagation Model

We select the nearest LEO satellite for a UE for the propagation of contents assuming that the LEO satellite is in the range and Line of Sight (LoS) with the UE. If a UE is in the vicinity of multiple LEO satellites, the model selects the LEO satellite with minimum propagation delay. Mathematically, the propagation delay is expressed as

$$T_{prop} = \frac{d_{ul}}{c}, \quad (2)$$

where d_{ul} is the distance between the UE and LEO satellite and c is the speed of light. The d_{ul} is expressed by $d_{ul} = \sqrt{(x_l^t - x_u)^2 + (y_l^t - y_u)^2 + (z_l^t - z_u)^2}$. UEs are assumed to be stationary at a location and LEO satellite positions are estimated according to the orbital model. The locations of UE and LEO satellite are represented using 3-D Cartesian coordinate system as mentioned in Sections 3.3 and 3.4. The two-way propagation delay is calculated in Section 6.1 by considering: (i) request forwarding distance from UE to LEO satellite and (ii) content forwarding distance from LEO satellite to UE.

4.2 Transmission Model

The transmission model provides a brief idea about the transmission of UE requests and the contents communicated between UE and LEO satellites. The round-trip transmission delay is calculated by considering the uplink and downlink transmission delays. The uplink transmission delay is the time required to transmit the content request message from UE to the LEO satellite. The downlink transmission time is the time required for transmitting the content from the LEO satellite to the UE. From Section 3.4, UE requests the LEO satellite for the content of size c_u bytes. Within the visibility window of LEO, the uplink physical data-rate from the UE to the LEO satellite is assume to be R_{ul}^{up}. The uplink transmission delay is calculated as

$$T_{trans}^{up} = \frac{c_u}{R_{ul}^{up}}. \quad (3)$$

The downlink transmission delay is the time taken for transmitting the content from the LEO satellite to the UE. In our SBHS, the size of the content is modeled using Normal distribution. When a UE requests content from a LEO satellite, the content is retrieved and the size of content P_u (in bytes) is assumed to be drawn from the Normal distribution. Within the visibility window of LEO, the downlink physical data-rate from the LEO satellite to the UE is assumed to be R_{lu}^{down}. The downlink transmission delay is calculated as

$$T_{trans}^{down} = \frac{P_u}{R_{lu}^{down}}. \quad (4)$$

Therefore, the total transmission time is calculated as

$$T_{trans} = T_{trans}^{up} + T_{trans}^{down}. \quad (5)$$

We note that power is expended in transmissions at the UE as well as the LEO satellite. We assume that the transmission power expended at the LEO satellite is constant across different computation schemes. Thus, we do not consider the transmit power expenditure optimization in our model.

4.3 Computation Model

For the the computation model in SBHS, we propose a B+ tree-based LEO satellite database. Using the B+ tree framework, we estimate the computational delay (T_{comp}) and computational energy consumption (ϵ_s) at the LEO satellite. The computational delay is the time required to search and retrieve the requested content in the database. The computational energy consumption is the energy consumed during the computational operation in the LEO satellite-based server.

4.3.1 B+ tree

The LEO satellite-based database follows the B+ tree framework. B+ tree is a balanced multiway search tree and is a popular data structure used for efficient retrieval of data in the filesystems [31]. The nodes in a B+ tree store values/information called keys. In Fig. 3, the nodes $1, 2, \ldots, m-1$ form the keys of the proposed B+ tree. In SBHS, the keys in the B+ tree form the title of the content in the LEO satellite database. The keys are used as the lookup logic for the contents in the database. The internal nodes of the B+ tree contain the key in a specific order. Every adjacent

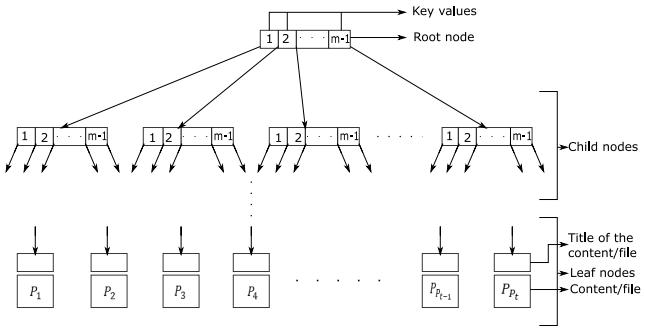


Fig. 3: Data structure of an m-order B+ tree.

key has a sub-tree with the same or different key. The leaf node represents a unique key with a data pointer to the content in the LEO database. An example of m-order B+ tree is illustrated in Fig. 3. The root node of an m-order B+ tree has children between 2 and m keys. The internal nodes except leaf nodes of the B+ tree contain children between $\lceil \frac{m}{2} \rceil$ and m keys. The leaf node contains the key and data pointer to the content. The search complexity of the B+ tree is described in Section 4.3.2.

4.3.2 Computational Delay

Assume P_t is the total number of contents present in the LEO satellite database. The height of the B+ tree can be estimated as $\log_m P_t$. For a content lookup in the LEO satellite database, a binary search is performed at each node in the B+ tree to find the appropriate key. The selection of key leads to selecting the sub-tree and the previous procedure is repeated until the sub-tree is a leaf node. The worst-case complexity for performing a binary search in a node is $O(\log_2 m)$. Therefore, searching a content in the B+ tree-based LEO database requires $O(\log_2 m \times \log_m P_t)$ searches. If β represents the CPU cycles required to process one bit of content, the number of CPU cycles required to search a content can be estimated between $2 \times \log_m P_t \times \beta$ and $\log_2 m \times \log_m P_t \times \beta$, where $O(2 \times \log_m P_t)$ is the best case search and $O(\log_2 m \times \log_m P_t)$ is the worst case search in a B+ tree. Let β_s represents the number of CPU cycles needed to perform a search operation in a B+ tree for one-bit content and the value of β_s is selected from a discrete uniform distribution between $2 \times \log_m P_t \times \beta$ and $\log_2 m \times \log_m P_t \times \beta$. If the computational capability of a single-core server in the LEO satellite is F_1 GHz, the searching delay for a requested content with c_u bytes request message in the B+ tree is calculated as

$$T_{comp}^{search} = \frac{c_u \times 8 \times \beta_s}{F_1 \times 10^9}. \tag{6}$$

Similarly, assuming β as the number of CPU cycles required to retrieve one bit of content from the B+ tree, the number of CPU cycles required to retrieve P_u bytes of content is $P_u \times \beta \times 8$. Computational delay in retrieving the content is calculated as

$$T_{comp}^{retrieve} = \frac{P_u \times \beta \times 8}{F_1 \times 10^9}. \tag{7}$$

Therefore, total computational delay for one request and retrieval message is calculated as

$$T_{comp} = T_{comp}^{search} + T_{comp}^{retrieve}. \tag{8}$$

4.3.3 Average Service Time to Serve a Request at the LEO Server

Service time at the LEO satellite content-server is defined as the time taken to search and retrieve the requested content from the LEO database, i.e., the computational delay (T_C). Therefore, we estimate the average service time t_s of LEO satellite content-server as the expectation of computational delay (T_{comp}).

$$\begin{aligned}
t_s &= E[T_{comp}] \\
&= E[T_{comp}^{search} + T_{comp}^{retrieve}] \\
&= E[T_{comp}^{search}] + E[T_{comp}^{retrieve}] \\
&= E\left[\frac{c_u \times 8 \times \beta_s}{F_1}\right] + E\left[\frac{P_u \times \beta \times 8}{F_1}\right] \\
&= \frac{8}{F_1}(E[c_u] \times E[\beta_s] + \beta \times E[P_u]) \\
&= \frac{8 \times \beta}{F_1}\left(\sum_{i=1}^{300} \frac{1}{300} \times i \right. \\
&\times \left. \sum_{j=2\log_m P_t}^{\log_2 m \times \log_m P_t} \frac{1}{\log_2 m \times \log_m P_t - 2\log_m P_t + 1} \right. \\
&\left. \times j + E[P_u] \right).
\end{aligned} \tag{9}$$

4.3.4 Computational Energy Consumption

Similar to Section 4.3.2, the computational energy consumption depends on the number of CPU cycles needed to search and retrieve a request from the B+ tree. Assume e represents the computational energy consumption per CPU cycle. The computational energy consumption ϵ_s required to handle a request at the LEO server is estimated as

$$\epsilon_s = c_u \times 8 \times \beta_s \times e + P_u \times \beta \times 8 \times e. \tag{10}$$

4.3.5 Average Computational Energy Consumption to Serve a Request at the LEO Server

Similar to Section 4.3.3, the average computational energy consumption depends on the number of CPU cycles required to serve a request. Assuming that e represents the computational energy consumption per CPU cycle and ϵ_s is the energy required to serve a request, the average computational energy consumption is estimated as

$$\begin{aligned}
E[\epsilon_s] &= E[c_u \times 8 \times \beta_s \times e] + E[P_u \times \beta \times 8 \times e] \\
&= 8 \times e \times (E[c_u] \times E[\beta_s] + \beta \times E[P_u]) \\
&= 8 \times \beta \times e \times \left(\sum_{i=1}^{300} \frac{1}{300} \times i \right. \\
&\times \left. \sum_{j=2\log_m P_t}^{\log_2 m \times \log_m P_t} \frac{1}{\log_2 m \times \log_m P_t - 2\log_m P_t + 1} \right. \\
&\left. \times j + E[P_u] \right).
\end{aligned} \tag{11}$$

4.4 Queuing Model

The queuing system in a LEO satellite is divided into two categories depending on whether the LEO satellite possesses a single-core or multi-core processor. Section 3.3 discusses different LEO satellite server types. Fig. 4 shows the schematic of a multi-core satellite server. The requests arriving from the ground UEs are stored in the satellite's *request receive buffer* queue. The *request receive buffer* follows FIFO queue discipline. We assume the maximum number of server cores in a LEO satellite as k and the maximum length of *request receive buffer* as q. Therefore, from the *request receive buffer*, the earliest requests are moved to the free server cores. The job of the server cores is to find the requested content in the database and transmit the content to the UE.

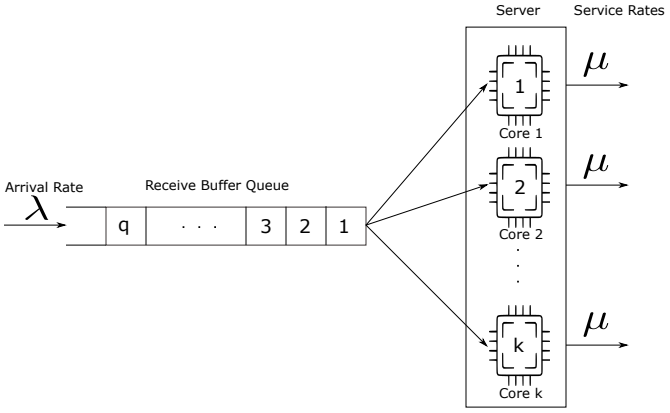


Fig. 4: A schematic of the multi-core LEO satellite server.

Queuing delay is defined as the average time (T_{queue}) a request remains in *request receive buffer* queue before getting served. We assume that U is the number of UE requests to a LEO satellite and λ_u is the average request rate for the content. Therefore, the request arrival rate at the LEO satellite is calculated as $\lambda = \sum_{i=1}^{U} \lambda_i$. To maintain uniformity among LEO satellites, we assume the request arrival rate at all LEO satellites as λ. The average service time (t_s) to serve a request using a single-core processor in the LEO server is estimated in Eqn. (9) and the average service rate is calculated as $\mu = \frac{1}{t_s}$. For a LEO server with a single-core processor, the average queue utilization of the LEO satellite queuing system is calculated as $\rho = \frac{\lambda}{\mu}$. According to Little's formula, the average number of requests present in the LEO satellite and the waiting in the LEO satellite *request receive buffer* queue are calculated as $L_{sys} = \frac{\lambda}{\mu - \lambda}$ and $L_{queue} = \rho L$, respectively. Similarly, the average time spent by a request waiting in the LEO satellite system and the LEO satellite *request receive buffer* queue are calculated as $W_{sys} = \frac{1}{\mu - \lambda}$ and $W_{queue} = \rho W$, respectively.

5 Dynamic control of processing cores using MDP in the multi-core Queuing System of SBHS

The length of the satellite *request receive buffer* queue depends on the mean request arrival rate (λ) and mean service rate (μ) at the LEO satellite. One of the objectives of our SBHS approach is to serve the maximum number of requests in order to ensure the availability of LEO satellites for the UE. To achieve the objective, we need to ensure that $\lambda \leq \mu$. If $\lambda > \mu$, the *request receive buffer* queue overflows, thereby leading to an unstable queuing system. If the queue size is finite as in our work, the requests are dropped when the queue is full. To reduce the possibility of queue overflow, we introduce a multi-core server in our LEO satellite system. We assume that there are k independent processor cores that are available in the LEO satellite. If all the available cores are used at all times, then the delay and packet drop probability would be reduced at the expense of larger power expenditure. Therefore, we propose a dynamic control strategy that controls the number of processor cores used to serve requests as a function of the number of requests in the buffer. The control strategy is obtained by formulating a Markov decision problem for an approximate model of the LEO service system.

We also note that the transmissions from the LEO satellite could be queued up in a transmit queue. Since the scheduling of this queue is handled by the LEO satellite servers, we approximate the transmit queuing delay as a fixed constant and do not consider it for optimization.

5.1 Assumptions in our MDP Model

In our SBHS, we formulate MDP with finite states and finite actions. The state space is defined by Eqn. (12) describing the satellite's *request receive buffer* queue length. Fig. 5 shows the queuing system in a LEO satellite of the SBHS. The maximum queue length q represents the maximum number of states in the system. An action is defined by the number of server cores used by the LEO satellite at a given time. A LEO satellite server possesses maximum k number of server cores and thus, the action space contains k actions. Eqn. (13) represents the action space of the MDP model. Action a_1 represents the service rate of a single-core processor. Similarly, action a_k represents combined service rates of k single-server cores. In the equation of the transition probability, the service rate of a k-core processor is assumed to be k times the service rate of a single-core processor. However, for simulation, instead of combined service rate, our SBHS considers independent server cores to serve the requests. Thus, it can be concluded that the solution to the proposed MDP model is a near-optimal solution to the SBHS system model.

$$S = \{0, 1, 2, \ldots, q\}. \tag{12}$$

$$A = \{a_1, a_2, a_3, \ldots, a_k\}. \tag{13}$$

$$a_i = \mu_i = \sum_{x=1}^{i} \mu, \forall a_i \in A. \tag{14}$$

5.1.1 Transition Probability Matrix Formulation

Fig. 5 represents the continuous Markov queuing system for the LEO satellite in SBHS. The continuous queuing system is modeled as a continuous-time Markov chain. For our MDP model, continuous-time Markov chain parameters such as arrival rate and service rate are converted to discrete Markov chain parameters through uniformization [32]. In the SBHS, the arrival rate (λ) and service rate (μ) represent the number of requests that arrive and depart per second,

$$TP[a_i ::] = \begin{bmatrix} 1-(1-\mu_{si})\lambda_s & (1-\mu_{si})\lambda_s & 0 & . & . & 0 \\ (1-\lambda_s)\mu_{si} & \lambda_s\mu_{si}+(1-\lambda_s)(1-\mu_{si}) & (1-\mu_{si})\lambda_s & 0 & . & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & . & . & . & (1-\lambda_s)\mu_{si} & 1-(1-\lambda_s)\mu_{si} \end{bmatrix}. \quad (15)$$

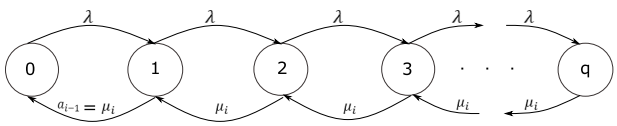


Fig. 5: State transition diagram of the multi-core server.

respectively. The arrival rate follows the Poisson arrival process with mean λ and the service rate follows a general distribution with mean μ. For the SBHS system model, we assume $\lambda \gg 1$ and $\mu \gg 1$. Such high rates of arrival and service increase the complexity while designing the transition probabilities. For example, the probability of transition from current state to the next state is infeasible to predict within one second if the number of requests arriving and departing are very high. Therefore, in the uniformization method, instead of the one second time-slot, we assume the time-slot to be one microsecond. The assumption is to make the arrival rate and service rate less than unity. Eqn. (16) and Eqn. (17) show the conversion of continuous arrival rate and service time to discrete ones by uniformization.

$$\lambda_s = \frac{\lambda \times U}{10^6}, \quad (16)$$

where U is the total number of UEs considered for the MDP model.

$$\mu_{si} = \frac{i}{t_s}, \quad (17)$$

where μ_{si} is the service rate at the LEO satellite with i-core server and t_s is the average service time from Eqn. (9).

The Transition Probability (TP) can be estimated for the MDP model as follows:

$$TP = [tp_{hij}]_{k \times q \times q}. \quad (18)$$

The matrix TP is of dimension $k \times q \times q$ and representing the number of actions, present states, and future states. From TP, we perceive k times of $q \times q$ 2-D matrix. A row in the $q \times q$ 2-D matrix indicates the present states, and a column indicates the next state. Each element in the TP represents the probability of moving from the present state to the next state by taking an action. Eqn. (15) shows the detailed transition probabilities for all actions in the action space A where $1 \leq i \leq k$.

Eqn. (15) represents the transition probabilities for taking action a_i where $1 \leq i \leq k$. The transition probabilities depend on the request arrivals and services. There exist three possibilities in the next time-slot: (i) the state remains in the same state in the next time-slot upon two events: (a) no arrival and service of requests and (b) one request arrival and one service, (ii) the state moves to next state in the next time-slot only in the case of one arrival and no service, and (iii) the state moves to the previous state in the next time-slot only in the case of no arrival and one service. In the first state of the queue, i.e., at state 0, there exist two possibilities in the next time-slot, states 0 and 1. In the last state q, there exist two possibilities in the next time-slot, states q and $(q-1)$. For the remaining states from 1 to $(q-1)$, all three possibilities exist in the next time-slot.

5.1.2 Reward Function Formulation

In our SBHS multi-core server, we concentrate on minimizing queuing delay and computational energy consumption in the LEO satellite server. Therefore, the reward function in the proposed MDP model includes parameters for both delay and computational energy consumption. The minimization of queuing delay in the queuing system leads to the use of more server cores to finish the tasks early. However, more use of server cores leads to more computational energy consumption. Therefore, we introduce a weight α, motivated by Lagrange Multipliers, to determine whether the policy is weighted towards minimizing queuing delay or the conservation of energy. The weight $\alpha = 1$ indicates the optimal policy is weighted towards minimizing queuing delay and $\alpha = 0$ shows the optimal policy is weighted towards minimizing the computational energy consumption in the server.

$$R = [r_{ij}]_{q \times k}, \quad (19)$$

where r_{ij} represents the immediate reward for state i by taking action j.

Eqn. (20) represents the reward matrix which is defined for each action in the MDP model. The immediate reward is a function of two weights W^1_{si} and W^2_{si}. In Eqn. (20), $W^1_{si} = c_1 \times \mu_{si}$ and $W^2_{si} = c_2 \times E[\epsilon_{si}]$. W^1_{si} indicates the reward with service rate μ_{si} and c_1 is a constant. W^2_{si} indicates the reward with average computational energy consumption due to the use of action a_i and c_2 is a constant.

$$R[:a_i] = \begin{bmatrix} \alpha \times (0 \times W^1_{si}) - (1-\alpha) \times W^2_{si} \\ \alpha \times (1 \times W^1_{si}) - (1-\alpha) \times W^2_{si} \\ \alpha \times (2 \times W^1_{si}) - (1-\alpha) \times W^2_{si} \\ \vdots \\ \alpha \times (s \times W^1_{si}) - (1-\alpha) \times W^2_{si} \end{bmatrix}, \quad (20)$$

where $1 \leq i \leq k$.

5.2 Policy Evaluation and Policy Improvement

Using the policy iteration method [33], the near-optimal policy is generated and is used by LEO satellites to control the number of server cores used. To formulate the MDP model for SBHS, Eqn. (12), (13), (15), and (20) represent the state-space S, action-space A, reward function R, and the transition probability matrix TP, respectively. The policy iteration is described in Algorithm 1. An arbitrary policy (Line 2) is used to initialize the policy iteration procedure. Policy $\pi : S \rightarrow A$ is defined for each state $s \in S$ and an

arbitrary policy $\pi(s)$ is selected from action space A. The arbitrary policy is known as the baseline policy. In Line 9, the expected reward is calculated using the baseline policy and reward function R using Eqn. (21). Further, the baseline policy is improved with a policy improvement method in Line 15 using Eqn. (22). In Eqn. (21) and Eqn. (22), γ is the discounted factor. The process of improving policy continues till $\theta > \Theta$ (Line 11). In Lines 16 and 17, the resultant value function is the near-optimal value function $V^*(s)$, and the resultant policy is the near-optimal policy $\pi^*(s)$ for the SBHS system. The near-optimal value function and near-optimal policy can be estimated as follows:

$$V(s) \leftarrow \sum_{s' \in S} TP(\pi(s), s, s')[R(s, \pi(s)) + \gamma V(s')], \forall s \in S \tag{21}$$

and

$$\pi(s) \leftarrow \arg\max_{A} \sum_{s' \in S} TP(a, s, s')[R(s, a) + \gamma V(s')]. \tag{22}$$

Algorithm 1: Estimating optimal policy for LEO satellite *request receive buffer* queue.

Data: S, A, R, TP
Result: Optimal Policy ($\pi^*(s)$)
1 $V(s) = 0, \forall s \in S$
2 Initialize $\pi(s) \in A(s)$ arbitrarily $\forall s \in S$
3 Initialize θ (A small positive number)
4 Initialize $\Theta \leftarrow 0$
5 **while** *(True)* **do**
6 **# Policy Evaluation**
7 **for** $(s \in S)$ **do**
8 $v(s) \leftarrow V(s)$
9 $V(s) \leftarrow \sum_{s' \in S} TP(\pi(s), s, s')[R(s, \pi(s)) + \gamma V(s')]$
10 $\Theta \leftarrow |v(s) - V(s)|$
11 **if** $\theta > \Theta$ **then**
12 Break
13 **else**
14 **# Policy Improvement**
15 $\pi(s) \leftarrow \arg\max_A \sum_{s' \in S} TP(a, s, s')[R(s, a) + \gamma V(s')]$
16 $V^*(s) \leftarrow V(s)$
17 $\pi^*(s) \leftarrow \pi(s)$

6 WIKIPEDIA: A CASE-STUDY OF SBHS

In order to study the efficacy of our SBHS, we use the English Wikipedia scenario where the content is hosted in LEO satellites. That is, in our case-study, we assume the entire English Wikipedia content is hosted in LEO and GEO satellites in addition to the terrestrial content-servers. In the existing Wikipedia data-centers, the English Wikipedia database is included along with other databases of the Wikimedia [10] such as Wikicommons, DE wiki, and other language Wikis. However, we considered only the English Wikipedia database as the content-server for our case-study.

The workflow of searching a page in Wikipedia is carried out in different stages. UEs send the Wikipedia page/article requests to the nearest LEO satellite. The keyword used to search the page in the Wikipedia *search box* is called the *search string* [34]. Wikipedia database searches the content from the search string throughout the LEO satellite server and returns the appropriate Wikipedia page/article to the UE.

6.1 Propagation Model

We consider the round trip propagation delay for end-to-end delay estimation (Eqn. 1) using the case-study of English Wikipedia. Assuming the distance between UE and the nearest LEO satellite for a request message is d_{ul} and the distance of the requested page from the LEO satellite to the UE is d_{lu}. In case the LEO satellite serves the UE within the same time-slot, $d_{ul} = d_{lu}$, the propagation delay (T_{prop}) is calculated as

$$T_p = \frac{d_{ul} + d_{lu}}{c}, \tag{23}$$

where d_{ul} is same as in Eqn. (2) and $d_{lu} = \sqrt{(x_l^{t_2} - x_u)^2 + (y_l^{t_2} - y_u)^2 + (z_l^{t_2} - z_u)^2}$. The LEO satellite position at time-slot t_2 is represented by ($x_s^{t_2}$, $y_l^{t_2}$, $z_s^{t_2}$).

6.2 Transmission Model

Similar to the transmission model in Section 4.2, in Wikipedia, the transmission delay is divided into two parts. The first part is the calculation of the uplink transmission time, i.e., the transmission of *search string* in the Wikipedia *search box* from the UE to the LEO satellite. The *search string* contains a maximum of 300 characters [34], i.e., 300 bytes of the request message. Therefore, each time a UE requests a Wikipedia page name in the *search box*, the size of the *search string* varies from 1–300 bytes. Assume c_u bytes is the size of *search string* which selected from discrete uniform distribution between 1 and 300 bytes. With an uplink physical data-rate of R_{ul}^{up} from UE to LEO satellite, the uplink transmission time is calculated using Eqn. (3).

The second part of transmission delay constitutes the downlink transmission time of the requested Wikipedia page from the LEO satellite to the UE. The Wikipedia database contains approximately 6,170,000 English articles [35] [36]. As of year 2020, the size of English Wikipedia text database is 30.619 GB [37]. Therefore, the average text size of each article is calculated as 4.9 KB. We assume the size of English articles is modeled by Normal distribution with a mean of 4.9 KB and a standard deviation of 4 KB. Therefore, each time a user requests a Wikipedia page, a page size of P_u bytes is drawn from the Normal distribution with a mean of 4.9 KB and a standard deviation of 4 KB. Assuming the downlink physical data-rate from LEO satellite to UE is R_{lu}^{down}, the downlink transmission time is calculated using Eqn. (4). Eqn. (5) calculates the total transmission time.

6.3 Computation Model

For modelling computation, we assume that the English Wikipedia database follows a B+ tree structure with 1000

(m) keys. The leaf nodes of the B+ tree contain the unique keys and the corresponding data pointers to the Wikipedia page present in the LEO satellite database. The English Wikipedia contains 6,710,000 pages in the LEO satellite and is represented as P_t. The height of the B+ tree is calculated as $\log_{1000} P_t$ = 2.58. Keeping the assumption in Section 4.3, the computational delay and computational energy consumption to search and retrieve a Wikipedia page in the B+ tree are calculated using Eqn. (8) and Eqn. (10), respectively.

6.4 Queuing Model

The queuing delay in the LEO Wikipedia server is the average duration (T_{queue}) a page request from the UE waits in the LEO satellite *request receive buffer* to get served. The assumption of the case-study is same as the queuing model described in Section 4.4.

7 Simulation Setup

In this section, we describe our simulation setup designed for evaluating the performance of the SBHS. We consider the Iridium-NEXT satellite constellation for the SBHS LEO satellite system. We use the Python library *Skyfield* [38] to track the live locations of the Iridium-NEXT satellites. The mathematical models discussed in Section 4 are implemented using Python, and the MDPToolbox library [39] is used to realize Algorithm 1. The network parameters used in the simulation are described in TABLE 2.

TABLE 2: Network parameters.

Parameter	Value
U	624
L	75
c_u	1 – 300 bytes
F_1	500 MHz, 1 GHz, 5 GHz
k	8
c	300,000 km/sec
r_{ul}^{up}	100 MBps
r_{lu}^{down}	200 MBps
P_t	6,170,000
m	1000
β	10 CPU cycle/bit
e	4×10^{-9} J/cycle
q	100
c_1, c_2	1
Simulation time	90 minutes

We use the Iridium-NEXT constellation consisting of 75 satellites revolving around the Earth in polar orbits of altitude 780 km and an inclination of $86.4°$. In addition, 624 UEs are distributed spatially across the surface of the Earth in different countries. Each satellite in the Iridium-NEXT constellation has an orbital period of approximately 100 minutes. The uplink and downlink physical data-rates between UE and LEO satellites are set to 100 MBps and 200 MBps, respectively. The maximum number of server cores in a LEO satellite is considered to be eight. The computational capacity of each server core is considered in terms of frequency and varies among 500 MHz, 1 GHz, and 5 GHz. In the B+ tree, the number of key pairs is set to 1000. The number of CPU cycles needed for one bit of operation (β) is 10 and the energy consumption per one CPU cycle (e) is 4×10^{-9} J. The total number of content pages present in a LEO satellite server is assumed to be the number of Wikipedia web-pages present in the English Wikipedia database, i.e., P_t. The simulation time is set to 90 minutes, which is approximately equal to the orbital period of the Iridium-NEXT satellite constellation. The remaining simulation settings were mentioned in Section 6.

Different request generation rates of UEs are taken into consideration for our simulation environment depending on the countries considered. For our work, we considered real-time traffic of Wikipedia usage for November 2020 [40]. The first column of TABLE 3 shows seven countries having the highest English Wikipedia usage. The second column ($R_{t_{country}}$/month (M)) of TABLE 3 shows the number of requests made in terms of millions per month to LEO Wikipedia server by the UEs from different countries [41]. The third column ($R_{t_{country}}$/sec) represents the number of requests made in millions per second to the LEO satellite Wikipedia server by the UEs. We considered 14 active hours per day for the UEs while calculating $R_{t_{country}}$/sec. The fourth column ($P_{E_{country}}$ (M)) represents the total number of English speaking UEs in seven countries [42]. In order to estimate the number of UEs for simulation, we divide the total number of UEs of all countries by one million which is represented by the fifth column ($P_{t_{country}}$/1M). We select a 10×10 km area for each country to deploy the UEs at random locations. The sixth column represents the inter-arrival rate (per second) of message generation for a UE derived by the ratio between the fourth and third columns. The seventh column represents the request generation rates (λ_u) for a UE per second in all seven countries.

8 Performance Evaluation

In this section, we evaluate the performance of our proposed SBHS through four different schemes. The schemes depend on three factors: (i) LEO satellite server with the single-core or multi-core process, (ii) text-based or multimedia-based data traffic types, and (iii) infinite or finite queue length. Further, two benchmark schemes are defined for comparison with the proposed schemes. The schemes we considered include:

1) **Text-based Request with Traditional Scheme (TRTS):** The scheme is similar to the traditional web-based communication where the content resides in the terrestrial-based server. The LEO satellites act as relay nodes between the UE and terrestrial content-servers. The UE requests only text-based articles/webpages to the Wikipedia servers located in the terrestrial content-servers. Each terrestrial content-server consists of eight cores with a queue length of 100. We consider five Wikipedia data-centers for terrestrial content-servers located at Ashburn, Carrollton, San Francisco, Haarlem, and Singapore [10]. The location and request rates of UEs are considered according to TABLE 3. TRTS scheme acts as one of the benchmark schemes for comparison with our proposed SBHS approach.
2) **Multimedia-based Request with Traditional Scheme (MRTS):** The scheme is similar to TRTS, where UE requests multimedia-based articles/webpages with both images and texts from the

TABLE 3: English Wikipedia traffic traces, UE distributions, and request rates among different countries.

Country Names	$R_{t_{country}}$/month (M)	$R_{t_{country}}$/sec	$P_{E_{country}}$ (M)	$P_{E_{country}}$/1M	Inter Arrival Rates/UE	Request Rates/UE
USA	3000	1985	283	283	0.142	7.1
UK	918	608	60	60	0.098	10.33
India	667	442	125	125	0.28	3.53
Canada	407	269	30	30	0.11	8.96
Australia	237	157	17	17	0.108	9.23
Germany	162	108	45	45	0.41	2.4
Philippines	154	102	64	64	0.62	1.59

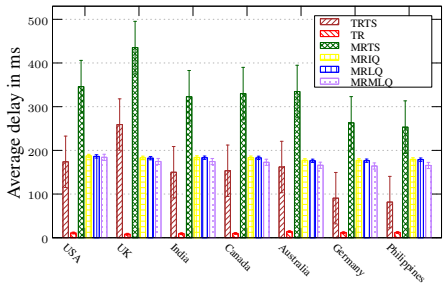


Fig. 6: 500 MHz single-core capacity.

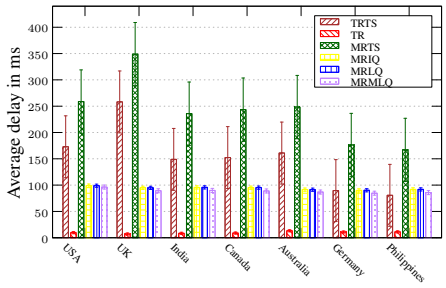


Fig. 7: 1 GHz single-core capacity.

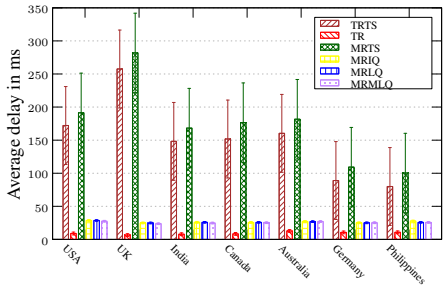


Fig. 8: 5 GHz single-core capacity.

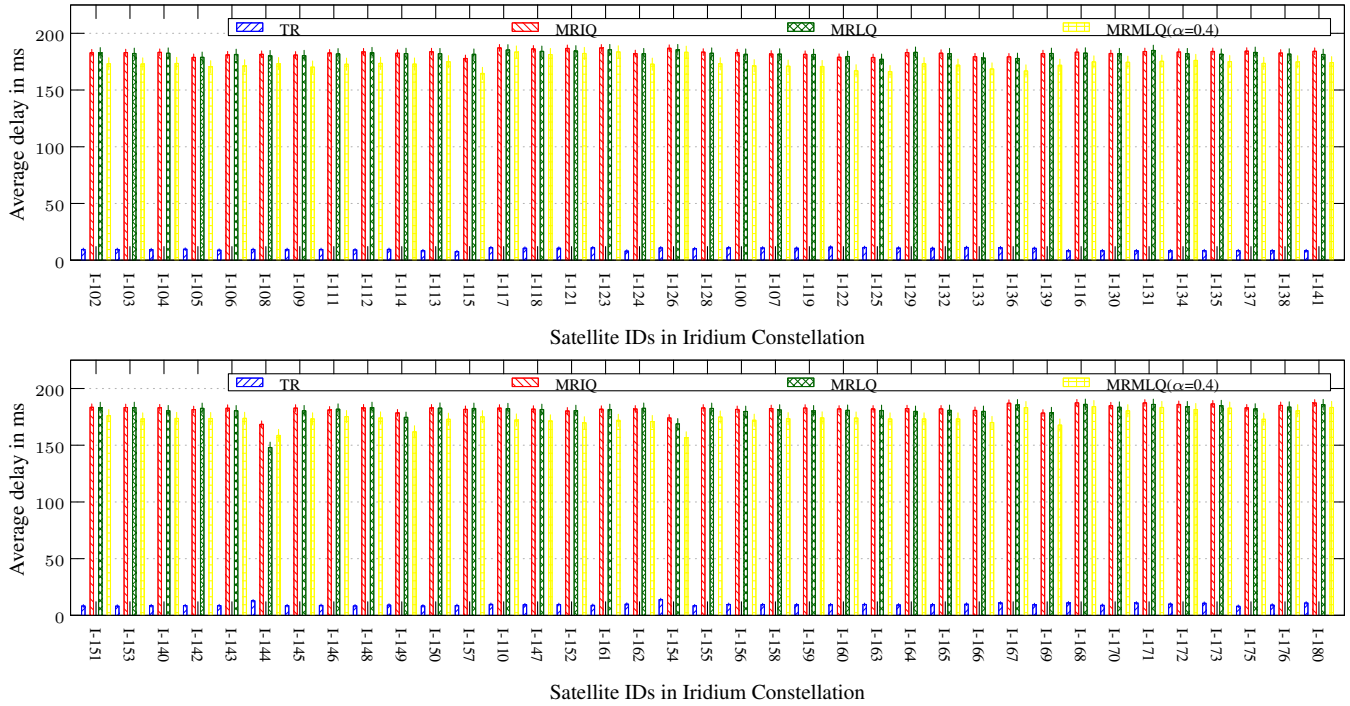


Fig. 9: Delay experienced by LEO satellites in Iridium-NEXT constellation with 500 MHz single-core capacity.

Wikipedia servers located in the terrestrial content-servers. This scheme acts as the second benchmark scheme for comparison with our proposed SBHS schemes.

3) **Text-based Request (TR):** The scheme is based on the SBHS approach where the UE requests only text-based Wikipedia articles/webpages to the Wikipedia servers located in the LEO satellites. The LEO satellite server consists of single-core processor with a queue length of 100. We do not consider the multi-core process for the TR scheme due to the very low service time for the text-based articles/webpages. Such a very low service time leads to approximately zero queue length. Thus, single-core is sufficient to serve all the requests of UEs.
4) **Multimedia-based Request with Infinite Queue length (MRIQ):** The scheme is based on our SBHS approach where UE requests multimedia-based Wikipedia articles/webpages involving both images and texts to the Wikipedia server located in the LEO satellite. In this category, the LEO satellite consists of single-core processor with infinite queue length.
5) **Multimedia-based Request with Limited Queue length (MRLQ):** The scheme is similar to MRIQ. However, in MRLQ the LEO satellite consists of single-core processor with a finite queue length of 100.
6) **Multimedia-based Request with Multi-core processor and Limited Queue length (MRMLQ):** Here, the scheme based on SBHS approach where UE requests multimedia-based Wikipedia arti-

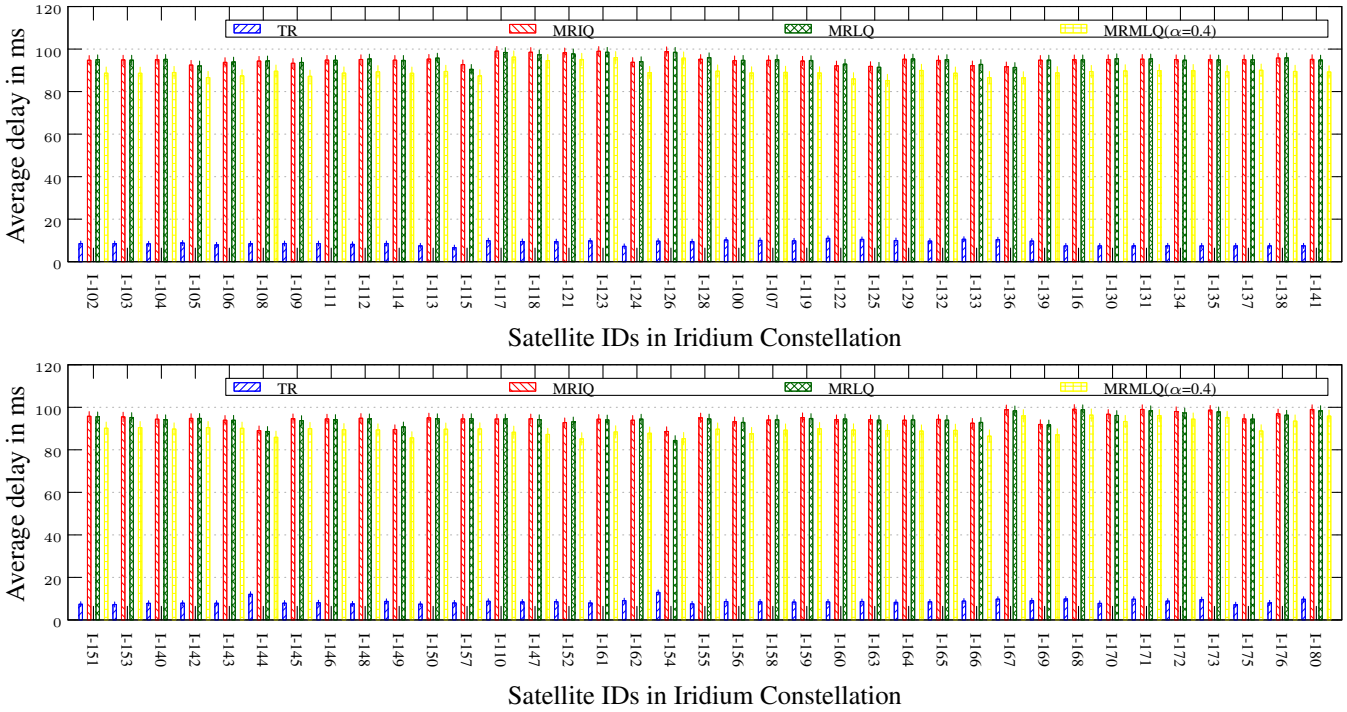


Fig. 10: Delay experienced by LEO satellites in Iridium-NEXT constellation with 1 GHz single-core capacity.

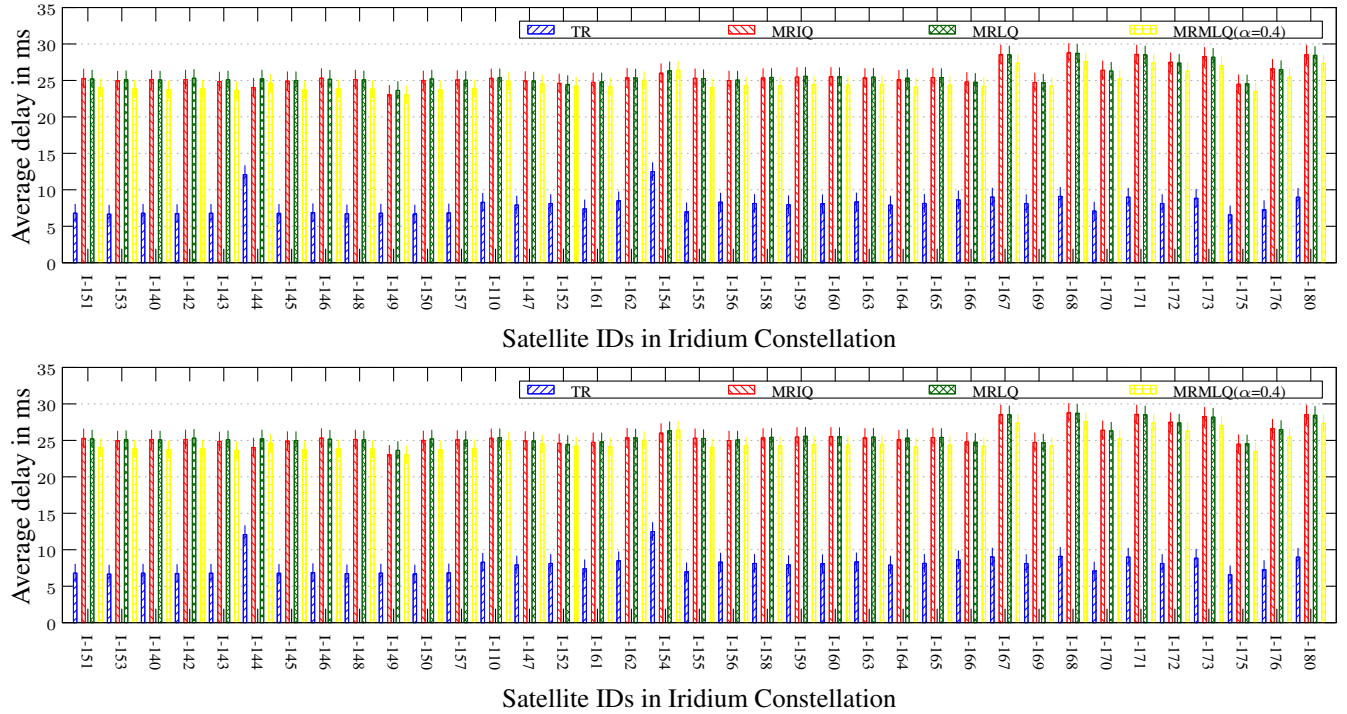


Fig. 11: Delay experienced by LEO satellites in Iridium-NEXT constellation with 5 GHz single-core capacity.

cles/webpages involving both images and texts to the Wikipedia server located in the LEO satellite. In this category, the LEO satellite has a multi-core processor with a queue length of 100 and follows the MDP optimization algorithm.

The average end-to-end delay experienced by seven countries using six different schemes is shown in Figures 6-8. Compared to the two traditional schemes TRTS and MRTS, our proposed schemes for the SBHS approach provides a very low average end-to-end delay. We simulate the six schemes with three different single-core server capacities of 500 MHz, 1 GHz, and 5 GHz for the Wikipedia case-study. Figures 6-8 show that for all single-core server capacities, the lowest average end-to-end delay is experienced by the TR-based scheme. The computational delay depends on the LEO server's computational capacity and the requested web-page's size. Therefore, the computational and transmission delays are significantly reduced due to the transmission of text-based webpages in the TR scheme. The reduction in computational delay reduces the queue length of LEO

TABLE 4: Total number of requests served and dropped at LEO servers for Wikipedia case-study.

Single-core capacity	500 MHz		1 GHz		5 GHz	
Schemes	**Request served**	**Request dropped**	**Request served**	**Request dropped**	**Request served**	**Request dropped**
TR	16448021	0	16466261	0	16457256	0
MRIQ	16468530	0	16483118	0	16508942	0
MRLQ	8563365	7871711	11380478	5055809	16480607	76
MRMLQ ($\alpha = 0.1$)	16490189	11	16429228	0	16449855	0
MRMLQ ($\alpha = 0.2$)	16467673	40	16454558	0	16483735	0
MRMLQ ($\alpha = 0.3$)	16474783	0	16486467	0	16491159	0
MRMLQ ($\alpha = 0.4$)	16448546	0	16486586	0	16426419	0

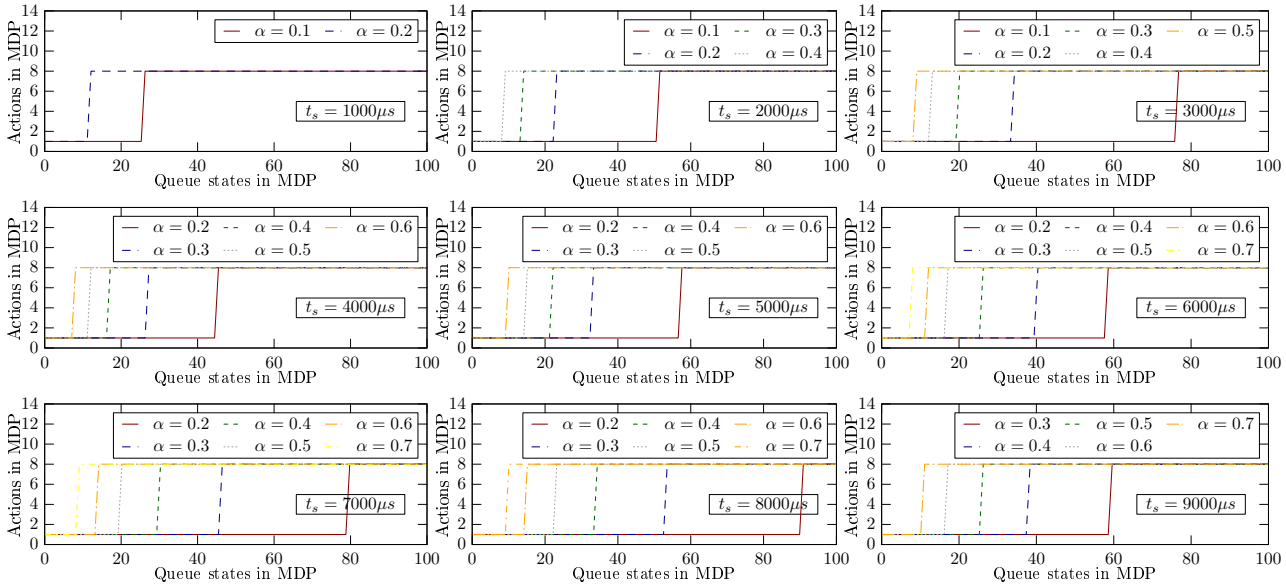


Fig. 12: Optimal policies for different values of service time (t_s).

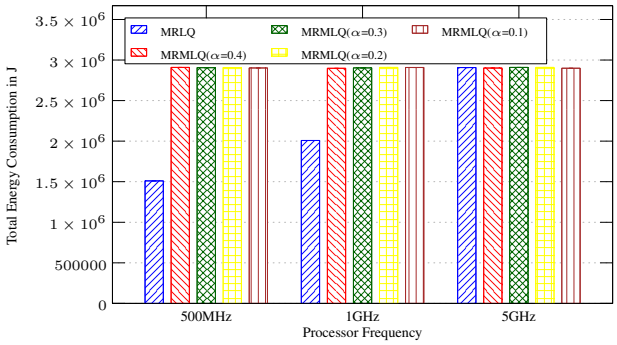


Fig. 13: Total computational energy consumption in all LEO satellites.

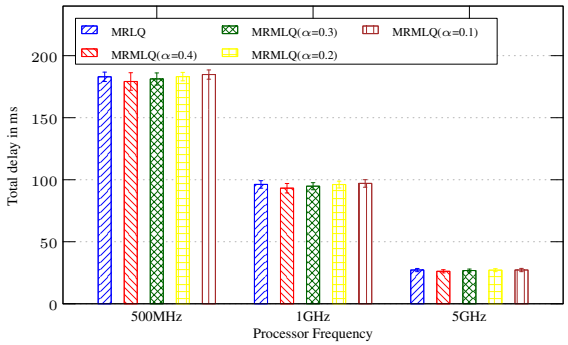


Fig. 14: Average end-to-end delay in SBHS approach.

satellites and results in less queuing delay. The reduction in computational, transmission, and queuing delays leads to minimum average end-to-end delay. Using the text-based requests, the UK experience the lowest average end-to-end delay of 6.56 ms (TR scheme in Figure 8) where the UK experienced the highest delay of 259.12 ms (TRTS scheme in Figure 6). In the TRTS and MRTS schemes, the long path between UEs and terrestrial content-servers results in very high propagation delays and subsequently very high average end-to-end delay.

The average size of text-based web-pages is 4.9 KB whereas the average size of multimedia-based web-pages is 500 KB. Therefore, in the multimedia-based request schemes, the transmission and computational delay increases due to the size of multimedia-based web-pages. MRIQ and MRLQ schemes are based on single-core LEO servers. From Figures 6-8, we observe that the MRLQ schemes provide lesser average end-to-end delay compared to MRIQ. Due to the infinite queue length in the MRIQ scheme, the average queuing delay is higher. On the other hand, in the MRLQ scheme, the requests are dropped due to queue overflow, thereby leading to lesser average queuing delay.

Among the four schemes that are based on multimedia-based requests, MRMLQ with 5 GHz single-core capacity provides the minimum average end-to-end delay. Further, the MRMLQ scheme is categorized into four according to different values of α. In Figures 6-8, we consider only the MRMLQ scheme with α = 0.4 which provides minimum delay. In MRMLQ, due to the parallel processing of incoming UE requests, the queuing delay reduces substantially in LEO servers. The use of eight 5 GHz parallel cores leads to a very low computational delay in the LEO server. We achieved a minimum end-to-end delay of 23.66 ms for the

UK (Figure 8) using the MRMLQ scheme with $\alpha = 0.4$. The maximum end-to-end delay of 435.23 ms is experienced by the MRTS scheme for the UK (Figure 6). The average end-to-end delays for other countries are shown in Figures 6-8.

Figures 9-11 represent the average end-to-end delays of all LEO satellite servers according to 500 MHz, 1 GHz, and 5 GHz single-core server capacities. In the traditional schemes, the LEO satellites act as relay nodes between UEs and terrestrial content-servers. Therefore, the delays experienced by the individual LEO satellites for TRTS and MRTS schemes are not included in the figures. From the simulation, we observed that a LEO satellite serves a maximum of four countries within the simulation period while the I-120 LEO satellite does not serve any country. For Figures 9-11, we consider the MRMLQ scheme with $\alpha = 0.4$. In Figure 11, minimum average end-to-end delays of 6.095 ms and 22.718 ms are experienced by the I-115 LEO satellite for the TR scheme and MRMLQ scheme, respectively.

Figure 12 represents the optimal policies for different values of service time (t_s) with a constant value of α. The graph behaves as a step function in each figure where the optimal actions are actions shift directly from action one (a_1) to action eight (a_8) after a specific queue length. In the reward function design, the average computational energy consumption and service rate are added linearly (Section 5.1.2). The linearity that leads to the optimal policies as shown in Figure 12. Further, the optimal policy changes depending on the value of α. For some higher values of α, the optimal policies are not realizable. For example, the systems are not realizable if the optimal action involves the utilization of eight cores (i.e., the action a_8) for a queue length below eight. We omitted optimal policies for such high values of α. Similarly, we do not consider the policies with low values of α that provide only one action (a_1) irrespective of the queue length.

Figure 13 represents the total computational energy consumption in Joules by the SBHS network for different single-core server capacities within the simulation period. We elaborate five methods for different single-core server capacities (500 MHz, 1 GHz, and 5 GHz), i.e., MRLQ and MRMLQ with α values 0.1, 0.2, 0.3, and 0.4. In the Wikipedia case-study, we estimate the maximum service time by a LEO server is 2000 μs. Therefore, we consider the decisions of Figure 12 with t_s = 2000 μs as the optimal policy for our simulation. In Figure 13, the total computational energy consumption of the MRLQ scheme is varied according to the server capacity. The total computational energy consumption of a LEO server is directly proportional to the total request served by the LEO server. In TABLE 4, the number of requests dropped due to queue overflow are 7,871,711 and 5,055,809 for 500 MHz and 1 GHz, respectively. However, the number of requests served by MRLQ (5 GHz single-core capacity) and MRMLQ (α = 0.1, 0.2, 0.3, and 0.4) approximately remain the same. Therefore, in Figure 13, the total computational energy consumption is approximately the same for MRLQ (5 GHz single-core capacity) and MRMLQ (α = 0.1, 0.2, 0.3, and 0.4).

Figure 14 represents the average end-to-end delay in milliseconds by the SBHS network for different single-core server capacities within the simulation period. From Figure 14, the average end-to-end delay varies according to five schemes for a specific single-core server capacity. For all single-core server capacities, the MRMLQ ($\alpha = 0.4$) gives minimum average end-to-end delay. According to the optimal policy for MRMLQ ($\alpha = 0.4$) scheme in Figure 12 with t_s = 2000 μs, the average queue length is minimum compared to other schemes. The minimum queue length leads to minimum queuing delay which subsequently minimizes average end-to-end delay. The combined results from Figure 14, Figure 13, and TABLE 4 show that MRMLQ scheme reduces the queue overflow and optimizes both average end-to-end delay and total computational energy consumption with the help of the MDP model.

9 Conclusion

Satellite communication is considered as an important technology to achieve the universal goal of *connecting the digitally unconnected population* through global coverage. We proposed a Space Based Hosting Service (SBHS) approach to deploy the copy of terrestrial content-servers in space, i.e., in GEO and LEO satellites. Mathematical models are formulated for UE, LEO, and GEO satellites as part of an SBHS system. We introduced a B+ tree data structure in the SBHS to organize and store data to reduce the computational delay and computational energy consumption in the LEO satellite server. An optimization algorithm with the help of Markov Decision Process (MDP) for the LEO satellite queuing system is designed to optimize queuing delay and computational energy consumption. Further, we conducted a case-study of hosting the English Wikipedia server in the Iridium-NEXT satellite constellation using the SBHS. We achieved minimum end-to-end delays of 6.56 ms and 23.66 ms for the United Kingdom (UK) using text-based and multimedia-based traffic, respectively. The simulation results demonstrated the feasibility of hosting a content-server in space and achieving ultra-low latency compared to the traditional satellite-based web services. The use of our SBHS approach to reduce the digital divide around the world is proved by studying the case study of hosting the English Wikipedia server in LEO and GEO satellites and achieving ultra-low latencies for different countries. Future work includes the hosting of multiple services in the LEO satellites where the services can be categorized according to the popularity, latency requirements, and computational resource requirements. Further, inter-LEO communication to achieve the continuity of services in the handoff situations can also considered.

References

[1] L. Pietrewicz, "Coordination in the age of Industry 4.0," *International journal of management science and business administration*, vol. 6, pp. 24–32, 01 2020.

[2] UNESCO, "New report on global broadband access underscores urgent need to reach the half of the world still unconnected," 2019. [Online]. Available: https://en.unesco.org/news/new-report-global-broadband-access-underscores-urgent-need/-reach-half-world-still-unconnected(accessed:02-25-2021)

[3] International Telecommunication Union World Telecommunication/ICT indicators database, "Individuals using the Internet (percentage of population)," 2020. [Online]. Available: https://data.worldbank.org/indicator/IT.NET.USER.ZS(accessed:02-25-2021)

[4] International Telecommunication Union (ITU) publications, "Measuring digital development facts and figures 2019," 2020. [Online]. Available: https://www.itu.int/en/ITU-D/Statistics/Documents/facts/FactsFigures2019.pdf(accessed:02-25-2021)

[5] UNISEF-ITU, "Two thirds of the world's school-age children have no Internet access at home, new UNICEF-ITU report says," 2020. [Online]. Available: https://www.unicef.org/press-releases/two-thirds-worlds-school-age-children-have-no-internet-access/-home-new-unicef-itu(accessed:02-25-2021)

[6] International Telecommunication Union (ITU) publications, "Measuring digital development facts and figures 2020," 2020. [Online]. Available: https://www.itu.int/en/ITU-D/Statistics/Documents/facts/FactsFigures2020.pdf(accessed:02-25-2021)

[7] International Telecommunication Union (ITU) publications, "Measuring digital development facts and figures 2017," 2020. [Online]. Available: https://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2017.pdf(accessed:02-25-2021)

[8] International Telecommunication Union (ITU) publications, "Measuring digital development facts and figures 2016," 2020. [Online]. Available: https://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2016.pdf(accessed:02-25-2021)

[9] International Telecommunication Union (ITU) publications, "Measuring digital development facts and figures 2015," 2020. [Online]. Available: https://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2015.pdf(accessed:02-25-2021)

[10] A Wikimedia project, "Wikimedia servers," 2020. [Online]. Available: https://meta.wikimedia.org/wiki/Wikimedia_servers(accessed:11-01-2020)

[11] S. L. Kota, K. Pahlavan, and P. A. Leppanen, *Broadband Satellite Communications for Internet Access*. 101 Philip Drive Assinippi Park Norwell, MA, USA: Kluwer Academic Publishers, 01 2004.

[12] E. Lutz, M. Werner, and A. Jahn, *Satellite Systems for Personal and Broadband Communications*. Heidelberg, Berlin, Germany: Springer Berlin Heidelberg, 12 2012.

[13] A. Fidler, G. Hernandez, M. Lalovic, T. Pell, and I. Rose, "Satellite—A new opportunity for broadband applications," *BT technology Journal*, vol. 20, no. 1, pp. 29–37, 01 2002.

[14] M. Fitch, "The use of satellite for multimedia communications," *BT technology journal*, vol. 21, no. 3, pp. 90–101, 07 2003.

[15] D. McKague, T. H. Zurbuchen, T. Donajkowski, J. Ervin, D. Heckathorn, and K. Moran, "IMAGINE Africa: Providing Internet to the developing world," in *2009 IEEE Aerospace conference*, 03 2009, pp. 1–9.

[16] A. Botta and A. Pescape, "On the performance of new generation satellite broadband Internet services," *IEEE Communications Magazine*, vol. 52, no. 6, pp. 202–209, 06 2014.

[17] M. Harris, "Tech giants race to build orbital Internet news," *IEEE Spectrum*, vol. 55, no. 6, pp. 10–11, 06 2018.

[18] Y. Henri, "The OneWeb satellite system," *Handbook of Small Satellites: Technology, Design, Manufacture, Applications, Economics and Regulation*, pp. 1–10, 02 2020.

[19] J. Foust, "Spacex's space-Internet woes: Despite technical glitches, the company plans to launch the first of nearly 12,000 satellites in 2019," *IEEE Spectrum*, vol. 56, no. 1, pp. 50–51, 01 2019.

[20] Telesat, "LEO satellite — Telesat lightspeed." [Online]. Available: https://www.telesat.com/leo-satellites/(accessed:03-30-2021)

[21] E. Howell, "The FCC has approved Amazon's plan for its Kuiper satellite constellation," 2020. [Online]. Available: https://www.space.com/amazon-kuiper-satellite-constellation-fcc-approval.html(accessed:03-30-2021)

[22] C. Qiu, H. Yao, F. R. Yu, F. Xu, and C. Zhao, "Deep Q-learning aided networking, caching, and computing resources allocation in software-defined satellite-terrestrial networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 6, pp. 5871–5883, 06 2019.

[23] F. Wang, D. Jiang, S. Qi, C. Qiao, and H. Song, "Fine-grained resource management for edge computing satellite networks," in *2019 IEEE Global Communications Conference (GLOBECOM)*, 12 2019, pp. 1–6.

[24] D. Dalai, S. Babu, and B. S. Manoj, "On using edge servers in 5G satellite networks," in *2020 IEEE 3rd 5G World Forum (5GWF)*, 09 2020, pp. 553–558.

[25] Y. Wang, J. Zhang, X. Zhang, P. Wang, and L. Liu, "A computation offloading strategy in satellite terrestrial networks with double edge computing," in *2018 IEEE International Conference on Communication Systems (ICCS)*, 12 2018, pp. 450–455.

[26] Y. Wang, J. Yang, X. Guo, and Z. Qu, "A game-theoretic approach to computation offloading in satellite edge computing," *IEEE Access*, vol. 8, pp. 12 510–12 520, 12 2019.

[27] S. Cao, Y. Zhao, J. Wei, S. Yang, H. Han, X. Sun, and L. Yan, "Space-based cloud-fog computing architecture and its applications," in *2019 IEEE World Congress on Services (SERVICES)*, vol. 2642-939X, 07 2019, pp. 166–171.

[28] G. Cui, X. Li, L. Xu, and W. Wang, "Latency and energy optimization for MEC enhanced sat-IoT networks," *IEEE Access*, vol. 8, pp. 55 915–55 926, 03 2020.

[29] H. Huang, S. Guo, W. Liang, K. Wang, and Y. Okabe, "Coflow-like online data acquisition from low-earth-orbit datacenters," *IEEE Transactions on Mobile Computing*, vol. 19, no. 12, pp. 2743–2760, 12 2020.

[30] D. Jiang, F. Wang, Z. Lv, S. Mumtaz, S. Al-Rubaye, A. Tsourdos, and O. Dobre, "QoE-aware efficient content distribution scheme for satellite-terrestrial networks," *IEEE Transactions on Mobile Computing*, 04 2021.

[31] D. Comer, "Ubiquitous B-tree," *ACM Comput. Surv.*, vol. 11, no. 2, p. 121–137, 06 1979. [Online]. Available: https://doi.org/10.1145/356770.356776

[32] W. J. Stewart, *Probability, Markov Chains, Queues, and Simulation: The Mathematical Basis of Performance Modeling*. Princeton, New Jersey, USA: Princeton University Press, 07 2009.

[33] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, ser. Athena Scientific optimization and computation series. 225 School St, Belmont, MA, USA: Athena Scientific, 2005, no. v. 1.

[34] A Wikimedia project, "Help:Searching," 2020. [Online]. Available: https://en.wikipedia.org/wiki/Help:Searching(accessed:11-01-2020)

[35] A Wikimedia project, "English Wikipedia," 2020. [Online]. Available: https://en.wikipedia.org/wiki/English_Wikipedia(accessed:11-01-2020)

[36] A Wikimedia project, "Wikipedia:Size of Wikipedia," 2020. [Online]. Available: https://en.wikipedia.org/wiki/Wikipedia:Size_of_Wikipedia(accessed:11-01-2020)

[37] A Wikimedia project, "Wikipedia:Size in volumes," 2020. [Online]. Available: https://en.wikipedia.org/wiki/Wikipedia:Size_in_volumes(accessed:11-01-2020)

[38] B. Rhodes, "Skyfield: High precision research-grade positions for planets and Earth satellites generator," p. ascl:1907.024, 07 2019.

[39] I. Chadès, G. Chapron, M.-J. Cros, F. Garcia, and R. Sabbadin, "MDPtoolbox: a multi-platform toolbox to solve stochastic dynamic programming problems," *Ecography*, vol. 37, no. 9, pp. 916–920, 07 2014. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/ecog.00888

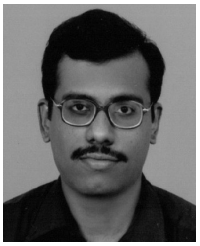
[40] Wikimedia statistics, "Wikipedia – English, Total page views," 2020. [Online]. Available: https://stats.wikimedia.org/#/en.wikipedia.org/reading/total-page-views/normal|table|all|~total|monthly(accessed:11-01-2020)

[41] Wikimedia statistics, "Wikipedia – English, Page views by country," 2020. [Online]. Available: https://stats.wikimedia.org/#/en.wikipedia.org/reading/page-views-by-country/normal|table|2020-11-01~2020-12-01|(access)~|monthly(accessed:11-01-2020)

[42] Wikimedia statistics, "List of countries by English-speaking population," 2020. [Online]. Available: https://en.wikipedia.org/w/index.php?title=List_of_countries_by_English-speaking_population&oldid=998888975(accessed:11-01-2020)

Debabrata Dalai (Graduate Student Member, IEEE) received the bachelor's degree in electronics and telecommunication engineering from Krupajal Engineering College, Bhubaneswar, India, and the master's degree in computer science and engineering (advanced network) from the Indian Institute of Information Technology and Management, Gwalior, India. He is currently a Research Fellow with the Department of Avionics, Indian Institute of Space Science and Technology, Thiruvananthapuram, India. His research interests include advanced wireless networks, satellite-IoT networks, satellite edge computing, and 5G satellite integration.

Sarath Babu (Graduate Student Member, IEEE) received the bachelor's degree in information technology from Mahatma Gandhi University, Kottayam, India, and the master's degree in computer science and engineering (information security) from the National Institute of Technology, Calicut, India. He is currently a Senior Research Fellow with the Department of Avionics, Indian Institute of Space Science and Technology, Thiruvananthapuram, India. His research interests include delay tolerant networks, software defined networks, wireless mesh networks, and complex networks.



Vineeth B.S. received the B. Tech degree in electronics and communication engineering from the College of Engineering, Thiruvananthapuram, and the Ph.D. degree in electrical communication engineering from the Indian Institute of Science, Bengaluru, India. He is currently an Assistant Professor with the Department of Avionics, Indian Institute of Space Science and Technology, Thiruvananthapuram. His research interests include communication networks, stochastic modeling and performance analysis, and reinforcement learning.

B. S. Manoj (Senior Member, IEEE) received the Ph.D. degree in computer science and engineering from the Indian Institute of Technology Madras in 2004. He was an Assistant Research Scientist and a Lecturer with the Electrical and Computer Engineering Department, University of California at San Diego. He is currently a Professor with the Department of Avionics, Indian Institute of Space Science and Technology, Thiruvananthapuram, India. His current research interests include complex networks, ad hoc wireless networks, delay tolerant networks, next generation wireless architectures, wireless sensor networks, wireless mesh networks, signal processing over networks, satellite networks, and quantum computing.