Delay Minimization for Progressive Construction of Satellite Constellation Network

Zhe Liu, Wei Guo, Weisheng Hu, and Ming Xia

***Abstract*—During the construction of satellite constellation, communication relay service could be provided by a partially constructed constellation through buffering data and forwarding it at a later time. However, this will introduce extra end-to-end delay during the construction period. We propose a scheme to determine an optimal construction process, which is used to insert satellites into appropriate positions in each stage of construction. Simulation results show that our scheme can reduce the average end-to-end delay by 70.1% compared to the traditional scheme.**

***Index Terms*—Progressive construction, satellite constellation network, end-to-end delay.**

I. Introduction

Satellite constellation networks, such as Iridium-Next [1], Globalstar-2 [2] and Orbcomm-2 [3], typically need several stages to complete the construction. Each stage usually takes several months, during which only a few satellites could be inserted into target positions. In the constellation construction, a partially formed constellation could buffer data on certain satellites, and forward the data at a later time when the links between satellites and users become available [4]–[6]. This strategy could provide relay services to users in areas where terrestrial communication infrastructure is not available. However, extra end-to-end delay is introduced in this period. Moreover, different schemes for constructing constellation may result in different intermediate satellite networks, hence yielding different end-to-end delay. In fact, a carefully planned process for the satellite network will serve users with low end-to-end delay, which is an important service target in [7].

It is necessary to understand how a constellation should be progressively constructed over time. For example, Fig. 1(a) and (b) show two partially formed constellations with different construction processes. In each stage, the process in Fig. 1(a) inserts satellites into different planes, while the process in Fig. 1(b) inserts satellites into one plane. By the end of the first stage, Fig. 1(c) and (d) show the time windows of ground-satellite links (GSLs) within the case of Fig. 1(a) and (b), respectively. In Fig. 1(c), u_1 will wait until t_1 to upload data to s_{21} when the data arrive at u_1 on t_0. Then, the uploaded data needs to be carried and stored on s_{21} for a time period T_1 due to such long distance between u_1 and u_2. Finally, the data can be downloaded to u_2 as early as at t_2. Similarly, if the process in Fig. 1(b) is taken, data will arrive

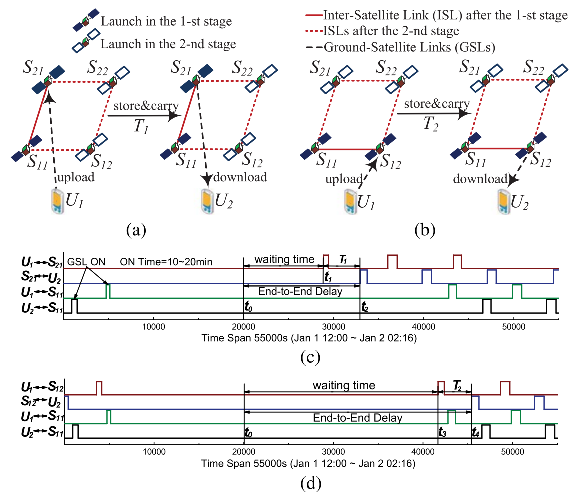

Fig. 1. Example of constellation construction.

at u_2 as early as at t_4. Due to $t_4 \gg t_2$, it is reasonable to claim that the former construction process is better than the latter.

In this letter, we propose a scheme to minimize the delay by designing an optimal construction process. We also evaluate the delay performance in the construction period. In each stage, our scheme decides the positions which satellites should be inserted into, and we analyze its fuel cost and time cost. Finally, simulation results indicate that our scheme can reduce average end-to-end delay by 70.1%, compared to the traditional scheme which inserts the satellites into one orbital plane in each stage.

II. Problem Statement

A. Network Model

Time-varying graph (TVG) is a useful high level abstraction for studying connectivity over time in the satellite network. Recently, the authors in [8] have proposed a unified framework for TVGs, and following this framework, we will use their definitions and notations in our analysis.

Let U be a set of M users, namely $U = \{u_1, \cdots, u_M\}$. Satellites will be launched and injected into N positions which are evenly distributed among P orbits. Satellite in the j-th position of the i-th orbit is denoted as $s_{(i,j)}$. As a result, the set of these N satellites is denoted as $S = \{s_{(i,j)} | 1 \le i \le P, i \le j \le N/P\}$. Vertices representing the satellites and users are denoted as $V = U \cup S$. There are no edges between users. Users do not relay messages between satellites, not like the users in [4]. Thus, the set of possible edges between vertices and satellites is denoted as $E = \{(v, s) : v \in V, s \in S\}$. Events occur over a time span $\mathcal{T} \subseteq \mathbb{T}$, where $\mathbb{T}$ is the temporal domain. In a general case, a TVG is denoted as a tuple:

$$\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta). \quad (1)$$

$\rho : E \times \mathcal{T} \to \{0, 1\}$ is called presence function of edge, which indicates whether a given edge $e = (v, s)$ exists at a given time.

Manuscript received February 14, 2015; revised July 11, 2015; accepted July 24, 2015. Date of publication July 29, 2015; date of current version October 8, 2015. This work was supported by the National Science Foundation of China under Grants 61471238, 61433009 and by CAST Foundation. The associate editor coordinating the review of this paper and approving it for publication was G. Giambene.

Z. Liu, W. Guo, and W. Hu are with the State Key Lab of Advanced Optical Communication Systems and Networks, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: wguo@sjtu.edu.cn).

M. Xia is with Ericsson Research, San Jose, CA 95134 USA.

Digital Object Identifier 10.1109/LCOMM.2015.2462358

$\zeta : E \times \mathcal{T} \to \mathcal{T}$ is called latency function of edge, which indicates the transmission latency over a given edge at a given time. $\psi : V \times \mathcal{T} \to \{0, 1\}$ is called presence function of vertex, which indicates whether a given vertex exists at a given time. The bandwidth of edge e at a given time t is assumed to be $B(e, t)$.

The construction of satellite constellation network includes K stages, where $K \geq 1$. The time for k-th stage is denoted as $[t_k^1, t_k^2)$ with a duration $T_k = t_k^2 - t_k^1$. During the k-th stage, a set of L_k satellites will be launched at time t_k^1, and will be inserted into their target positions at time t_k^2, namely $S_k = \{s_1, \cdots, s_{L_k}\}$. These satellites will be able to provide communication relay service at time t_k^2, namely $\psi(s, t) = 1, t \geq t_k^2, s \in S_k$. Furthermore, each user exists during the whole time span, namely $\psi(u, t) = 1, \forall t \in \mathcal{T}$.

When satellite s is available at t_k^2, there exist n time windows on edge e connecting satellite s and vertex v, namely:

$$W_{t_k^2}(e) = [t_1, t_2) \cup \cdots \cup [t_{2n-1}, t_{2n}), t_1 \geq t_k^2. \tag{2}$$

We define the gap between two neighbor windows as:

$$g_i(e) = t_{2i+1} - t_{2i}, i \leq i \leq n - 1. \tag{3}$$

B. Routing in the Satellite Network

In TVG $\mathcal{G}$, there are $|R|$ requests generated during time span $\mathcal{T}$. Each request $r \in R$ is represented as $r = (s, d, t_s, TTL, m)$, where s and d are the source and the destination, t_s is the arrival time, TTL is the time-to-live parameter, and m is the message size.

We use a modification of Dijkstra's shortest path routing algorithm to compute Earliest Delivery paths, named DED [9]. DED algorithm uses the knowledge of time windows on all edges. In the algorithm description, $W_{c(u)+t_s}^{t_s+TTL}(e)$ is the time windows of e in $[c(u)+t_s, t_s+TTL)$. Line 8 checks each time window $[t_{2i}, t_{2i+1})$ which is belonged to $W_{c(u)+t_s}^{t_s+TTL}(e)$. In line 9, $t_{2i} - c(u) - t_s$ is the period of data stored at vertex u, namely the waiting time for a valid time window of edge e, while $\zeta(e, t_{2i})$ is the propagation delay of edge e at time t_{2i}. Thus, a message will reach vertex v at time $t_{2i} + \zeta(e, t_{2i})$ if edge e is chosen. This algorithm differs from the original Dijkstra's algorithm in lines 7 and 8, as well as no replacement waiting cost and cost function in lines 9 and 10.

Algorithm 1 DED Algorithm

Input: $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta)$, $r = (s, d, t_s, TTL, m)$.
Output: $\mathcal{J} = (e_1, t_1), (e_2, t_2), \cdots, (e_h, t_h)$.
1: $V_1 \leftarrow V$.
2: $c_s \leftarrow 0, c_v \leftarrow \infty, \forall v \in V$ s.t. $v \neq s$.
3: **while** $V_1 \neq \emptyset$ **do**
4: Let $u \in V_1$ be the vertex s.t. $c(u) = \min_{x \in V_1} c(x)$.
5: $V_1 = V_1 - \{u\}$.
6: **for** each edge $e \in E$ s.t. $e = (u, v)$ **do**
7: **if** $W_{c(u)+t_s}^{t_s+TTL}(e) \neq \emptyset$ **then**
8: **for** each $[t_{2i}, t_{2i+1}) \in W_{c(u)+t_s}^{t_s+TTL}(e)$ **do**
9: **if** $c(v) > t_{2i} - t_s + \zeta(e, t_{2i})$ **then**
10: $c(v) \leftarrow c(u) + (t_{2i} - c(u) - t_s) + \zeta(e, t_{2i})$.
11: **end if**
12: **end for**
13: **end if**
14: **end for**
15: **end while**

DED calculates a journey $\mathcal{J}$ for each r, namely a sequence of h couples (e, t) between vertices s and d:

$$\mathcal{J} = (e_1, t_1), (e_2, t_2), \cdots, (e_h, t_h). \tag{4}$$

The difference between $\mathcal{J}$ and conventional path $e_1 \to e_2 \to \cdots \to e_h$ is that $\mathcal{J}$ must satisfy the constraints of $t_{i+1} \geq t_i + \zeta(e_i, t_i)$ for all $i < h$. The time needed for a message travel along journey is:

$$|\mathcal{J}|_t = t_h + \zeta(e_h, t_h) - t_s. \tag{5}$$

Journey $\mathcal{J}$ is valid for request r if $|\mathcal{J}|_t \leq TTL$. For each tuple (e_i, t_i), we can find a time window $[t_i^1, t_i^2)$ for edge e_i such that $\rho_{[t_i^1, t_i^2)}(e_i, t_i) = 1, t_i \in [t_i^1, t_i^2)$. Thus, the maximum capacity volume of edge e_i during time window $[t_i^1, t_i^2)$ is:

$$C(e_i) = \int_{t_i}^{t_i^2} B(e_i, t)\, dt. \tag{6}$$

The capacity of journey $\mathcal{J}$ is denoted as:

$$C(\mathcal{J}) = \min_{1 \leq i \leq h} C(e_i). \tag{7}$$

After DED calculates a valid journey for request r, the time windows of edge e_i along journey $\mathcal{J}$, used to send message of $C(\mathcal{J})$ size, is removed from the available time windows of edge e_i along journey $\mathcal{J}$ in TVG $\mathcal{G}$. As a result, the time windows of edge e_i will not be shared by other journeys. If $C(\mathcal{J})$ is smaller than m, DED routes another valid journey for r based on the updated TVG $\mathcal{G}$. Therefore, request r can be satisfied if there exist n valid journeys such that $\Sigma_{i=1}^{n} C(\mathcal{J}) \geq m$. The size of message travel along the n-th journey is $m - \Sigma_{i=1}^{n-1} C(\mathcal{J})$. The end-to-end delay of request r is calculate as the time of n valid journeys ($n > 1$), namely:

$$D = \Sigma_{i=1}^{n-1} \frac{C(\mathcal{J}_i)}{m} |\mathcal{J}_i|_t + \frac{m - \Sigma_{i=1}^{n-1} C(\mathcal{J}_i)}{m} |\mathcal{J}_n|_t. \tag{8}$$

Thus, the objective of progressive construction of satellite constellation network is to determine an optimal construction process $\{s_1, \cdots, s_{L_1}\}_1, \cdots, \{s_1, \cdots, s_{L_K}\}_K$ such that the overall end-to-end delay for all requests is minimized.

III. Progressive Construction Scheme

In this section, we describe our scheme to insert satellites into appropriate target positions belonging to different orbital planes. And then, we analyze the extra-fuel cost and the time deployment cost to achieve our scheme.

A. Progressive Construction Scheme

If the time windows on edges between users and satellites could be distributed more evenly, the waiting time for users to access to satellites, as the main source of end-to-end delay, can be reduced. Thus, it is important to insert satellites into their target positions with a better time window distribution in each stage. In order to achieve this goal, we will use the standard deviation (STD) of gaps between the time windows, which has the same units as the end-to-end delay and can reflect the distribution situation of the time windows, on edge e after time t_{k-1}^2 to characterize $W_{t_{k-1}^2}$:

$$\sigma_{t_{k-1}^2}(e) = \sqrt{\frac{\Sigma_{i=1}^{n-1}\left(g_i(e) - \overline{g(e)}\right)^2}{n-1}}, \tag{9}$$

where $\overline{g(e)}$ is equal to $\frac{\Sigma_{i=1}^{n-1} g_i(e)}{n-1}$.

$|S'_{k-1}|$ satellites can provide relay services at t_{k-1}^2:

$$S'_{k-1} = \begin{cases} \emptyset, & k = 1, \\ S_1 \cup \cdots \cup S_{k-1}, & k > 1. \end{cases} \tag{10}$$

The edges connecting user u and satellite s in S'_{k-1} are denoted as $E^u = \{(u, s) : s \in S'_{k-1}\}$. Therefore, after time t_{k-1}^2, the time windows of E^u becomes $W_{t_{k-1}^2}(E^u) = W_{t_{k-1}^2}(e_1) \cup \cdots \cup W_{t_{k-1}^2}(e_{|S'_{k-1}|})$. Furthermore, all the time windows of edge E^u after time t_{k-1}^2 can be represented as $W_{t_{k-1}^2}(E^u) = W_{S'_{k-1}}(u)$. Therefore, the STD of gaps between time windows on E^u after time t_{k-1}^2 is represented as $\sigma_{S'_{k-1}}(u)$. The average STD of gaps for M users after time t_{k-1}^2 is calculated as follows:

$$\overline{\sigma_{S'_{k-1}}(u)} = \frac{\Sigma_{i=1}^{M}\sigma_{S'_{k-1}}(u_i)}{M}. \tag{11}$$

We use a scheme called min σ, to decide the L_k positions belonged to N_P different orbital planes that satellites should be inserted into during stage k.

Step 1: Available satellites in subset S_k are distributed among $P(S_k)$ orbital planes. For example, satellite s_{11} and s_{31} are distributed among $P(S_k) = 3$ adjacent planes: planes 1, 2 and 3. Initially, we set $S_k = \emptyset$, resulting in $P(S_k) = 0$.

Step 2: There exists a subset $S^* \subseteq S$ at time t_k^2 such that any position $s \in S^*$ satisfies the constraint $P(S_k \cup \{s\}) \leq N_P$. For any $s \in S^*$, we can find $s' \in S'_{k-1}$ such that an ISL $e = (s, s')$ could be maintained after time t_k^2. If $S'_{k-1} = \emptyset$, S^* is set as S. If position $s \in S^*$ is injected into a satellite during stage k, the satellite subset available at time t_k^2 is $S'_k = S'_{k-1} \cup S_k \cup \{s\}, s \in S^*$. Time windows on $E^u = \{(u, s) : s \in S'_k\}$ after time t_k^2 are $W_{S'_k}(u) = W_{t_k^2}(e_1) \cup \cdots \cup W_{t_k^2}(e_{|S'_k|})$, where $e_{|S'_k|-1} = (u, s_{|S'_{k-1}|})$ and $e_{|S'_k|} = (u, s)$.

Step 3: s^* is the position such that average STD of gaps between time windows for M users after time t_k^2 is minimized, namely $\overline{\sigma_{S'^*_k}(u)} = \min_{s \in S^*} \overline{\sigma_{S'_k}(u)}$. Thus, s^* is the position that satellite should be inserted into during stage k, namely $S_k = S_k \cup \{s^*\}$.

Step 4: Redo **Steps 2** and **3** until $|S_k| = L_k$.

Fig. 2 shows the time windows of four edges. For simplicity, 1 satellite is launched during each stage. According to **Step 2**, we can set $S'_0 = \emptyset$ and $S^* = \{s_1, s_2\}$. Thus, $S'_1 = \{s_1\}$ will result in $\overline{\sigma_{S'_1}(u)} = 10.39$, while $S'_1 = \{s_2\}$ will result in $\overline{\sigma_{S'_1}(u)} = 0$. According to **Step 3**, satellite should be inserted into s_2 during stage 1. Similarly, satellite should be inserted into s_1 during stage 2.

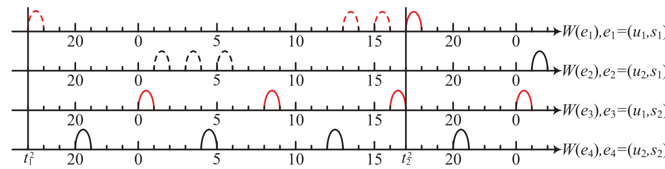

Fig. 2. The time windows of four edges.

TABLE I
THE TIME DEPLOYMENT AND FUEL COST

N_P	1	2	3	4	5	6
$m_f/m_s(\%)$	0	5.8	5.8	5.8	5.8	5.8
T_D(day)	0	57.79	115.58	173.37	231.16	288.95

B. Cost Analysis of Extra-Fuel and Time Deployment

In each stage, target positions in our scheme may belong to N_P different orbital planes. To achieve this goal, the oblateness of the Earth gravity could be used to achieve right ascension ascending node (RAAN, Ω) separation. Thus, all the satellites can be injected into the same drifting orbit, and each satellite raises itself to its target positions when the drifting and the target orbital planes overlap.

The drift rate of Ω for a near-circular orbit is model [10]:

$$\Omega' \cong -2.0647 \times 10^{14} \cdot a^{-3.5} \cdot \cos i, \tag{12}$$

where Ω' is the drift of RAAN per day, a is the semi-major axis in kilometers, and i is the inclination. Thus, time T_P for achieving desired RAAN separation $\Delta\Omega$ is calculated as: $T_P = \Delta\Omega/\Delta\Omega'$, where $\Delta\Omega' = \Delta\Omega'_d - \Delta\Omega'_t$ is the difference in the drift rate between the drifting and target orbits. Moreover, the difference in the orbital velocity between the drifting orbit and the target orbit is $\Delta V_{d\to t} = \sqrt{\mu_E/a_d} - \sqrt{\mu_E/a_t}$, where μ_E is the gravitational constant $3.986 \times 10^5 \text{km}^3/\text{s}^2$. Thus, the minimum additional mass of fuel m_f to achieve $\Delta V_{d\to t}$ from a specific impulse I_{sp} of propulsion system is:

$$\frac{m_f}{m_f + m_s} = \exp\left[\frac{-\Delta V_{d\to t}}{g \cdot I_{sp}}\right], \tag{13}$$

where $g = 9.8$ m/s^2 is the gravitational acceleration and m_s is dry mass of satellite. Time cost T_H for orbit transferring is $T_H = \pi\sqrt{(a_d + a_t)^3/8\mu_E}$.

Satellite should be injected into an appropriate position in the drifting orbit at the beginning of each stage such that the difference in phasing between target position and the satellite in drifting orbit at the beginning of orbit transferring is $\Delta\theta = \pi - \pi\sqrt{((a_t + a_d)/2a_t)^3}$. As a result, the satellite will be exactly inserted into target position in target orbit at the end of orbit transferring. Moreover, the minimum time duration needed to insert L_k satellites into N_P different orbital planes is $T_D = (N_P - 1) \cdot T_P + T_H$.

IV. Numerical Analysis

We firstly analyze the cost for fuel and time deployment to insert satellites into different planes in each stage, and then evaluate the performance in the construction period.

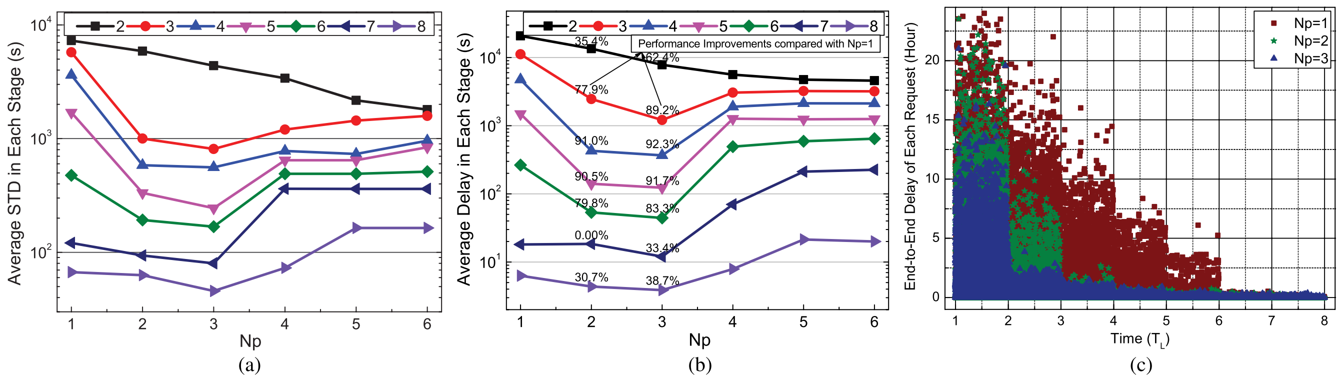

Fig. 3. The results of min schemes with different N_P: (a) average STD in each stage for min σ scheme with $N_P = 1 \sim 6$; (b) average delay in each stage for min σ scheme with $N_P = 1 \sim 6$; (c) end-to-end delay of each request for min σ scheme with $N_P = 1 \sim 3$ during $[T_L, 8T_L)$, $T_L = 120$ days.

We consider a satellite constellation that eventually consists of 48 satellites with an even distribution among 8 orbits within $i = 52°$ and altitude $h = 1410$ km [2]. Each satellite could establish 4 ISLs at most [11]. The time duration of each stage is the same, namely $T_L = t_k^2 - t_k^1$. There are 8 stages, and in each stage, 6 satellites will be launched at time t_k^1, and provide relay service for users at time t_k^2. Requests arrival follows a Poisson distribution with $\lambda = 100$/h. Source and destination are uniformly selected among 109 gateways, following the same geographical distribution as described in [4]. We set *TTL* as 24 hours, message size of request as 20 MB, bandwidth of GSLs as 8 Mbps, and bandwidth of ISLs as 100 Mbps.

During each stage, 6 satellites will be firstly injected into a drifting orbit with $h = 920$ km and $i = 52°$, and each satellite will raise itself into its target position [2]. Since each satellite is equipped with 100% extra-fuel [12], it is acceptable to achieve orbit transferring by consuming 5.8% extra-fuel ($I_{SP} = 430$ s). Moreover, the time needed to achieve RAAN separation of 45° is 57.79 days. As a result, Table I shows the detailed time deployment and fuel cost.

Supposed that T_L is long enough to insert 6 satellites into 6 orbital planes during each stage. Fig. 3(a) show the average STD in each stage by min σ schemes with $N_P = 1 \sim 6$. Because there are no satellites available in stage 2, using more planes will reduce the average of STD. As a result, the average STD will reduce with the increase of $N_P = 1 \sim 6$ in stage 2. After stage 2, more planes are used and more satellites are inserted into one plane simultaneously, hence achieving a smaller average STD. The min schemes with $N_P = 1 \sim 2$ mainly insert satellites into one plane without using more planes, while the min schemes with $N_P = 4 \sim 6$ mainly use more planes without inserting as many satellites into one plane as possible. However, the min scheme with $N_P = 3$ could strike a balance between above two factors. As a result, the average STD of min scheme with $N_P = 3$ will the minimum among $N_P = 1 \sim 6$. Moreover, the smaller average STD is, the more evenly distributed time windows on edges between users and satellites can be achieved, hence yielding lower waiting time, which is the main part of end-to-end delay. Thus, the pattern of curves within Fig. 3(b) is similar to that within Fig. 3(a), meaning that the minimization of the STD of gaps leads to minimization of the delay.

However, time duration between satellite launches is usually months. Thus, it is reasonable to set $T_L = 120$ days, which makes $N_P \leq 3$. Fig. 3(c) shows the end-to-end delay of each request generated during $[T_L, 8T_L)$, where claret squares with $N_P = 1$ represents a traditional scheme which insert satellites into the same plane in each stage. With the increase of available satellites, there are more time windows to forward data. As a result, end-to-end delays of $N_P = 1 \sim 3$ become smaller.

V. Conclusion

In this letter, we propose a novel scheme to progressively construct satellite constellation network. Our scheme inserts satellites into the positions, that could belong to different orbital planes. The target positions are at most distributed among 3 orbital planes subject to time deployment constraint. Our simulation results indicate that, the end-to-end delay with 2 and 3 orbital planes can be reduced by 57.6% and 70.1% in average, in comparison with the single orbital plane.

References

[1] Iridium-Next launch schedule, 2015. [Online]. Available: http://www.skyrocket.de/space/doc_dat/iridium-next.htm
[2] Globalstar-2 launch schedule, 2010–2015. [Online]. Available: http://www.skyrocket.de/space/doc_sdat/globalstar-2.htm
[3] Orbcomm-2 launch schedule, 2012–2014. [Online]. Available: http://www.skyrocket.de/space/doc_sdat/orbcomm-2.htm
[4] C. Hugo *et al.*, "Operator calculus approach to minimal paths: Precomputed routing in a store and forward satellite constellation," in *Proc. IEEE Globalcom*, Anaheim, CA, USA, Dec. 2012, pp. 3431–3436.
[5] C. Caini *et al.*, "Application of contact graph routing to LEO satellite DTN communications," in *Proc. IEEE ICC*, Ottawa, ON, Canada, Jun. 2012, pp. 3301–3305.
[6] W. Ivancic *et al.*, "Experience with Delay-Tolerant Networking from orbit," *Int. J. Satellite Commun. Netw.*, vol. 28, no. 5/6, pp. 335–351, Sep.–Dec. 2010.
[7] I. Psaras *et al.*, "Delay-/disruption-tolerant networking: State of the art and future challenges," Univ. Surrey, Surrey, U.K., Tech. Rep., 2010.
[8] C. Arnaund *et al.*, "Time-varying graphs and dynamic networks," in *Ad-hoc, Mobile, Wireless Netw.* Berlin, Germany: Springer-Verlag, 2011, pp. 346–359.
[9] S. Jain *et al.*, "Routing in a delay tolerant network," in *Proc. SIGCOMM*, Portland, OR, USA, Aug. 30–Sep. 3, 2004, pp. 145–158.
[10] N. J. Beidleman *et al.*, "Scheme and Apparatus for Deploying a Satellite Network," U.S. Patent No. 5 199 672, Apr. 6, 1993.
[11] Z. Liu *et al.*, "Perfect match model based link assignment for optical satellite network," *Proc. IEEE ICC* Sydney, NSW, Australia, Jun. 2014, pp. 4149–4153.
[12] *The Information of Artificial Satellites 1957–2015*. [Online]. Available: http://www.satlist.nl/