Temporal Netgrid Model based Routing Optimization in Satellite Networks

Jian Li, Hancheng Lu, Yali Wang
University of Science and Technology of China, Hefei, China, 230027
lijian9@mail.ustc.edu.cn, hclu@ustc.edu.cn, ylwang15@mail.ustc.edu.cn

***Abstract*—With global coverage abilities, satellite networks are expected to provide users with ubiquitous data services. However, routing in satellite networks faces more challenges due to the satellite movement. Fortunately, the satellite movement can be predicted by orbit calculation. Based on this characteristic, a lot of existing routing algorithms use Temporal Graph Model (TGM) to calculate instantaneous satellite network topologies at discrete times as priori knowledge for routing decision. In this case, high computation cost will be involved. In this paper, we propose a novel Temporal Netgrid Model (TNM) to represent the time-varying satellite network topology. In TNM, the whole space is divided into small cubes (i.e. netgrids) and then satellites can be located by netgrids instead of coordinates. By doing so, TNM reduces the computation complexity from $\mathcal{O}(N^2)$ to $\mathcal{O}(N)$ compared with TGM. Furthermore, an Earliest Arrival Space Routing (EASR) algorithm is proposed, which attempt to find the earliest arrival paths from the source node to any other reachable nodes with low computation cost. Simulations are performed to validate the effectiveness of the proposed routing algorithm. Results show that EASR algorithm achieves a significant reduction in computation complexity as well as acceptable routing performance in terms of data delivery ratio.**

***Index Terms*—Satellite Networks, Temporal Netgrid Model, Temporal Graph Model, Routing**

I. Introduction

Satellite networks are widely regarded as an essential component in next-generation information networks due to the ability of achieving ubiquitous communication with low cost. In regional areas where network facilities are not available, such as desert, ocean and polar region, satellite networks can assure users with a consistent level of services and are expected as an indispensable supplement of providing seamless service [1][2].

In recent years, the research on Low Earth Orbit (LEO) small satellites has emerged as a hot topic due to economical prospect and convenience in launching and maintenance of such kind of satellites. It is a good choice to construct the constellation of LEO small satellites in order to achieve tasks such as navigation, communications and scientific research [3]. Therefore, the scale of satellite networks is increasingly extended because of the large number of small satellites, and the interconnectivity pattern of small satellites becomes more complicated. In this case, how to deliver data over time-varying links becomes a challenging problem, which determines the performance of LEO satellite networks [4]. Considering the resource constraints on the small satellite, it is an important issue to provide an efficient routing algorithm which can find the optimum path consisting time-varying links.

The dynamic nature of the satellite network topology brings more challenges in routing. Due to the intermittent characteristic of links in satellite networks, the routing algorithms in terrestrial networks are inefficient because they do not consider the temporal aspect of the network topology. In some cases, the end-to-end path even does not exist at a moment. Therefore the routing algorithm applied in satellite networks must consider the temporal aspect of the network topology.

Recently, Contact Graph Routing (CGR) is widely used in Delay Tolerant Networks (DTN) [5], the performance evaluation on CGR in LEO satellite communications and a survey of CGR in DTN space networks were made in [6] and [7], respectively. However, CGR follows the Dijkstra's greedy approach of routing, and this greedy algorithm does not result in optimal delivery ratio in some cases [8]. Inspired by this, [9] proposed an EAODR routing algorithm using temporal graph to solve the inefficiency of greedy approach in choosing the earliest transmission time adopted in CGR. The temporal graph, also called time-varying graph [10], is also adopted by lots of works and can provide the detailed information of the network topology, [11] mathematically formalized the computation of end-to-end delay based on time-varying graph, and propose a routing policy to optimize traffic delivery delay.

As we described, lots of existing works use the contact graph or temporal graph to describe the time-varying topology of satellite networks, most of them consider the graph as priori information. It is not practical to make such assumption in large-scale satellite networks especially that consisting of resource-constrained small satellites.

To address this issue, we propose a Temporal Netgrid Model (TNM) to describe the time-varying satellite network topology which can decease the computation complexity to $\mathcal{O}(N)$. Based on TNM, we also propose an efficient routing algorithm which can find the earliest arrival path. The main contributions of this paper are described as follows.

1) In order to describe the time-varying network topology, TNM is proposed. This model uses cubes (i.e. netgrids) to replace the coordinate of satellite nodes, so it can decrease the computation complexity significantly compared to the Temporal Graph Model (TGM). We also propose a multi-layer netgrid partition method and analyse the performance of TNM in a theoretical manner.
2) Based on TNM, a routing algorithm is proposed in order to solve the optimal path-finding problem in satellite

978-1-4673-8999-0/17/$31.00 ©2017 IEEE

networks. We prove that the proposed routing algorithm can find the earliest arrival path in time-varying networks.
3) We perform simulations to evaluate the proposed algorithm and compare with traditional routing algorithms based on TGM. Results show that the algorithm based on TNM can approach the optimal performance and achieve a significant reduction in computation complexity.

The rest of this paper is organized as follows. In Section II, problems about existing works are described. In Section III, we describe the principle of TNM and analyse the computation complexity. Then a routing algorithm based on TNM is proposed in Section IV, and simulation results are shown in Section V. At last, conclusions are drawn in Section VI.

II. Problem Statement

Due to the dynamic network topology of satellite networks, traditional terrestrial routing strategies, such as AODV and OSPF, are not applicable in satellite networks. In order to find an optimum path, the routing algorithm must consider the temporal aspect of satellite networks. Thus, a network topology model which can provide detailed topology information is required. Based on the predictable characteristic of satellite orbits, there exist several network topology models which can describe time-varying satellite networks, such as contact graph and temporal graph. Relying on the topology information provided by these models, the routing algorithm can find optimal path from source to destination, which makes the topology model directly influence the performance of routing algorithms.

In order to describe time-varying satellite networks, graph based network topology models such as contact graph and temporal graph are widely used [7][9]. Due to the fact that the communication contact of CGR is scheduled in practice, CGR does not need to calculate contact graph. Since we consider onboard topology computation and routing in satellite networks, only TGM is considered in this paper.

By adding time as another dimension in the static graph, TGM can capture the state of time-varying networks at any specific time. It can also be represented as an ascending time ordered set of sub-graphs, which can be written as $\{G_1, G_2, ...G_{M'}\}$ and each sub-graph $G_m = \{V, E\}$, where M' is the number of time spans through which the graph changes consecutively and that are not overlapping [9]. Intuitively, vertices and edges in the sub-graph are static in each time span. In order to get one sub-graph G_m, the distance between each two vertices should be calculated iteratively based on their coordinates to get all available edges in G_m. Assuming the number of vertices as N, then the computation complexity of a sub-graph and the whole temporal graph calculation should be $\mathcal{O}(N^2)$ and $\mathcal{O}(N^2 \cdot M')$, respectively. With the increase of N, the computation cost will rise rapidly.

However, high computation complexity of temporal graph calculation in TGM hinders the application of this model in practice. As the scale of satellite networks is increasingly extended due to the large number of small satellites, considering temporal graph as priori information in existing works is unreasonable regarding resource constraints on small satellites. In this case, a network topology model which can describe the time-varying topology with low computation complexity is in need.

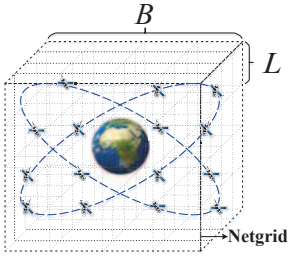


Fig. 1. The illustration about the basic idea of TNM.

III. Temporal Netgrid Model

The basic idea behind TNM is to divide the whole space into cubes (i.e. netgrids) and then locate satellites by netgrids instead of coordinations in TGM. By doing so, the computation complexity of TNM is reduced to $\mathcal{O}(N)$. In this section, the definition and partition method of netgrid are described.

A. Definition for Netgrid

In TNM, the whole space where satellites' orbits exist is divided into a number of cubes, then satellite nodes are allocated into netgrids according to their coordinates. After allocation of all satellite nodes, the three-dimensional coordinates of satellite nodes is replaced by coordinate ranges of netgrids.

Considering a satellite network as shown in Fig.1, the whole space where satellites' orbits exist is represented by a giant cube, and the side length of this giant cube is represented as B, which is associated with orbit radius, representing the boundary of this space. The shape of each netgrid in TNM is also adopted as a cube in order to easily distinguish the boundary, and the side length of each cube is assumed as L. Let K represents the number of cubes inside the three-dimensional space in one dimension, then the whole space can be divided into $K^3 = \lceil \frac{B}{L} \rceil^3$ cubes (i.e. netgrids). In order to store the containment relationship between satellite nodes and netgrids, we adopt a netgrid table $\mathcal{G}$ which can be represented as an ascending time ordered set of sub-tables, i.e., $\mathcal{G} = \{\mathcal{G}_1, \mathcal{G}_2, ..., \mathcal{G}_M\}$, where M is the number of time spans which is same with TGM. In order to identify their location in the space, x, y, z is allocated as the discrete coordinate of each netgrid and $x, y, z \in \mathbb{N}$, then all nodes contained by one netgrid can be identified by one discrete coordinate. Hence, the containment relationship between netgrids and satellite nodes can be represented as

$$\mathcal{C}_{x,y,z} = \{v_1, v_2, ..., v_n \mid v_n \in V, n < N\}, x, y, z \leq K,$$

and the a sub-table of the netgird table can be represented as

$$\mathcal{G}_m = \{\mathcal{C}_{x,y,z} \mid x, y, z \leq K\},$$

where m denotes the specific time span and $m \in \{1, 2, ..., M\}$.

In this way, the sub-table $\mathcal{G}_m$ replaces the sub-graph in temporal graph and the netgrid table $\mathcal{G}$ contained by an ascending time ordered set of sub-tables replaces the whole temporal graph G in TGM. Note that in different time spans, the partition of netgrids is the same and the only difference of each sub-table is that the containment relationship between netgrids and satellite nodes $\mathcal{C}_{x,y,z}$ is changing due to the satellite movement. Algorithm 1 describes the detailed calculation process of the netgrid table.

After the partition of netgrids, for a satellite node contained by a netgrid, we approximate its neighbor netgrids to spherical communication coverage of this node. In this case, if neighbor netgrids contain another satellite node, we can assure that these two satellite nodes can communicate with each other. Above all, we replace distance calculation in TGM with netgrid table reading operation to reduce the complexity. The specific partition method of TNM is described in Section III-B.

Algorithm 1: Calculation process of the netgrid table

Input: Satellite nodes set V, time spans $\mathrm{T} = \{[t_1, t_2), [t_2, t_3), ..., [t_{M-1}, t_M)\}$;
Output: Netgrid table $\mathcal{G}$;

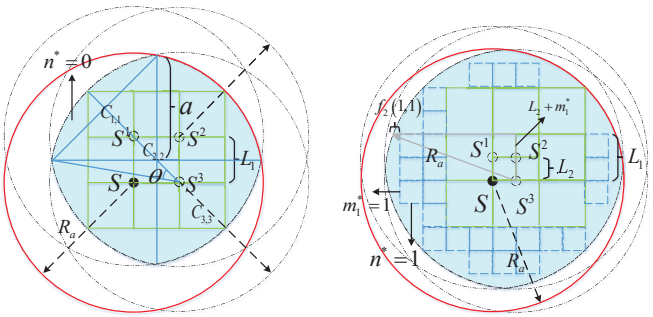
```
foreach [t_m, t_{m+1}) ∈ T do
    foreach v ∈ V do
        Calculate the location of node v, and allocate v
        into netgrid C_{x,y,z};
    end
    Store non-empty netgrids in the sub-table G_m;
end
Return G = {G_1, G_2, ..., G_M}
```

B. Multi-layer Netgrid Partition

Although the principle of TNM and the definition of the netgrid table are given, the value of L is still uncertain. By replacing the coordinate of satellite nodes by netgrids, the side length L of each netgrid determines the accuracy of TNM to describe the time-varying network topology. As the routing algorithm relies on the information provided by TNM to find path from source to destination, the routing performance will be influenced by the value of L. In this way, we will propose a multi-layer netgrid partition method and give the definition about the accuracy of TNM.

We take an example to explain the principle of our multi-layer partition method in Fig.2. For clarity, the example is shown as a two-dimensional form which can be seen as cross-section drawing of Fig.1. Squares which are drawn in Fig.2 represent cubes (i.e. netgrids) in TNM.

As we can see in Fig.2(a), the cube in the center of netgrids is represented by $\mathcal{C}_{2,2}$ and a satellite node S is located at one vertex of cube $\mathcal{C}_{2,2}$. We don't know the exact location of S inside the cube according to the netgrid table. So we take all vertices of $\mathcal{C}_{2,2}$ which are S, S^1, S^2 into consideration, S^3 and separately draw circle in the centre of them. Then we can get an overlapping area which is shown as colorful area in Fig.2(a). This area represents the effective communication coverage of S due to the fact that the exact location of S inside $\mathcal{C}_{2,2}$ is unknown. In this way, no matter where exactly S is, we can assure that all satellite nodes contained by neighbor cubes, which are inside the overlapping area, are in the communication coverage of S. Compared to the actual communication coverage of S, which is drawn by red circle in Fig.2(a), the overlapping area is much smaller. Hence, our method can not find nodes inside the non-overlapping area.



(a) Netgrid partition in layer 1 (b) Netgrid partition in layer 2

Fig. 2. Example about netgrid partition method in different layer.

In this case, the overlapping area of all nodes contained by one netgrid is the same. We define *effective coverage* of a satellite node inside a netgrid, which is the overlapping area of communication coverage in each extreme location of the netgrid. *effective netgrids* is defined as neighbor netgrids contained by *effective coverage* of a satellite node. In Fig.2(a), *effective coverage* of S is the colorful area and *effective netgrids* of S are the cubes contained by the *effective coverage*.

Considering the three-dimensional space as shown in Fig.1, we assume the volume of *effective netgrids* and *effective coverage* as V_{cube}^l and V_{ol}^l, respectively. The communication radius for all satellite nodes is assumed as R_a, then the side length of a cube in Fig.2(a) should be $\frac{\sqrt{3}}{6}R_a$. We also give a definition about accuracy of the netgrid.

Definition 1. The accuracy of the netgrid is defined as the ratio between the volume of *effective netgrids* V_{cube}^l and the volume of spherical communication coverage.

However, the partition method in Fig.2(a) still sacrifices the accuracy too much as *effective netgrids* of S is much smaller than the actual communication coverage of S. If we reduce the size of each cube in Fig.2(a), the volume of *effective netgrids* will increase and become closer to the actual communication coverage of S and the accuracy can be improved. In order to achieve this purpose, the partition method in Fig.2(a) is regarded as layer 1 of netgrid partition, and dichotomy method is adopted based on partition method in layer 1 to reduce the size of netgrid and improve the accuracy of netgrid partition method. Then the partition method in Fig.2(b) represents the layer 2 of netgrid partition. Let L_l represent the side length of a cube in layer l. Then the side length of a cube in different layer l should be $L_l = 2L_{l+1} = \frac{\sqrt{3}}{3 \cdot 2^l}R_a$.

Lemma 1. In multi-layer netgrid partition method, when $l \to \infty$, the volume of each netgrid in layer l approaches zero, and

the volume of *effective netgrids* V_{cube}^{l} and *effective coverage* V_{ol}^{l} approaches spherical communication coverage.

Proof. As shown in Fig.2(a), θ can be calculated by

$$\theta = \arccos[3 \cdot 2^{l-1} + \frac{1}{2^{l+2}} - \frac{1+\sqrt{1+\frac{2^{l+1}-1}{2^{l}} \cdot R_a}}{2^{1-l} \cdot R_a} - \frac{1}{4}],$$

and the side length of a can be calculated by

$$a = \Big(\sqrt{1+\frac{2^{l+1}-1}{2^{l}}R_a^2} - 1\Big)/2.$$

Depending on the knowledge of solid geometry, we can calculate the volume of *effective coverage* in layer l by

$$V_{ol}^{l} = \tfrac{\theta}{180}2\pi \cdot \tfrac{4}{3}\pi R_a^3 + \tfrac{1}{3}\pi R_b^2(\sqrt{R_a^2 - R_b^2} - \sqrt{a^2 - R_b^2}),$$

where $R_b = R \cdot \sin\theta$. The volume of *effective netgrids* is

$$\begin{aligned} V_{cube}^{l} &= \Sigma_{n=1}^{n^*} V_n^l + (3L_1 + L_2 + ... + L_l)^3 \\ &= \Sigma_{n=1}^{n^*} V_n^l + (\frac{2^{l+1}-1}{2^{l}} \cdot R_a)^3, l \in \mathbf{Z}, \end{aligned}$$

and the volume of cube V_n^l in level n is

$$V_n^l = \Sigma_{n=1}^{n^*}[6 \cdot (2m_n + 1)^2] \cdot (L_l)^3,$$

for a given l, there exists a equation

$$f_l(n, m_n) = \sqrt{R_a^2 - \sqrt{(m_n+1) \cdot L_l^2}} - (2^l + n - \tfrac{1}{2}) \cdot L_l,$$

and it must satisfy $f_l(n, m_n) > 0$, then we assume the maximum level n^* and maximum number of cubes m_n^* in level n, they can be calculated by

$$m_n^* = argmin[f_l(n, m_n)], n, m_n \in \mathbb{N},$$
$$n^* = argmin[f_l(n, 0)], n \in \mathbb{N},$$

an example about the maximum level n^* and m_n^* in layer 2 is shown as Fig.2(b). When $l \to \infty$ then $n \to \infty$, we have $\lim_{l\to\infty} V_{ol}^{l} = \lim_{l\to\infty, n\to\infty} V_{cube}^{l} = \frac{4}{3}\pi \cdot R_a^3$. □

C. Complexity Analysis

As we analyzed above, there exist two steps to obtain a sub-graph in TGM. First, calculating the coordinate of N vertices; Second, calculating the distance between each two vertices according to their coordinates to get all available edges. The computation complexity of former step is $\mathcal{O}(N)$, and latter step is $\mathcal{O}(N^2)$. Although this method can provide detailed information of the network topology, it can not scale up as the computation complexity of each sub-graph calculation is $\mathcal{O}(N^2)$. The rising of N rapidly increases computation cost.

In TNM, there also exist two steps to obtain a sub-table. First, calculating the coordinate of N nodes; Second, allocating nodes into the netgrid according to their coordinates. Thus, the computation complexity of these two steps in TNM both is $\mathcal{O}(N)$, which makes the computation complexity of each sub-graph in TGM decrease to $\mathcal{O}(N)$. Note that different partition layer does not influence the computation complexity of TNM but influence the routing performance and the number of netgrid table read operations. In this way, TNM can scale up with low computation cost compared to TGM.

Algorithm 2: Earliest Arrival Space Routing Algorithm

Input: Vertex Set V, Source node $s \in V$, Source set $S^* \subset V$, TTL of data T^L, Searching starttime t_s, Maximum integer value $\mathcal{M}$;

1 **Initialize:** Searching endtime $t_e = t_s + T^L$, $S^* = \{s\}$, RouterTable $\mathcal{R} = \{time(n), p(n) | n \in V\} = \oslash$, $\mu = true$, $T_{min} = \mathcal{M}$, $p_{min} = \oslash$, $time(s) = t_s$;
2 **while** $\mu == true$ **do**
3 $\mu = false, T_{min} = \mathcal{M}$;
4 **foreach** $n_s \in S^*$ **do**
5 $\mathcal{N}(n_s) \leftarrow$ Search available neighbor nodes of n_s from netgrid table $\mathcal{G}$ during $[time(n_s), t_e]$;
6 **foreach** $n_n \in \mathcal{N}(n_s)$ **do**
7 $\varepsilon = 0$;
8 **if** $n_n \in S^*$ **then**
9 **Continue** loop in line 6;
10 **end**
11 **if** $time(n_s) \geq t_e$ **then**
12 **Continue** loop in line 6;
13 **end**
14 **if** $time(n_s) < starttime(e(n_s, n_n))$ **then**
15 $\varepsilon = starttime(e(n_s, n_n)) - time(n_s)$;
16 **end**
17 $time(e(n_s, n_n)) = \frac{B}{R(n_s, n_n)} + \varepsilon$;
18 $time(n_n) = time(n_s) + time(n_s, n_n)$;
19 **if** $time(n_n) > endtime(e(n_s, n_n))$ **then**
20 **Continue** loop in line 6;
21 **end**
22 **if** $time(n_n) < T_{min}$ **then**
23 $T_{min} = time(n_n)$, $n_{min} = n_n$, $\mu = true$;
24 $p_{min} = p(n_s) \cup e(n_s, n_n)$;
25 **end**
26 **end**
27 **end**
28 **if** $\mu == true$ **then**
29 $time(n_{min}) = T_{min}$, $p(n_{min}) = p_{min}$;
30 $S^* = S^* \cup \{n_{min}\}$;
31 **end**
32 **end**
33 **Return** RouterTable $\mathcal{R}$

IV. Earliest Arrival Space Routing Algorithm

In order to overcome the weakness of terrestrial routing algorithms in satellite networks, we propose a routing algorithm based on TNM which considers temporal aspect of the network topology. And we also prove that the proposed algorithm can find the earliest arrival path in time-varying networks.

A. Algorithm Description

In this section, an Earliest Arrival Space Routing (EASR) algorithm is proposed to solve optimal path-finding problem in time-varying networks. Our method borrows ideas from Dijkstra Algorithm to calculate the earliest path from a given

source node to any other nodes based on TNM. The process of EASR is described in Algorithm 2.

EASR will be executed when a satellite needs to transmit data. In order to the find earliest arrival path, our method borrows Dijkstra's greedy approach, and acquires the information of network topology based on TNM. For each node n_s in line 4, EASR reads netgrid table to find the discrete coordinate of the netgrid which contains n_s. Then EASR will read neighbor netgrids based on this discrete coordinate in each sub-table and find available neighbor nodes of n_s in line 5. Note that the number of neighbor netgrids that EASR reads in a sub-table is rising with the increase of netgrid partition layer.

Let B and $R(s, n_n)$ represent the size of data and transmission rate between node s and node n_n, respectively. For each data D, it has a duration of Time To Live (TTL). Let T^L represent the duration of TTL, when D expires T^L and it still does not reach destination node, D will be abandoned by satellite nodes. In this case, EASR only considers the network topology during $[t_s, t_e]$, which means EASR does not need to access all sub-tables in the netgrid table.

B. Proof of Earliest Arrival Path

Lemma 2. If there is at least one path in timed evolving graph G linking source node s to destination node d, then, among all paths linking s to d, there exists at least one earliest path that all its prefix paths are themselves earliest paths.

Proof. The proof process is described in [12]. □

Lemma 3. EASR routing algorithm finds the earliest arrival path that all its prefix paths are themselves earliest paths.

Proof. At first, we prove that EASR finds the earliest arrival path. The procedure is similar to Dijkstra Algorithm in [13]:

1. At the beginning, source set S^* only contains source node $s \in V$ and the start time of searching is t_s. After searching available neighbor nodes of s in line 5, we will get neighbor set $\mathcal{N}(s)$, which contain all reachable edges from s to the next hop during $[t_s, t_s + T^L]$. Assuming node n_n is the neighbor node of s and its time cost is minimum among all neighbor nodes in $\mathcal{N}(s)$. Then, at the end of first iteration in line 30 to line 34, source set will add n_n into S^* and path $p = \{e(s, n_n)\}$ is an earliest arrival path from s to n_n.

2. After $k-1$ iterations in line 2, source set already has k nodes in S^* with $k < N$. Assume that the algorithm finds k-th node n_k and path $p_s(n_k)$ has minimum time cost in k-th iteration. If p^* is the earliest arrival path from s to n_{k+1} and $p_s(n_k)$ is not, let n^* represent the first node which is not included by S^* in the earliest arrival path p^*. Considering nodes in S^* must have found the earliest arrival path, then the time cost of $p_s(n_j)$ and $p_s(n_k)$ must have $time(p_s(n^*)) \leq time(p_s(n_k)) < time(p^*)$, thus we obtain a contradiction from the assumption that node n_k has minimum time cost among all nodes from source node s in $V - S^*$. Then, the path $p_s(n_k)$ that EASR found must be the earliest arrival path from s to n_k.

Second, we prove that all earliest arrival paths found by EASR must satisfy the earliest prefix paths property:

EASR finds an earliest arrival path $p_s(d)$ from s to d and $p_s(d)$ does not satisfy the property that all its prefix paths are earliest paths. According to Lemma 2, we assume path p_1^* is the only path which satisfies the property from s to d, and $p_s(d) \neq p_1^*$. At the middle of path $p_s(d)$, there must exist two continuous nodes, n_i and n_{i+1}, and the prefix path of $p_s(d)$ from s to n_i satisfies the earliest prefix paths property but from s to n_{i+1} does not, which means the path from n_i to n_{i+1} that EASR found is not earliest path. Due to process of EASR, after n_i is added into S^*, n_{i+1} must have minimum time cost among all neighbor nodes of n_i then it can be added into S^*. In this case, the path from n_i to n_{i+1} that EASR found must be the earliest path, which is a contradiction from the assumption. Therefore, the path $p_s(d)$ found by EASR must satisfy the earliest prefix paths property.

□

V. Simulation

The analysis in Section III-C shows that the proposed TNM can decrease the computation complexity from $\mathcal{O}(N^2)$ to $\mathcal{O}(N)$ compared with TGM, which is realized by sacrificing the information of the satellite network topology. It is no doubt that the sacrifice will influence the performance of EASR. In this section, we evaluate the performance of routing algorithms based on TNM and TGM in the simulation, and also make the comparison about computation cost.

A. Configuration

In our simulation, ONE simulator is adopted and simulation platform is: Core i3-4150 CPU, 3.50GHz, 12G RAM. We adopt the scenario of LEO satellite network and construct walker constellation with parameters $18/3/0°$. The computation model of satellite orbits is two body model and transmission rate is 100Kb/s. The communication radius and orbit radius are 500km and 340km, respectively. The update interval of simulator is 1 minute, and total simulation time is 12 hours.

We fix source node and destination node in two separate orbit planes and mainly evaluate the performance of 3 routing schemes in terms of delivery ratio. The prediction time of TNM and TGM is set as the same as T^L, and the number of time spans $M' = M = T^L$. Note that 3 routing schemes are:

Scheme 1: EASR based on TGM with detailed satellite network topology information.

Scheme 2: EASR based on TNM in partition layer 1.

Scheme 3: EASR based on TNM in partition layer 2.

B. Simulation Results

In the analysis of computation complexity before, we only consider the computation cost of a temporal graph or a netgrid table and ignore the cost of memory reading operation. Here, we consider the CPU cycle consumption of calculating a whole temporal graph or netgrid table and the consumption of reading operation when performing the routing algorithm once.

Fig.3 plots the CPU cycle consumption of 3 routing schemes. As we can see, our schemes are rising linearly with the increase of network scale, and the consumption of

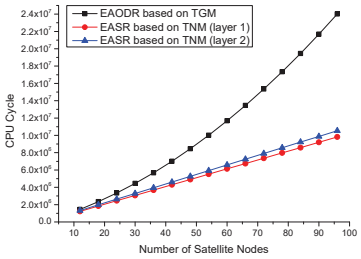
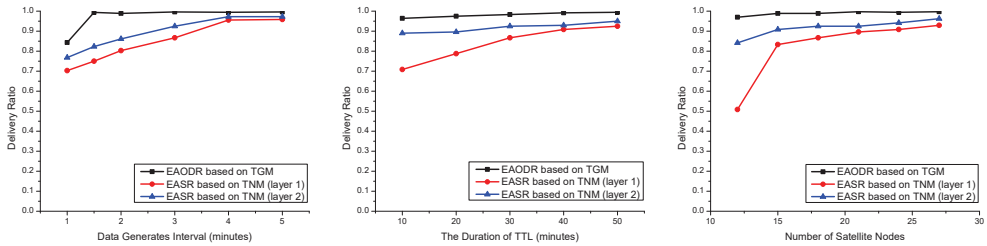


Fig. 3. Comparison performance of CPU cycle consumption.



(a) versus Data generation interval (b) versus Duration of TTL (c) versus Number of satellites

Fig. 4. Comparison of delivery ratio among 3 routing schemes. (Data Size= $5M$, $T^L = 30m$ and $N = 18$ in (a), Data generation interval= $3m$ and $N = 18$ in (b), Data generation interval= $3m$ and $T^L = 30m$ in (c).)

Scheme 1 is rising rapidly with the increase of network scale. This is due to the fact that the computation complexity of TGM calculation is $\mathcal{O}(N^2)$, and the reading consumption from temporal graph is associated with number of edges when the routing algorithm is performed. However, computation complexity of TNM calculation is $\mathcal{O}(N)$ and the reading consumption from the netgrid table is constant as neighbor netgrids are fixed in a specific partition layer.

The simulation results in Fig.4 show the routing performance in terms of delivery ratio among 3 routing schemes. Intuitively, Scheme 1 always has the best performance compared with Scheme 2 and 3. However, compared to our schemes, which are Scheme 2 and 3, high computation complexity of TGM is shown in Fig.3 as expected. In our schemes, Scheme 3 always has better performance than Scheme 2, this is due to the fact that higher partition layer in TNM can obtain more detailed information about the network topology.

As T^L equals the prediction time of TNM and TGM, the probability that routing algorithms find a path to the destination is rising with the increase of T^L, and results in Fig.4(b) verifies our explanation. In Fig.4(a), traffic loads influence our schemes significantly since EASR only considers the shortest path and does not consider traffic loads.

Fig.4(c) shows the influence about the number of satellite nodes. Although TNM sacrifices the information about the network topology compared with TGM, when network scale becomes large, our schemes can approach the optimal scheme and the gap between Scheme 1 and 3 is only 4% when the number of satellites increases to 27. This can be explained as follows. With the increase of network scale, transmission opportunities will rise and our schemes based on TNM will have more probability to obtain a path from source to destination. Considering the computation complexity in Fig.3, Scheme 2 and 3 can provide near-optimal performance with low computation cost, so our schemes are more applicable for large-scale satellite networks.

VI. Conclusion

In this paper, we propose a novel model named TNM to describe the time-varying network topology. Our model, compared with traditional model such as TGM, can reduce computation cost significantly. That is to say, the computation complexity is reduced from $\mathcal{O}(N^2)$ to $\mathcal{O}(N)$. Then we propose a routing algorithm based on TNM to solve the path-finding problem in satellite networks, and also prove that the proposed routing algorithm can obtain the earliest arrival path. The simulation results show that the proposed algorithm achieves a significant reduction in computation complexity as well as acceptable routing performance in terms of delivery ratio compared with traditional TGM based routing algorithm.

VII. Acknowledgement

This work was supported in part by the National Science Foundation of China (No.91538203, No.61390513) and the Fundamental Research Funds for the Central Universities.

References

[1] H. Nishiyama, Y. Tada, and N. Kato, "Toward optimized traffic distribution for efficient network capacity utilization in two-layered satellite networks," in *IEEE Trans. Veh. Technol.*, vol. 62, pp. 1303–1313, March 2013.

[2] E. K. Jaff, P. Pillai, and Y. F. Hu, "IP multicast receiver mobility support using PMIPv6 in a global satellite network," in *IEEE Commun. Mag.*, vol. 53, pp. 30–37, March 2015.

[3] D. K. Geller, D. Crocket, R. Christensen, and A. Shelley, "Advantages of small satellite carrier concepts for LEO/GEO inspection and debris removal missions," in *International Journal of Space Science and Engineering*, vol. 2, pp. 115–134, 2014.

[4] Y. Hongcheng, Q. Zhang, and Y. Sun, "A novel routing scheme for LEO satellite networks based on link state routing," in *Proc. IEEE CSE*, pp. 876–880, December 2014.

[5] N. Bezirgiannidis, C. Caini, and D. D. P. Montenero, "Contact graph routing enhancements for Delay Tolerant Space communications," in *Proc. IEEE ASMS*, pp. 17–23, September 2014.

[6] C. Caini and R. Firrincieli, "Application of contact graph routing to LEO satellite DTN communications," in *Proc. IEEE ICC*, pp. 3301–3305, June 2012.

[7] G. Araniti, N. Bezirgiannidis, and E. Birrane, "Contact graph routing in DTN space networks: overview, enhancements and performance," in *IEEE Commun. Mag.*, vol. 53, pp. 38–46, March 2015.

[8] S. E. Alaoui, S. Palusa, and B. Ramamurthy, "The interplanetary internet implemented on the geni testbed," in *Proc. IEEE GLOBECOM*, pp. 1–6, 2015.

[9] S. E. Alaoui and B. Ramamurthy, "Routing optimization for DTN-based space networks using a temporal graph model," in *Proc. IEEE ICC*, pp. 1–6, May 2016.

[10] J. Du, C. Jiang, S. Yu, and Y. Ren, "Time cumulative complexity modeling and analysis for space-based networks," in *Proc. IEEE ICC*, pp. 1–6, May 2016.

[11] L. Lei and H. Li, "A routing policy based on time-varying graph for predictable delay tolerant networks," in *Proc. IEEE WCSP*, pp. 1–6, October.

[12] B. B. Xuan, A. Ferreira, and A. Jarry, "Evolving graphs and least cost journeys in dynamic networks," in *Proc. WiOpt*, March 2003.

[13] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to algorithms*. Cambridge: MIT press, 2001.