Collaborative Data Downloading by using Inter-Satellite Links in LEO Satellite Networks

Xiaohua Jia, *Fellow, IEEE*, Tao Lv, Feng He, Hejiao Huang, *Member, IEEE*

***Abstract*—Satellite systems have attracted great attention from academic and industrial communities in the recent years. There are many satellites that have been launched for weather forecast, environment monitoring, and target surveillance. One important task of the satellites is to download the data they have collected in the space to the ground servers via Earth Stations (ESs). Since satellites move at high speed along their own orbits and have very limited contact time with ESs, satellites may not be able to download all the data they have to the ground on time. In this paper, we propose a collaborative scheme that allows satellites to offload data among themselves using inter-satellite links (ISLs) before they come to the contact with the ES, such that satellites will carry the right amount of data according to the length of their contact time with the ES and the throughput of data downloading at ES is maximized. We develop an iterative optimization algorithm that jointly schedules data offload among the satellites and data downloading from satellites to the ES. Extensive simulations have been conducted to evaluate the effectiveness of our proposed method. The simulation results show that the data downloading throughput by using ISL data offload can be increased significantly. In many cases, the throughput reaches close to 100% of the capacity of the ES.**

***Index Terms*—LEO satellite network, satellite data downloading, collaborative data downloading, inter-satellite links, satellite routing.**

I. Introduction

SATELLITE systems have attracted great attention in recent years due to the rapid development of LEO (Low Earth Orbit) satellite systems. LEO satellites, typically operating at the altitude of 500-1500 km with system period of less than 2 hours [1], have many advantages of better signal quality and shorter propagation latency, compared with MEO (Medium Earth Orbit) and GEO (Geostationary Earth Orbit) satellite systems. Many LEO satellites have been launched for weather forecast, environment monitoring, target surveillance, and so on. One important task for these satellites is to download the data they have collected in the space to the ground servers via Earth Stations (ESs). Since satellites fly in the space at high speed along their own orbits and have very limited contact time with ESs, satellites may not have sufficient contact time with the ES to download all their data to the ground. Extensive research has been done on the scheduling of data exchange between satellites and the ESs during the limited time windows that satellites are visible to the ESs, such as [2]–[7]. Most of the existing works focus on developing optimal scheduling algorithms to schedule data exchange (or download) from satellites to a single or multiple ESs, such that the total throughput of data downloading is maximized. However, what the scheduling algorithms can do is very much limited in this scenario due to the disparity of the contact time of a satellite with the ES (or ESs) and the amount of data it has for downloading. A satellite that has a large amount of data for downloading may have a very short contact time with the ES. In such a case the satellite is not able to download all its data to the ES no matter how good the scheduling algorithm is. In this paper, we propose a collaborative scheme that allows satellites to offload data among themselves using inter-satellite links (ISLs) before they come to the contact with the ES. A satellite with a large amount of data but a small contact time can offload its data to other satellites that have spare contact time to help downloading the data to the ES. By offloading data among satellites on-the-fly in the space, each satellite can adjust the amount of data it carries for download to the ES, such that its contact time with the ES can be fully utilized. To our best knowledge, there is no work in the literature that has considered using data offloading among satellites in space to help data downloading to the ES.

It is a challenging issue to make use of the on-the-fly data offloading among satellites to help data downloading to the ES. The difficulties are: 1) Disparity of the amount of data a satellite has for downloading and the length of contact time it has with the ES. A satellite may have a large amount of data, but a little contact time with ES; vice versa, a satellite may have a long contact time with the ES, but a small amount of data to be downloaded. 2) High Dynamics of ISLs. Since satellites move at high speed along their own orbits, the communication links between two satellites are highly dynamic and may exist only for a very short period of time. It is difficult to make use of this short period time for data offloading between two satellites. 3) Joint scheduling of data offloading among satellites and data downloading to the ES. Since there could be multiple satellites that are in contact with the ES simultaneously, the allocation download time with the ES should be jointly done with the scheduling of data offloading among the satellites. This joint scheduling is

This work was financially supported by National Natural Science Foundation of China with Grant No. 61672195 and No.11371004, National Key Research and Development Program of China with Grant No. 2016YFB0800804, and Shenzhen Science and Technology Plan with Grant No. JCYJ20160318094336513, No. JCYJ20160318094101317 and No. KQCX20150326141251370.

X. Jia is with the Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen 518055, China and also with the Department of Computer Science, City University of Hong Kong, Hong Kong, China (e-mail: csjia@cityu.edu.hk).

T. Lv and F. He are with the Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen 518055, China (e-mail: lvtaoleo@gmail.com; hefengair@gmail.com).

H. Huang is with the Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen 518055, China (Corresponding author: hjhuang@hitsz.edu.cn).

further complicated by the requirement that the data offloaded to a satellite from others must be done before this satellite comes to download its data to the ES.

We first introduce an ES time sharing graph that describes the contact time between satellites and ES to allocate download time to the satellites when multiple satellites share the contact time with the ES. Then, we employ the space-time topology graph to model the spatial and temporal connectivity between the satellites for the data offloading among satellites. Our algorithm follows the iterative scheme. In each iteration, we start with an allocation of download time to the satellites according to their contact time with the ES and the amount of data they carry. Then, we schedule data offloading among the satellites by using ISLs, such that the download time allocated to satellites is maximally utilized. Finally, we adjust the amount of download time allocated to each satellite according to how much data it actually carries through the ISL data offloading from (or to) others. This iteration is repeated until the download time allocated to satellites cannot be improved anymore.

Extensive simulations have been conducted to evaluate the effectiveness of our proposed method. The simulations run on the Globalstar Constellation [8] and the Iridium Constellation [9]. The simulation results show that the data downloading throughput by using ISL data offloading can be increased significantly. In many cases, the throughput reaches close to 100%.

The rest of this paper is organized as follows. Section II is about the related work. In Section III, we present the system model and the problem definition. In Section IV, we present our proposed algorithm. We give the analysis of the algorithm in Section V. Simulations are discussed in Section VI. Finally, we draw our conclusions in Section VII.

II. Related Work

LEO satellite networks have received significant attention for some decades due to their ability to capture more detailed data about earth in comparison with MEO and GEO satellite networks. Two typical regular LEO satellite constellations, Walker delta and Walker polar, were discussed in [10], and the dynamical activities of Walker delta and polar constellations were described in details in [1].

The periodic and dynamic nature of satellite networks poses new challenges to the design of routing protocols, and the routing protocols for Mobile Ad-Hoc Networks (MANETs) or delay tolerant Vehicular Networks (VENET) networks cannot be effectively applied to satellite networks. Some new routing protocols were proposed specifically for satellite networks, such as those in [11]–[15]. In [11], [12], a virtual grid structure was introduced to model the logical locations of satellites. This virtual grid covers the entire Earth, where the nodes in the grid (representing logical locations) do not move and each node is filled by the nearest satellite. This virtual grid structure hides the mobility of satellites and transforms the routing between two moving satellites to the regular routing two static nodes in the grid. However, this virtual grid routing scheme requires there should be a sufficient number of satellites that cover the earth in an interconnected grid structure. In [13]–[15], the network topology is represented by a sequence of snapshots of topologies. The period T of a satellite network system is divided into a series of time slots, denoted by $[t_0, t_1), [t_1, t_2), [t_{n-1}, t_n)$. The topology change only occurs at the start of each time slot, i.e., at the time point $t_1, t_2, \ldots, t_n$, and the topology remains unchanged throughout the period of the time slot. Based on the snapshot representation of dynamic topology changes, some routing protocols were proposed in [16], [17] by using space-time topology graphs. More recently, Contact Graph Routing (CGR) [18]–[20] and Enhancing Contact Graph Routing (ECGR) [21] were proposed to find the shortest delay path between any pair of satellites in the network. The authors in [22] concluded the CGR style algorithms are efficient for multi-path interplanetary routing in satellite networks.

There are some other works in the literature that studied the performance of routing by ISL under different scenarios, such as [23]–[25]. Based on the snapshot topology representation, an ISL reassignment method was presented in [23] to optimize the snapshot routing performance for the polar-orbit LEO satellite networks. The performance of satellite routing algorithms under node failure situations was discussed in [24]. Janhunen *et al.* in [25] studied the impact of terrestrial network interference to a satellite uplink transmission and found that the satellite links can operate on the same frequency band as the terrestrial networks in spite of the long satellite links and strictly power limited systems.

The problem of scheduling data communication between satellites and the ES was first introduced by Gooley *et al.* in [2]. It was defined as the Satellite Range Scheduling (SRS) problem that aims to schedule a set of satellites communication requests with a set of ESs during satellites window time with ESs. The objective is to maximize the total number of communication requests that can be scheduled. Gooley *et al.* first formulated the problem for low-altitude satellites using Mixed Integer Programming (MIP), and proposed a heuristic algorithm by partitioning the requests into smaller subsets and solving them individually. Then, for medium and high-altitude satellites, an insertion method was proposed to insert as many requests as possible into the previous solution for low-altitude satellites. Barbulescu *et al.* in [3] presented a systematic analysis on the hardness of the SRS problem. They first analysed the Single-Resource Range Scheduling (SiRRS) problem and showed the algorithms based on machine scheduling methods outperform the genetic methods proposed in [2], where $resource$ means ES. They then studied the Multi-Resource Range Scheduling (MuRRS) problem, and proposed a genetic algorithm that obtains the best performance through their simulations. Some follow-up works can be found in [4]–[7]. In [4], the problem of selecting and scheduling a subset of satellite tasks constrained by the capacity of the ES was studied. Each task has a weight, time-window and expected gain when it is performed. The objective is to maximize the total gain by scheduling the subset of tasks. Marinelli *et al.* in [5] formulated the data exchange problem between satellites and multi-ESs as a multiprocessor task scheduling problem, and a time-indexed 0,1-linear programming formulation was

introduced to represent all complex constraints. A Lagrangian version of the Fix-and-Relax heuristic was developed to solve the problem and the solution is able to provide near optimal results. Spangelo *et al.* in [6] formulated the problem of scheduling data downloading from a single satellite to multi-ESs using constained MIP, and proposed an iterative algorithm that progressively tightens the constraints to obtain a feasible and optimal solution. Castaing presented a model in [7] to model data downloading from multi-satellites to multi-ESs, where satellites continuously collect data at a constant rate. A greedy algorithm was proposed for the test of the model in various scenarios of the simulations.

In the previous works about satellite task scheduling with the ESs, no ISL data offloading among the satellites was considered. By using ISLs, data offloading among the satellites can be done when the satellites are not immediately in connection with the ESs, and thus the throughput of data download can be significantly increased. To the best of our knowledge, there is no literature working on downlink data downloading using ISLs data offloading. In this work, we present the first effort that combines the inter-satellite data offloading with the data downloading to the ESs.

III. System Model and Problem Definition

The LEO satellite system of our concern consists of n satellites, denoted by $V = \{v_1, v_2, \ldots, v_n\}$, and an ES (Earth Station). Each satellite v_i has an amount of data to be downloaded to the ES, denoted by O_i. A satellite can only download data to the ES when it is in the line-of-sight of the ES (called time window for downloading). Since all satellites move along their own orbits, their contact time windows with the ES (i.e., the start-time and end-time of windows) are known. The ISLs (Inter-Satellite Links) are highly dynamic. A link between two satellites is up when they move into each other's transmission range, and the link is down when they move away from each other. Since satellites moves along their orbits periodically, the pattern of network topology in terms of the interaction among satellites and the interaction between satellites and the ES is periodical and predictable. We consider our algorithm in a time period T.

Given a set of satellites, an ES, and their contact time windows in a period of T, the problem of our concern is to design an algorithm that enables the satellites to use ISLs to offload data from satellites with heavy data load to those with light load, such that satellites can maximally download their data within their contact window with the ES and the system throughput of downloading is maximized. Let O_i^* denote the amount of data that satellite v_i has actually downloaded to the ES. The system throughput of data downloading is defined as:

$$Throughput = \sum_{i=1}^{n} O_i^* / \sum_{i=1}^{n} O_i \tag{1}$$

The objective of our algorithm is to maximize this throughput defined above.

For simplicity, we make the following assumptions: 1) The data transmission between all satellites and between the satellites and the ES has the same data rate and uses the same packet size. That is, the data rates on all ISLs and between satellites and the ES are all the same. Under this assumption, the data size and transmission time are interchangeable. The amount of data that a satellite carries can be represented as the number of packets, which can be further represented as the amount time that is required for transmission. 2) The satellites and the ES are equipped with a single pair of transceivers for communication. That is, a satellite (or the ES) can only do one operation, transmitting or receiving, at a time. 3) The switching time that the ES switches from one satellite to another for data downloading is negligible. That is, it takes no time for the ES to switch communication from one satellite to another.

We also assume the communication on ISLs and between satellites and the ES is error free. The detection of transmission errors and retransmission can be dealt with by the underneath link layer protocols, which is beyond the scope of this paper.

IV. Joint scheduling of Offloading and Downloading

A. Overview of Our Solution

To maximize the overall system throughput of downloading data to the ES, our strategy is to offload data among satellites by using ISLs before the satellites come to the contact with the ES. Since there could be multiple satellites that are visible to the ES at the same time, i.e., the contact windows of satellites with the ES may overlap with each other, the download time to the ES is shared by the satellites that have overlapping contact windows. We need to answer two questions: 1) How much download time that should be allocated to a satellite when it shares the contact windows with others? 2) How much data that should be offloaded from one satellite to another, such that the download time allocated to each satellite is fully utilized? The answers to these two questions are inter-dependent with each other. In order to allocate the download time to a satellite properly, we need to know how much data this satellite can offload to (or receive from) others; however, to optimally offload data from one satellite to others (or receive data from others), we need to know how much download time that is allocated to each satellite. To break this cycle, we adopt an iterative method that asymptotically reaches the optimum. We initially split the download time equally among the satellites that share the contact time with the ES. Then, we design an algorithm to offload data among the satellites in the space such that this allocated download time is maximally utilized. At the end of this offloading, some satellites may have surplus of download time, but others may have remaining data yet to be downloaded. We readjust the download time allocated to the satellites. This operation is repeated, until either all satellites can have their data downloaded to the ES during their contact windows or data offloading among the satellites cannot improve the system throughput of downloading anymore.

The allocation of download time of the ES among the satellites is done by the help of the ES Time Sharing Graph, and the data offloading algorithm uses the satellite Space-Time Topology Graph. Both graphs will be discussed in the next subsections.

B. ES Time Sharing Graph

Let V_{ES} denote a set of satellites that have contact time with the ES, $V_{ES} \subseteq V$. Since multiple satellites may share the contact time with the ES, we need to allocate the download time properly to each satellite according to how much data it carries for downloading. We introduce the ES time sharing graph for the allocation of download time to satellites. Each satellite in V_{ES} has a contact window with the ES, defined by the start-time and the end-time. The time axis is divided by the start-time and the end-time of contact windows of all satellites, as shown in Fig. 1. Two adjacent time points on the time axis form a segment, denoted by $T_i, i = 0, \ldots, m$. As in the example of Fig. 1, segment T_0 is between the start-time of v_1 (or v_4) and the start-time of v_2; segment T_1 is between the start-time of v_2 and the end-time of v_1 (or the start-time of v_3). A time segment is shared by the satellites whose contact windows overlap with each over this segment. Taking Fig. 1 as the example again, segment T_0 is shared by v_1 and v_4. Note that the number of satellites that share a time segment remains unchanged throughout this segment and time segments have different lengths.

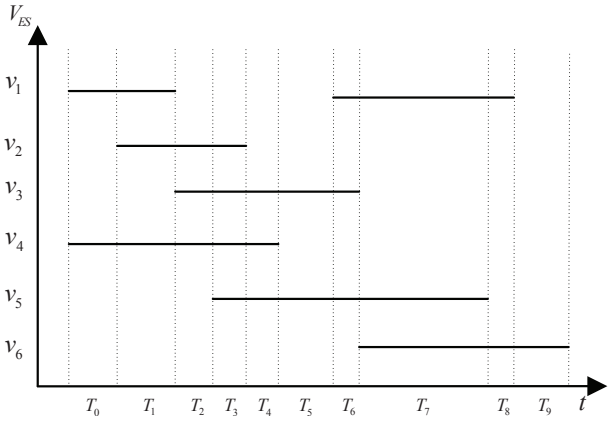


Fig. 1. ES Time Sharing Graph

As the initial allocation of ES' download time to the satellites, we equally split the time among the satellites that share the contact window with the ES. As shown in Fig. 2, segment T_1 is shared by v_1, v_2 and v_4, so the download time allocated to them is $\frac{1}{3}T_1$ each. Let T_i^{DL} denote the total download time allocated to satellite v_i. T_i^{DL} is the sum of time allocated to v_i in all segments. For example, the contact window of v_2 has three segments, T_1, T_2 and T_3. Segments T_1, T_2 and T_3 are shared by 3, 3 and 4 satellites, respectively. Thus, $T_2^{DL} = \frac{1}{3}T_1 + \frac{1}{3}T_2 + \frac{1}{4}T_3$.

There are some cases where a satellite contacts the ES several times during the period T. For simplicity of the algorithm design, we regard each time that a satellite contacts the ES as a new virtual satellite, and we divide the original data carried by the satellite into several parts that are in proportion to the download time initially allocated to the virtual satellites. For example, v_1 has two contact windows with the ES in Fig. 2, v_1 will be represented by two virtual satellites in the algorithm. The original amount of data carried by v_1 will be divided in proportion with the download time allocated to the two contact windows and assigned to the two virtual nodes of v_1. Thus, in the rest of the paper, we assume that one satellite only has only one contact window with the ES during period T.

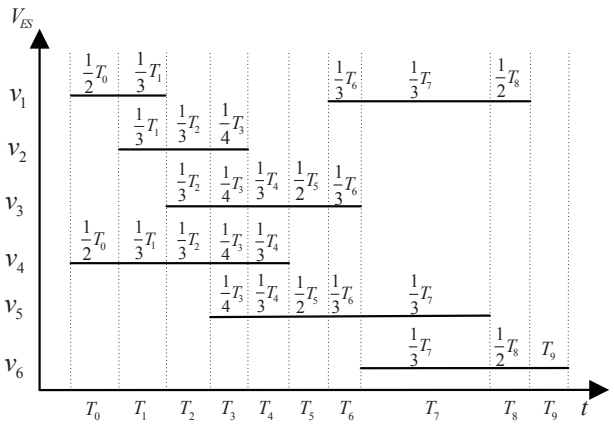


Fig. 2. Initially allocate the ES time sharing graph

C. Space-time Topology Graph

In the above subsection, we introduced the ES time sharing graph to help allocating download time to satellites. Once a satellite has allocated a download time, we need to fill it with the right amount of data through ISL data offloading among the satellites. Since satellites are moving at high speeds along their orbits, the communication links between satellites are up and down frequently and the topology of the satellite network changes dynamically. We construct the space-time topology graph to describe the connectivity of satellites at different time points as time progresses. The time is equally divided into small slots with the same size τ. The length τ is the exact time required for transmitting one packet of data. Since we assume all satellites have the same data rate and use the same packet size, the transmission time of one packet is the same for all satellites and τ can thus be used as time unit. The time line is represented as $\{0, 1\tau, 2\tau, \ldots, T\}$, as shown in Fig. 3.

The space-time topology graph consists of columns of satellite nodes at different time points, as shown in Fig. 3. Let $V_{k\tau}$ denote the set of satellites at time $k\tau$ ($V_{k\tau} = V$). Links in this topology graph only exist between two adjacent columns of nodes. For node v_i in $V_{k\tau}(k \geq 0)$ and node v_j in $V_{(k+1)\tau}$, a link (v_i, v_j) exists if and only if v_i and v_j are within each other's transmission range during the time from $k\tau$ to $(k+1)\tau$. That is, v_i is able to transmit one packet of data to v_j at time point $k\tau$. We will use this space-time topology graph for ISL data offloading among satellites.

D. Notations and Initial Allocation of Download Time

Our optimization goal is to maximize the throughput of data downloading at the ES. This throughput depends on two factors: 1) how much download time that is allocated to each satellite, and 2) how much data offloading that can be done among the satellites. As we discussed before, these two factors

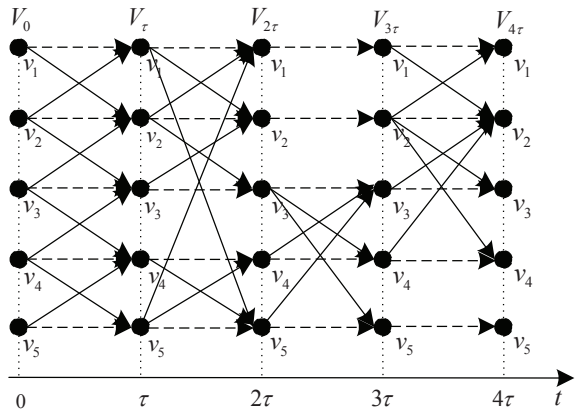


Fig. 3. Space-time Topology Graph.

TABLE I
NOTATIONS USED IN OUR ALGORITHM

Symbol	*Definition*
V	set of satellites $\{v_1, v_2, \ldots, v_n\}$
V_{ES}	subset of V that have contact time with the ES
T	scheduling period of satellites
τ	length of timeslot in space-time topology graph
O_i	amount of data carried by v_i
T_i	time segment i in ES time sharing graph, $i = 0$ to m
T_i^{DL}	amount of download time allocated to v_i
T_i^{rm}	amount of remaining download time of v_i
O_i^{rm}	amount of remaining data of v_i

are inter-dependent with each other. We propose an iterative optimization scheme as following.

For each satellite, we need to keep track of the situation whether the allocated download time is sufficient for it to download all its data. If the allocated time is less than the data amount it has, it needs to offload its data to others; otherwise if it has more time it needs, it can receive data from others to help them for downloading. For this purpose, we introduce the following notations (A summary of notations can be found in Table.I):

T_i^{DL}: the amount of download time allocated to satellite i. For satellite i, if it does not have contact time with the ES, its $T_i^{DL} = 0$.

O_i^{rm}: the current amount of remaining data of satellite i that cannot be downloaded within its allocated download time. That is, $O_i^{rm} = O_i - T_i^{DL}$. Note that the unit of download time and unit of data amount are now interchangeable. In our algorithm, one unit of time is the time required for transmitting one data packet. Thus, the length of time is measured in terms of the number of packets that can be transmitted, which is interchangeable with the amount of data to be transferred or downloaded. O_i^{rm} is always greater than or equal to zero. When O_i^{rm} reaches 0, it means that satellite i is able to download all its carried data to the ES within the download time allocated to it.

T_i^{rm}: the current amount of remaining download time that satellite i can help others for downloading. That is, satellite i can still have room to receive T_i^{rm}-equivalent amount of data from others to help their data downloading. We have, $T_i^{rm} = T_i^{DL} - O_i$. T_i^{rm} is always greater than or equal to zero. When $T_i^{rm} = 0$, it means that satellite i does not have any more time for downloading to the ES.

E. Iterative Optimization of ISL Data Offloading and Download Time Allocation

The data offloading algorithm runs on the space-time topology graph. At each iteration, the offloading starts from time 0, and progresses forward step by step along the timeline of the space-time topology graph as shown in Fig. 3. Initially, each satellite v_i is allocated with an amount of download time T_i^{DL}, as discussed in section IV.B. For each v_i, its O_i^{rm} is initialized to $max\{0, O_i - T_i^{DL}\}$; and its T_i^{rm} is initialized to $max\{0, T_i^{DL} - O_i\}$. When $O_i^{rm} > 0$, it means that v_i has data to be offloaded to others; when $T_i^{rm} > 0$, it means that v_i has more download time to help others. The length of each step is τ. At each step we compute the optimal data offloading from $k\tau$ to $(k+1)\tau$, supposing we are currently at step k. Note that even though data offloading can take several hops from a source node to a destination node, to simplify the offloading algorithm, we limit the offloading operation occurring in one hop. That is, a node can only offload its data to another node in one hop. By limiting this one-hop data offloading, we do not need to consider the routing issue for the multi-hop cases. We have done many simulations to evaluate the effect of this one-hop offloading, and our simulation results show that with one-hop offloading it can achieve over 95% of the download capacity of the ES for almost all different load situations.

We introduce a bipartite graph to compute the data offloading at each step. Suppose we are at the k-th step. The node set at left-hand side of the bipartite graph are source nodes of data offloading, denoted by V_L, $V_L \subseteq V_{k\tau}$. V_L consists of nodes that have data to be offloaded. That is, for any node v_i in V_L, $O_i^{rm} > 0$. The node set at right-hand side are the destination nodes of the data offloading, denoted by V_R, $V_R \subseteq V_{(k+1)\tau}$. V_R consists of nodes that have spare time to receive data from others. That is, for any node v_i in V_R, $T_i^{rm} > 0$. The bipartite graph is an induced subgraph of the space-time topology graph that contains only nodes in $V_L \cup V_R$. Fig. 4 is an example of a bipartite graph, induced from space-time topology graph in Fig. 3.

There is a time constraint from satellites to offload data among themselves. A satellite must complete its receiving data from others before it is its turn to download data to the ES. Let ST_i denote the start of download time for satellite i. For v_i in V_R in the bipartite graph, the data offloading to v_i must happen before its start time of downloading. That is, $(k+1)\tau < ST_i$.

The optimal data offloading at step k is equivalent to finding the maximal match in the bipartite graph, where no two edges share an endpoint and the number of edges in the match is maximized. Ford-Fulkerson algorithm for maximum flows problem [26] is employed to compute the maximal match. When the maximal match $M = \{(v_i, v_j) : v_i \in V_L, v_j \in V_R\}$ is found, an edge, say (v_i, v_j) in this match, represents that node v_i can offload one packet to node v_j. We update O_i^{rm}

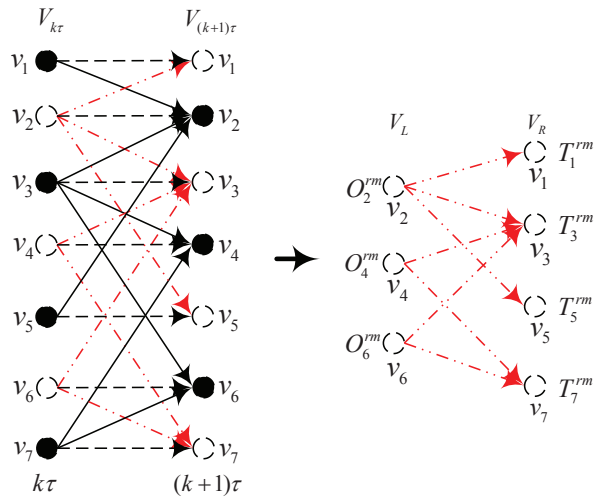


Fig. 4. Bipartite Graph Induced from Space-time Topology Graph.

by decreasing 1, indicating that v_i has one packet less. When O_i^{rm} reaches 0, it means v_i now can download all its data to the ES with the allocated download time, and it will not participate in data offloading for the rest of the time. We also update T_j^{rm} by decreasing 1, indicating that v_j has 1 packet less time to help others. When T_i^{rm} reaches 0, it means v_j cannot receive data anymore from others. The algorithm then moves to step $k+1$, $k+2$, ..., until reaching step T/τ, where this iteration completes.

At the end of each iteration, for any node, say v_i, if $T_i^{rm} > 0$, it means satellite v_i has surplus of download time allocated to it, even with the data offloaded from others. We need to reallocate T_i^{rm} to others. The nodes that need more download time are those nodes that still have remaining data to be downloaded, i.e., their $O_i^{rm} > 0$. If there is no such node that needs more download time, the surplus download time will be cleared, i.e., the T_i^{rm} is set to 0.

To allocate surplus of download time T_i^{rm} of v_i, $T_i^{rm} > 0$, suppose T_i^{rm} falls into several consecutive segments (as shown in Fig. 2). For each such segment, we split the surplus time of v_i and equally distribute to the satellites that share this segment with v_i and need more download time. At the end of redistribution of surplus download time, each node v_i has a new amount of download time T_i^{DL}. Then, the next iteration starts. This iteration process is repeated until for all satellites, v_i, $T_i^{rm} = 0$. This termination condition indicates that all satellites use up their allocated time and the allocation of the download time is optimal. Theorem 1 states that the termination condition is reachable.

Algorithm Description. Algorithm 1 describes the process of the iterative optimization algorithm. First, the algorithm initializes the T_i^{DL} for all satellites (lines 1-3). The main-loop of the algorithm (lines 4-20) is to iteratively offload data among satellites by using ISLs according to the amount of data and the download time each satellite has, and further adjust the allocation of download time based on the result of ISL data offloading. Inside this main-loop, it first initializes O_i^{rm} and T_i^{rm} for each satellite (lines 5-8). It then constructs a bipartite graph and computes the maximal match in the bipartite graph for each timeslot (lines 9-16). Each edge in the maximal match represents the transfer of one data packet between two satellites in the timeslot (lines 12-15). After one iteration of data offloading, the algorithm then readjust the download time allocated to each satellite T_i^{DL} (lines 17-20). The algorithm terminates when $T_i^{rm} = 0$ for all v_i, which is the case that the download time allocated to all satellites are fully utilized. The throughput of data downloading to the ES cannot be improved anymore in this case.

Algorithm 1 Iterative Offloading Algorithm.

```
    //Initialize T_i^DL
1:  for v_i ∈ V do
2:      Initial allocation of T_i^DL;
3:  end for
    //Interative data offloading and reallocation of download time
4:  repeat
5:      for v_i ∈ V do
6:          O_i^rm = max{0, O_i − T_i^DL};
7:          T_i^rm = max{0, T_i^DL − O_i};
8:      end for
9:      for k = 0 to (T/τ − 1) do
10:         Construct a bipartite graph for node sets V_kτ and
            V_(k+1)τ;
11:         Find the maximal match M in the bipartite graph;
12:         for each (v_i, v_j) ∈ M do
13:             O_i^rm = O_i^rm − 1;
14:             T_j^rm = T_j^rm − 1;
15:         end for
16:     end for
        //Update T_i^DL
17:     for any T_i^rm > 0 do
18:         Split T_i^rm and distribute it to the nodes that share the
            time segment with v_i and need more download time;
19:         T_i^rm = 0;
20:     end for
21: until for ∀i : T_i^rm = 0
```

V. Algorithm Analysis

Theorem 1. Algorithm 1 will eventually terminate.

Proof. Since after each iteration, for the nodes whose $T_i^{rm} > 0$, their remaining download time will be reallocated to other nodes that need it (lines 17-20). Thus, in each iteration, there is at least one node whose T_i^{rm} becoming zero. It takes at most n iterations for all nodes to reach $T_i^{rm} = 0$, where the Algorithm terminates. □

Theorem 2. Algorithm 1 produces the optimal result with 1-hop offloading when it terminates.

Proof. We introduce T_{ES}^{DL} to denote the total time that the ES is in contact with all satellites. T_{ES}^{DL} is the maximum amount of time that the ES can receive data from satellites. When the algorithm terminates, i.e., for all v_i: $T_i^{rm} = 0$, the system is in one of the following three cases:

1) $O_i^{rm} = 0$ for all v_i: this is the case that data carried by all satellites are successfully downloaded to the ES.

2) $\sum_{i=1}^{n} T_i^{DL} = T_{ES}^{DL}$: this is the case that the total download time used by all satellites is equal to the total contact time that the ES has with all satellites. This means the ES works at its full capacity and no more data can be downloaded to the ES.

3) $\sum_{i=1}^{n} T_i^{DL} < T_{ES}^{DL}$, but $O_i^{rm} > 0$ for some v_i: this is the case that some satellites download time is not fully utilized while some other satellites still have data not being completely downloaded. However, we argue that if there is an optimal solution that can achieve the maximal throughput by using only 1-hop offloading, then Algorithm 1 can find this optimal solution. Suppose there is a node v_i with $O_i^{rm} > 0$ and another node v_j with $T_j^{rm} > O_i^{rm}$, and there is sufficient 1-hop bandwidth for v_i to offload its excessive amount of data to v_j before $v_j^{'}$s contact time with the ES. Our maximal matching algorithm will be able to find the offloading schedule to offload $v_i^{'}$s data to v_j step by step until all $v_i^{'}$s excessive data is completely offloaded to v_j. This statement is also true when v_i offloads its data to multiple destination nodes, provided there is sufficient 1-hop bandwidth from v_i to these nodes for offloading. Thus, Algorithm 1 achieves the optimal throughput with 1-hop offloading.

Note that in the above case 3, to 100% utilize the download time allocated to all satellites, 1-hop offloading is not enough. It requires a multi-hop offloading scheme. As we discussed before, the experiment results show us that the 1-hop offloading strategy is a good tradeoff between the system throughput and the overhead of ISL communication. □

Theorem 3. Algorithm 1 has time complexity $O(n^4T/\tau)$.

Proof. The complexity of Algorithm 1 consists of two major components: the construction of the space-time topology graph and the iterative optimization method. We first look at the construction of the space-time topology graph. Since there are in total T/τ timeslots in the space-time topology graph, there are (T/τ +1) columns of nodes in the space-time graph and there are at most n^2 edges between two columns of nodes. Thus, the complexity of constructing the space-time topology graph is $O(n^2T/\tau)$ in the worst case.

As for the complexity of the iterative optimization, since after each iteration, there is at least one node whose T_i^{rm} becoming zero, there are at most n iterations before the algorithm terminates. In each iteration, the 1-hop data offloading takes T/τ steps, and in each step, the maximal matching algorithm is employed to compute the optiaml data offloading over a bipartite graph. It takes $O(n^3)$ to compute the maximal matching at each step. Thus, the time complexity of the iterative optimization method is $O(n^4T/\tau)$. Summing up with the complexity of constructing the space-time topology graph, the total complexity of Algorithm 1 is $O(n^2T/\tau) + O(n^4T/\tau) = O(n^4T/\tau)$. □

VI. Simulations and Analysis

We conduct our simulations by using Satellite Tool Kit (STK) to evaluate the performance of our proposed algorithm. STK [27] is an analytical tool from the AGI company (Analytical Graphics, Inc.), which allows scientists and engineers to perform complex analysis of land, sea, air and space assets in an integrated solution. Walker constellations [1] are a typical design of LEO satellite networks. In Walker constellations, all satellites in a network are organized into planes. The total number of satellites in the network is $N_p \times M_p$, where N_p denotes the number of planes in LEO constellation, and M_p the number of satellites per plane. There are two types of Walker constellations, called Walker delta constellation and Walker polar constellation. For the Walker delta constellation, the planes are separated from each other with the same angular distance of $360°/N_p$; and for the Walker polar constellation, the planes are separated with the angular distance of $180°/N_p$. Satellites in the same plane are separated from each other with equal angular distance of $360°/M_p$.

Our simulations are performed on the Globalstar constellation (a typical system of the Walker delta constellation) and the Iridium constellation (a typical system of the Walker polar constellation). The length of a timeslot is set to 5ms, and the system period of the LEO satellite network T is set to two hours.

A. Globalstar constellation

In the Globalstar constellation, there are 8 orbital planes with 6 satellites each and inter-plane of $60°$. Satellite orbits are 1414 km in altitude with an orbit inclination angle of $52°$. Globalstar satellites are assigned a bandwidth of 16.5 MHz in the L-band (1610.0-1626.5 MHz) for ISL communication, and a bandwidth of 33 MHz in the C-band (5199.5-5126.0 MHz and 6250.0-6541.5 MHz) is used for satellite-ES communication [28]. We set that mission started Jul 1, 2015 at 12:00:00 UTC. The orbital period is 114 minutes. The ES is located in latitude $34.5°$ and longitude $109.5°$.

The data in satellites is collected in advance and the data distributed in the network satisfies the normal distribution. The mean value of the normal distribution μ is the average of the amount of the data of all satellites. The variance σ is from 0.05μ to 0.5μ. We evaluate our algorithm in Globalstar constellation under various load situations of the data to be downloaded to the ES. We compare our method with the method that do not use ISLs for inter-satellite data transfer, known as SiRRS (Single-Resource Range Scheduling) [3]. In SiRRS, the problem is transformed to the classic scheduling problem. We call our method CoDld (Collaborative downloading) in all simulation figures.

We simulate the performance of the algorithms under three different load situations of data downloading, light, medium and heavy. By light load, the total amount of data carried by all satellites is set to around 40-50% of the capacity of the ES. By medium and heavy load, the total amount of data to be download is set to around 70-80% and 90-100% respectively of the capacity of the ES. Fig.5), Fig.6) and Fig.7), are the results under the light load, medium load and heavy load, respectively. In each figure, we draw a curve of total data load as a benchmark. The total data load is the ratio of total amount of data carried by all satellites over the capacity of the ES. From Fig. 5-7, the following observations can be made:

1) The performance of our method (CoDld) is significantly better than the SiRRS method, and it can achieve almost 100% of all the data to be downloaded. The curve of CoDld completely overlaps with the benchmark line ("total data load") in both light load and medium load, and it almost overlaps with the benchmark line in situation of heavy load. As for the SiRRS method, it can only download around 60% of the total amount of data in all three different load situations.

2) The performance gap between CoDld method and the SiRRS method becomes wider when the download data load is heavier. This can be seen from Fig.6) to Fig.7). This shows the advantages of collaborative downloading are more significant than the method without using ISL data transfer when the download traffic gets heavier.

3) The throughput of CoDld and SiRRS do not change much as the increase of disparity of the initial amount of data assigned to satellites (i.e., the variance). This indicates that the download performance is more sensitive to the traffic load; but not sensitive to the variance of data amount carried by satellites. That is, both methods are effective in scheduling of data downloading.

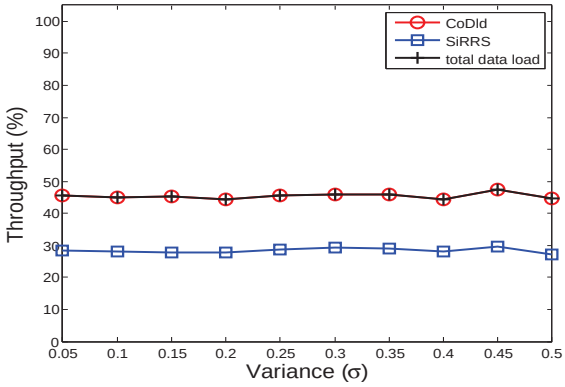


Fig. 5. Throughput of Light Data Load (40-50% of the ES' Capacity).

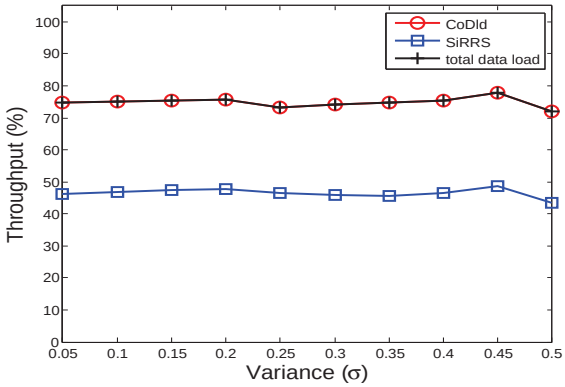


Fig. 6. Throughput of Medium Data Load (70-80% of the ES' Capacity).

B. Iridium constellation

In the Iridium constellation, there are 6 orbital planes with 11 satellites each. Satellites are at a height of approximately 780 km and inclination of 86.4°. In Iridium constellation, both uplink and downlink occupy 100 MHz bandwidth in the K-band (27.5-30.0 GHz and 18.8 - 20.2 GHz). The ISL communication uses 200 MHz bandwidth in band 22.55 - 23.55 GHz [29]. We set that mission started Jul 1, 2015 at 12:00:00 UTC. The orbital period is 100 minutes. The ES is also in latitude 34.5° and longitude 109.5°. The data distribution is the same as the Globalstar constellation. We compare the CoDld algorithm with the SiRRS algorithm in Iridium constellation under various load situations of the data to be downloaded to the ES, which are light load, medium load and high load.

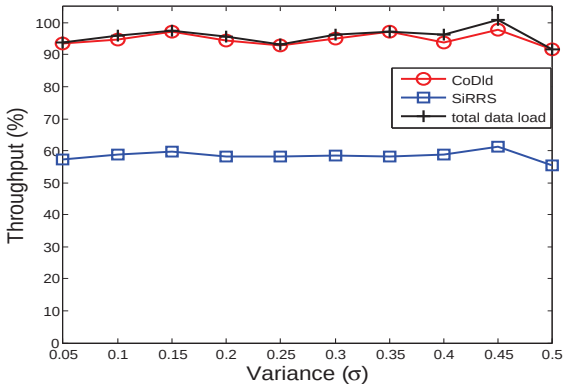


Fig. 7. Throughput of Heavy Data Load (90-100% of the ES' Capacity).

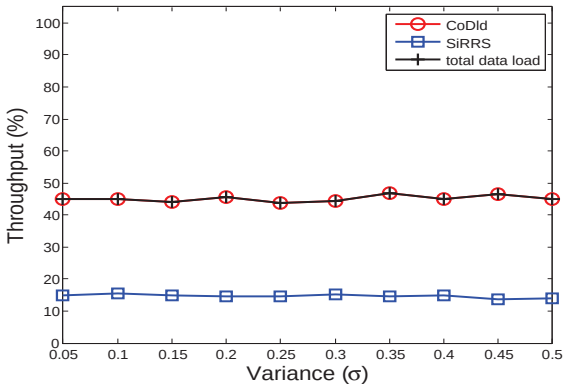


Fig. 8. Throughput of Light Data Load (40-50% of the ES' Capacity).

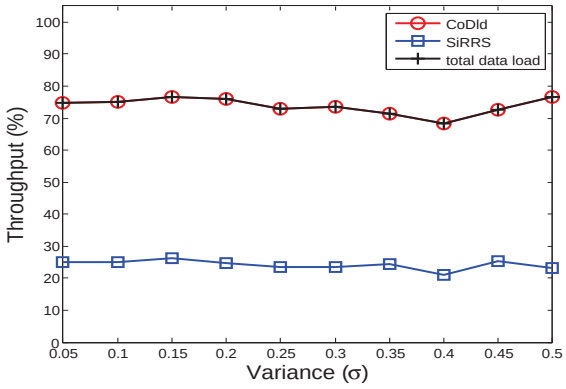


Fig. 9. Throughput of Medium Data Load (70-80% of the ES' Capacity).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TWC.2017.2647805, IEEE Transactions on Wireless Communications

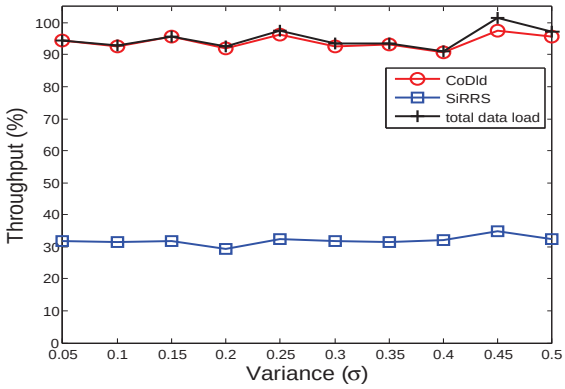


Fig. 10. Throughput of Heavy Data Load (90-100% of the ES' Capacity).

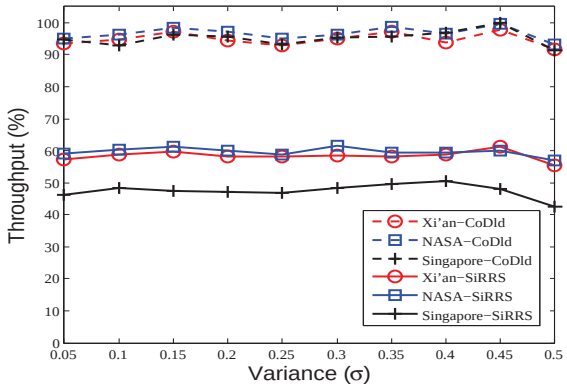


Fig. 11. Throughput for Different Locations of ESs in Globalstar

Fig. 8, Fig. 9 and Fig. 10 show the results of our CoDld algorithm comparing with SiRRS method under three different degrees of data load, the same as the Globalstar constellation. The total amount of data carried by all satellites in the Iridium networks is 40-50% (shown in Fig. 8), 70-80% (shown in Fig. 9) and 90-100% (shown in Fig. 10) of the capacity of the ES, respectively. As shown in these three figures, the results of the Iridium constellation are similar to the Globalstar constellation. The difference is that the throughput of the Iridium constellation is less when the data is in high load among satellites. This is because that the Iridium constellation has 66 satellites, which is more than 48 satellites in Globalstar constellation but the number of the visible satellites in Iridium is almost the same as the Globalstar constellation in two hours. This causes heavier traffic load on the visible satellites for data downloading to the ES, which leads to a slight lower download throughput.

C. Different Locations of ESs

We choose three different locations of ESs to compare the performance of our algorithm and SiRRS under different ES locations. They are Xi'an (34.5°,109.5°), NASA (35.6°,−117.4°), and Singapore (1.3°,103.8°). Xi'an is the most important area of China's aerospace industry and NASA is an important aeronautics and space institution in the world. They locate at the similar latitude. Singapore locates almost on the equator and it has the similar longitude with Xi'an. We choose the heavy load situation (i.e., the total amount of data to be downloaded is around 90-100% of the ES' capacity) for this session of simulations. The Fig.11 and Fig.12 show the results of our CoDld algorithm comparing with SiRRS method in the Globalstar and Iridium constellations respectively.

From Fig. 11 and Fig.12, the following observations can be made:

1) The performance of our CoDld method in different ESs locations is very stable and it achieves over 90% of the throughput. The three curves for CoDld method (solid lines) almost overlap with each other, and they are significantly higher than the curves of SiRRS method. This indicates that the CoDld method can achieve almost full system throughput through ISL data offloading regardless of the locations of the ES.

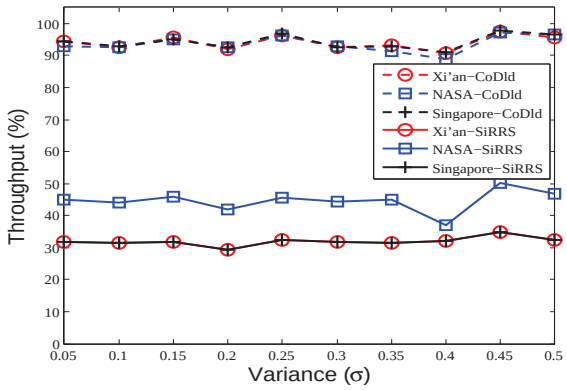


Fig. 12. Throughput for Different Locations of ESs in Iridium

2) The performance of SiRRS is unstable. SiRRs performance better with some ES locations (e.g., NASA) than others (e.g., Singapore). This is because Singapore is close to the equator and satellites move faster over the equator. Thus satellites have less contact time with the ES located close to the equator. Since SiRRS does not transfer data among satellites and thus it performs bad for the ESs located close to the equator. From Fig. 11, we can see the two curves of SiRRS for Xian and NASA almost overlap with each other for the Globalstar (the same phenomenon for SiRRS with Xian and Singapore in Fig. 12 in the Iridium). This is because: In the Globalstar, the orbit inclination is 52° not 90°. So ESs located at similar latitude, such as Xi'an and NASA in Fig. 11, will have similar contact time with satellites; and b) In the Iridium, the orbit inclination is 90°. So, ESs located at the similar longitude, such as Xi'an and Singapore in Fig. 12, will have similar contact time with satellites.

3) The performances of CoDld method for the Globalstar and Iridium constellations are almost the same, but the performances of the SiRRS method for the Globalstar and Iridium constellations have some significant difference, which is exhibited from the curves in Fig. 11 and those in Fig.12. This is again because the CoDld method adjusts the amount of data carried by satellites through ISL data offloading regardless where the ESs are located. But, the performances of SiRRS are affected by the orbit inclinations of constellations.

VII. Conclusion

In this paper, we proposed a collaborative scheme that allows satellites to offload data among themselves by using the ISLs before they come to the contact with the ES, such that satellites will carry the right amount of data according to the length of their contact time with ES and the throughput of data downloading at ES is maximized. We developed an iterative optimization algorithm that jointly schedules data offloading among the satellites and data downloading from satellites to the ES. Extensive simulations have been conducted to evaluate our proposed scheme by using STK (Satellites Took Kit). The simulations are performed on two well-known LEO satellite constellations, Globalstar and Iridium. Simulation results have shown that our scheme can increase the throughput of data downloading significantly compared with the method that does not use the ISL data transfer.

References

[1] J. Wang, L. Li, and M. Zhou, "Topological Dynamics Characterization for LEO Satellite Networks," *Computer Networks*, vol. 51, no. 1, pp. 43–53, 2007.

[2] T. Gooley, J. Borsi, and J. Moore, "Automating Air Force Satellite Control Network (AFSCN) Scheduling," *Mathematical and computer modelling*, vol. 24, no. 2, pp. 91–101, 1996.

[3] L. Barbulescu, J.-P. Watson, L. D. Whitley *et al.*, "Scheduling Space–Ground Communications for the Air Force Satellite Control Network," *Journal of Scheduling*, vol. 7, no. 1, pp. 7–34, 2004.

[4] J. Meng-Gérard, P. Chrétienne, P. Baptiste, and F. Sourd, "On Maximizing the Profit of a Satellite Launcher: Selecting and Scheduling Tasks with Time Windows and Setups," *Discrete Applied Mathematics*, vol. 157, no. 17, pp. 3656–3664, 2009.

[5] F. Marinelli, S. Nocella, F. Rossi, and S. Smriglio, "A Lagrangian Heuristic for Satellite Range Scheduling with Resource Constraints," *Computers & Operations Research*, vol. 38, no. 11, pp. 1572–1583, 2011.

[6] S. Spangelo, J. Cutler, K. Gilson, and A. Cohn, "Optimization-Based Scheduling for the Single-Satellite, Multi-Ground Station Communication Problem," *Computers & Operations Research*, vol. 57, pp. 1–16, 2015.

[7] J. Castaing, "Scheduling Downloads for Multi-Satellite, Multi-Ground Station Missions," 2014.

[8] D. Smith, "Operations Innovations for the 48-Satellite Globalstar Constellation," in *Proceedings of International Communications Satellite Systems Conferences (ICSSC)*, 1996, pp. 537–542.

[9] S. R. Pratt, R. Raines, C. E. Fossa Jr *et al.*, "An Operational and Performance Overview of the IRIDIUM Low Earth Orbit Satellite System," *IEEE Communications Surveys*, vol. 2, no. 2, pp. 2–10, 1999.

[10] L. Wood, "Satellite Constellation Networks," in *Internetworking and Computing over Satellite Networks*. Springer, 2003, pp. 13–34.

[11] E. Ekici, I. Akyildiz, and M. Bender, "A Distributed Routing Algorithm for Datagram Traffic in LEO Satellite Networks," *IEEE/ACM Transactions on Networking*, vol. 9, no. 2, pp. 137–147, Apr 2001.

[12] R. Mauger and C. Rosenberg, "QoS Guarantees for Multimedia Services on a TDMA-Based Satellite Network," *IEEE Communications Magazine*, vol. 35, no. 7, pp. 56–65, Jul 1997.

[13] V. Gounder, R. Prakash, and H. Abu-Amara, "Routing in LEO-based satellite networks," in *Proceedings of Emerging Technologies Symposium Wireless Communications and Systems*, 1999, pp. 1–6.

[14] H. S. Chang, B. W. Kim, C. G. Lee *et al.*, "Topological Design and Routing for Low-Earth Orbit Satellite Networks," in *Proceedings of Global Telecommunications Conference (GLOBECOM)*, Nov 1995, pp. 529–535.

[15] M. Werner, "A Dynamic Routing Concept for ATM-Based Satellite Personal Communication Networks," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 8, pp. 1636–1648, 1997.

[16] S. Merugu, M. H. Ammar, and E. W. Zegura, "Routing in Space and Time in Networks with Predictable Mobility," Georgia Institute of Technology, Technical Report GIT-CC-04-7, 2004.

[17] M. Huang, S. Chen, Y. Zhu *et al.*, "Topology Control for Time-Evolving and Predictable Delay-Tolerant Networks," *IEEE Transactions on Computers (TOC)*, vol. 62, no. 11, pp. 2308–2321, 2013.

[18] C. Caini and R. Firrincieli, "Application of Contact Graph Routing to LEO Satellite DTN Communications," in *Proceedings of IEEE International Conference on Communications (ICC)*, June 2012, pp. 3301–3305.

[19] N. Bezirgiannidis, C. Caini, D. Padalino Montenero *et al.*, "Contact Graph Routing enhancements for delay tolerant space communications," in *Proceedings of Advanced Satellite Multimedia Systems Conference and the 13th Signal Processing for Space Communications Workshop (ASMS/SPSC)*, Sept 2014, pp. 17–23.

[20] G. Araniti, N. Bezirgiannidis, E. Birrane *et al.*, "Contact Graph Routing in DTN Space Networks: Overview, Enhancements and Performance," *IEEE Communications Magazine*, vol. 53, no. 3, pp. 38–46, March 2015.

[21] J. Seguí, E. Jennings, and S. Burleigh, "Enhancing Contact Graph Routing for Delay Tolerant Space Networking," in *in procedings of IEEE Global Telecommunications Conference (GLOBECOM)*, 2011, pp. 1–6.

[22] E. Birrane, S. Burleigh, and N. Kasch, "Analysis of the Contact Graph Routing Algorithm: Bounding Interplanetary Paths," *Acta Astronautica*, vol. 75, pp. 108–119, 2012.

[23] Z. Tang, Z. Feng, W. Han, W. Yu, B. Zhao, and C. Wu, "Improving the Snapshot Routing Performance through Reassigning the Inter-Satellite Links," in *2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2015, pp. 97–98.

[24] Z. Liu, J. Han, Y. Wang, X. Li, and S. Chen, "Performance Analysis of Routing Algorithms in Satellite Network Under Node Failure Scenarios," in *2014 IEEE Global Communications Conference*, 2014, pp. 2838–2843.

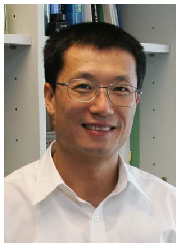
[25] J. Janhunen, J. Ketonen, A. Hulkkonen, J. Ylitalo, A. Roivainen, and M. Juntti, "Satellite Uplink Transmission with Terrestrial Network Interference," in *2015 IEEE Global Communications Conference (GLOBECOM)*, 2015, pp. 1–6.

[26] L. R. Ford Jr and D. R. Fulkerson, *Flows In Networks*. Princeton University Press, 2015.

[27] https://en.wikipedia.org/wiki/Systems_Tool_Kit.

[28] P. Metzen, "Globalstar satellite phased array antennas," in *Phased Array Systems and Technology, 2000. Proceedings. 2000 IEEE International Conference on*, 2000, pp. 207–210.

[29] K. Maine, C. Devieux, and P. Swan, "Overview of iridium satellite network," in *WESCON/'95. Conference record.'Microelectronics Communications Technology Producing Quality Products Mobile and Portable Power Emerging Technologies'*, 1995, p. 483.



Xiaohua Jia received his BSc (1984) and MEng (1987) from University of Science and Technology of China, and DSc (1991) in Information Science from University of Tokyo. He is adjunct with the Harbin Institute of Technology Shenzhen Graduate School, China while performing this work, and he is Chair Professor in the Department of Computer Science, City University of Hong Kong. His research interests include cloud computing and distributed systems, computer networks and mobile computing. Prof. Jia is an editor of IEEE Internet of Things, IEEE Trans. on Parallel and Distributed Systems (2006-2009), Wireless Networks, Journal of World Wide Web, Journal of Combinatorial Optimization, etc. He is the General Chair of ACM MobiHoc 2008, TPC Co-Chair of IEEE GlobeCom 2010 Ad Hoc and Sensor Networking Symp, Area-Chair of IEEE INFOCOM 2010, 2015-2017. He is Fellow of IEEE.

Tao Lv received the B.Sc (2014) degree in Computer Science from the Department of Information and Science in Nanjing Audit University. He is currently working toward the M.Eng degree at the Department of Computer Science, Harbin Institute Of Technology Shenzhen Graduate School, China. His current research interests include wireless ad-hoc networks and satellite networks.

Feng He received his B.Sc (2008) degree and M.Eng (2011) from Harbin Institute of Technology. He is currently working toward the Ph.D. degree at the Department of Computer Science, Harbin Institute of Technology Shenzhen Graduate School, China. His research interests include wireless ad-hoc networks and satellite networks.

Hejiao Huang received the Ph.D. degree in computer science from the City University of Hong Kong, Hong Kong, in 2004. She is currently a Professor with the Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, China, and was an Invited Professor with the Institut National de Recherche en Informatique et Automatique, Rennes, France. Her research interests include cloud computing, trustworthy computing, formal methods for system design, and wireless networks.